Probing the Stellar Initial-Final Mass Relation

By

Richard Bruce Baxter

A thesis submitted to Macquarie University
for the degree of
Master of Philosophy
Department of Physics
July 2011



© Richard Bruce Baxter, 2017.

Typeset in LaTeX 2$_\varepsilon$.

Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree in any university.

Richard Bruce Baxter

Acknowledgements

I would like to thank my supervisors Dr Paul Dobbie and Professor Quentin Parker for their careful guidance and constant support.

This thesis is dedicated in memory of my grandmother Mrs Betty Baxter who died in 2010.

- Richard

Abstract

I have tested our current understanding of the form of the stellar initial-final mass relation (IFMR) by performing a preliminary study of twelve (12) wide double-degenerate binaries. I identified these systems by applying colour and magnitude selection criteria, in software, to identify blue object close pairs in the imaging data of the Sloan Digital Sky Survey (SDSS). A selection of these were then targeted for spectroscopic follow-up while performing astrometric analysis to determine if the components in each had significant, common proper motions. Subsequently, I used existing grids of white dwarf synthetic spectra and evolutionary models to interpret the spectroscopic data. The white dwarf parameters of mass, $M_{\rm f}$, and cooling time, $t_{\rm cool}$, were then used to probe the form of the IFMR. Using the white dwarfs in my sample whose derived parameters confirmed they were coeval and physically associated, I have also constructed a preliminary mass distribution for the components of these binary systems.

Contents

Acknowledgements **v**

Abstract **vii**

List of Figures **xiii**

List of Tables **xix**

1 Introduction **1**
1.1 Overview . . . 1
1.2 White Dwarf and their fundamental Physics . . . 2
1.2.1 White Dwarf Cooling . . . 3
1.2.2 White Dwarf Spectral Classification . . . 4
1.3 Stellar Evolution . . . 4
1.3.1 Main sequence (MS) . . . 4
1.3.2 Red Giant Branch (RGB) . . . 5
1.3.3 Horizontal Branch . . . 6
1.3.4 Asymptotic Giant Branch (AGB) . . . 7
1.3.5 White Dwarf Cooling Sequence . . . 7
1.3.6 White Dwarf Survey Statistics . . . 8
1.4 The stellar initial-final mass relation (IFMR) . . . 11
1.4.1 A light introduction to the stellar IFMR . . . 11
1.4.2 Using open clusters to map the IFMR . . . 13
1.4.3 IFMR Calculations To Date . . . 14
1.4.4 Sources of Uncertainty . . . 19
1.5 Thesis Outline . . . 21

2 Data **23**
2.1 The identification of candidate wide double-degenerate systems . . . 23
2.1.1 Summary . . . 23
2.1.2 The SDSS Data Release 7 . . . 23
2.1.3 Initial Identification of candidate wide DD systems . . . 25
2.1.4 Simulations . . . 28
2.2 Follow-up spectroscopic observations . . . 38
2.2.1 Prioritisation of candidate wide DD systems for follow-up . . . 38

2.2.2 The spectroscopic follow-up observations . . . 45
2.3 Data Reduction . . . 49
2.3.1 Spectroscopic Data Reduction . . . 49
2.4 Data modelling . . . 57
2.4.1 Atmospheric modelling . . . 57
2.4.2 Evolutionary modelling . . . 61
2.4.3 Proper motion verification of physical association . . . 67

3 Results **75**
3.1 Properties of the components of the wide double-degenerate systems. . 75
3.1.1 Hot DA + hot DA systems . . . 75
3.1.2 Cool DA + hot DA systems . . . 76
3.1.3 Cool DA + cool DA systems . . . 76
3.1.4 HFMWD (DAH) + hot DA systems . . . 76
3.1.5 HFMWD (DAH) + cool DA systems . . . 77
3.1.6 Estimation of the uncertainties in derived parameters . . . 77
3.2 The double-degenerate systems on a case-by-case basis . . . 77
3.2.1 DD-01 . . . 78
3.2.2 DD-02 . . . 79
3.2.3 DD-03 . . . 80
3.2.4 DD-04 . . . 81
3.2.5 DD-05 . . . 81
3.2.6 DD-06 . . . 82
3.2.7 DD-07 . . . 83
3.2.8 DD-08 . . . 84
3.2.9 DD-09 . . . 85
3.2.10 DD-10 . . . 86
3.2.11 DD-11 . . . 87
3.2.12 DD-12 . . . 88
3.2.13 My systems overall and understanding of the IFMR . . . 88
3.3 IFMR probing . . . 94
3.4 A preliminary investigation of the double degenerate mass distribution 105
3.4.1 Comparing the DD mass distribution to the field WD mass distribution . . . 110
3.4.2 Why might the DD mass distribution be skewed to higher mass? 114
3.5 A preliminary investigation of the double degenerate magnetic WD distribution . . . 114

4 Moving forward **115**
4.1 Further optical spectroscopy . . . 115
4.1.1 Non-magnetic systems . . . 115
4.1.2 Systems containing HFMWDs . . . 116
4.2 Preliminary investigation of IFMR form derivation . . . 119
4.3 Conclusion . . . 121

A Appendix **131**
A.1 Overview . . . 131
A.1.1 Standard Libraries . . . 131
A.2 Target Selection Code and Data . . . 131
A.2.1 SDSS SQL Scripts for blue object list generation . . . 131
A.2.2 Count Close Pairs within a BO List Code . . . 132
A.2.3 Count Close Pairs within a Random Distribution Code . . . 142
A.2.4 Plot Clustering Code . . . 153
A.2.5 Create Blue Object Close Pair List (BOCPL) . . . 153
A.2.6 Create BOCPL Subset using Is Quasar Data . . . 160
A.2.7 create BOCPL Subset using Visual Check Bad List . . . 160
A.2.8 create BO Subset Using Generic Ra and Dec Data . . . 160
A.2.9 create BOCP Subset Using Generic Ra and Dec Data . . . 164
A.2.10 create BOCPL Subset Using Bergeron Theoretical Data . . . 167
A.2.11 Calculate probability of chance alignment using distance limited volume method . . . 172
A.3 Proper Motions Code and Data (Target Selection Part II) . . . 174
A.3.1 Proper Motion Calculation Procedure . . . 174
A.3.2 Proper Motion Calculations in Pixels . . . 176
A.3.3 Proper Motion Calculations in RA/Dec (Calibration of pixel values) . . . 179
A.3.4 Calculation Probability of Chance Proper Motion Alignment - PDD Method . . . 187
A.3.5 Calculation Probability of Chance Proper Motion Alignment - RBB Method . . . 198
A.4 Observations Code and Data . . . 205
A.4.1 calculate the angle between WD components in double degenerate system (for aligning slit) . . . 205
A.4.2 Flux Calibrated WD Data . . . 207
A.4.3 WHT Data Reduction . . . 208
A.4.4 Gemini South/North Data Reduction . . . 228
A.4.5 VLT Data Reduction . . . 241
A.4.6 Flux Calibration . . . 241
A.4.7 Signal to Noise Calculations . . . 242
A.5 Stellar Modelling Scripts . . . 245
A.5.1 Balmer Line Extraction . . . 245
A.5.2 White Dwarf Parameter Extraction . . . 253
A.5.3 Tlusty And Koester ASCII Model Grid Comparison Using GDL 269
A.5.4 Interpolate ASCII Spectroscopic Model Grid and create ASCII model using Best Fit Temperature and Surface Gravity values . 271
A.5.5 White Dwarf Parameter Extraction - including gravitational redshift values - NOT USED . . . 274
A.5.6 Flux Recalibration using Best Fit ASCII Spectroscopic Model . 274

A.5.7 Interpolate ASCII Main Sequence Stellar evolutionary model grid using theoretical main sequence mass or age values to calculate a best fit main sequence age or mass . . . 277
A.5.8 ASCII Model Grid Creation Software - performed by PDD and Koester . . . 278
A.5.9 XSpec Software . . . 279
A.5.10 PDD IDL ASCII Model Interpolation Software Pre-requisites - NOT USED . . . 294
A.5.11 Model Grids . . . 294
A.5.12 Xspec Model Files . . . 295
A.5.13 Operational Manuals . . . 297
A.6 Data Analysis . . . 300
A.6.1 Calculate Main Sequence Ages . . . 300
A.6.2 compare DD System Age Difference Distribution Across Published IFMRs . . . 323
A.6.3 Distance Moduli Comparison of DD components . . . 334
A.6.4 DD Mass Distribution Comparison with Field WD Survey . . . 337
A.7 List of Documentation . . . 346
A.7.1 Object Table . . . 347
A.7.2 Target Selection Mass Bias Confirmation . . . 389
A.7.3 Target Selection Basic WD Population Synthesis . . . 391
A.7.4 Private Projects . . . 391
A.7.5 WHT Observing Proposals . . . 391
A.7.6 WHT 2008A Proposal . . . 391
A.7.7 Gemini Observing Proposals . . . 398
A.7.8 VLT Observing Proposals . . . 416
A.8 original analytical method for determining the properties of the components of the wide double-degenerate systems . . . 425
A.8.1 Uncertainty Calculations . . . 425
A.9 example IDL routine . . . 427

List of Symbols **431**

References **433**

List of Figures

1.1 Diagram of Stellar Evolution from Herwig (2005). This diagram shows the evolution of a low-intermediate mass star, from the Main-sequence, to the Red Giant Branch (RGB), Horizontal Branch (ZAHB), Asymptotic Giant Branch (AGB), and finally the post-AGB phase with the formation of a central star of a planetary nebula (CSPN) afterwards becoming a WD. 6

1.2 HST Spectrum of the heliux nebula (NGC7293) planetary nebula central star, compared with three models with different effective temperatures (Traulsen et al. 2005). 8

1.3 The SDSS WD mass distribution as determined by Kepler et al. (2007). Histogram for the 1859 DA stars brighter than $g = 19$ and hotter than $T_{\rm eff} = 12000K$, compared to the PG survey published by Liebert et al. (2005a) and the SDSS DR1 sample published by Madej et al. (2004). Gaussian fits are also shown. The bins are $0.025M_{\odot}$ wide. 10

1.4 The IFMR (Casewell et al. 2009). A state-of-the-art representation of the known form of the IFMR, showing the revised locations of the Praesepe white dwarfs in initial mass-final mass space (filled circles). The locations of the degenerate members of the NGC 6633 (square star), NGC 7789 (cross), NGC 6819 (asterisk), Hyades (open triangles), Sirius binary system (open circle), the Pleiades (open stars), NGC 2168 (open diamonds), NGC 1039 (open squares), NGC 6791 (filled triangles) and NGC 2516 (open '+' signs) are also shown. The relation of Weidemann (2000) (dotted line) and a revised linear fit to 41 white dwarfs (dashed line) are overplotted. The radial velocity variable, WD0837+185 is labelled. 12

1.5 The IFMR (Dobbie et al. 2009). Another state-of-the-art representation of the known form of the IFMR, showing that the bulk of stars appear to follow fairly closely a monotonic relation. The locations of the white dwarf members of NGC 3532 and NGC 2287 in initial mass-final mass space. Data points from a number of other populations with metallicities close to the solar value are also shown. The theoretical IFMR of Marigo and Girardi (2007) (dot - dashed heavy line), the semi-empirical IFMR of Weidemann (2000) (heavy dotted line) and the initial mass-core mass at first thermal pulse relation from Karakas et al. (2002) (medium solid line) are overplotted. The peak in the field white dwarf mass distribution ($\pm 1\sigma$) is represented by the band of grey shading. 16
1.6 Ferrario et al. (2005) volume corrected mass distribution of field WDs in the Palomar-Green survey (Liebert et al. 2005a) (shaded histogram) for stars hotter than $13000K$ with the best linear IFMR fit model (top panel) and the IFMR with curvature (bottom panel). WDs with masses less than $0.5M_{odot}$ are believed to be helium WDs resulting from close binary evolution and are not modelled in this paper 17
1.7 The Liebert et al. (2005a) comparison of their T_{eff} and $\log g$ determinations of PG stars in common with the data sets of Finley and Koester (1997), Vennes et al. (1997), Marsh et al. (1997), Homeier et al. (1998), and Koester et al. (2001). In each panel they plot the differences between these investigations and their results (PG) as a function of their determinations of T_{eff} or $\log g$. Effective temperatures are in units of 10^3K. 20

2.1 A graph of the passbands of the SDSS filters (u, g, r, i, z) with a WD spectrum overlayed 24
2.2 WD photometry selection criteria diagram, from Harris et al. (2003). Colour-colour diagrams ($u-g$ vs. $g-r$) showing the WDs and hot subdwarfs. Different types are shown with different symbols. Small dots show all objects with stellar image profiles with $15 < g < 20$ in a region of $25deg^2$. The curves show the colours of WD model atmospheres Bergeron et al. (1995b) of pure H (solid curves) and pure He (dashed curves) with $\log g = 7$, 8, and 9, where the $\log g = 7$ curve is toward the lower right and the $\log g = 9$ curve is toward the upper left. The dotted lines with labels connect models with the same effective temperature. . 26
2.3 All WD Candidates (Blue Objects from SDSS DR7 Survey) shown with Aitoff projection of galatic coordinates 27
2.4 Plot showing the number of BOCPs found in my simulated random distribution of single objects as a function of separation in arc-seconds (lower line) overlayed with the analytical estimate of Struve 1852. The number of BOCPs measured in the SDSS DR7 to a given maximum separation is also shown (upper line). The difference between these two curves provides a handle on the frequency of physically associated BOCPs found in the SDSS within a given maximum separation. 30

2.5 Same as Figure 2.4 but shown out to a much larger angular separation ($\sim 6'$) . . . 30
2.6 The likelihood (expressed as a percentage) that a BOCP is a physically associated system within a given maximum separation (solid curve). The probability that a BOCP is physically associated at a given separation is also overplotted (dashed curve). . . . 31
2.7 Examples of "bad" objects within my initial sample. These were rejected by my visual inspection of all pairs selected initially. . . . 31
2.8 Finder charts (part 1/3) for the candidate wide double-degenerate systems found in SDSS DR7 ($< 30''$). Note that these are arc $30'' \times 30''$, with North at top end and East at the left. . . . 33
2.9 Finder charts (part 2/3) for the candidate wide double-degenerate systems found in SDSS DR7 ($< 30''$). Note that these are arc $30'' \times 30''$, with North at top end and East at the left. . . . 34
2.10 Finder charts (part 3/3) for the candidate wide double-degenerate systems found in SDSS DR7 ($< 30''$). Note that these are arc $30'' \times 30''$, with North at top end and East at the left. . . . 35
2.11 SDSS Double-Degenerate Survey - Target Selection (1x1$'$). Systems which have been followed up spectroscopically. . . . 39
2.12 Relative response of the WHT ISIS blue arm . . . 43
2.13 VLT FORS2 red and blue CCD quantum efficiency curves . . . 43
2.14 GMOS North (top) and South (bottom) CCD quantum efficieny curves 44
2.15 Flux Calibrated Spectrum - DD01 to DD02 . . . 51
2.16 Flux Calibrated Spectrum - DD03 to DD04 . . . 52
2.17 Flux Calibrated Spectrum - DD05 to DD06 . . . 53
2.18 Flux Calibrated Spectrum - DD07 to DD08 . . . 54
2.19 Flux Calibrated Spectrum - DD09 to DD10 . . . 55
2.20 Flux Calibrated Spectrum - DD11 to DD12 . . . 56
2.21 Bergeron Theoretical WD Cooling Tracts - Temperature versus Age for a given mass. The cooling tracts for both a low mass ($0.5 M_{\odot}$) and a high mass ($1.0 M_{\odot}$ WD are shown. . . . 62
2.22 "Christmas Tree" graphs of my spectra with models overlayed (DD-01 to DD-03). Note DD-02B and DD-08A are missing (as they have been modeled externally using magnetic models). . . . 63
2.23 "Christmas Tree" graphs of my spectra with models overlayed (DD-04 to DD-06). . . . 64
2.24 "Christmas Tree" graphs of my spectra with models overlayed (DD-07 to DD-09). Note DD-02B and DD-08A are missing (as they have been modeled externally using magnetic models). . . . 65
2.25 "Christmas Tree" graphs of my spectra with models overlayed (DD-10 to DD-12). . . . 66
2.26 Proper motion diagrams for the 12 blue object close pairs for which I observed spectroscopic follow-up data (DD01 to DD06). The DD system components are marked as stars. . . . 70

2.27 Proper motion diagrams for the 12 blue object close pairs for which I observed spectroscopic follow-up data (DD07 to DD12). The DD system components are marked as stars. 71

2.28 Colour-Colour for my 12 DD candidates (u-g vs g-r). Bergeron cooling tracts are shown with specific ages ($t_{cool} = 1 \times 10e^4 yrs - t_{cool} = 1 \times 10e^9 yrs$) and temperatures ($T_{\rm eff} = 30000K - T_{\rm eff} = 5000K$) marked. . . 73

3.1 A plot in which my effective spectroscopic mass offsets (to align cool component distances with their hot components) are compared to that of Tremblay et al. (2011) across temperature. The blue curve is that of Tremblay et al. (2011), while the red data points are calculated from my Double Degenerate survey. The error in my derived mass offset is displayed with error bars, while my derived mass offset itself is displayed as a dashed red line about $dM = -0.17 \pm 0.04 M_{\odot}$. 86

3.2 Distance Modulii (m - M) for all DD systems (u, g, r), with cool component mass corrections / distance matching 90

3.3 Distance Modulii (m - M) for DD systems with at least one cool component (u, g, r), without cool component mass corrections / distance matching 90

3.4 Distance Modulii (m - M) for DD systems with at least one cool component (u, g, r), with cool component mass corrections / distance matching 91

3.5 Distance Modulii (m - M) for DD systems with two cool components or a HFMWD system (u, g, r), without cool component mass corrections / distance matching 91

3.6 Distance Modulii (m - M) for DD systems with two cool components or a HFMWD system (u, g, r), with cool component mass corrections / distance matching 92

3.7 Distance Modulii (m - M) for DD systems with hot components (u, g, r) 92

3.8 Plots of all published IFMRs used in this analysis; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009). . . . 96

3.9 Calculated System Age for SDSS DD systems, using known IFMR from Dobbie et al. (2006). 96

3.10 Calculated System Age for SDSS DD systems, using known IFMR from Kalirai et al. (2008). 97

3.11 Calculated System Age for SDSS DD systems, using known IFMR from Williams et al. (2009). 97

3.12 Plots of the system age difference distribution for DD-01 to DD-06, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009). . . . 98

3.13 Plots of the system age difference distribution for DD-07 to DD-12, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009). . . . 99
3.14 Plots of the system age difference distribution for external WDs PG0922+162 (DD-13) and HS2240+125A (DD-14) respectively, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009). 100
3.15 Plots of the system age difference distributions for all DD systems, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For this plot the following published IFMR form has been assumed; Dobbie et al. (2006). 100
3.16 Plots of the system age difference distributions for all DD systems, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For this plot the following published IFMR form has been assumed; Kalirai et al. (2008). 101
3.17 Plots of the system age difference distributions for all DD systems, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For this plot the following published IFMR form has been assumed; Williams et al. (2009).101
3.18 Plots of the initial-final mass for each component WD (of DD-01 through to DD-06) according to the (Dobbie et al. 2006) IFMR (A/B). 103
3.19 Plots of the initial-final mass for each component WD (of DD-07 through to DD-12) according to the (Dobbie et al. 2006) IFMR (A/B). 104
3.20 Kepler SDSS V_{max} Corrected Mass Distribution (Kepler et al. 2007) . . 106
3.21 Liebert PG Survey V_{max} Corrected Mass Distribution (Liebert et al. 2005a)107
3.22 DD mass distribution (Solid) versus the PG Survey Vmax corrected mass distribution (Dotted), the Kepler SDSS DR4 WD mass distribution (Kepler et al. 2007) (Dashed), and the Kepler SDSS DR4 WD Vmax corrected mass distribution (Kepler et al. 2007) (Dash Dot) 108
3.23 Kepler SDSS Survey Mass Distribution Modelled With Gaussians . . . 111
3.24 PG Survey $V_{\max}$ Corrected Mass Distribution Modelled With Gaussians 111
3.25 KS Test Method 2 - Comparison of DD masses with masses derived from Kepler theoretical SDSS WD mass distribution modelled with Gaussians 112
3.26 KS Test Method 6 - Comparison of DD masses with masses derived from Kepler theoretical $V_{\max}$ corrected SDSS WD mass distribution modelled with Gaussians 112
3.27 KS Test Method 5 - Comparison of DD masses with masses derived from Liebert theoretical PG Survey $V_{\max}$ corrected mass distribution modelled with Gaussians 113

4.1 SDSS z band image of two candidate spatially resolved magnetic/non-magnetic double-degenerate systems, CBS 229 (Gianninas et al. 2009) and SDSS J074853.07+302543.5. Images are approximately 1'×1' with N at the top and E to the left. . . . 116
4.2 DR8 Only DD Candidates . . . 118
4.3 A plot of the derivation of the form of the IFMR for each DD system individually, by selecting a linear IFMR form (m and c parameters) resulting in a common system age. A statistical derivation of the form of the IFMR is shown by averaging these parameters (with error bars showing min/max mass/cooling age assumption extremes). This graph also shows the Dobbie et al. (2006) linear IFMR form for comparison (with cited errors). . . . 120
4.4 A second plot of the derivation of the form of the IFMR as in Figure 4.3 - excluding systems with an uncommon distance between components (DD-04 and DD-09). . . . 120
4.5 La Palma - Cloud Valley . . . 122
4.6 La Palma - Volcano . . . 122
4.7 La Palma - INT . . . 123
4.8 La Palma - From Top . . . 123
4.9 La Palma - Sunset . . . 124
4.10 La Palma - Forest Mountain . . . 124
4.11 Cerro Paranal in Chile - Accomodation Paronama . . . 125
4.12 Cerro Paranal in Chile - From Top . . . 125
4.13 VLT Unit Telescope view 1 . . . 126
4.14 VLT Unit Telescope view 2 . . . 126
4.15 VLT Unit Telescope view 3 . . . 126
4.16 VLT Unit Telescope view 4 . . . 127
4.17 VLT Unit Telescope view 5 . . . 127
4.18 VLT Platform 1 . . . 128
4.19 VLT Platform 2 . . . 128
4.20 VLT Platform 3 . . . 129
4.21 Cerro Paranal in Chile - Sunset . . . 130

List of Tables

1.1 A summary of the current white dwarf spectral classification scheme. (The P, H, Z, or X designator gets added after the main classification). Percentage representation is also displayed from Eisenstein et al. (2006). 5

2.1 Blue Object List (BOL) and Blue Object Close Pair List (BOCPL) Statistics - this table identifies the percentage contamination of my sample of blue objects due to quasars. "Bad" objects have not been removed from this sample. Note the 'ideal' case column assumes that the photometric target selection of quasars for spectroscopic follow-up within the SDSS program has been perfect (i.e. every quasar within the sample of blue objects has been targeted for spectroscopic follow-up based on its photometric properties and has been subsequently identified). The 'Worst' case column assumes that the spectroscopic follow-up of quasars within the SDSS program has not been biased based upon photometric data (any more than the effective biasing/probability of spectroscopic follow-up for my sample of blue objects). This table also includes an indication of the expected percentage of WDs within my sample based upon known WDs from Eisenstein et al. (2006), and assumes that the spectroscopic follow-up of objects within the SDSS program has not been biased in any way based upon photometric data towards the selection of WDs (i.e. constraints were purely chosen based upon expected quasar photometric signatures). 32

2.2 Blue Object Close Pair Data - SDSS 36

2.3 Blue Object Close Pair Data - SDSS (continued) 37

2.4 Double-Degenerate Data - SDSS 40

2.5 Telescope Instrument Properties 42

2.6 Double-Degenerate Data - Observations and Signal to Noise 48

2.7 Standard WD Stars 50

2.8 Double-Degenerate Data - Proper Motions 72

2.9 Double-Degenerate Data - Probability of Chance Alignment 73

3.1 Double Degenerate Data - Results 93

3.2 Published forms of the IFMR (linear gradient m and y intersection c parameters). It displays statistics calculated from a monte carlo generated distribution of 25000 hypothetical WDs with parameters selected within their error limits. Firstly it displays the number of DD systems determined to have components with a common system age; [1] using an independent system age comparison, and [2] using the interdependent/-covariant system age difference derivation - from Figures 3.9, 3.10, and 3.11. Secondarily, it displays the median system age difference, averaged across all systems, and the number of DD systems best satisfied by the published IFMR (i.e. having the lowest system age difference with the given published IFMR). 95
3.3 Additional (third party) Double Degenerate systems added to my mass distribution 109

4.1 A table of my preliminary IFMR linear parameter derivations, [1] for all systems, and [2] excluding systems with an uncommon distance between components (DD-04 and DD-09) 119

"He had achieved harmony simply by accepting it."
from Frank Herbert's Children of Dune

1 Introduction

1.1 Overview

The stellar initial-final mass relation (IFMR) describes a theoretical correlation between small to intermediate mass stars on the main sequence and their final mass in the white dwarf (WD) state at the end of their evolution. It defines how much mass is lost during the lifetime of a star in this mass range, i.e. through the difference between the initial mass and the final mass.

The aim of this project is to probe our current estimate of the stellar IFMR using new observations of wide double-degenerate (white dwarf binary) systems uncovered in the Sloan Digital Sky Survey (SDSS) imaging. Current constraints in the form of the IFMR are almost exclusively derived from white dwarf members of open star clusters. These new data will provide an independent verification of the results obtained from open clusters.

The primary tasks in this research project involved, firstly, identifying double-degenerate candidates in imaging data from a pre-existing sky survey through the application of judiciously chosen colour, magnitude and angular separation selection criteria. Secondly, confirmatory follow-up spectroscopy of a selection of the brightest systems was obtained and these data processed using common reduction utilities. Thirdly, white dwarf effective temperatures, surface gravities, masses and cooling times were obtained from these data by comparing them to grids of synthetic white dwarf spectra and evolutionary models. Finally, these data were interpreted in the context of our current understanding of the form of the IFMR.

1.2 White Dwarf and their fundamental Physics

White dwarfs (WDs) are the end product of approximately 98% (Wood 1992) of main sequence (MS) stars. WDs do not generate energy by nuclear fusion, but instead dissipate their stored energy through thermal radiation released over a long period of time (Kippenhahn and Weigert 1990). WDs are extremely dense with $\rho_{centre} \approx 10^7 gcm^{-3}$. They are supported from further gravitational collapse by electron degeneracy pressure. Electron degeneracy pressure is a consequence of Pauli's exclusion principle. Pauli's exclusion principle states that particles of half integer spin (e.g. electrons) cannot occupy the same space (quantum cell), whose dimensions are constrained by Heisenberg's uncertainty relation to be $\approx h^3$, where h is Planck's constant. During the evolution of a star, when the core's primary fuel in the core has been spent, and it does not have a high enough central temperature to generate further energy from nuclear reactions, the core contracts to a higher density. When the particle density in the centre of the star is so great that free electrons (free from their nuclei, ionised under extreme pressure) fill all of the lowest energy quantum cells (limited by the exclusion principle), the remaining electrons are forced into higher momentum states, which results in a degeneracy pressure which halts further collapse.

It has been calculated that the pressure provided by the non-degenerate ions is negligible compared to that of the electrons (Fowler 1926). As degenerate electron velocities do not exhibit significant temperature dependency, the pressure countering gravitational collapse arises primarily due to the degenerate nature of the electrons and not the temperature of the WD (Chandrasekhar 1931). It was also calculated, counter-intuitively, that in WDs, stellar radius is inversly proportional to mass ($R \propto M^{-1/3}$); this contrasts to stellar radius being positively correlated with mass, as is the case for MS stars.

The electron velocities increase in WDs with greater mass, to the level where they become relativistic. The star's hydrostatic balance ultimately becomes independent of radius. In cores of mass greater than the Chandrasekhar limit ($> M_{\mathrm{Ch}} \approx 1.459 M_{\odot}$), the electron degeneracy pressure is unable to support against further collapse. The Chandrasekhar limit is defined by Equation 1.1 (Chandrasekhar 1939) below, where μ_e is the average molecular weight per electron, ≈ 2.

$$M_{\mathrm{Ch}} = (2/\mu_e)^2 \times 1.459 M_{\odot} \tag{1.1}$$

Chandrasekhar made the simplifying assumption that WDs were zero temperature objects ($T = 0K$). More recent zero temperature structural calculations involving a more accurate treatment of the way in which gas particles interact (Salpeter 1961) indicate that for lower densities ($\rho \approx 10^5 gcm^{-3}$) corresponding to lower mass WDs, the Coulomb electrostatic interactions between ions and electrons become significant resulting in less of a reduction in radius as proposed by Chandrasekhar's theory (Hamada and Salpeter 1961). Moreover, for higher densities ($\rho \approx 10^9 gcm^{-3}$) corresponding to higher mass WDs, inverse beta decays occur, where electrons and protons merge to form neutrons, reducing the number of supporting electrons, and therefore lowering the limiting WD mass (Hamada and Salpeter 1961).

1.2.1 White Dwarf Cooling

WDs contain a degenerate interior with a non-degenerate ionised surface for the majority of their observable evolution ($t_{\rm cool} \approx 8$Gyr), during which time their effective temperatures fall from $100000K$ to $5000K$. The interior of most WDs in the Galaxy, whose MS progenitors have had high enough mass to have already evolved into WDs and are not the byproducts of interacting binaries, and many WDs to be formed in the future (MS mass $\gtrsim 0.5M_{\odot}$), are composed of carbon and oxygen. However, the visible exteriors of the majority of WDs are either predominately hydrogen or helium.

The interior of a WD is isothermal, due to the high heat conductivity of the degenerate electron gas. Gravitational contraction or the thermal motions of the degenerate electrons are not the source of WD radiation. Thermal energy cannot be extracted from degenerate electrons to produce the observed radiation, since they already occupy the lowest quantum states limited by the exclusion principle. The energy emitted by a WD comes from the non-degenerate nuclei in the core.

The cooling rate of a WD is dependent upon its radius and its chemical composition, including the thickness of the atmosphere. The size of the WD determines the surface area across which radiation may escape into space. Since higher mass WDs have smaller radii, they take longer to cool (decreased luminosity). However, WDs also lose energy through neutrino emission early in their life times (O'Brien and Kawaler 1996), when their temperatures are still very high, $T_{eff} \approx 100000K$.

For high mass WDs, the amount of energy lost through neutrino emission can exceed the thermal radiative luminosity by a factor of $5 - 10$, so these cool more rapidly in their initial stages than lower mass objects. For example, a $1M_{\odot}$ WD reaches $\log[L_*/L_{\odot}] \approx -0.5$ or $T_{eff} \approx 40000K$ after 3×10^6yr where as a $0.4M_{\odot}$ WD will take 1.5×10^7yr to reach $\log[L_*/L_{\odot}] \approx -0.5$ or $T_{eff} \approx 40000K$ (Wood 1990).

Once the white dwarf has cooled significantly ($T_{eff} \lesssim 5000K$), the ions in the central regions of the WD begin to crystallise (Lamb and van Horn 1975). Both the latent energy and the gravitational energy released during crystallisation (due to changes in the distribution of carbon and oxygen atoms as the oxygen crystallises at higher temperatures and shifts towards the centre of the star) reduces the cooling rate temporarily. For example, theoretically a $0.6M_{\odot}$ CO WD takes $\approx 20\%$ longer to cool to $\log[L_*/L_{\odot}] = -4.5, T_{eff} \approx 3800K$ (Isern et al. 1997) if crystallisation is taken into account. After crystallisation, the cooling rate is greater for higher mass WDs, where the heat capacity of the non-degenerate ions is proportional to T^x where $x > 1$.

As the degeneracy pressure support of the electrons is relatively weakly dependent on temperature, WDs cool at approximately constant radii. However, a $0.6M_{\odot}, T = 80000K$ WD with a (non-degenerate) helium surface layer of $0.02M_{\odot}$ has a radius 36% larger than a pure C degenerate model (Hamada and Salpeter 1961). This effect is even greater with a surface envelope composed of hydrogen.

It has proved difficult to confirm the theoretical WD mass-radius relationship (e.g. Schmidt 1996; Provencal et al. 1997). This is because it is challenging to measure the mass and the radius independently with current instrumentation and the WD mass distribution is sharply peaked at around $0.6M_{\odot}$ (Gennaro et al. 2008). The Hipparcos satellite has been used to establish very accurate distances to some of the closest WDs

in binaries, including Sirius B, 40 Eri, and Procyon B. This has allowed mass and radius to be reliably determined for these objects. The radius/mass values calculated are in reasonable support of theoretical mass-radius relations (Provencal et al. 1998).

1.2.2 White Dwarf Spectral Classification

WDs are primarily classified according to their optical spectra. The current classification scheme uses the prefix character 'D' to specify the degenerate nature of the object, which takes into account all WDs with $\log g \gtrsim 7.0$ (McCook and Sion 1999, Wesemael et al. 1993). The next characters specify the composition of the atmosphere, from the most dominant element observable to the least dominant element observable. The character 'A' is used to designate hydrogen exhibited by strong Balmer lines, observed with temperatures $5000K \lesssim T \lesssim 80000K$. The character 'O' is used to designate helium where the `HeII` line at 4686Å is dominant with possible weak `HeI` lines at 4471Å and 4921Å including weak `H`, observed with temperatures $T \gtrsim 50000K$. The character 'B' is used to designate helium where only `HeI` lines are present, observed with temperatures $12000K \lesssim T \lesssim 30000K$ (Sion et al. 1983). The character 'C' is used to specify atmospheres lacking in distinct hydrogen and helium features, observed with temperatures $5000K \lesssim T \lesssim 12000K$. It is presumed many of these are low temperature helium atmosphere WDs. In this scheme, for example, a hydrogen rich WD is classified as a DA, and a table giving the full classification scheme here is shown in Table 1.1.

A number ranging from 1 to 10, immediately after the atmospheric composition identification character, may be used to specify the temperature, where it approximates to $50400/T_{eff}$. T_{eff} here is generally calculated based upon the colour of the star (Wesemael et al. 1993). For example, DA4 would indicate a 12600K DA WD. Additional letters are used to specify secondary spectral characteristics, indicating the presence of additional but less abundant elements, and/or the presence of an observed magnetic field. The character Z is generally used to designate weak metal features. P or H are used to designate a polarised and unpolarised magnetic field respectively (Sion et al. 1983). The character Q is used to designate the presence of carbon.

For example, a DAO object exhibits strong `H` Balmer lines with a weak `HeII` absorption line at 4686Å. For example also, a hydrogen rich WD with a magnetic field is classified as a DAH. Objects with strange or unidentifiable spectra are given the classification X.

1.3 Stellar Evolution

1.3.1 Main sequence (MS)

Stars fuse hydrogen into helium in their cores for approximately 90% of their lives, where an equilibrium is maintained between the gravity of the star pulling its mass inwards, and the energy and pressure generated in the core by the nuclear fusion effectively pushing its mass outward. This phase is known as the Main Sequence

Classification	Observed spectral lines	Percentage Representation
DA	H Balmer only	85.9%
DB	HeI only (4471Å and 4921Å)	7.7%
DO	Strong HeII (4686Å), possible weak HeI (4471Å and 4921Å) and H Balmer	0.3%
DC	No features exhibited stronger than 5% of continuum flux level	3%
DQ	Carbon features present	3.1%
DZ	Metallic features	1.4%
DH	Magnetic, no observable polarisation	0.1%
DAB	Strong H Balmer with HeI (4471Å and 4921Å)	-
DAO	Strong H Balmer with HeII (4686Å)	-
P	Magnetic, observable polarisation	
H	Magnetic, no observable polarisation	
X	Peculiar or unidentifiable	
Z	Metallic features	

Table 1.1: A summary of the current white dwarf spectral classification scheme. (The P, H, Z, or X designator gets added after the main classification). Percentage representation is also displayed from Eisenstein et al. (2006).

(MS) on the Hertzsprung-Russell (HR) diagram (see Figure 1.1). The mass of a star determines its hydrogen burning MS, including stellar temperature, luminosity, and age. The nuclear reaction rates, being temperature dependent, are higher in the cores of more massive stars which have hotter and denser cores. For example, $1.0M_{\odot}$ stars like our Sun have MS lifetimes of 1.0×10^{10}yr , while $4.0M_{\odot}$ stars have MS lifetimes of only 1.5×10^{8}yr (Girardi et al. 2000). As a consequence of this and the finite age of the Galaxy, stars less than $\approx 0.8M_{\odot}$ cannot yet have completed their main sequence lives and should not yet have evolved into WDs.

1.3.2 Red Giant Branch (RGB)

After most of the hydrogen has been transformed into helium in the core, nuclear energy release ceases, the radiative pressure decreases, and the core starts to resume gravitational contraction. The gravitational potential energy released in turn raises the temperature such that hydrogen fusion continues in a thin shell around the inert helium core. The stellar envelope expands as a result of the shell burning. The theoretical basis for this expansion of the star is still to be fully established, e.g. see Iben and Livio (1993). As the core of the star continues to contract, the shell luminosity increases and the envelope of the star expands. As the star's radius increases, its surface temperature decreases, and the star becomes a Red Giant on the Red Giant Branch (RGB) of the HR diagram (see Figure 1.1).

The outer layers of the star are cooled by the expansion of the envelope, which results in a change in its transparency to radiation. The envelope becomes convective under the buoyancy of its gas (Loore and Doom 1992, Stothers and Chin 1995). The cooling of the stellar layers reduces the number density of free electrons in the envelope gas, where they recombine with the hydrogen and helium nuclei. The number of H^{-} ions therefore decreases rapidly, and the stellar luminosity increases linearily with respect to T_{eff} due to the convection and its opacity lowers. Consequently, as the red giant star evolves it moves upwards in the HR diagram (Iben 1965). During this phase of evolution, a star may loose a signifant amount of mass through a stellar wind, e.g. $10^{-6}M_{\odot}yr^{-1}$ (Dupree 1986).

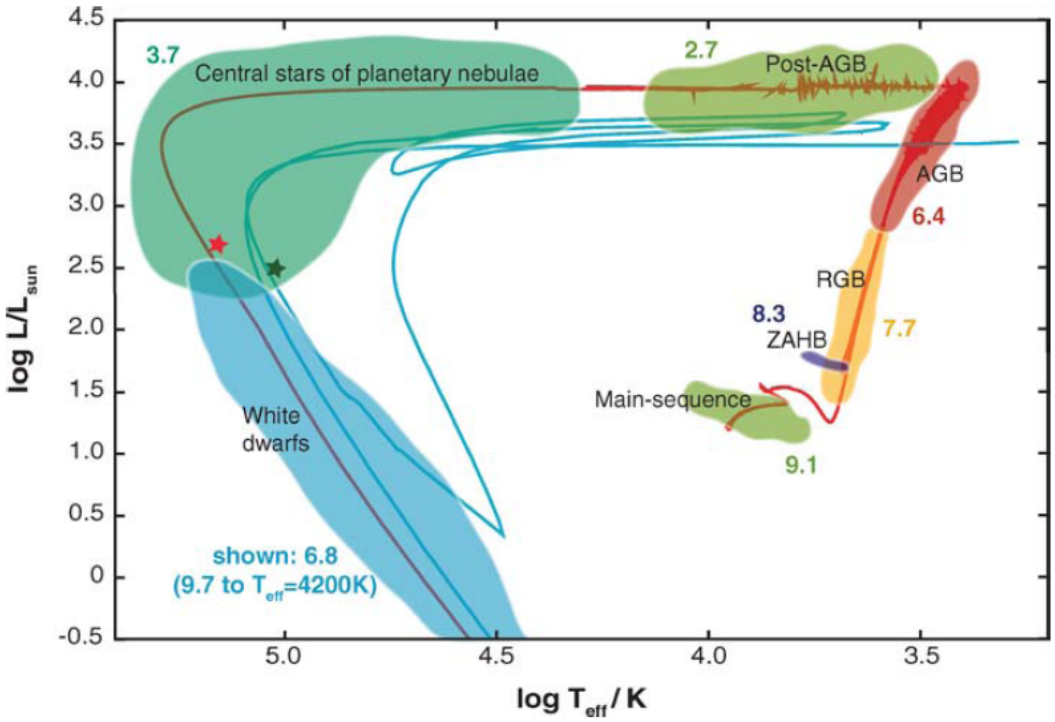


FIGURE 1.1: Diagram of Stellar Evolution from Herwig (2005). This diagram shows the evolution of a low-intermediate mass star, from the Main-sequence, to the Red Giant Branch (RGB), Horizontal Branch (ZAHB), Asymptotic Giant Branch (AGB), and finally the post-AGB phase with the formation of a central star of a planetary nebula (CSPN) afterwards becoming a WD.

The method of ignition of helium in the core is dependent upon the mass of the star, where if the initial mass is less than approximately $2.0M_{\odot}$ the helium becomes electron degenerate by the time the star reaches the tip of the RGB. In these stars when the mass of the core eventually exceeds $\approx 10\%$ of the total stellar mass, which is known as the Schonberg-Chandrasekar limit, the helium is ignited explosively (the helium flash) to form carbon and oxygen. In stars $\gtrsim 2.0M_{\odot}$, the core continues increasing in mass and temperature but avoids degeneracy. When it attains T$\gtrsim$100 million K, the helium ignites and begins burning to carbon. At this point, the star moves onto the core helium burning sequence.

1.3.3 Horizontal Branch

In both the lower mass and higher mass cases, stars then settle on either the horizontal branch (low metallicity population II) or the red clump (high metallicity population I) regions of the HR diagram, depending upon their primordial metallicity. The helium core burning phase of the star is the second longest after the hydrogen core burning phase of the star, lasting approximately one tenth of the hydrogen burning phase (Loore and Doom 1992). The majority of a star's luminosity on the horizontal branch

is provided by the hydrogen burning shell which remains active in this phase.

1.3.4 Asymptotic Giant Branch (AGB)

When the helium in the core is exhausted, the core contracts, and the rise in temperature due to release of gravitational potential energy increases the temperature of the surrounding helium. This ignites helium in a shell around an inert CO core. For stars $\leq 8M_{\odot}$ carbon does not ignite, and the CO core contracts and becomes increasingly electron degenerate. The envelope then increases in radius, and decreases in temperature as a consequence. The star reaches the early-Asymptotic Giant Branch (AGB), where the outer hydrogen burning shell is initially dormant. During this phase, the helium shell dominates the nuclear energy release, burning outwards until it approaches the hydrogen/helium discontinuity. However, beyond this point the hydrogen shell re-ignites and becomes the dominant source of the stars luminosity (e.g. $\approx 90\%$ Schoenberner 1983). During the thermally pulsing (TP)-AGB phase the burning of the hydrogen shell adds fuel to the now dormant helium shell. However, periodically, the star is interrupted by a thermonuclear runaway helium shell flash event. It is theorised that a significant number of these TPs occur during the late stages in the evolution of low and intermediate mass stars. During a TP the star is temporarily ($\approx$ 500yr) taken out of thermal equilibrium and the hydrogen burning shell is extinguished. When the star regains thermal equilibrium ($\approx$ 20000yr), the hydrogen shell starts to burn again. The peak flash luminosity increases with every cycle (Bloecker 1993).

During the evolution up the AGB, the envelope mass loss rate increases significantly. This ultimately results in an overall radius decrease and temperature increase, without a significant change in luminosity. As the remaining star is observed to have $T_{eff} \gtrsim 30000K$ during the final envelope ejection and formation of a planetary nebula (PN) (50000yr), it must loose envelope mass at an extremely high rate (e.g. $10^{-3} - 10^{-4} M_{\odot} yr^{-1}$). Superwinds lasting $\approx$ 1000yr have been proposed as providing these high rates of mass loss (Renzini and Voli 1981, Schoenberner 1983), where shock waves driven by stellar pulsation instabilities may explain their origin (Bowen 1988, Wachter et al. 2002). Indeed, CO line emission from circumstellar clouds support theoretical mass loss rates of $10^{-4} M_{\odot} yr^{-1}$ at the end of the AGB (Knapp 1986).

1.3.5 White Dwarf Cooling Sequence

After the final envelope ejection, the remaining stellar cores exhibit high levels of elemental purity where the atmospheres are observed to be either hydrogen rich or helium rich. Strong gravitational fields are expected to cause elements heavier than hydrogen and helium to sink inwards in a relatively short period of time - as compared to the cooling time of the WD (Schatzman 1958). The ratio of the number of hot DAs to DOs observed ($T_{eff} \gtrsim 40000K$) is $7:1$, which may be compared with the number of hydrogen abundant to non-hydrogen abundant planetary nebula central stars (CSPN), $\approx 2:1$. Considering stars resulting in hydrogen rich WDs evolve approximately 3 times faster than the stars resulting in non-hydrogen rich WDs, these ratios are well matched. Figure 1.2 displays the spectrum of a CSPN for reference.

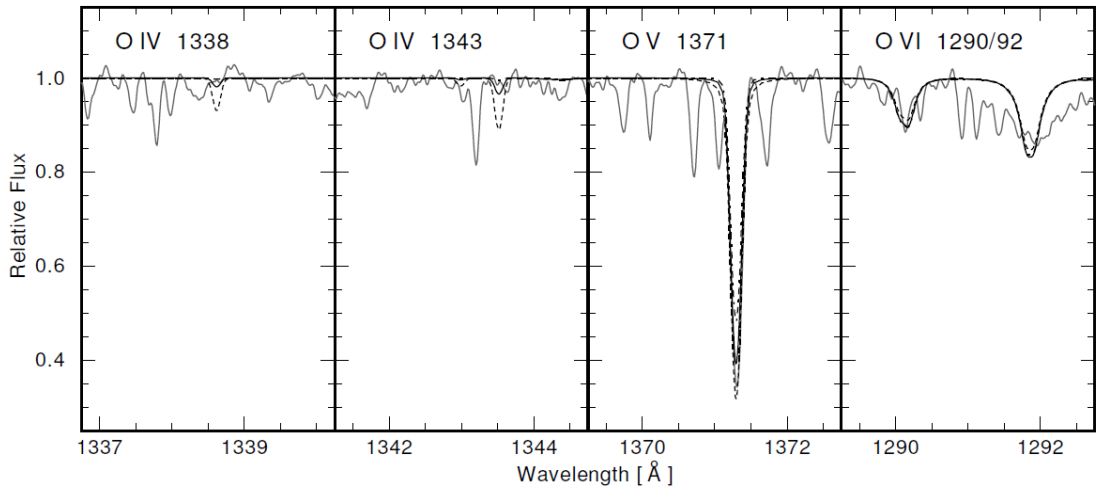


FIGURE 1.2: HST Spectrum of the heliux nebula (NGC7293) planetary nebula central star, compared with three models with different effective temperatures (Traulsen et al. 2005).

Under a primordial evolution model, these two classes of WDs may be explained by two different AGB termination scenarios. If the star undergoes high mass loss rates on the AGB while burning hydrogen in its shell, as it is theorised it should do 80% of the time (Schoenberner 1983), then it will most likely evolve to become a hydrogen rich planetary nebula central star (CSPN), with a thick hydrogen layer ($10^{-4}M_{\odot}$).

Alternatively, if the star undergoes high mass loss rates on the AGB during or immediately after thermal pulse before thermal equilibrium has been reestablished (without an active hydrogen shell), then the majority of the stellar hydrogen may be dispersed in the stellar wind. The resultant object is classified as a PG1159 star based upon the prototype PG1159 - a helium rich pre-degenerate star (in transition between being a CSPN and a hot WD). PG1159 type stars are extremely hot and have spectra showing helium as well as carbon and oxygen absorption lines. Upon cooling, gravitational settling is expected to remove remaining C and O in the atmosphere, resulting in first a DO star ($45000 - 100000K$) - hydrogen deficient with strong `HeII` absorption lines, a DB star ($12000 - 30000K$) - hydrogen deficient with strong `HeI` absorption lines, and then a DC ($\lesssim 12000K$) - hydrogen deficient without helium absorption lines. In support of this theory, DOs are observed to have C and O abundances less than 0.1% of that of helium (Dreizler and Werner 1996).

1.3.6 White Dwarf Survey Statistics

WD Luminosity Function

The WD Luminosity function (LF) or number of WDs per unit interval in absolute magnitude for a given volume of space can be used to constrain the age and the star formation history of the Galactic disk (Winget et al. 1987, Wood 1990). The WD luminosity function and space density has been measured numerous times. One of the most widely referenced efforts involved the Palomar-Green (PG) survey of 1986

and yielded a space density of $0.00049 \pm 0.00005 pc^{-3}$ (Fleming et al. 1986). Another, which was a by-product of the AAT faint QSO survey, yielded a DA space density of $0.00060 \pm 0.00009 pc^{-3}$ and a scale height of $275 \pm 50 pc$ (Boyle 1989). More recently, the WD space density has been estimated as $0.0034 pc^{-3}$ (Leggett et al. 1998), $0.0050 pc^{-3}$ (Holberg et al. 2002) and $0.0046 \pm 0.0005 pc^{-3}$ (Harris et al. 2006). A latest estimate provided by Holberg et al. (2008) gives a WD space density of $0.0048 \pm 0.0005 pc^{-3}$ and is directly measured from a volume-limited sample of local WDs, as opposed to other determinations of the local density

The estimated WD space density is a function of both the birth rate and the theoretical cooling time for a WD to fade below the limiting magnitude/colour of the sample. The DA birth rate in the local disk has previously been calculated as $\chi_{WD} = 0.39 - 0.61 \times 10^{-12} yr^{-1} pc^{-3}$ (Fleming et al. 1986), with a total WD birth rate including helium WDs as $\chi_{WD} = 0.49 - 0.75 \times 10^{-12} yr^{-1} pc^{-3}$. More recent determinations are $\chi_{WD} = 0.75 \pm 0.25 \times 10^{-12} pc^{-3} yr^{-1}$ (Liebert et al. 2005a) using the PG survey, and $\chi_{WD} = 0.75 \pm 0.25 \times 10^{-12} pc^{-3} yr^{-1}$ (Vennes et al. 2002).

Early estimates of the formation rates of PNe were substantially higher than this. For example Pottasch (1996) calculated $\chi_{WD} = 3 \times 10^{-12} pc^{-3} yr^{-1}$. Frew (2008) however summarises the historical evolution of both planetary nebula formation rates and WD formation rates, while providing his latest estimates of the former, $0.8 \pm 0.3 \times 10^{-12} pc^{-3} yr^{-1}$. He finds that his value is in very good agreement with the WD birthrates determined by both Vennes et al. (2002) and Liebert et al. (2005a). Taking into account the assumption that WD production also involves other post-RGB processes which do not involve creation of a PNe, the WD birth rate is expected to be slightly higher than the PNe birth rate but within the errors of these PNe/WD birth rate estimates. These channels include AGB-manqué evolution, where H-deficient stars do not pass through the AGB phase and are characterised by a lack of thermal pulses, and 'lazy PNe', with central stars that evolve too slowly to ionise the ejected nebula shell.

It is known that the space density of WDs drops off steeply below $M_v \approx 16.0$. This is attributed to the finite age of the Galactic disk, where WDs have not had sufficient time to yet cool below these luminosities (Liebert et al. 1988). Using WD cooling models and the observed WD LF, it has been calculated that star formation began in the Galactic disk $8 \pm 1.5 \times 10^9$ years ago (Leggett et al. 1998), or $10^{+3}_{-1} \times 10^9$ years ago (Knox et al. 1999) with a broader estimate of $7 - 10$Gyr Liebert et al. (1988).

WD Mass Function

Latest determinations of the field white dwarf mass distribution (for single stars), or number of objects per unit interval in mass, have been obtained using SDSS data (Kepler et al. 2007) and confirm earlier results that it peaks sharply at $0.6 M_{\odot}$. See Figure 1.3, where it can be noted also that a low mass peak is present, and is product of He-cores arising from close binary evolution. The shape of the WD mass function is believed to be due to the form of both the stellar initial mass function and the stellar initial-final mass relation (IFMR). The initial mass function favours the production of lower mass stars, while the IFMR links these to the formation of less massive WDs.

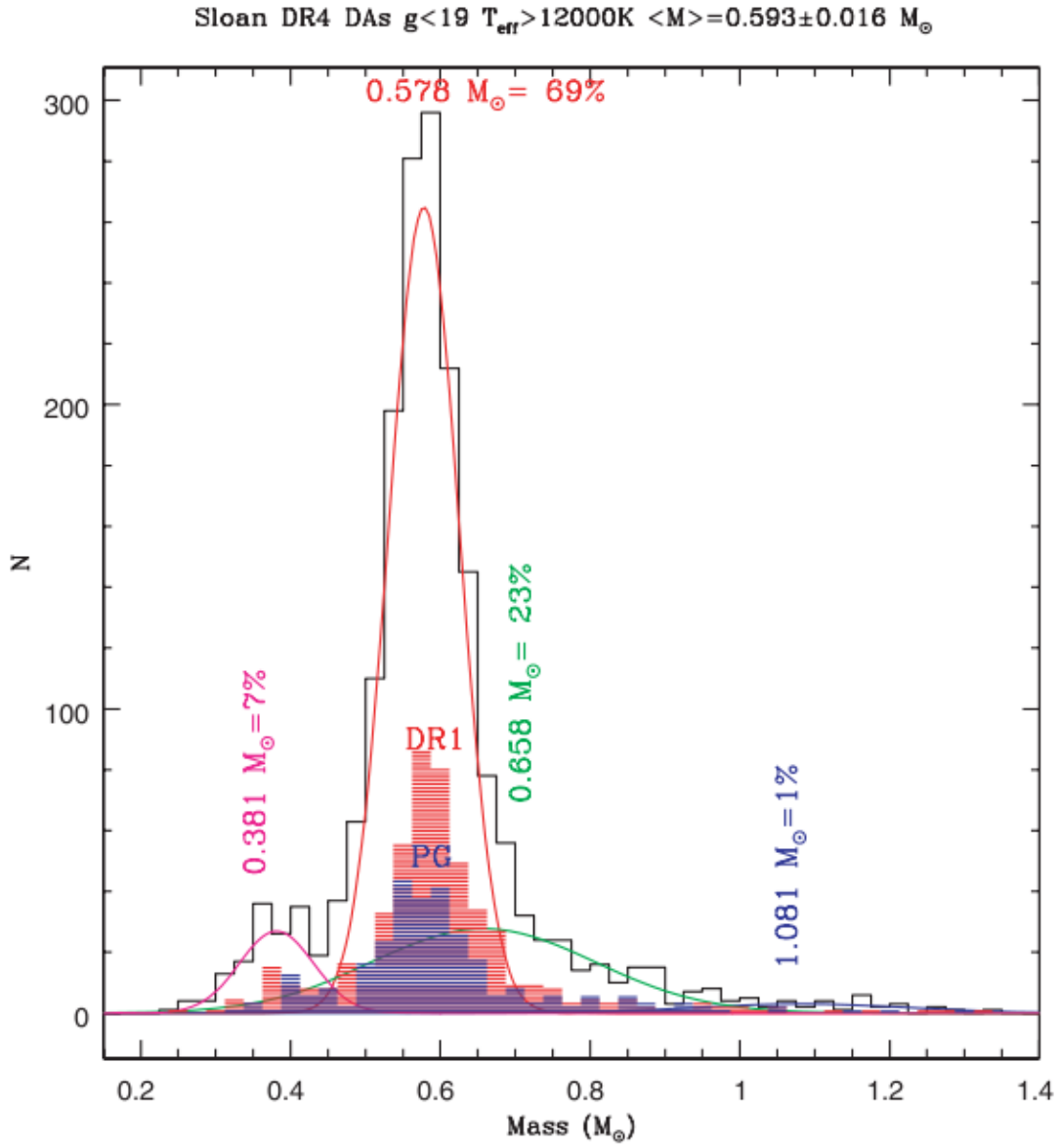


FIGURE 1.3: The SDSS WD mass distribution as determined by Kepler et al. (2007). Histogram for the 1859 DA stars brighter than $g = 19$ and hotter than $T_{\rm eff} = 12000K$, compared to the PG survey published by Liebert et al. (2005a) and the SDSS DR1 sample published by Madej et al. (2004). Gaussian fits are also shown. The bins are $0.025M_{\odot}$ wide.

1.4 The stellar initial-final mass relation (IFMR)

1.4.1 A light introduction to the stellar IFMR

The benefits of a solid understanding of the form of the stellar initial-final mass relation (IFMR) have been well documented in the literature. The nature of the relationship between the initial mass $M_{\rm init}$ of a star and the final mass $M_{\rm final}$ of the star has been of interest for many decades (Weidemann 1981). The hypothesis that stars with MS mass much greater than the Chandrasekhar limit $M_{\rm init} \gg 1.4 M_{\odot}$ can still lead to the formation of WDs was confirmed by Weidemann (1977). Koester and Reimers (1981) conducted a search for WDs in young open clusters where only stars with $M_{\rm init} > 4 - 5 M_{\odot}$ had yet evolved beyond the main sequence. Infering that WDs produced within the cluster NGC 2516 would generally have to have come from stars at least as massive as the current highest mass (most luminous) stars within this population, they concluded that they had discovered a number of WDs whose progenitors were approximately $M_{\rm init} \approx 6 - 8 M_{\odot}$ (Reimers and Koester 1982).

As discussed previously, the majority of a star's envelope is lost after the main sequence phase, primarily during the ascent of the asymptotic giant branch (AGB) (Herwig 2005, Iben and Renzini 1983). As these late stages in a star's evolution remain extremely challenging to model (Marigo and Girardi 2007) empirical data are required to tease out the details of this mass loss process. Observations of circumstellar shells and planetary nebulae around pre-WD stars provide some handles on the rate of AGB mass loss (Weidemann 2000). The mapping of the masses of WDs to their progenitor masses through the IFMR is the method of choice for assessing quantitatively, at least, the total stellar mass loss and can also be useful for gaining some insight into the mechanisms behind this mass loss.

The total amount of mass lost by stars has a strong bearing on our understanding of the minimum mass of a Type II (core collapse) supernova (SNe) progenitor (Williams et al. 2009) and thus the frequency of these events. Moreover, the vast quantity of envelope gas expelled by stars during their final evolutionary phases contains the products of nucleosynthesis (e.g. C, N, O and s-process elements) so a sound understanding of the form of the IFMR is very important for refining models of stellar and Galactic chemical evolution. The form of the top end of the IFMR also has some relevance to cosmology through type Ia SNe which are generally considered to involve the thermonuclear detonations of high mass CO white dwarfs (Howell et al. 2006).

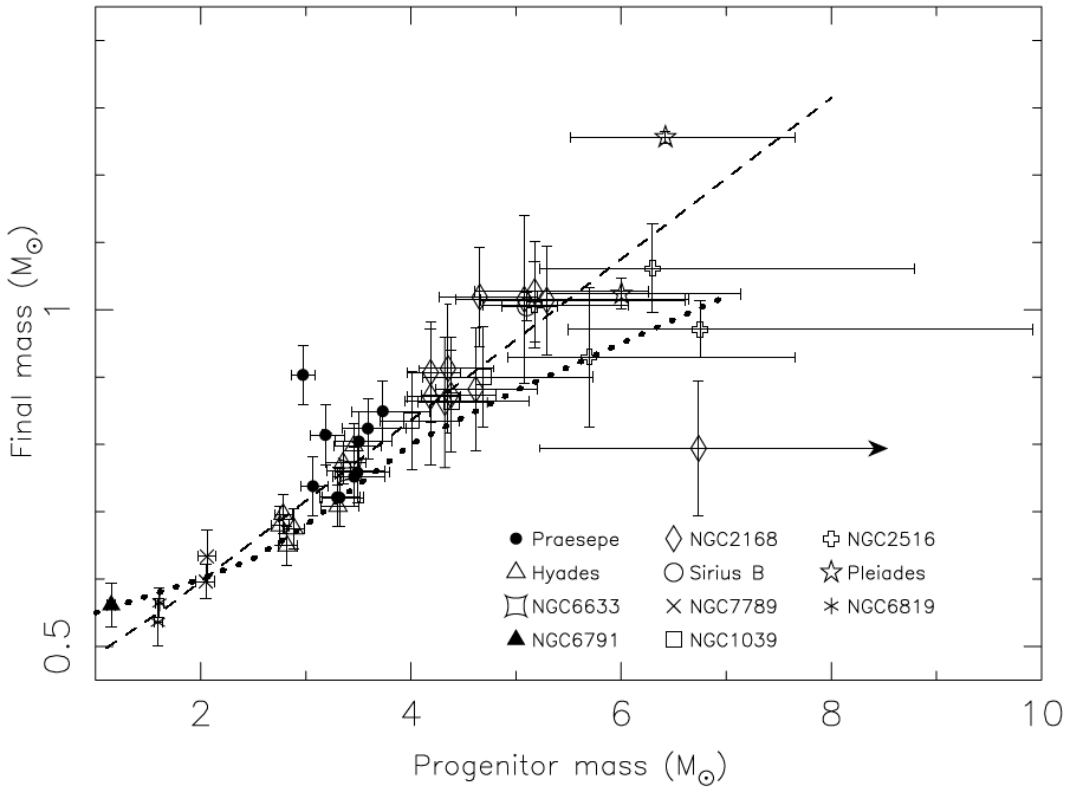


FIGURE 1.4: The IFMR (Casewell et al. 2009). A state-of-the-art representation of the known form of the IFMR, showing the revised locations of the Praesepe white dwarfs in initial mass-final mass space (filled circles). The locations of the degenerate members of the NGC 6633 (square star), NGC 7789 (cross), NGC 6819 (asterisk), Hyades (open triangles), Sirius binary system (open circle), the Pleiades (open stars), NGC 2168 (open diamonds), NGC 1039 (open squares), NGC 6791 (filled triangles) and NGC 2516 (open '+' signs) are also shown. The relation of Weidemann (2000) (dotted line) and a revised linear fit to 41 white dwarfs (dashed line) are overplotted. The radial velocity variable, WD0837+185 is labelled.

Stars on the tip of the AGB

The relationship between the AGB core mass and the AGB tip luminosity, which is predicted by standard stellar evolutionary models, has been used in the past to place constraints on the form of the IFMR (Weidemann 1987). In this approach, the brightnesses of stars towards the tip of the AGB in very rich clusters are measured and their absolute magnitudes and luminosities then derived. Subsequently, assuming strict adherence to a core - mass luminosity relation, the masses of their cores is calculated. A WD final mass of $M_{\rm final} = 0.8 M_{\odot}$ was estimated for a progenitor of $M_{\rm init} = 4.5 M_{\odot}$ in the NGC 1866 cluster of the LMC using this method (Weidemann 1987). However, the assumption of a classical core - mass luminosity relation is stretched somewhat by the levels of overshoot adopted in modern theoretical AGB models (Weidemann 2000). Other methods have since taken precedence in the determination of the IFMR, such as the open cluster method described below.

1.4.2 Using open clusters to map the IFMR

Arguably the best current way to map the form of the IFMR is through studying WDs in open cluster systems. Here, all stars have a common age, and both this and the distance to these stars can be determined from the brighter members of the population. Deriving the initial progenitor mass, $M_{\rm init}$, of a cluster WD first involves obtaining optical spectroscopy so that the relative shape of its broad Balmer lines (due to the atmospheric distortion under gravity of the absorption of light through its hydrogen atmosphere) can be assessed. Generally, the lines β through to 8 are chosen for this purpose (i.e. β, γ, δ, ϵ, and 8). These observed Balmer lines are compared to synthetic profiles (Bergeron et al. 1992) to extract estimates of the surface gravity, $\log g$, and effective temperature, T_{eff}. WD evolutionary mass-radius models are then used to determine the mass $M_{\rm final}$ and cooling age $t_{\rm cool}$ of the WD. $M_{\rm final}$ is strongly dependent upon $\log g$ while $t_{\rm cool}$ is more dependent upon T_{eff}. Finally, the cooling age is subtracted from the cluster age to obtain an estimate of the progenitor lifetime, and theoretical stellar evolutionary models are then used to determine the initial mass $M_{\rm init}$ of the star (Weidemann 2000).

There are a number of potential sources of error in this method of mapping the IFMR, ranging from uncertainties in the open cluster ages to the limitations in the physical realism of the WD atmosphere and evolution models. It is known that there are still significant systematic discrepancies in the atmospheric models in certain effective temperature ranges e.g. spectroscopic surface gravity measurements at $T_{eff} \lesssim 12000K$ (Kepler et al. 2007). Open clusters also must be chosen judiciously as they vary in their usefulness for constraining the form of specific areas of the IFMR. Younger populations are most useful for constraining the high mass end of the IFMR, as their high mass white dwarfs are still relatively hot and therefore bright ($M_{WD} \gtrsim 4 M_{\odot}$). Older open clusters, $\gtrsim 1$ Gyr, are better suited for constraining the intermediate and lower mass end of the IFMR, as they contain WDs which have had relatively large main sequence life times, and are therefore of relatively low mass ($M_{WD} \lesssim 2 M_{\odot}$).

The age of a cluster, which is usually known a priori, is defined assuming that

the vast majority of stars within the system have evolved from a single burst of star-formation in a molecular cloud of near uniform composition (Bate et al. 1995). It is generally calculated using photometric data, taking into account the distance and the reddening due to foreground interstellar dust and gas, where observed colours and magnitudes (e.g. B-V,V) at which stars move off the main sequence are matched to the predictions of stellar evolutionary models. Similarly, the distance can be derived from the offsets between the apparent magnitudes of stars as predicted by the model and their observed magnitudes (e.g. V magnitude), allowing for any foreground reddening.

1.4.3 IFMR Calculations To Date

Empirical based determinations of the form of the IFMR have not always sat well with expectations based on stellar evolutionary models. For example, estimates based on stars at the tip of the AGB, discussed in section 1.4.1, failed to lead to mass loss measurements in agreement with theoretical predictions. Furthermore, empirical determinations based on different approaches produced mixed results e.g. mass loss estimated from planetary nebula (PNe) formation led to final masses that were inconsistent with those obtained from open clusters (Weidemann 1981). In the early 1990s WD evolutionary models saw substantial improvements in their physical accuracy (Wood 1992) which allowed for more robust determinations of their ages/cooling times and masses from spectroscopic effective temperature and surface gravity measurements. The revised models were widely adopted in the study of WDs within open cluster systems. Volker Weidemann published a summary of the state of empirical based knowledge of the IFMR to 2000 (Weidemann 2000). By this time the WD populations of several open clusters had been included in attempts to map the relation. First, the Hyades WDs had been the subject of multiple studies and successfully harvested for data within the range $2.9M_{\odot} \leq M_{\rm init} \leq 3.5M_{\odot}$. Second, the WDs of the NGC 3532 cluster had been targeted and provided data for the progenitor range $3.5M_{\odot} \leq M_{\rm init} \leq 5M_{\odot}$. Third, the WD members of the Pleiades and NGC 2516 had been studied and found to have progenitor masses in the range $6M_{\odot} \leq M_{\rm init}7M_{\odot}$. Fourth, studies of the 4Gyr old cluster M67 had resulted in a single additional data point at $M_{\rm init} \approx 1.35M_{\odot}$.

However, since Weidemann's 2000 summary on the IFMR, there has been a substantial increase in the number of known cluster WDs, which are suitable for further advancing our understanding of the relation's form. In particular, new datapoints have been added to the following mass ranges: 1) $3_{\odot} \leq M_{\rm init} \leq 4_{\odot}$ using Praesepe cluster WDs (Casewell et al. 2009, Claver et al. 2001, Dobbie et al. 2004; 2006, Williams et al. 2004), 2) $3.6_{\odot} \leq M_{\rm init} \leq 4.6_{\odot}$ and $4.4_{\odot} \leq M_{\rm init} \leq 4.6_{\odot}$ using NGC 3532 and NGC 2287 cluster WDs respectively (Dobbie et al. 2009), 3) $3.5_{\odot} \leq M_{\rm init} \leq 4.75_{\odot}$ using NGC1039 WDs (Rubin et al. 2008), and 4) $M_{\rm init}$ between $2.8_{\odot}$ and $3.4_{\odot}$ using 18 NGC 2099 (M37) WDs (Kalirai et al. 2005a). Williams and Bolte (2007) have also spectroscopically studied WD candidates of the NGC 6633 and NGC 7063 clusters providing a single additional point to the IFMR at $M_{\rm init} \approx 3.5M_{\odot}$.

These has also been a substantial number of new data points added towards the upper end of the IFMR, which is particularly pertinent to clarifying the supernova progenitor lower mass limit. Williams et al. (2009) have identified and spectroscopically

investigated twelve WDs in the young, rich cluster NGC 2168. Assuming a simple linear form to the IFMR, they extrapolated to $M_{\rm final} \simeq 1.4M_{\odot}$ to infer an upper initial mass limit for the production of a WD of $6M_{\odot} \leq M_{up} \leq 9M_{\odot}$ taking into account all available open cluster IFMR data. Dobbie et al. (2006) had previously established an upper mass limit of $6.8M_{\odot} \leq M_{up} \leq 8.6M_{\odot}$. At the low mass end of the IFMR ($M_{\rm init} < 2.5M_{\odot}$, $M_{\rm final} < 0.6M_{\odot}$), Kalirai et al. (2007) and Kalirai et al. (2008) have added a small number of important new data points based on observations of WDs in NGC 6791, NGC 7789 and NGC 6819 e.g. $M_{\rm init} \approx 1.16M_{\odot}$ –, $M_{\rm final} \approx 0.53M_{\odot}$ and $M_{\rm init} \approx 1.6M_{\odot}$ – $M_{\rm final} \approx 0.54M_{\odot}$.

Limitations on IFMR calculations to date

This solid recent progress in mapping the form of the IFMR is due to greater access to mosaic imagers and 8/10m telescopes with blue sensitive spectrographs. This has allowed a larger number of clusters to be investigated to very faint magnitude limits (Kalirai et al. 2007). Recent work by Dobbie et al. (2009), using data from approximately 50 WDs in 11 open clusters, suggests that the bulk of stars follow closely a monotonic relation (see Figure 1.5). Tentative evidence has also been found suggesting that the IFMR is somewhat steeper in the range $3M_{\odot} < M_{\rm init} < 4M_{\odot}$ than elsewhere. This is consistent with theoretical expectations and the sharp drop seen in the number density of objects on the high mass side of the main peak in the field WD mass distribution (Ferrario et al. 2005). However, it has been questioned as to whether such an inference can be drawn from the data subject to its current uncertainties (Salaris et al. 2009).

Despite clear headway, the IFMR remains sparsely sampled in several important initial mass regimes. For example, there are only two secure data points at $M_{\rm init} >\approx 5.5 - 6M_{\odot}$. Thus, the form of the upper IFMR remains somewhat uncertain. Furthermore, there are still only a handful of objects at $M_{\rm init} \lesssim 2.5 - 3M_{\odot}$ yet understanding the fate of the numerous stars with masses more closely resembling that of our Sun is of significant interest.

In the open cluster approach, mapping the higher initial mass regime of the IFMR requires targeting young ($\approx 100-300$ Myr) populations that are sufficiently rich to have harboured a substantial number of stars with $M_{\rm init} \gtrsim 5-6M_{\odot}$ and where the progeny of these stars are still relatively young, hot and luminous. Investigation of the lower initial mass regime requires targeting old ($> 1 - 2$ Gyr) open clusters, where populations are sufficiently mature such that stars with $M_{\rm init} \lesssim 2.5 - 3M_{\odot}$ have evolved beyond the main sequence. Such clusters are comparatively rare so to identify good examples it is necessary to probe large volumes of space. Consequently, while high signal-to-noise optical spectroscopy is critical to determining WD masses and cooling times, the great distances involved make it extremely challenging to obtain the necessary data even with todays large telescopes.

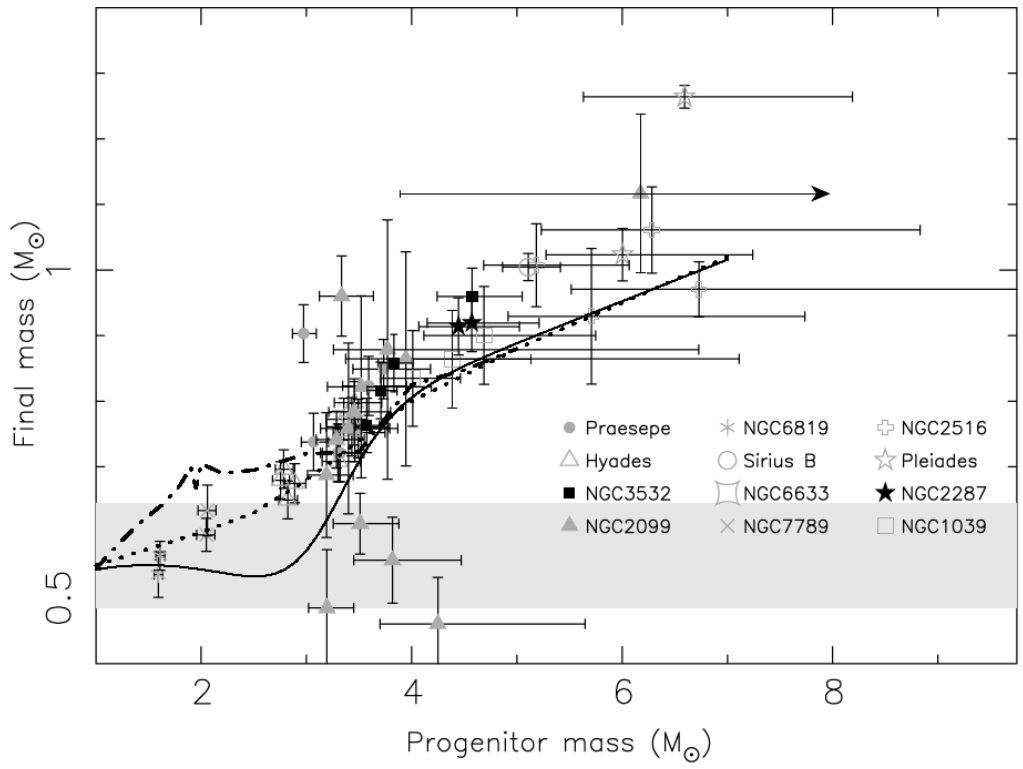


FIGURE 1.5: The IFMR (Dobbie et al. 2009). Another state-of-the-art representation of the known form of the IFMR, showing that the bulk of stars appear to follow fairly closely a monotonic relation. The locations of the white dwarf members of NGC 3532 and NGC 2287 in initial mass-final mass space. Data points from a number of other populations with metallicities close to the solar value are also shown. The theoretical IFMR of Marigo and Girardi (2007) (dot - dashed heavy line), the semi-empirical IFMR of Weidemann (2000) (heavy dotted line) and the initial mass-core mass at first thermal pulse relation from Karakas et al. (2002) (medium solid line) are overplotted. The peak in the field white dwarf mass distribution ($\pm 1\sigma$) is represented by the band of grey shading.

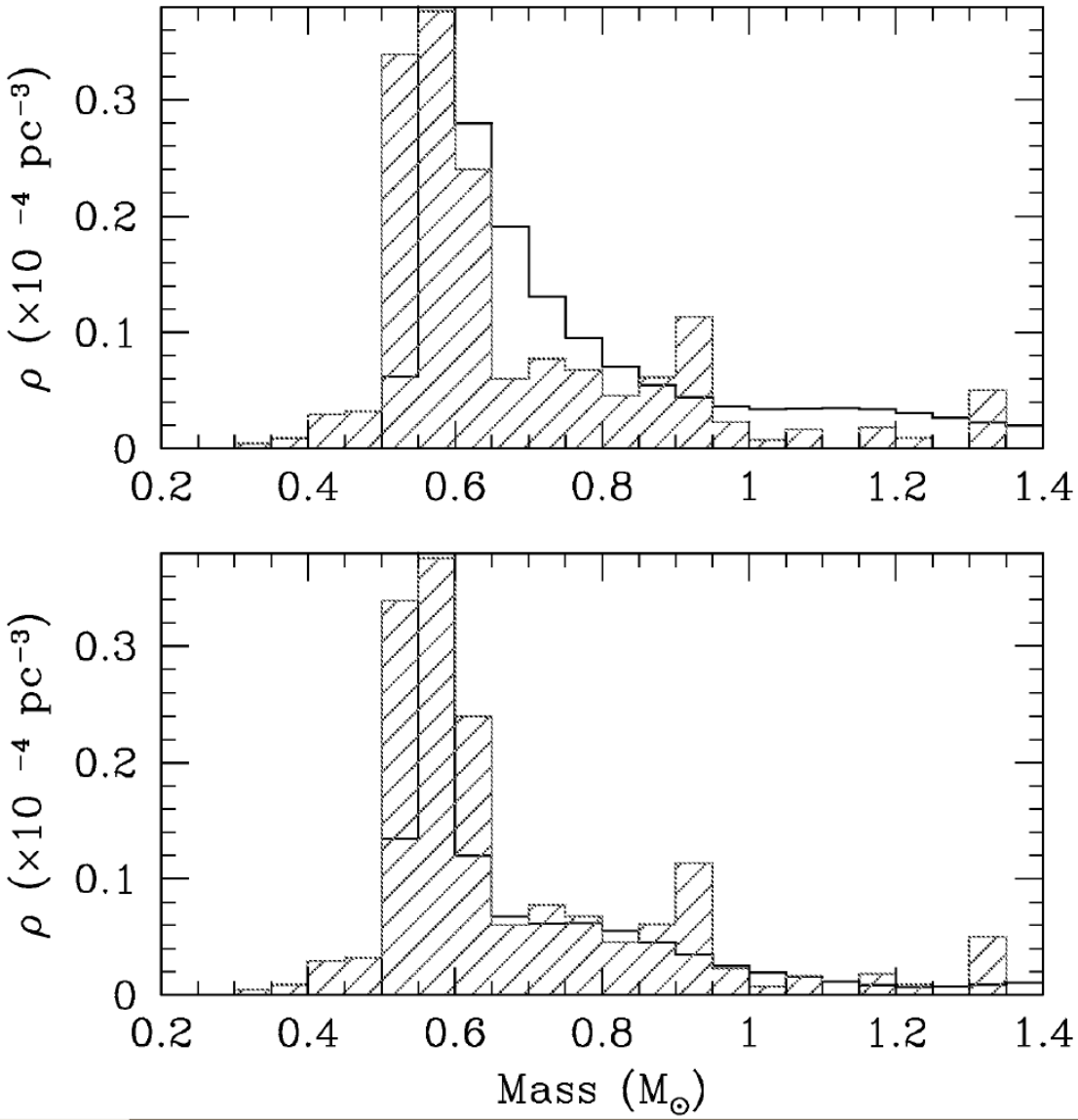


Figure 1.6: Ferrario et al. (2005) volume corrected mass distribution of field WDs in the Palomar-Green survey (Liebert et al. 2005a) (shaded histogram) for stars hotter than $13000K$ with the best linear IFMR fit model (top panel) and the IFMR with curvature (bottom panel). WDs with masses less than $0.5M_{odot}$ are believed to be helium WDs resulting from close binary evolution and are not modelled in this paper

Using Double-Degenerate systems to probe the IFMR

Multiple stellar systems have the potential to improve our understanding of the evolution of stars, including testing our knowledge of the form of the IFMR (Greenstein et al. 1983). Like open cluster systems, their components share a common system age and this factor can, in suitable cases, be exploited for similar purposes. They also have the capacity to provide constraints on the birth and dynamical evolution of stellar binaries and the mechanisms behind the production of magnetic WDs (Girven et al. 2010).

Greenstein et al. (1983) attempted to use a wide WD+WD binary system to constrain the IFMR but both the quality of their data and models at this time led to results with very large uncertainties. A small number of additional hot double-degenerate (DD) systems have been discovered since but generally the component masses and their ratios have not been particularly suitable for placing meaningful constraints on the system age (Jordan et al. 1998b). Alternatively, the systems have not been wide enough to be confident that the components have evolved essentially as separate entities (Holberg et al. 1995). The radii of AGB stars can extend up to 4-5AU so in binary systems with primordial separations of a$\lesssim$10AU the components could potentially experience Roche Lobe overflow and exchange significant quantities of mass. Nevertheless, Finley and Koester (1997) were able to demonstrate the potential of wide DD systems for probing the form of the IFMR by obtaining a firm data point in the central region of the IFMR with $3.6M_{\odot} \leq M_{\rm init} \leq 4M_{\odot}$ and $0.76M_{\odot} \leq M_{\rm final} \leq 0.82M_{\odot}$, through their study of PG 0922+16.

The identification of additional wide WD+WD pairs which exhibit common proper motions, such that they are highly likely to be physically associated and thus co-eval, is highly desirable so that this method can be more fully exploited. Noting that mass loss, primarily during the AGB phase (and being more pertinent for higher mass WDs, subject to the form of the IFMR), could lead to the expansion of the orbit by upto a factor 5-10, so to be confident that two WDs are unlikely to have exchanged mass in the past, I seek systems with a minimum projected separation of a$\approx 100AU$. This is satisfied by visual binary pairs with an angular separation $> 1''$ and which lie at $d \gtrsim 100pc$.

The most interesting systems for investigating the form of the IFMR are those in which the ratio of the masses of the component WDs is relatively low, for example, a high ($M_{\rm final} > 0.9M_{\odot}$) and low ($M_{\rm final} < 0.7M_{\odot}$) mass pairing. However, the details of this approach depend upon the masses of the components. In all cases, the mass $M_{\rm final}$ and cooling age $t_{\rm cool}$ for both components are first derived using theoretical WD atmospheric and cooling models, where the effective temperature and surface gravity are measured from the line profiles of their spectroscopic observation. If one of the component WDs is found to have a mass corresponding to a part of the IFMR which is currently well delineated by data, its initial mass, main sequence lifetime and the total system age can be estimated. Subsequently, the main sequence life time and hence initial mass of the second component is calculated by subtracting its cooling time from the system age. Alternatively, if one of the component WDs is found to be of particularly high mass (ie. $M_{\rm final} > 1M_{\odot}$), the DD system age is largely constrained

by the cooling age $t_{\rm cool}$ of the higher mass WD component, since the main sequence life time is significantly shorter than that of the lower mass component, in some cases almost negligible (e.g. Finley and Koester 1997). The main sequence lifetime of the younger component is approximately the difference between the cooling ages of the two WDs ($t_{\rm MS} = T_{\rm cool1} - t_{\rm cool2}$). Finally, the main sequence (initial) mass $M_{\rm init}$ of the younger component is inferred from its estimated main sequence life time $t_{\rm MS}$ using main sequence stellar evolution models.

1.4.4 Sources of Uncertainty

Theoretical models of stars and WDs

Current techniques for determining the form of the IFMR rely upon theoretical stellar evolutionary models to determine the mass of a star for a given lifetime. Poorly understood processes occuring within stars, especially the level of convective overshooting, affect the precision to which we can derive the form of the IFMR. This limits our ability to study in detail the dependencies of the form of the IFMR on parameters such as metallicity (Salaris et al. 2009). It is suggested that systematic uncertainties of this nature impact our ability to make secure assertions that the IFMR departs from a non-linear form. The main source of uncertainty highlighted by Salaris et al. (2009) is that related to the cluster age estimates.

A number of investigations have also examined the level of uncertainty which stems from the choice of synthetic line profile grid used in determining the effective temperatures and surface gravities of WDs. These have looked at magnitudes of the systematic offsets in $\log g$ and T_{eff} estimates for samples of WDs that are common to multiple studies. For example, Liebert et al. (2005a) undertook a comparison between their effective temperature and surface gravities measurements for WDs in the PG survey and those derived independently by Finley and Koester (1997), Vennes et al. (1997), Marsh et al. (1997), Homeier et al. (1998), and Koester et al. (2001). Figure 1.7 shows the results of this procedure, in which they found systematic departures in $\log g$ / T_{eff} from their estimates of 0.06 dex / 1.7%, 0.03 dex / 0.6%, 0.1 dex / 3.3%, 0.08 dex / 0.3%, and 0.08 dex / 0.6%, respectively.

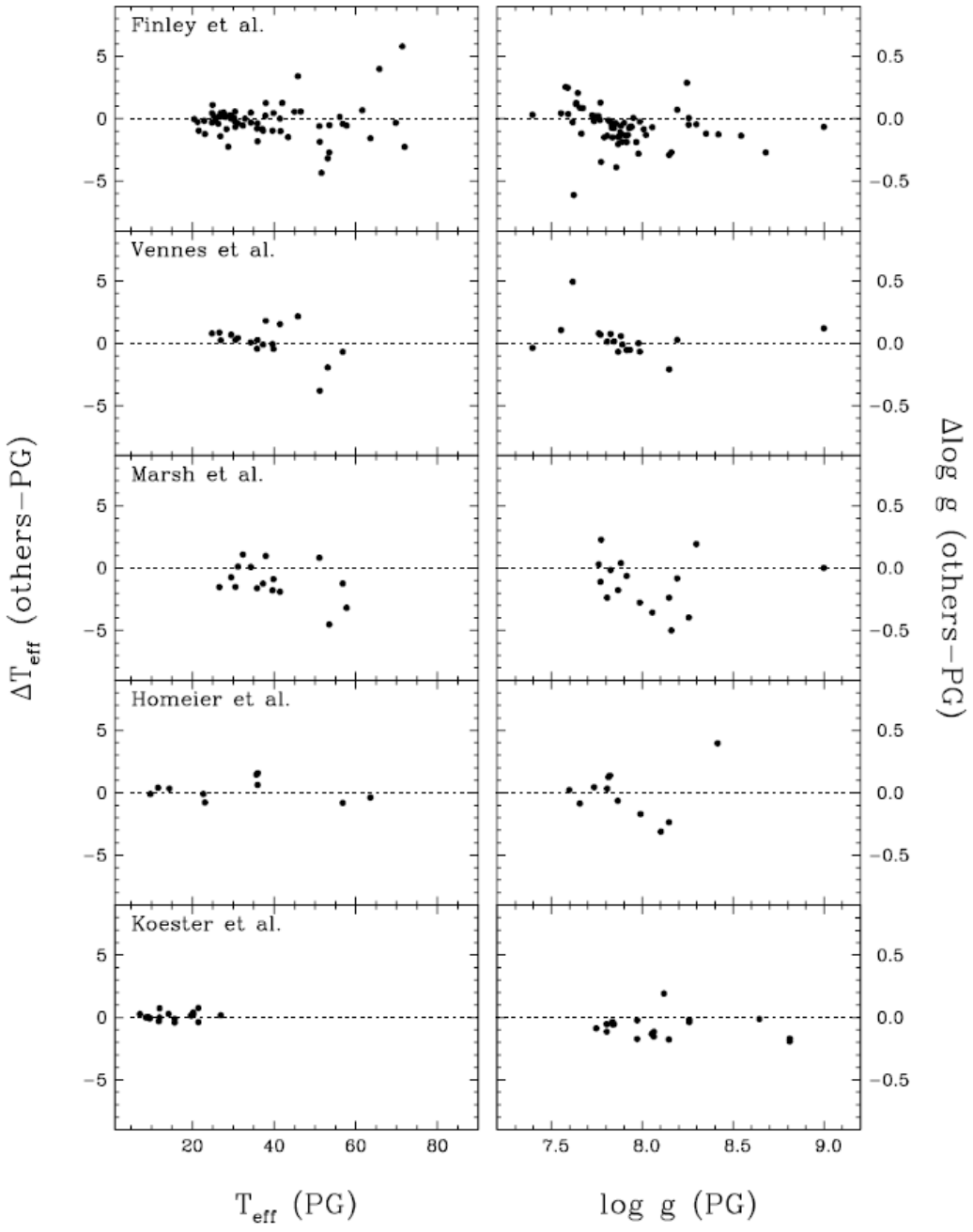


FIGURE 1.7: The Liebert et al. (2005a) comparison of their T_{eff} and $\log g$ determinations of PG stars in common with the data sets of Finley and Koester (1997), Vennes et al. (1997), Marsh et al. (1997), Homeier et al. (1998), and Koester et al. (2001). In each panel they plot the differences between these investigations and their results (PG) as a function of their determinations of T_{eff} or $\log g$. Effective temperatures are in units of $10^3 K$.

Open Cluster age and distance calculations

Robust open cluster age and distance estimates are integral to our understanding of the IFMR as the majority of IFMR points are derived using these populations. Distance calculations must be adjusted for estimated interstellar reddening, which can add significantly to their uncertainties. With the increasing availability of more powerful ground based telescopes however, refinement of some cluster age estimations using the lithium depletion boundary technique (Stauffer et al. 1998) will be possible, where the depletion results from the slow nuclear burning of this element deep within cores of the lowest mass stellar and largest mass substellar members.

Metallicity and Magnetism effects on IFMR

Theoretical models of stellar evolution predict some dependence of the form of the IFMR on stellar metallicity (ratio of hydrogen versus heavier elements) based upon subtle differences in the processes that occur during the AGB phase (e.g. third dredge-up, mass loss). Exploring observationally the relationship between the IFMR and the system metallicity is therefore a goal of astronomers working on understanding the IFMR. It is quite likely that the uncertainties in our current understanding of the form of the IFMR exceed the effects which arise as a result of differing metallicities. It has been argued that there is no evidence yet for a metallicity dependence of the IFMR to within the current observational errors, at least for $M_{\rm init} \approx 3$ (Williams 2007).

Magnetism of WDs has also been proposed as a potential contributor to variations in the IFMR. Wickramasinghe and Ferrario (2005) surmise that the presence of a magnetic field may affect the convective dredge-up process which retards the growth of the CO core during the AGB phase of evolution. This could lead to larger white dwarfs than expected from an IFMR which is delineated largely by non-magnetic stars. Indeed, the apparently single high field magnetic WD EG 59 ($B\sim$3MG), which is a member of Praesepe, appears to be somewhat too massive, relative to similarly aged WDs in the cluster (e.g. Claver et al. 2001).

1.5 Thesis Outline

This project involves the identification of candidate wide DD systems within the Sloan Digitial Sky Survey imaging data. For a selection of bright candidates I have obtained follow-up optical spectroscopy, which has been reduced using common utilities, and interpreted with the help of pre-existing WD atmospheric and cooling models.

The low resolution follow-up spectra are neccessarily of high signal to noise and span the range 3700Å to 5200Å (which encapsulates the $\beta - 9$ Balmer lines of a DA WD). Subsequently, I have used the SDSS DR7 imaging, Palomar Sky Survey photographic data and frames obtained with the Wide Field Camera (WFC) on the Isaac Newton Telescope to assess the proper motions of the components of each double-degenerate system. In addition to using some of these binaries to probe the IFMR, I have also constructed a preliminary DD mass function for the components of the systems.

In the next chapter, I outline my target selection process, my observations, data reduction, and data modelling processes. In Chapter 3 I analyse each system in turn, discussing my results in the context of the IFMR and stellar evolution. In the final Chapter I summarise my main results, and outline future work to extend my initial findings.

"All proofs inevitably lead to propositions which have no proof! All things are known because we want to believe in them!"

The Lady Jessica, to Bene Gesserit delegation, from Frank Herbert's Children of Dune

2 Data

2.1 The identification of candidate wide double-degenerate systems

2.1.1 Summary

A list of candidate wide double-degenerate systems has been extracted from the SDSS DR7 sky survey database using an SQL query. This was filtered by comparing it to a list of known quasars within the SDSS database and by undertaking a visual inspection to remove spurious pairings. Subsequently, high signal-to-noise, low resolution optical spectroscopy was obtained for these systems which ultimately allowed me to both probe the IFMR and to construct a preliminary mass distribution for WDs in wide double-degenerate systems. The spectroscopic follow-up was prioritised, by constraints on time of year, the location of the object in the sky, weather, seeing, and the brightness/exposure times. As a result, brighter, more widely separated systems were favoured. A total of 12 pairs were observed. Each were confirmed to contain WDs and are likely to be physically associated double-degenerate binaries.

2.1.2 The SDSS Data Release 7

The SDSS is a large scale imaging survey with fiber spectroscopic follow-up (Abazajian et al. 2009). The 7th public data release of the data (DR7) includes imaging for a total of 11663 sq. deg. of sky and has photometry in five bands, u (3551Å), g (4686Å), r (6165Å), i (7481Å), and z (8931Å), for 357 million unique cataloged objects. Figure 2.1 displays the passbands of the SDSS filter set. The SDSS data extends to 5σ limits at u, g, and r of 22.0, 22.2, and 22.2 respectively. However, my faint magnitude selection criteria was much brighter than these limits. The imaging pixel size is $0.396''$ and the

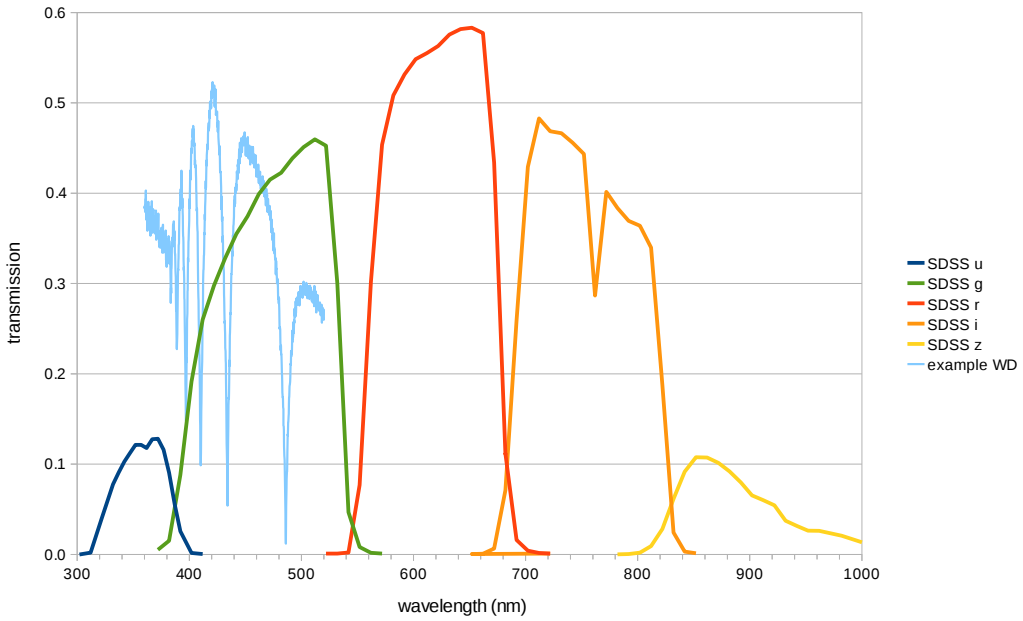


FIGURE 2.1: A graph of the passbands of the SDSS filters (u, g, r, i, z) with a WD spectrum overlayed

typical width of the point spread function (PSF) is $\sim$1.4″. The SDSS DR7 also includes spectroscopic data for 1.6 million objects. This data was used only to confirm (when available) that my blue candidates were WDs.

The SDSS DR7 mainly covers the northern Galactic cap region of the sky which is not heavily affected by the interstellar dust of the Milky Way. This is greatly beneficial for galaxy studies which was one of the SDSS's primary science missions. As the SDSS also was designed for quasar detection its wavelength coverage extends to the blue end of the visible spectrum, making it well suited to the photometric identification of WDs with $T_{\rm eff}$ >8000K. The SDSS data can be accessed in a number of different ways, all of which operate through a dedicated SQL query server (SDSS Sky Server). This included the SDSS DR7 Image List Tool (http://cas.sdss.org/dr7/en/tools/chart/list.asp) - for generating coloured images of target lists, the SDSS DR7 Finding Chart Tool (http://cas.sdss.org/dr7/en/tools/chart/chart.asp) - for providing database lookup on individual SDSS objects, and their science front end (CAS). The various SQL queries I used in the extraction of SDSS photometry are outlined in the Appendix Section A.2.1 and one of which is shown below in Section 2.1.3.

2.1.3 Initial Identification of candidate wide DD systems

The selection of Blue Objects (BO)

The SDSS has unearthed a vast number of spectroscopically confirmed WDs and several studies have been conducted to determine their masses and temperatures (Harris et al. 2003, Kepler et al. 2007). Figure 2.2 from Harris et al. (2003) shows the location of WDs in a $u-g$, $g-r$ colour-colour diagram and highlights how well separated those with $T_{\rm eff} > 8000-9000K$ are from the bulk of main sequence stars. I confine my investigation here to objects with $T_{\rm eff} > 8000-9000K$ since, in addition to being more readily identified using the SDSS photometry, at lower temperatures, the atmospheres are considerably more difficult to model reliably due to the emergence of more physically complex sources of opacity (Kepler et al. 2007). I also imposed moderately bright constraints upon the minimum observed brightnesses of the candidates so that these could be observed spectroscopically on an 8m class in a reasonable time (e.g. 15 minutes in photometric conditions, or 2 hours in bad weather conditions).

An SQL query (see below) was executed on the SDSS DR7 sky server to search for close pairs of blue objects (BOCP) with clean photometry and a maximum angular separation of $d \leq 60''$ while passing the following criteria: $u-g \geq -0.7$, $u-g \leq 0.5$, $g-r \geq -0.7$, $g-r \leq 0.0$, $r-i < 0.0$, $r \leq 20$, and $g \geq 15.3$.

```
SELECT P.ObjID as SDSSID, P.ra as RIGHT_ASCENSION, P.dec as DECLINATION, P.b as GALATIC_LATITUDE, P.
    l as GALATIC_LONGITUDE, P.psfmag_u as u, P.psfmag_g as g, P.psfmag_r as r, P.clean as pok
FROM Star P
WHERE
   P.clean > 0
    and (P.psfmag_u-P.psfmag_g)<=0.5
    and (P.psfmag_u-P.psfmag_g)>=-0.7
    and (P.psfmag_g-P.psfmag_r)<=0.0
    and (P.psfmag_g-P.psfmag_r)>=-0.7
    and (P.psfmag_r-P.psfmag_i)<0.0
    and P.psfmag_r<=20
    and P.psfmag_g>=15.3
```

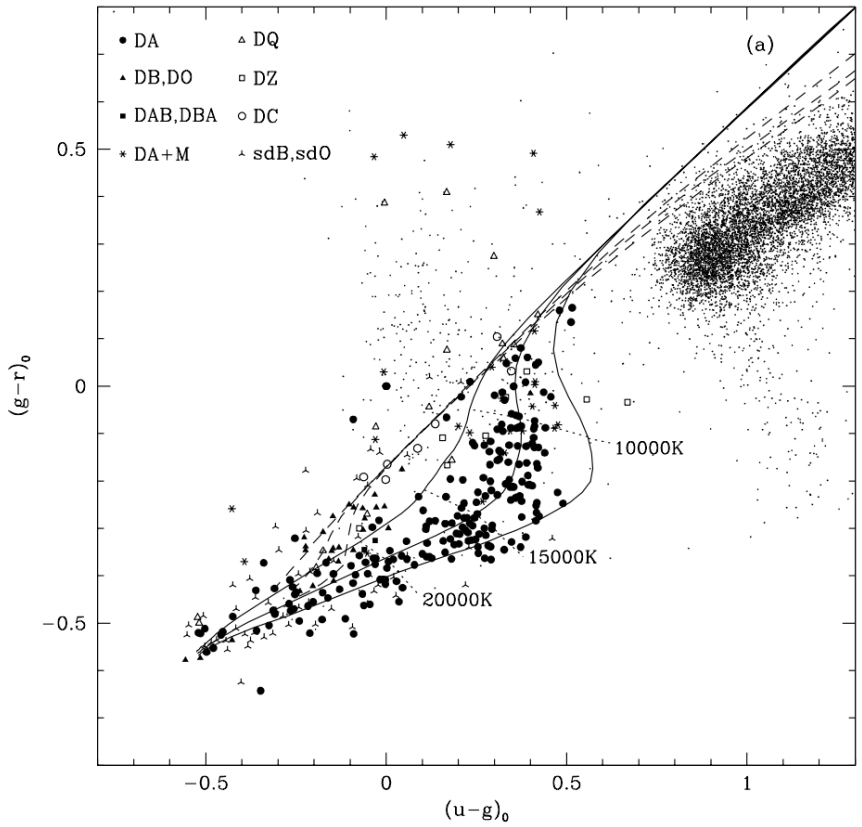


FIGURE 2.2: WD photometry selection criteria diagram, from Harris et al. (2003). Colour-colour diagrams ($u-g$ vs. $g-r$) showing the WDs and hot subdwarfs. Different types are shown with different symbols. Small dots show all objects with stellar image profiles with $15 < g < 20$ in a region of $25 deg^2$. The curves show the colours of WD model atmospheres Bergeron et al. (1995b) of pure H (solid curves) and pure He (dashed curves) with $\log g = 7$, 8, and 9, where the $\log g = 7$ curve is toward the lower right and the $\log g = 9$ curve is toward the upper left. The dotted lines with labels connect models with the same effective temperature.

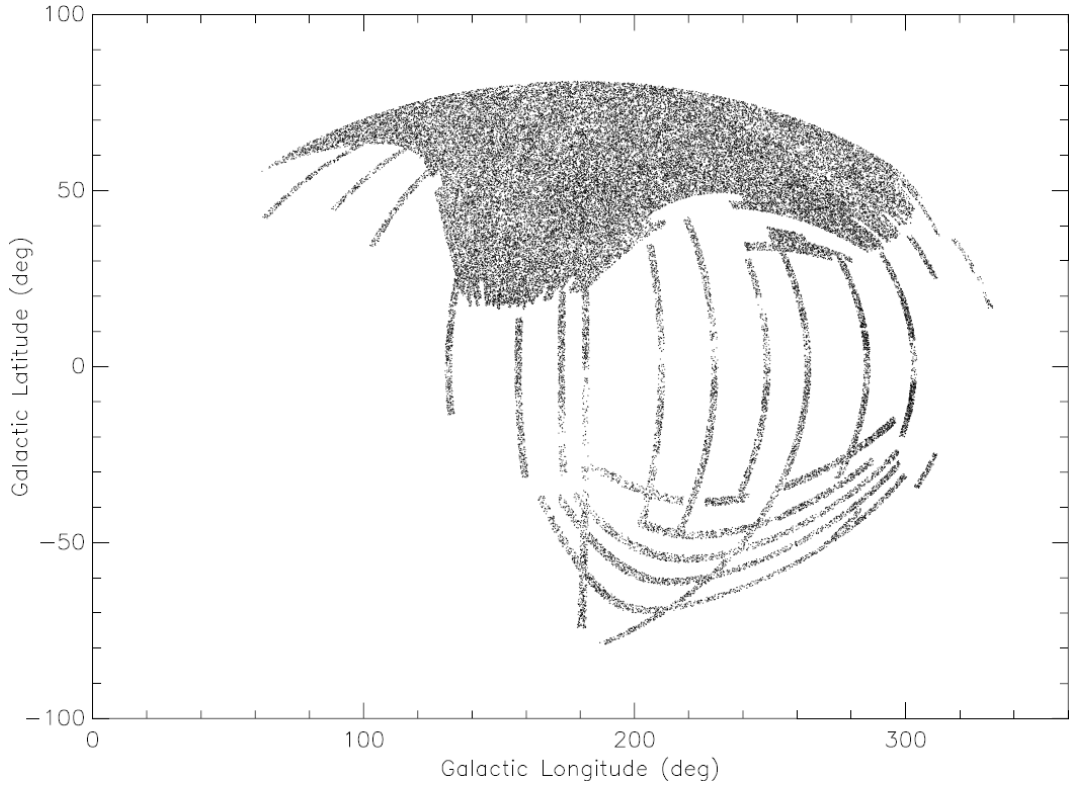


Figure 2.3: All WD Candidates (Blue Objects from SDSS DR7 Survey) shown with Aitoff projection of galatic coordinates

Blue Object Close Pair (BOCP) selection

I initially drew together a list of candidate double-degenerate systems based on all pairs with separations $d \leq 30''$. Routines written in Interactive Data Language (IDL) were used to analyse the SDSS data after performing the query to further refine the selection of WD candidates. For example, sources that were spectroscopically confirmed to be quasars were eliminated from this sample using a list based upon SDSS DR7 (Schneider et al. 2010). In addition, obviously "bad" systems, such as spurious pairings in nearby galaxies, were eliminated using a visual check. Figure 2.7 below gives examples of these "bad" objects. The codes used to generate the list of close pairs, to eliminate quasars and to eliminate "bad" objects are listed in Appendix Sections A.2.5, A.2.6 and A.2.7. I also excluded all BOCPs with angular separations $< 1''$, since given the typical spatial resolution of the SDSS imaging these were deemed most likely to be spurious.

Figures 2.8, 2.9, and 2.10 below display all wide DD candidates selected using this method. Tables 2.2 and 2.3 display their SDSS coordinates and photometry. Table 2.1 provides my estimates of the percentage quasar contamination of my BO and BOCP lists respectively. Generic BO list and BOCP list filtering code (for known "bad" objects, spectroscopically confirmed quasars, spectroscopically confirmed white dwarfs etc) was developed for the purposes of refining and confirming my simulations for this thesis.

2.1.4 Simulations

Monte-Carlo like simulations

Before applying for and conducting new spectroscopic follow-up of my candidate wide DD systems, I performed a basic statistical analysis on the spatial distribution of blue objects in the SDSS DR7 data to estimate the probability that a pairing was an actual physical system and not merely a chance alignment (ie. a visual binary). This was made with regards to the number of blue object pairs expected to be found in the SDSS sample where no physical association existed between objects (field WDs or quasars). A comparison was then made between this simulated random distribution and that observed.

In my Monte-Carlo like analysis, firstly I randomly distributed a specified number of simulated blue objects within an area of a particular size, such that the mean projected spatial density of these sources was consistent with that observed in the SDSS frames. For every object within this random field, the angular distance to its nearest neighbour was then calculated, and a frequency distribution was generated for each object within the random field having a nearest neighbour up to a chosen maximum separation. This distribution was found to be exponential in shape. The code used to generate this distribution and calculate the probability of close pairs is referenced in Appendix Section A.2.3. The distribution from my simulations was verified to be equivalent to that predicted using the analytical equation of Struve 1852 (Halbwachs 1988), shown in Equation 2.1.

$$n(\rho_{max}) = N(N-1)\pi\rho_{max}^2/(2A). \tag{2.1}$$

Next, for every blue object within the SDSS DR7 data, the angular separation to its nearest neighbour was calculated, and a second frequency distribution, based on the observed data, was generated. The code used to calculate the probability of close pairs within a BO list is referenced in Appendix Section A.2.2.

When performing these latter calculations, I used a cleaned list of observed sources (ie. excluding spurious, misclassified, pairings such as blue point-like sources within the same nearby galaxies and known quasars). By comparing these two frequency distributions, I was able to estimate the number of real (physically associated) BOCP in the DR7 data. The results of these comparisons are displayed in Figures 2.4 and 2.6. It was found that there was a clear excess of blue object pairings in the observed over the simulated (random) samples at smaller separations ($\sim$50 systems at separations less than $30''$). Indeed, from the simulations I estimated that within $30''$, there is a $\approx 80\%$ probability that a blue object close pair is physically associated (ie. not merely a chance alignment), and similarly within $10''$, a $\approx 95\%$ probability of a physical association. A slight discrepancy is apparent between the shape of the observed distribution and the theoretical distribution at large angular separations ($\gtrsim 100''$, see Figure 2.5). This is likely due to the breakdown of the assumption in my simulations of a uniform density of blue objects across the SDSS footprint. However, giving the limited time frame for this project and my desire that a high proportion of genuine systems are followed-up spectroscopically, I concentrated on pairings that had separations $< 30''$ (the vast majority actually have $\lesssim 10''$).

The probability of finding a chance alignment is one minus the probability of physical association of a single system to the power of the number of BOCPs in the sample. So in the case of 10 BOCPs (DD candidates), each with an angular separation of $10''$, given the probability of physical association of a single system (using the monte-carlo statistics alone) is $\approx 90\%$, the probability of at least one of these systems being a chance alignment (not associated) is 65% ($1 - \frac{9}{10}^{10}$) as per Equation 2.2, where x is the probability of any one of a number of DD candidates being a chance alignment, y is the probability of a system being a chance alignment, and z is the number of DD candidates.

$$x = 1 - (1 - y)^z \tag{2.2}$$

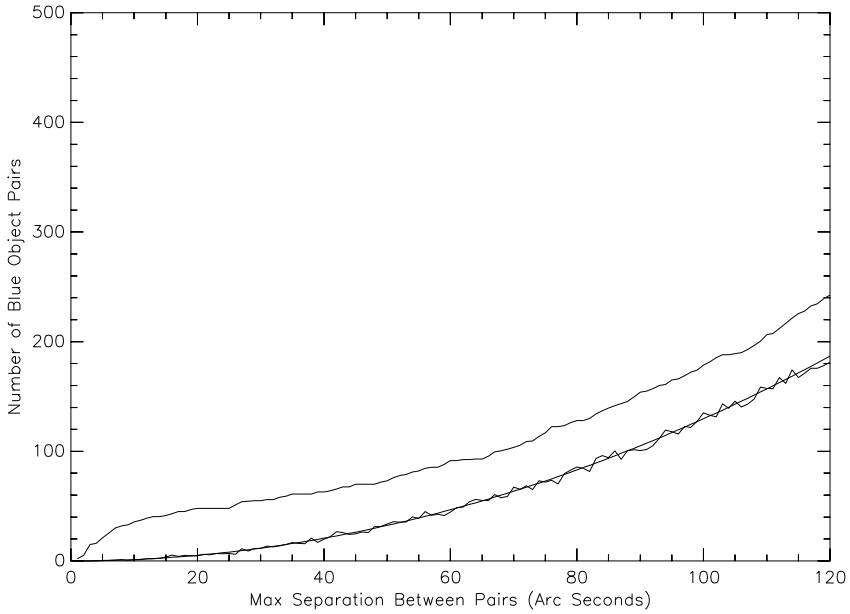


FIGURE 2.4: Plot showing the number of BOCPs found in my simulated random distribution of single objects as a function of separation in arc-seconds (lower line) overlayed with the analytical estimate of Struve 1852. The number of BOCPs measured in the SDSS DR7 to a given maximum separation is also shown (upper line). The difference between these two curves provides a handle on the frequency of physically associated BOCPs found in the SDSS within a given maximum separation.

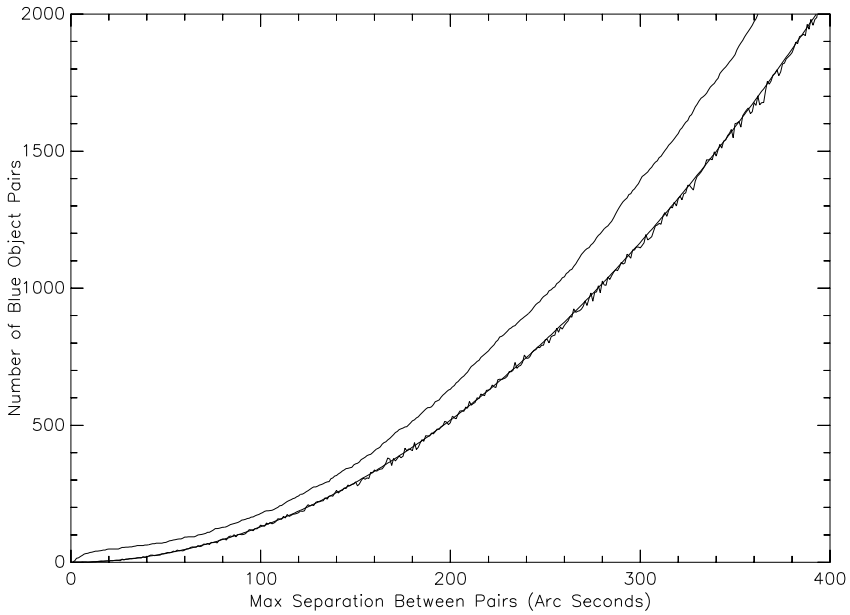


FIGURE 2.5: Same as Figure 2.4 but shown out to a much larger angular separation ($\sim 6'$)

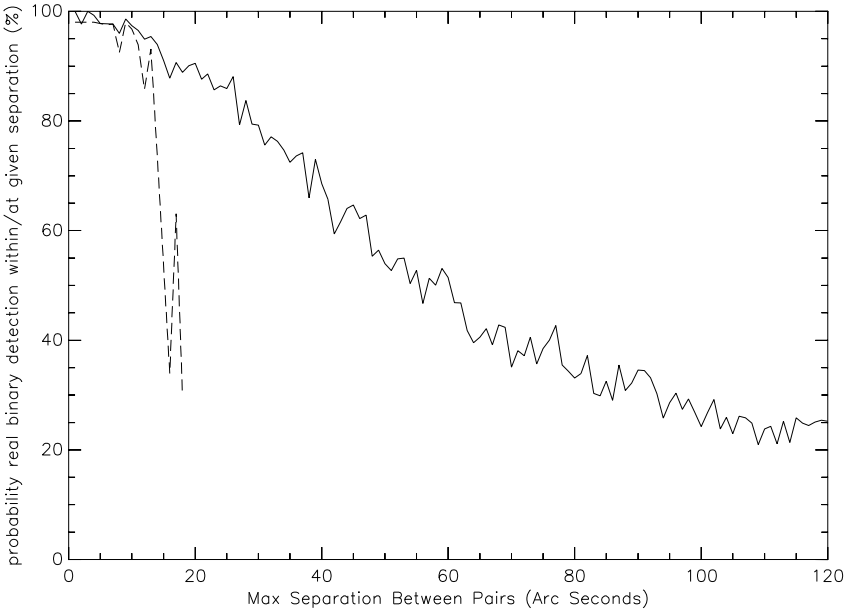


Figure 2.6: The likelihood (expressed as a percentage) that a BOCP is a physically associated system within a given maximum separation (solid curve). The probability that a BOCP is physically associated at a given separation is also overplotted (dashed curve).



Figure 2.7: Examples of "bad" objects within my initial sample. These were rejected by my visual inspection of all pairs selected initially.

Sample	number	DR	DR coverage (sq deg)	DR total num objects (million)	% representation (in sample)	% (DR corrected)	% (DR & have spectra corr. - worst case)	% (DR & have spectra corr. - ideal)
BOL Blue Objects	36681	DR7	11663	357	100	100	NA	NA
BOL SDSS Spectra available	9788	DR7	11663	357	26.7	26.7	NA	NA
BOL known Quasars	1291	DR7	11663	357	3.5	3.5	13.2	3.5
BOL known WDs	5350	DR4	6670	180	14.6	25.5	95.6	95.6
BOCPL Blue Object Close Pairs	53	DR7	11663	357	100	100	NA	NA
BOCPL BOCPs with SDSS Spec. for a component	13	DR7	11663	357	24.5	24.5	NA	NA
BOCPL BOCPs with a known Quasar component	0	DR7	11663	357	0	0	0	0
BOCPL BOCPs with a known WD component	12	DR4	6670	180	22.6	39.6	161.4	161.4

Table 2.1: Blue Object List (BOL) and Blue Object Close Pair List (BOCPL) Statistics - this table identifies the percentage contamination of my sample of blue objects due to quasars. "Bad" objects have not been removed from this sample. Note the 'ideal' case column assumes that the photometric target selection of quasars for spectroscopic follow-up within the SDSS program has been perfect (i.e. every quasar within the sample of blue objects has been targeted for spectroscopic follow-up based on its photometric properties and has been subsequently identified). The 'Worst' case column assumes that the spectroscopic follow-up of quasars within the SDSS program has not been biased based upon photometric data (any more than the effective biasing/probability of spectroscopic follow-up for my sample of blue objects). This table also includes an indication of the expected percentage of WDs within my sample based upon known WDs from Eisenstein et al. (2006), and assumes that the spectroscopic follow-up of objects within the SDSS program has not been biased in any way based upon photometric data towards the selection of WDs (i.e. constraints were purely chosen based upon expected quasar photometric signatures).



Figure 2.8: Finder charts (part 1/3) for the candidate wide double-degenerate systems found in SDSS DR7 ($< 30''$). Note that these are arc $30'' \times 30''$, with North at top end and East at the left.

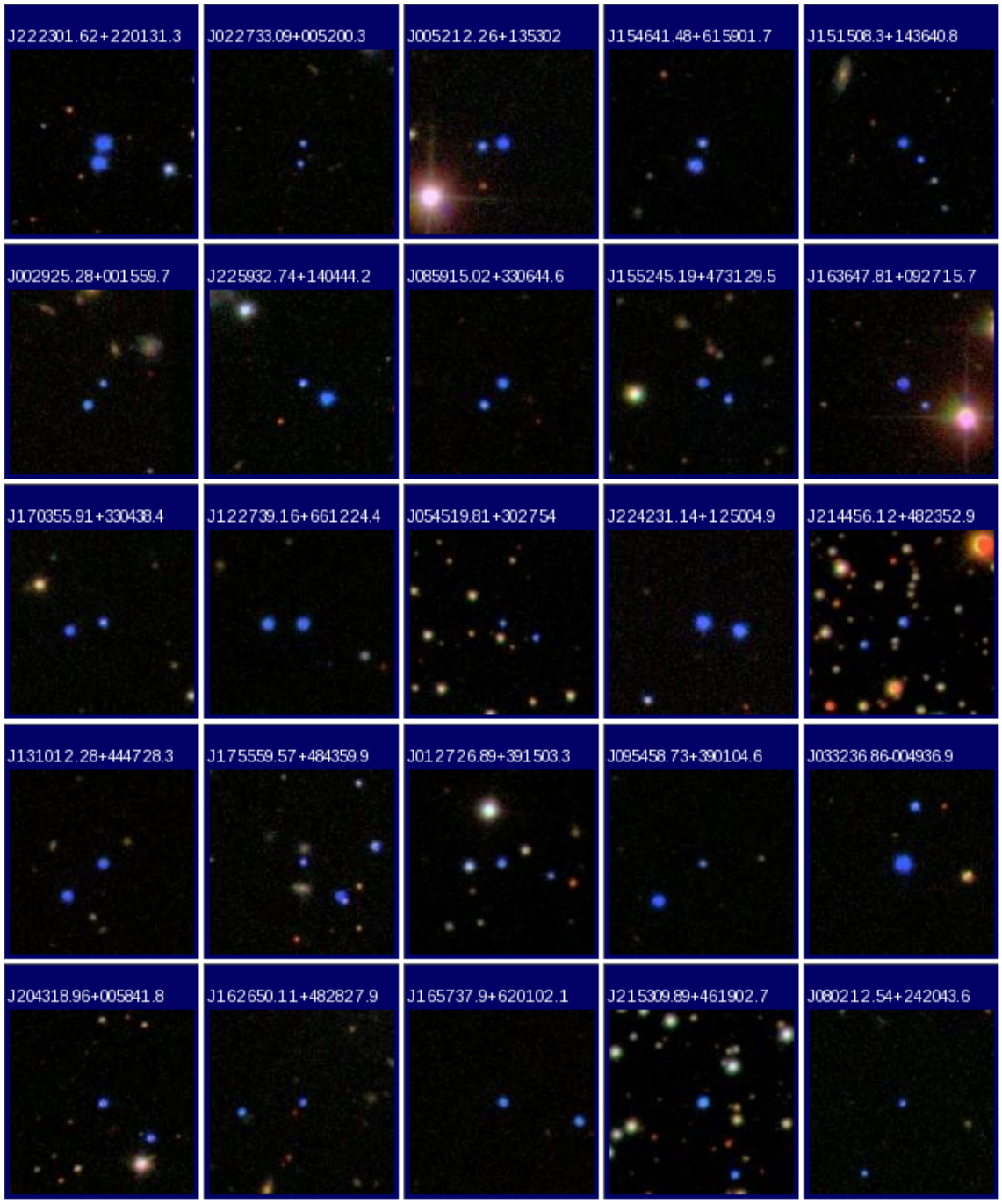


FIGURE 2.9: Finder charts (part 2/3) for the candidate wide double-degenerate systems found in SDSS DR7 ($< 30''$). Note that these are arc $30'' \times 30''$, with North at top end and East at the left.



Figure 2.10: Finder charts (part 3/3) for the candidate wide double-degenerate systems found in SDSS DR7 ($< 30''$). Note that these are arc $30'' \times 30''$, with North at top end and East at the left.

TABLE 2.2: Blue Object Close Pair Data - SDSS

BOCP ID	DD ID	RA Dec (A) (J2000)	u (A) (psf)	g (A) (psf)	r (A) (psf)	δu(A) (psf)	δg(A) (psf)	δr(A) (psf)	RA Dec (B) (J2000)	u (B) (psf)	g (B) (psf)	r (B) (psf)	δu(B) (psf)	δg(B) (psf)	δr(B) (psf)	Sep. ($''$)
BOCP1		J115305.54+005646.1	18.42	18.89	19.38	0.02	0.02	0.02	J115305.47+005645.8	18.5	18.91	19.34	0.02	0.02	0.02	1.22
BOCP2	DD-11	J132814.28+163151.5	16.34	16.27	16.63	0.02	0.02	0.02	J132814.36+163150.9	17.75	17.65	17.74	0.27	0.23	0.19	1.32
BOCP3		J074853.07+302543.5	17.41	17.59	17.88	0.07	0.05	0.05	J074852.95+302543.4	17.57	17.59	17.96	0.05	0.04	0.04	1.5
BOCP4		J075410.53+123947.3	19.23	18.76	18.96	0.06	0.06	0.08	J075410.58+123945.5	19.22	18.84	19.04	0.06	0.07	0.09	2
BOCP5	DD-06	J021131.51+171430.4	17.36	17.26	17.65	0.12	0.09	0.07	J021131.52+171428.3	16.69	16.76	17.19	0.03	0.03	0.03	2.04
BOCP6		J211607.27+004503.1	18.61	18.7	18.91	0.03	0.04	0.04	J211607.2+004501.3	19.43	18.96	19.05	0.1	0.07	0.09	2.06
BOCP7		J000142.84+251506.1	17.82	17.81	18.18	0.02	0.02	0.02	J000142.79+251504	19.16	18.7	19.01	0.19	0.16	0.16	2.16
BOCP8		J173249.57+563900	19.55	19.07	19.22	0.07	0.05	0.05	J173249.32+563858.8	18.99	19.12	19.27	0.04	0.05	0.06	2.36
BOCP9		J222427.07+231537.4	17.6	17.2	17.43	0.03	0.03	0.03	J222426.91+231536	18.22	17.77	17.92	0.08	0.07	0.07	2.64
BOCP10		J131421.7+305051.4	18.59	18.2	18.22	0.1	0.08	0.09	J131421.5+305050.5	18.23	17.86	17.88	0.04	0.04	0.05	2.76
BOCP11		J072147.38+322824.1	18.24	18.1	18.22	0.03	0.04	0.03	J072147.2+322822.4	18.73	18.28	18.32	0.08	0.04	0.05	2.77
BOCP12		J113928.52-001420.9	19.92	19.45	19.61	0.06	0.04	0.04	J113928.47-001418	20.13	19.8	19.85	0.07	0.06	0.06	2.95
BOCP13		J233246.27+491712	18.8	18.67	18.94	0.05	0.05	0.04	J233246.23+491709.1	19.02	18.76	19.04	0.06	0.04	0.04	2.96
BOCP14		J213648.79+064320.2	18.07	17.94	18.24	0.02	0.02	0.01	J213648.98+064318.2	19.72	19.35	19.39	0.04	0.02	0.03	3.44
BOCP15	DD-12	J222236.3-082808	16.68	16.41	16.67	0.02	0.02	0.03	J222236.56-082806	17.56	17.11	17.3	0.03	0.07	0.07	4.29
BOCP16	DD-08	J092647+132138.4	18.74	18.4	18.46	0.03	0.03	0.05	J092646.88+132134.5	18.46	18.34	18.39	0.02	0.02	0.02	4.35
BOCP17		J084952.87+471249.4	16.64	16.77	17.08	0.02	0.02	0.02	J084952.47+471247.7	18.14	17.77	17.79	0.14	0.08	0.06	4.37
BOCP18		J085917.36+425031.6	19.37	18.94	19.01	0.03	0.05	0.02	J085917.23+425027.4	18.83	18.38	18.53	0.02	0.04	0.02	4.39
BOCP19		J092513.18+160145.4	17.05	17.13	17.51	0.08	0.07	0.1	J092513.48+160144.1	16.05	16.13	16.55	0.02	0.02	0.02	4.51
BOCP20	DD-02	J150746.48+521002.1	17.14	16.91	17.29	0.02	0.03	0.01	J150746.8+520958	17.98	17.76	18.06	0.02	0.03	0.01	5.05
BOCP21		J011714.48+244021.5	19.94	19.63	19.77	0.03	0.02	0.02	J011714.12+244020.3	20.29	19.83	19.96	0.04	0.02	0.02	5.05
BOCP22		J080644.09+444503.2	18.54	18.14	18.32	0.02	0.02	0.01	J080643.64+444501.4	19.18	18.74	18.82	0.02	0.02	0.01	5.09
BOCP23	DD-09	J115937.81+134413.9	18.45	18.07	18.12	0.03	0.02	0.02	J115937.82+134408.7	18.42	18.28	18.52	0.03	0.02	0.02	5.18
BOCP24	DD-10	J131332.14+203039.6	18.13	17.8	17.98	0.02	0.02	0.01	J131332.56+203039.3	17.86	17.48	17.69	0.02	0.02	0.01	5.93
BOCP25		J115030.12+253210.1	20.43	19.95	19.97	0.05	0.02	0.02	J115030.48+253206	19.28	18.86	19.09	0.03	0.02	0.02	6.38
BOCP26	DD-05	J222301.62+220131.3	15.66	15.6	15.91	0.01	0.02	0.01	J222301.72+220124.9	16.37	16.01	16.2	0.01	0.03	0.03	6.56
BOCP27		J022733.09+005200.3	20.02	19.62	19.69	0.05	0.02	0.02	J022733.15+005153.6	20.3	19.86	19.91	0.06	0.02	0.02	6.72

Table 2.3: Blue Object Close Pair Data - SDSS (continued)

BOCP ID	DD ID	RA Dec (A) (J2000)	u (A) (psf)	g (A) (psf)	r (A) (psf)	δu(A) (psf)	δg(A) (psf)	δr(A) (psf)	RA Dec (B) (J2000)	u (B) (psf)	g (B) (psf)	r (B) (psf)	δu(B) (psf)	δg(B) (psf)	δr(B) (psf)	Sep ($''$)
BOCP28	DD-01	J005212.26+135302	17.79	17.71	17.98	0.02	0.03	0.02	J005212.73+135301.1	19.35	18.89	18.92	0.03	0.03	0.02	6.78
BOCP29		J154641.48+615901.7	19.07	18.63	18.75	0.03	0.02	0.02	J154641.79+615854.3	17.16	16.89	17.16	0.02	0.02	0.02	7.64
BOCP30		J151508.3+143640.8	18.38	18	18.2	0.02	0.02	0.01	J151507.9+143635.4	19.76	19.63	19.88	0.03	0.02	0.02	7.9
BOCP31		J002925.28+001559.7	20.02	19.59	19.59	0.05	0.02	0.02	J002925.62+001552.7	18.91	18.48	18.53	0.03	0.01	0.02	8.64
BOCP32	DD-04	J225932.74+140444.2	19.02	18.57	18.68	0.03	0.02	0.01	J225932.21+140439.2	16.16	16.36	16.78	0.02	0.01	0.01	9.14
BOCP33		J085915.02+330644.6	18.27	18.01	18.34	0.02	0.02	0.02	J085915.5+330637.6	19.07	18.7	18.87	0.03	0.02	0.02	9.29
BOCP34		J155245.19+473129.5	18.79	18.71	19.05	0.02	0.02	0.02	J155244.41+473124	19.21	18.99	19.3	0.04	0.03	0.02	9.65
BOCP35		J163647.81+092715.7	18.13	17.72	17.93	0.02	0.01	0.01	J163647.33+092708.4	19.98	19.54	19.54	0.04	0.02	0.02	10.12
BOCP36	DD-03	J170355.91+330438.4	19.16	18.81	18.86	0.02	0.01	0.01	J170356.77+330435.7	18.48	18.16	18.27	0.02	0.01	0.01	11.16
BOCP37		J122739.16+661224.4	17.72	17.86	18.13	0.02	0.02	0.02	J122741.05+661224.3	18.23	17.99	18.21	0.02	0.02	0.02	11.43
BOCP38		J054519.81+302754	20.19	19.82	19.96	0.05	0.02	0.02	J054518.98+302749.3	20.05	19.64	19.8	0.05	0.02	0.02	11.72
BOCP39		J224231.14+125004.9	16.48	16.23	16.5	0.01	0.02	0.01	J224230.33+125002.3	16.83	16.5	16.75	0.01	0.02	0.01	12.13
BOCP40		J214456.12+482352.9	19.19	18.74	18.83	0.03	0.01	0.02	J214457.39+482345.5	19.81	19.49	19.49	0.05	0.02	0.02	14.67
BOCP41		J131012.28+444728.3	17.88	17.84	18.02	0.01	0.02	0.01	J131013.38+444717.8	17.95	17.59	17.85	0.01	0.02	0.01	15.71
BOCP42		J175559.57+484359.9	19.04	19.21	19.39	0.03	0.02	0.02	J175558.35+484348.8	18.03	17.69	17.91	0.02	0.02	0.02	16.41
BOCP43		J012726.89+391503.3	19.16	18.7	18.83	0.03	0.02	0.02	J012725.51+391459.2	20.35	19.99	19.99	0.06	0.02	0.02	16.55
BOCP44		J095458.73+390104.6	20.31	19.86	19.95	0.05	0.02	0.03	J095459.97+390052.4	17.96	17.69	18.02	0.02	0.02	0.02	18.87
BOCP45	DD-07	J033236.86-004936.9	15.32	15.64	16.09	0.01	0.02	0.02	J033236.6-004918.4	18.64	18.2	18.3	0.03	0.02	0.02	18.91
BOCP46		J204318.96+005841.8	18.51	18.24	18.42	0.03	0.02	0.01	J204317.93+005830.5	18.96	18.59	18.75	0.03	0.02	0.01	19.13
BOCP47		J162650.11+482827.9	19.72	19.62	19.94	0.03	0.02	0.03	J162652.12+482824.7	19.14	18.98	19.3	0.02	0.01	0.02	20.22
BOCP48		J165737.9+620102.1	18.72	18.65	18.98	0.02	0.01	0.03	J165734.39+620055.9	18.88	18.53	18.76	0.02	0.01	0.02	25.47
BOCP49		J215309.89+461902.7	18.15	17.72	17.9	0.02	0.01	0.01	J215308.9+461839.1	18.88	19.08	19.36	0.03	0.01	0.02	25.68
BOCP50		J080212.54+242043.6	19.87	19.54	19.78	0.04	0.02	0.02	J080213.44+242020.9	20.24	19.85	19.97	0.05	0.02	0.02	25.85
BOCP51		J135713.14-065913.7	18.94	19.25	19.76	0.04	0.02	0.02	J135714.5-065856.9	18.58	18.16	18.35	0.04	0.02	0.02	26.29
BOCP52		J100245.86+360653.3	19.42	19.04	19.09	0.03	0.02	0.02	J100244.88+360629.6	19.32	18.92	19.01	0.03	0.02	0.02	26.53
BOCP53		J105306.13+025052.5	19.57	19.14	19.28	0.04	0.02	0.02	J105306.82+025027.9	19.37	18.98	19.18	0.03	0.02	0.02	26.6

2.2 Follow-up spectroscopic observations

To confirm my sources as WDs and to better determine their parameters (i.e. their surface gravities and effective temperatures) it was necessary to obtain high signal-to-noise low resolution optical spectroscopy in the range of $\lambda \approx 3800 - 5200$Å. Therefore a series of observing proposals were written and submitted to a number of different telescopes to secure these follow-up datasets.

2.2.1 Prioritisation of candidate wide DD systems for follow-up

I initially considered prioritising the spectroscopic follow-up of systems suspected on the grounds of their magnitudes and colours as having the most suitable mass ratios for probing the IFMR (Appendix Section A.2.10). Ultimately, due to observing constraints, I was forced not to implement this. However as a result, I can provide a first estimate of the mass distribution of WDs in wide binary systems. Targets were instead selected based upon brightness and separation. Figure 2.11 below gives a list of my candidates selected for spectroscopic follow-up, and Table 2.4 displays their SDSS coordinates and photometry.



Figure 2.11: SDSS Double-Degenerate Survey - Target Selection (1x1′). Systems which have been followed up spectroscopically.

Table 2.4: Double-Degenerate Data - SDSS

WD	SDSS ID	RA Dec (*J2000*)	Ra (*J2000*)	Dec (*J2000*)	psf u	psf g	psf r	psf u error	psf g error	psf r error	Separation (arc sec)
DD-01A	587724231565246516	005212.73+135301.1	13.05304438	13.88365572	19.35	18.89	18.92	0.029	0.033	0.019	6.78
DD-01B	587724231565246515	005212.26+135302.0	13.05112233	13.88390554	17.79	17.71	17.98	0.019	0.033	0.017	6.78
DD-02A	587733604259332157	150746.48+521002.1	226.9436913	52.16725612	17.14	16.91	17.29	0.023	0.028	0.013	5.05
DD-02B	587733604259332160	150746.80+520958.0	226.94503382	52.16612142	17.98	17.76	18.06	0.025	0.028	0.014	5.05
DD-03A	587729752747475228	170355.91+330438.4	255.98296977	33.07733943	19.16	18.81	18.86	0.023	0.013	0.012	11.16
DD-03B	587729752747475236	170356.77+330435.7	255.98656627	33.07660731	18.48	18.16	18.27	0.017	0.012	0.011	11.16
DD-04A	587727221943959713	225932.74+140444.2	344.88642274	14.07895147	19.02	18.57	18.68	0.025	0.015	0.015	9.14
DD-04B	587727221943959712	225932.21+140439.2	344.88422825	14.07756995	16.16	16.36	16.78	0.015	0.014	0.013	9.14
DD-05A	587740712426537180	222301.62+220131.3	335.75675515	22.02537688	15.66	15.6	15.91	0.012	0.016	0.009	6.56
DD-05B	587740712426537181	222301.72+220124.9	335.75717854	22.02359622	16.37	16.01	16.2	0.012	0.033	0.03	6.56
DD-06A	758881524261781580	021131.52+171428.3	32.88134033	17.24121308	16.69	16.76	17.19	0.026	0.029	0.028	2.04
DD-06B	758881524261781581	021131.51+171430.4	32.88133036	17.24177863	17.36	17.26	17.65	0.115	0.093	0.071	2.04
DD-07A	588015508218904600	033236.86-004936.9	53.1535948	-0.82692479	15.32	15.64	16.09	0.011	0.021	0.023	18.91
DD-07B	588015508218904597	033236.60-004918.4	53.15251716	-0.82178397	18.64	18.2	18.3	0.025	0.022	0.023	18.91
DD-08A	587745403612430400	092646.88+132134.5	141.69533369	13.35958843	18.46	18.34	18.39	0.02	0.022	0.017	4.35
DD-08B	587745403612430401	092647.00+132138.4	141.69584285	13.36069038	18.74	18.4	18.46	0.033	0.032	0.047	4.35
DD-09A	587735346962432035	115937.81+134413.9	179.90758042	13.73720478	18.45	18.07	18.12	0.03	0.025	0.018	5.18
DD-09B	587735346962432036	115937.82+134408.7	179.90760447	13.73576715	18.42	18.28	18.52	0.026	0.019	0.019	5.18
DD-10A	587742578058395699	131332.14+203039.6	198.38394049	20.5110005	18.13	17.8	17.98	0.02	0.021	0.015	5.93
DD-10B	587742578058395698	131332.56+203039.3	198.38569698	20.51094081	17.86	17.48	17.69	0.018	0.021	0.014	5.93
DD-11A	587742774031024218	132814.28+163151.5	202.05953933	16.53098677	16.34	16.27	16.63	0.017	0.024	0.018	1.32
DD-11B	587742774031024219	132814.36+163150.9	202.05987454	16.5308083	17.75	17.65	17.74	0.269	0.234	0.186	1.32
DD-12A	587726878883381389	222236.30-082808.0	335.65128196	-8.46888943	16.68	16.41	16.67	0.015	0.021	0.026	4.29
DD-12B	587726878883381390	222236.56-082806.0	335.65235447	-8.46834597	17.56	17.11	17.3	0.031	0.07	0.067	4.29

Gemini Observing Proposals

Four applications were submitted to the Australian Time Allocation Committee (ATAC) to obtain service mode observations for the components of a number of my pairings with the Gemini telescopes during periods of poorer weather in the 2009B and 2010A semesters. The following objects were included in the target lists of these proposals:

GDD-01A/B (00:29:25.62 00:15:52.74), GDD-02A/B (03:32:36.6 -00:49:18.42) and GDD-07A/B (22:22:36.31 -8:28:8.0), were submitted for Gemini South 2009B. GDD-03A/B (08:49:52.87 47:12:49.47), GDD-04A/B (09:25:13.48 16:01:44.15), GDD-05A/B (09:26:46.88 13:21:34.52), GDD-06A/B (10:02:44.88 36:06:29.68), GDD-08A/B (22:23:1.62 22:01:31.36), GDD-20A/B (02:11:31.52+17:14:28.3), were submitted for Gemini North 2009B. GDD-01A/B (00:29:25.62 00:15:52.74), GDD-07A/B (22:22:36.31 -8:28:8.0), were submitted for Gemini South 2010A. GDD-101 (07:48:53.07 30:25:43.56), GDD-102 (08:49:52.9 47:12:49.5), GDD-104 (12:27:39.17 66:12:24.46), GDD-105 (13:13:32.57 20:30:39.39), and GDD-106 (22:24:27.08 23:15:37.49), were submitted for Gemini North 2010A.

Two of these proposals were successful with 5.5 and 4.0 hours being awarded on Gemini-N (2009B) and Gemini-S (2010A) respectively. The two successful Gemini observing proposals are included in the Appendix Section A.7.7.

WHT Observing Proposal

A proposal was also submitted to the UK Panel for the Allocation of Telescope Time (PATT) prior to the commencement of this thesis for the purposes of obtaining 3 nights of time on the William Herschel Telescope (WHT) during the 2008A semester. This proposal was approved and observations were carried out by Paul Dobbie on 24-25 July 2008. The observations were originally proposed to test both our understanding of spectroscopic mass determinations at $T_{\rm eff}$ <12000K (e.g. Kepler et al. 2007) and to probe the IFMR across a broad mass range, by constraining their age using calculations from the higher mass components. This proposal successfully secured 2 nights of observing time and has been added to the Appendix Section A.7.5.

VLT Observing Proposal

An application for observing time on the European Southern Observatory's Very Large Telescope for another project on the IFMR was granted 2 nights in February 2010. The aims of the proposal were the targeting of 10 faint, suspected high mass ($M_i \geq 5{-}6M_{\odot}$) WDs with high probability of membership of the young open cluster systems, NGC2287 and NGC3532. A selection of double-degenerate systems were included as backup targets, in case observing conditions were not favourable for observing the extremely faint primary targets. The VLT observing proposal has been added to the Appendix Section A.7.8.

TABLE 2.5: Telescope Instrument Properties

Telescope	Instrument	Grating	Central λ (Å)	λ range (Å)	disp (Å/mm)	disp (Å/pixel)	$\delta\lambda$ (FWHM in Å) (with binning)	$\delta\lambda$ (FWHM in ch) (with binning)	pixel width (μm/pixel)	R ($\lambda/\delta\lambda$)	binning (1/2/4x)	Slit ($''$)
VLT	FORS2	600B	4650	3300 - 6210	50	0.75	3.9	5.2	15	600	2	1.3
Gemini	GMOS-N/S	B600	4100	2760	37	0.5	2.7	5.5	13.5	422	4	2
WHT	IRIS blue	R300B	4503	3539	64	0.86	3.4	4	13.5	1309	1	1

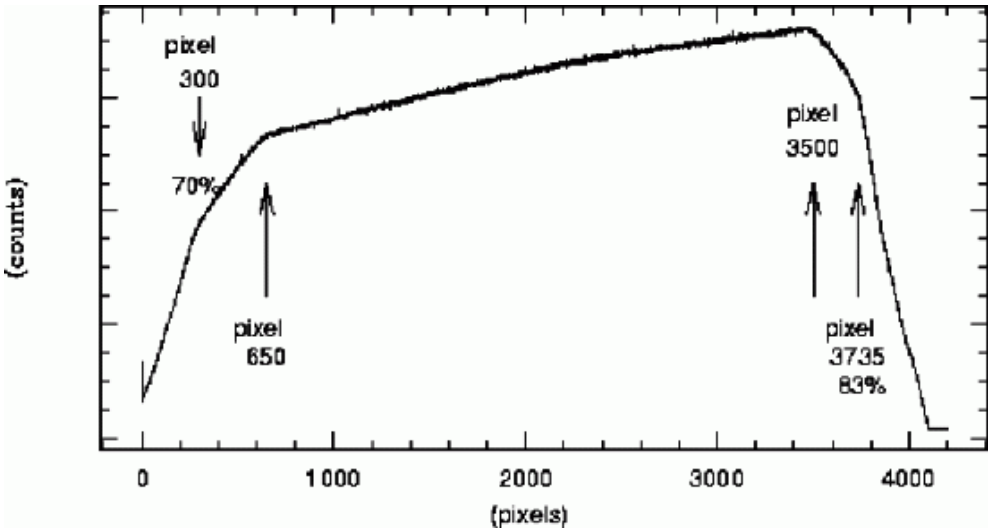


Figure 2.12: Relative response of the WHT ISIS blue arm

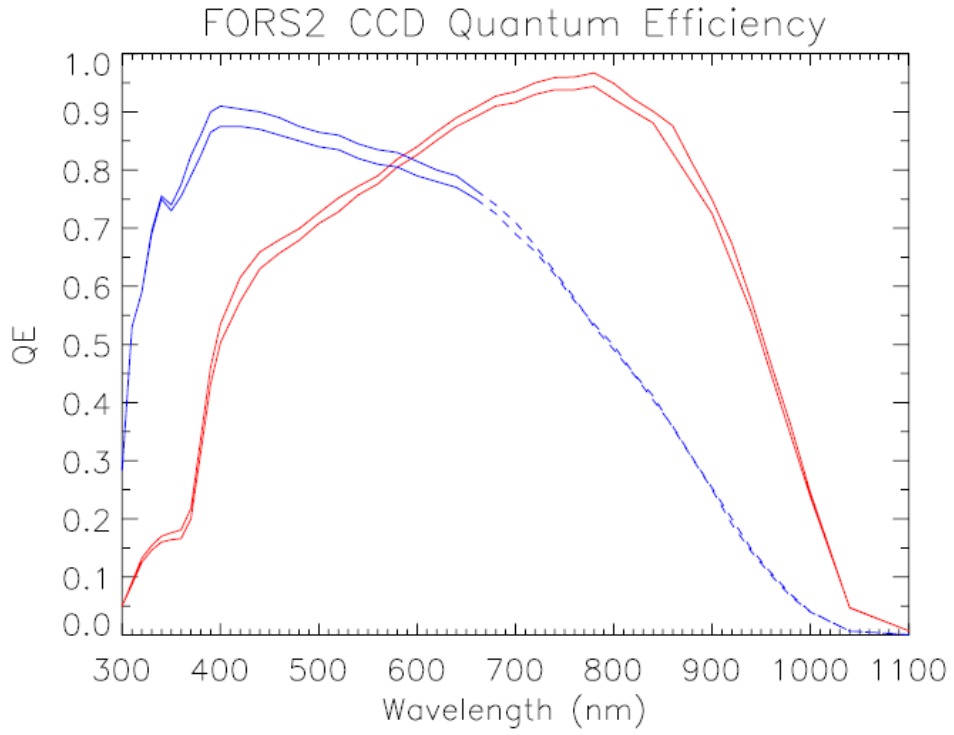
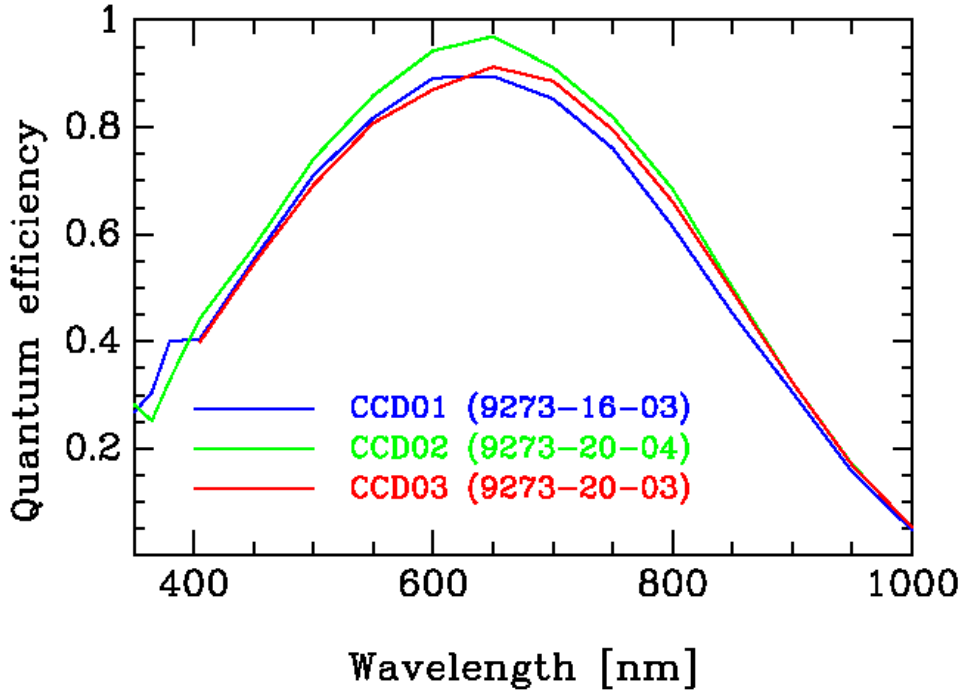


Figure 2.13: VLT FORS2 red and blue CCD quantum efficiency curves



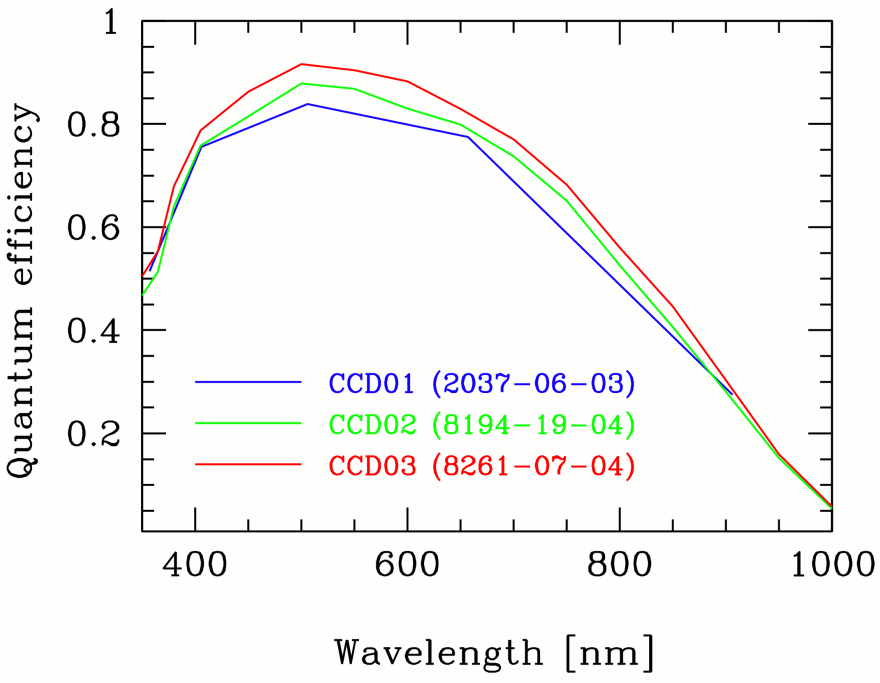


FIGURE 2.14: GMOS North (top) and South (bottom) CCD quantum efficieny curves

2.2.2 The spectroscopic follow-up observations

In the period July 2008 to May 2010 optical spectroscopy was obtained with various telescope and instrument combinations for the components of 12 blue object close pairs. The details of these telescopes and instruments are summarised in Table 2.5. All spectroscopically observed sources were confirmed as WD stars.

I adopted a common scheme for the reduction of the data from each observing run. A DC (virtually spectrally featureless) WD of known effective temperature was observed to remove instrument signature and to provide a first order flux calibration. Multiple exposures were taken (typically 3) of each system such that I could remove cosmic rays using either a $\approx$ 4 sigma-clipped mean or Laplacian "edge-detection" filtering of the data, which ever appeared to be more efficient. In general, the individual exposures were between 10 and 30 minutes long. This insured that the total counts in each CCD pixel from my brightest candidates remained well within the linear regime of the detectors and were also never close to their saturation levels.

WHT Observations

My first batch of follow-up observations were performed with the 4.2m William Herschel Telescope (WHT) and the Intermediate dispersion Spectrograph and Imaging System (ISIS) instrument prior to me starting this project. The 4.2 metre diameter WHT is located on the island of La Palma, Canary Islands, and is part of the Isaac Newton Group of Telescopes (ING). The ISIS spectrograph is mounted at the Cassegrain focus of the WHT and is of double-arm design, with the two channels being separately optimised for the blue and red spectral regions. Only the data from the blue arm is considered in this work. The relative response curve for the blue arm of ISIS is shown below in Figure 2.12.

Four double-degenerate (DD) systems were observed on 24-25 July 2008 with the WHT. The program ID for this observing run was WHT/08A/24. For the purposes of this thesis, these systems are labelled as DD-01A+B, DD-02A+B, DD-03A+B, and DD-04A+B. The slit was aligned to capture the light of component stars simultaneously. All observations were made in visitor mode, in good weather, with $\approx 0.6 - 0.9''$ seeing. They were made using the R300B grating, a $1.0''$ slit width, and with 5x2400s, 3x1800s, 4x1800s+3x1800s, and 4x1800s+1x1800s+1x1700s+1x1200s exposures respectively. Additionally, 3x540s exposures of the DC WD1917+386 (herein called 1918+38 based upon its current/J2000 coordinates) were also taken using a $1.0'$ slit for the purposes of removing instrument response.

Gemini Observations

My second batch of spectroscopic follow-up observations was obtained with the two 8.1m Gemini telescopes and their respective Gemini Multi-Object Spectrograph (GMOS) instruments (GMOS South and GMOS North). The Gemini South telescope resides on the summit of Cerro Pachón in Chile, and the Frederick C. Gillett Gemini North telescope resides on the summit of Mauna Kea in Hawaii. Like ISIS on the WHT, the

GMOS instruments are mounted at the Cassegrain foci of each telescope. GMOS provides long-slit and multi-slit spectroscopy over the range $0.36 - 1.10\mu m$. At the time of my observations, the GMOS-North CCDs favoured somewhat (in terms of response) the red end of the optical regime, whereas the response of those in GMOS-South was more suitable for observing the blue end. This can be seen clearly in Figure 2.14 below.

Two candidate DD systems were observed in service mode on 20 July 2009 and 25 July 2009 using GMOS-North in longslit mode. For the purposes of this thesis these pairings are refered to as DD-05A+B and DD-06A+B. The program ID for this observing run was GN-2009B-Q-80. The slit was aligned along the binary to capture the light of both component stars simultaneously. All observations were made using the B600/410.0 grating/disperser, a 2.0″ slit/mask width, using 3x1800s exposures for DD-05A+B and 3x2000s exposures for DD-06A+B. 3x750s exposures of the DC WD1918+38 were taken for the purposes of removing instrument response.

A further candidate DD system, DD-12A+B, was observed in service mode on 11 May 2010 using GMOS-South in long-slit mode. The program ID for this observing run was GS-2010A-Q-70. As before, the slit was aligned to capture the light of component stars simultaneously. All observations were made using the B600/410.0 grating/disperser, a 2.0″ slit/mask width, with 3x1800s exposures. Two 1560s exposures of the DC WD0000+345 were also acquired.

VLT Observations

My final new spectra were acquired with the European Southern Observatory's 8.2m Very Large Telescope UT1 (Antu) and the FOcal Reducer and low dispersion Spectrograph (FORS2). The Very Large Telescope (VLT) is located at the ESO Paranal Observatory, Chile. FORS2 is mounted at the Cassegrain focus of VLT Antu. FORS2 was designed for the wavelength range from 330 nm to 1100 nm and provides an image scale of 0:0025/pixel (and 0:00125/pixel) with the standard resolution collimator (and the high resolution collimator, respectively), in the default binned (2x2) read-out mode of its mosaic of two CCD chips. FORS2 supports two CCD chip generations in visitor mode, the current generation (FORS2) MIT CCDs and the previous generation (FORS1) E2V CCDs. Each configuration consists of two $2k \times 4k$ chips with pixel size of $15 \times 15\mu m$. The E2V CCDs are particularily sensitive to the blue end of the spectrum. The response curves of the E2V blue optimised CCDs are shown below in Figure 2.13, as compared with the red optimised MIT CCDs.

I observed five candidate DD systems on 6-7 February 2010 using the Long Slit Spectroscopy (LSS) mode of FORS2, before and after the primary science program. These systems are referred to as DD-07A+B, DD-08A+B, DD-09A+B, DD-10A+B, and DD-11A+B for the purposes of this thesis. The program ID for this observing run was 084.D-1097(A). The slit of FORS2 was also aligned to capture the light of both component stars simultaneously. The observations were made using the 600B grating, a 1.3′ slit width, using 2x600s, 1x500+1x360s, 2x600s, 2x300s, and 2x300s exposures respectively. 2x250s+2x180s exposures of DC LHS-2333 were taken such that I could remove the instrument response. During our observations at the VLT it became apparent that one of my DD targets featured both low and a high mass

components. At first glance, SDSS 1159+1344 (DD-09A+B) appears to be an ideal system for the purposes of refining the IFMR.

TABLE 2.6: Double-Degenerate Data - Observations and Signal to Noise

WD	Observation Run Target Name	Observatory	Spectrograph	Date	Exposure Time(s)	Slit (″)	Grating	R (fwhm channels)	Arc Lamp Type	DC WD	S/N
DD-01A	SDSS0052	WHT 4.2m	Long Slit	[25/07/2008]	5x2.4 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	12.2
DD-01B	SDSS0052	WHT 4.2m	Long Slit	[25/07/2008]	5x2.4 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	32.3
DD-02A	SDSS1507	WHT 4.2m	Long Slit	[24/07/2008]	3x1.8 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	64.3
DD-02B	SDSS1507	WHT 4.2m	Long Slit	[24/07/2008]	3x1.8 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	37.8
DD-03A	SDSS1703	WHT 4.2m	Long Slit	[24+25/07/2008]	4x1.8+3x1.8 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	26.8
DD-03B	SDSS1703	WHT 4.2m	Long Slit	[24+25/07/2008]	4x1.8+3x1.8 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	42
DD-04A	SDSS2259	WHT 4.2m	Long Slit	[24+25/07/2008]	4x1.8+1.8+1.7+1.2 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	31.8
DD-04B	SDSS2259	WHT 4.2m	Long Slit	[24+25/07/2008]	4x1.8+1.8+1.7+1.2 ks	1	R300B	3	Cu-Ne + Cu-Ar	WD1918+38	105.3
DD-05A	DD-08A+B	Gemini North	Long Slit	[20/07/2009]	3x0.75 ks	2	B600	4.5	Cu-Ar	WD1918+38	136.9
DD-05B	DD-08A+B	Gemini North	Long Slit	[20/07/2009]	3x0.75 ks	2	B600	4.5	Cu-Ar	WD1918+38	133.6
DD-06A	DD-20A+B	Gemini North	Long Slit	[20+25/07/2009]	3x2.0 ks	2	B600	4.5	Cu-Ar	WD1918+38	148.3
DD-06B	DD-20A+B	Gemini North	Long Slit	[20+25/07/2009]	3x2.0 ks	2	B600	4.5	Cu-Ar	WD1918+38	90.2
DD-07A	WDB02	VLT-ANTU	LSS	[02/07/2010]	2x0.6 ks	1.3	600B	3.5	He-HgCd	LHS-2333	89.5
DD-07B	WDB02	VLT-ANTU	LSS	[02/07/2010]	2x0.6 ks	1.3	600B	3.5	He-HgCd	LHS-2333	26.9
DD-08A	WDB04	VLT-ANTU	LSS	[02/07/2010]	1x0.5+1x0.36 ks	1.3	600B	3.5	He-HgCd	LHS-2333	32.1
DD-08B	WDB04	VLT-ANTU	LSS	[02/07/2010]	1x0.5+1x0.36 ks	1.3	600B	3.5	He-HgCd	LHS-2333	45.7
DD-09A	WDB10	VLT-ANTU	LSS	[02/06/2010]	2x0.6 ks	1.3	600B	3.5	He-HgCd	LHS-2333	23.4
DD-09B	WDB10	VLT-ANTU	LSS	[02/06/2010]	2x0.6 ks	1.3	600B	3.5	He-HgCd	LHS-2333	33.1
DD-10A	WDB11	VLT-ANTU	LSS	[02/07/2010]	2x0.3 ks	1.3	600B	3.5	He-HgCd	LHS-2333	43.2
DD-10B	WDB11	VLT-ANTU	LSS	[02/07/2010]	2x0.3 ks	1.3	600B	3.5	He-HgCd	LHS-2333	57.2
DD-11A	WDB12	VLT-ANTU	LSS	[02/07/2010]	2x0.3 ks	1.3	600B	3.5	He-HgCd	LHS-2333	81.2
DD-11B	WDB12	VLT-ANTU	LSS	[02/07/2010]	2x0.3 ks	1.3	600B	3.5	He-HgCd	LHS-2333	55.8
DD-12A	DD-07A+B	Gemini South	Long Slit	[11/05/2010]	3x1.8 ks	2	B600	4.5	Cu-Ar	WD0000+345	99.7
DD-12B	DD-07A+B	Gemini South	Long Slit	[11/05/2010]	3x1.8 ks	2	B600	4.5	Cu-Ar	WD0000+345	94.7

2.3 Data Reduction

2.3.1 Spectroscopic Data Reduction

Spectroscopic data was reduced using `IRAF` to follow a common methodology. Scripts were used to drive this process in each case. The specifics of all reduction scripts and external software are outlined in the Appendix Section A.4. However, in brief, the following reduction steps were applied to the data:

1. Bias Subtraction, to account for the ramp level added to the CCD (to prevent negative levels being fed to the analogue-to-digital converter). This was achieved by subtracting a stack of zero exposure time frames from the data.

2. Flat Fielding, to account for CCD pixel to pixel sensitivity variations. This was achieved by dividing the data by a normalised version of the image produced when illuminating the CCD with a quartz continuum lamp. CCD frames were debiased and flat-fielded using the `IRAF` procedure `CCDPROC`.

3. Cosmic ray removal using the LAplacian-COSmic ray remover, *lacos_spec.cl* created by Pieter van Dokkum, April 2001 (van Dokkum 2001).

4. 1D spectra were extracted using the `APEXTRACT IRAF` procedure.

5. Wavelength calibration using arc lamp exposures. The arc lamps used depended upon the particular spectrograph. On Gemini N+S Cu-Ar lamps were used, on the WHT Cu-Ne + Cu-Ar lamps were used, and on the VLT He-HgCd lamps were used.

6. A first order flux calibration was performed independently using a DC standard star of known effective temperature (see Appendix Section A.4.6).

Spectroscopic Flux Calibration Method

The flux calibration of my spectra was performed using observations of DC WDs. Three He atmosphere (no Balmer lines) DCs of known effective temperature were used for this purpose. For the northern hemisphere observations, the WHT and Gemini North, I adopted WD1918+38. For the southern hemisphere observations, I used WD0000-345 and WD LHS-2333 to calibrate the data from Gemini South and the VLT, respectively. The effective temperatures of WD0000-345 ($6240 \pm 140K$) and LHS-2333/WD1055-072 ($7420 \pm 200K$) were obtained from Bergeron et al. (2001) and that of WD1918+38 ($6470K$) was taken from Bergeron et al. (1997). Note that I specifically chose stars in the middle and towards the hotter end of the DC range because across the wavelength range of interest (between 3700Å and 5200Å) they have more even flux levels, allowing high signal-to-noise data to be acquired right across this regime in relatively short integration times.

The observed spectrum of each DC was fitted with a cubic spline in `IRAF` *splot*. This allowed me to efficiently eliminate the Calcium H and K lines (3968.5Å and 3933.7Å

TABLE 2.7: Standard WD Stars

WD	RA (*J2000*)	Dec (*J2000*)	
WD1918	19 18 58.0	38 43 35.0	$6470K$
WD0000-345	00 02 40.08	-34 13 38.7	$6240 \pm 140K$
WD1055-072 (LHS-2333)	10 57 35	-07 31.4	$7420 \pm 200K$

respectively) and any poor data channels, in the case of the failure in cosmic ray removal or bad CCD columns. Calcium H+K absorption is often seen in the spectra of He atmosphere WDs since the time scales for metals to diffuse below their photospheres are much larger than for DAs of similar effective temperature. Subsequently, each science spectrum was divided by this interpolating spline before being multiplied by a black body function with the same effective temperature as the DC WD. I performed this step using a custom written Perl script, where the details of this are provided in Appendix Section A.4.6. Figures 2.15, 2.16, 2.17, and 2.18 display my normalised flux calibrated spectra. Measurements of signal-to-noise were made on each of these spectra, where the code used to perform these calculations is outlined in the Appendix Section A.4.7

It should be made clear that the use of DC standards and blackbody spectral distributions yielded relative fluxes, not absolute fluxes. The flux calibration performed here was designed to compensate for variations in flux across wavelength due to imperfect calibration of the instruments (gradients in throughput).

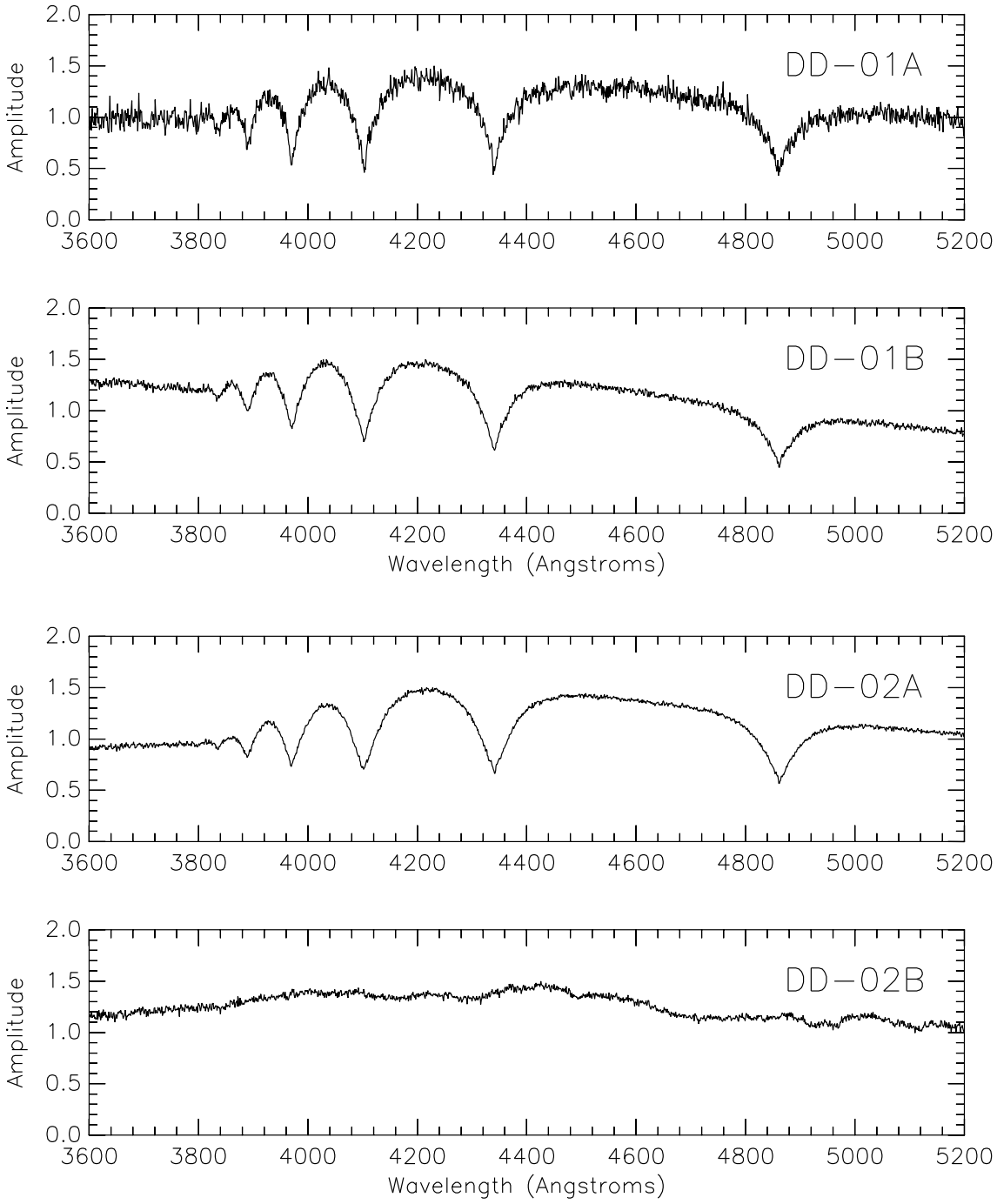


Figure 2.15: Flux Calibrated Spectrum - DD01 to DD02

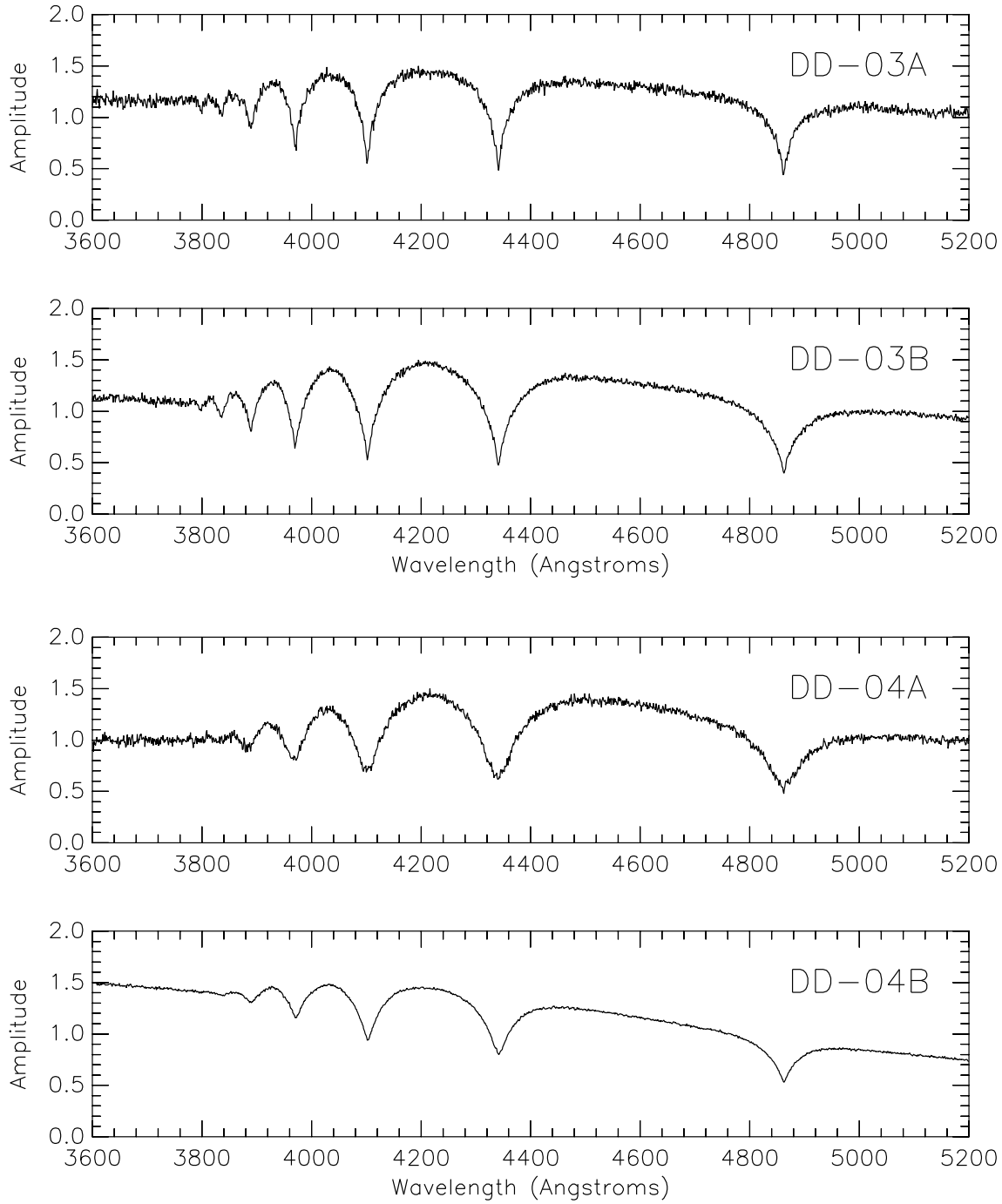


FIGURE 2.16: Flux Calibrated Spectrum - DD03 to DD04

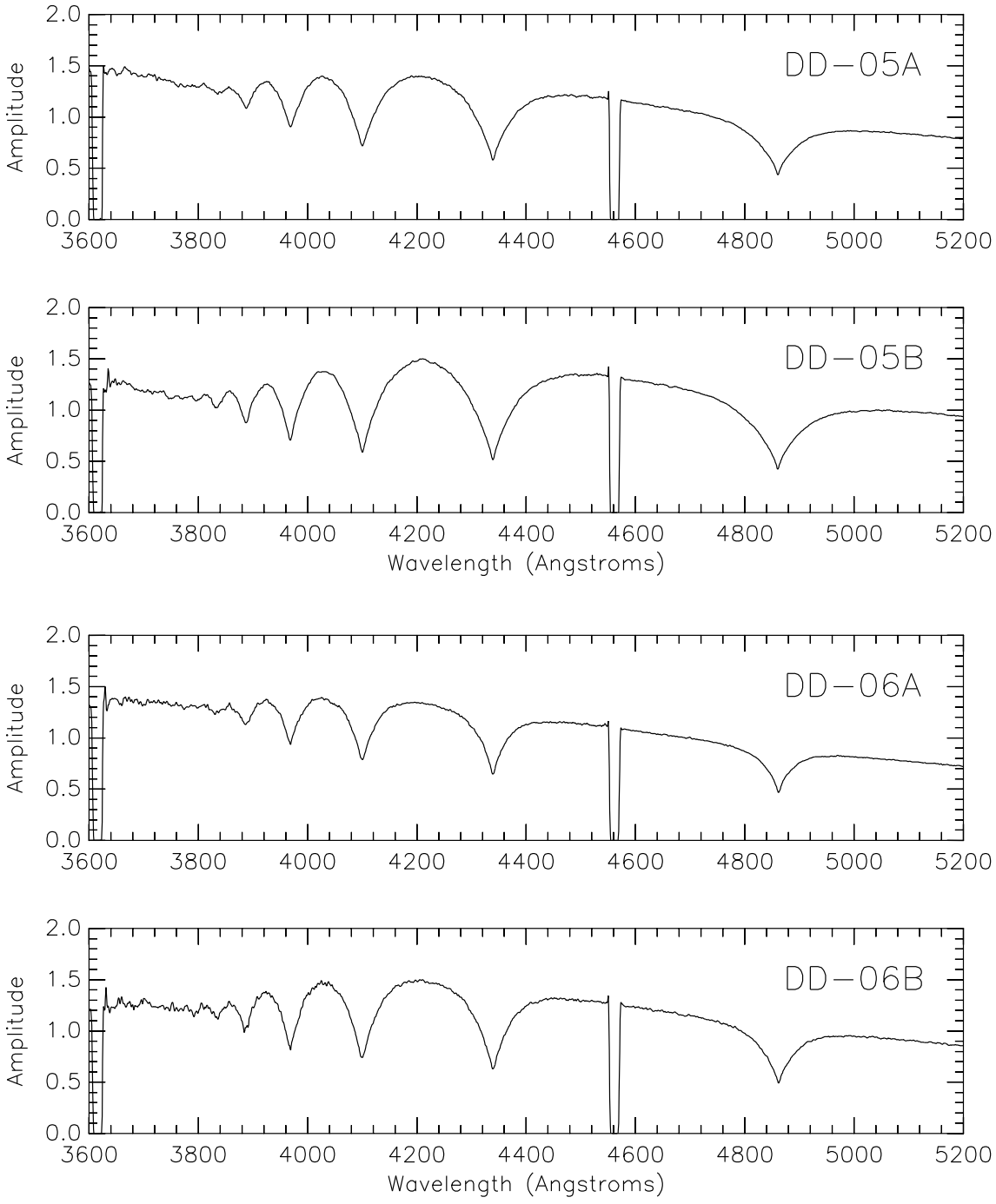


Figure 2.17: Flux Calibrated Spectrum - DD05 to DD06

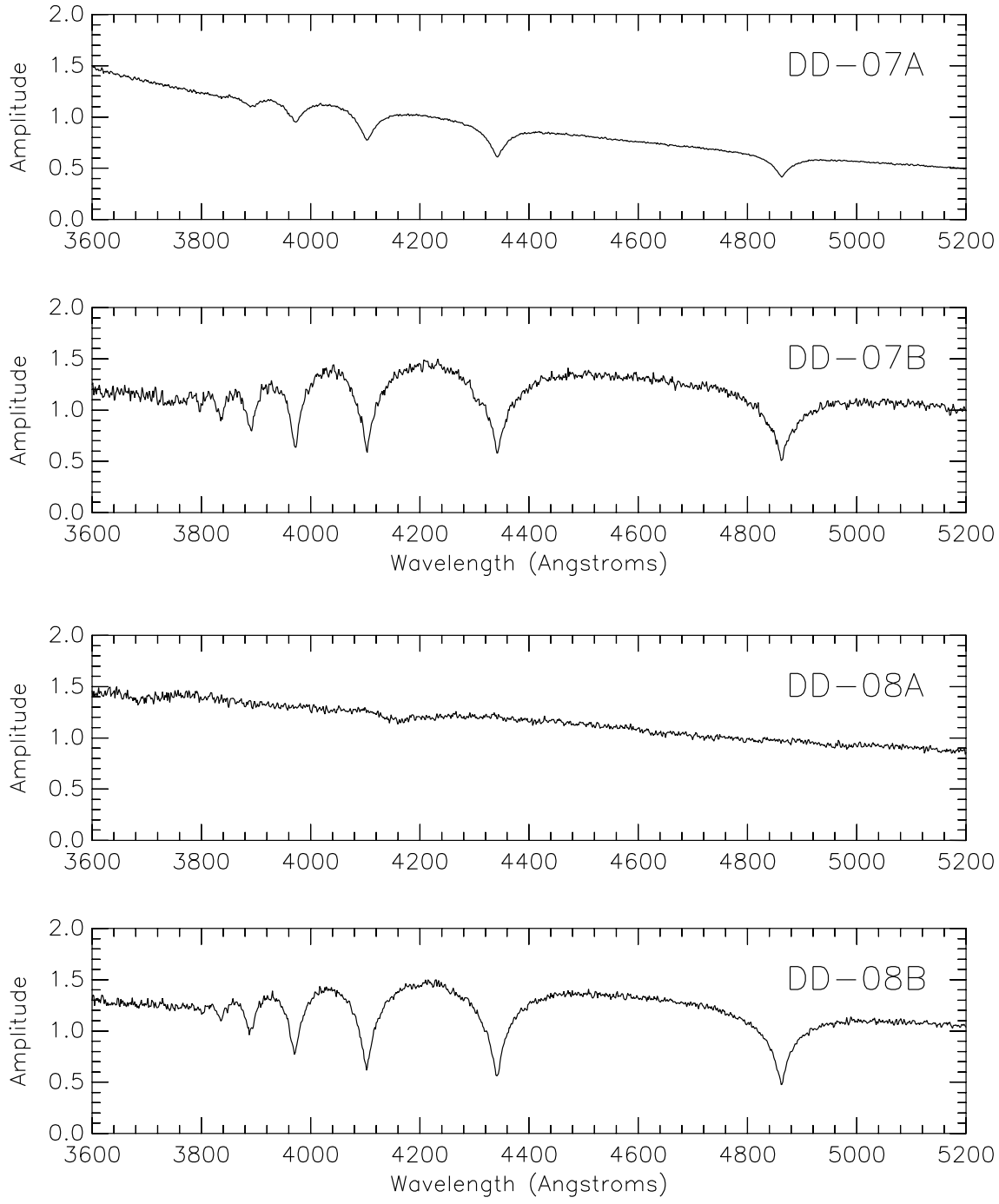


FIGURE 2.18: Flux Calibrated Spectrum - DD07 to DD08

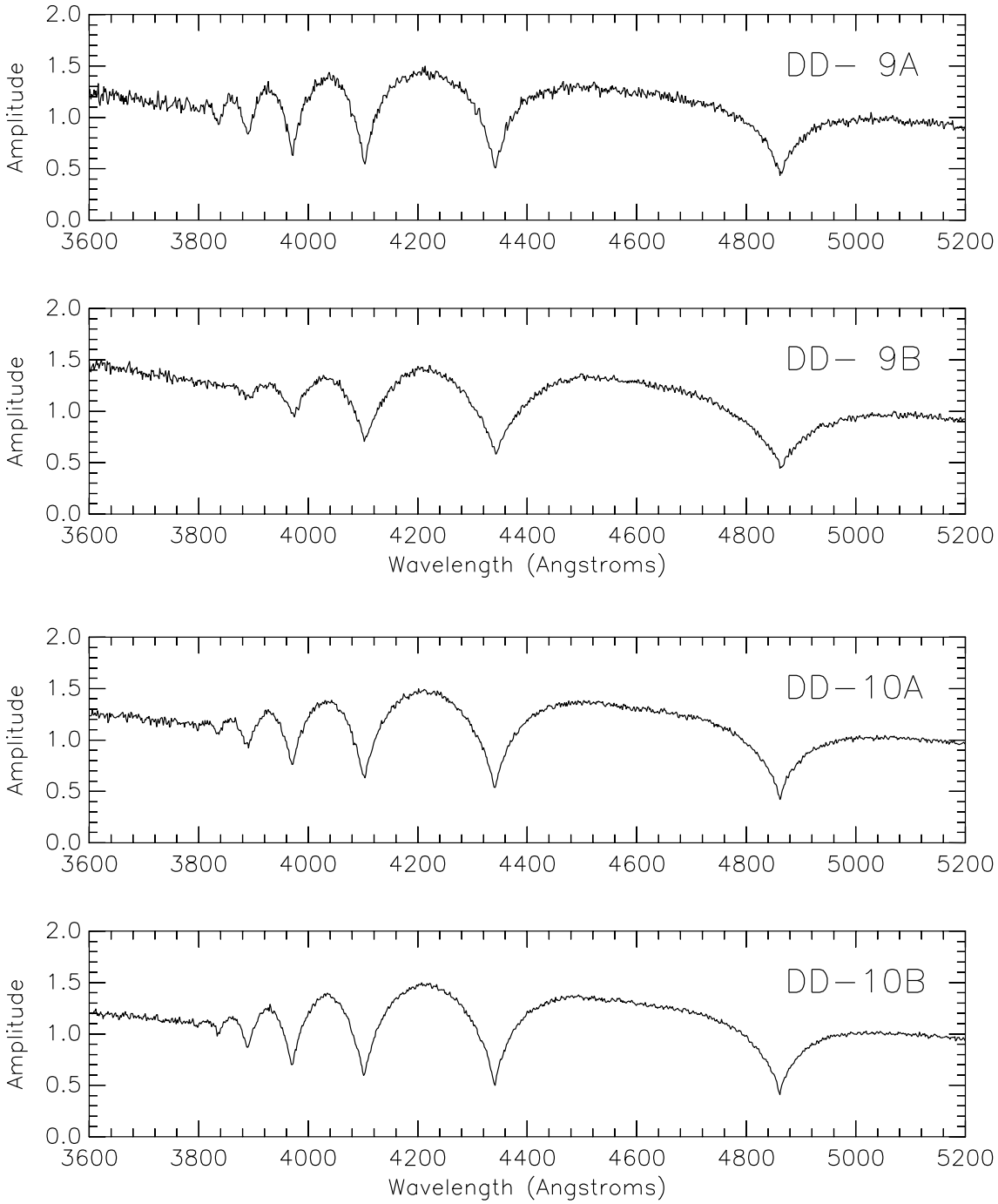


Figure 2.19: Flux Calibrated Spectrum - DD09 to DD10

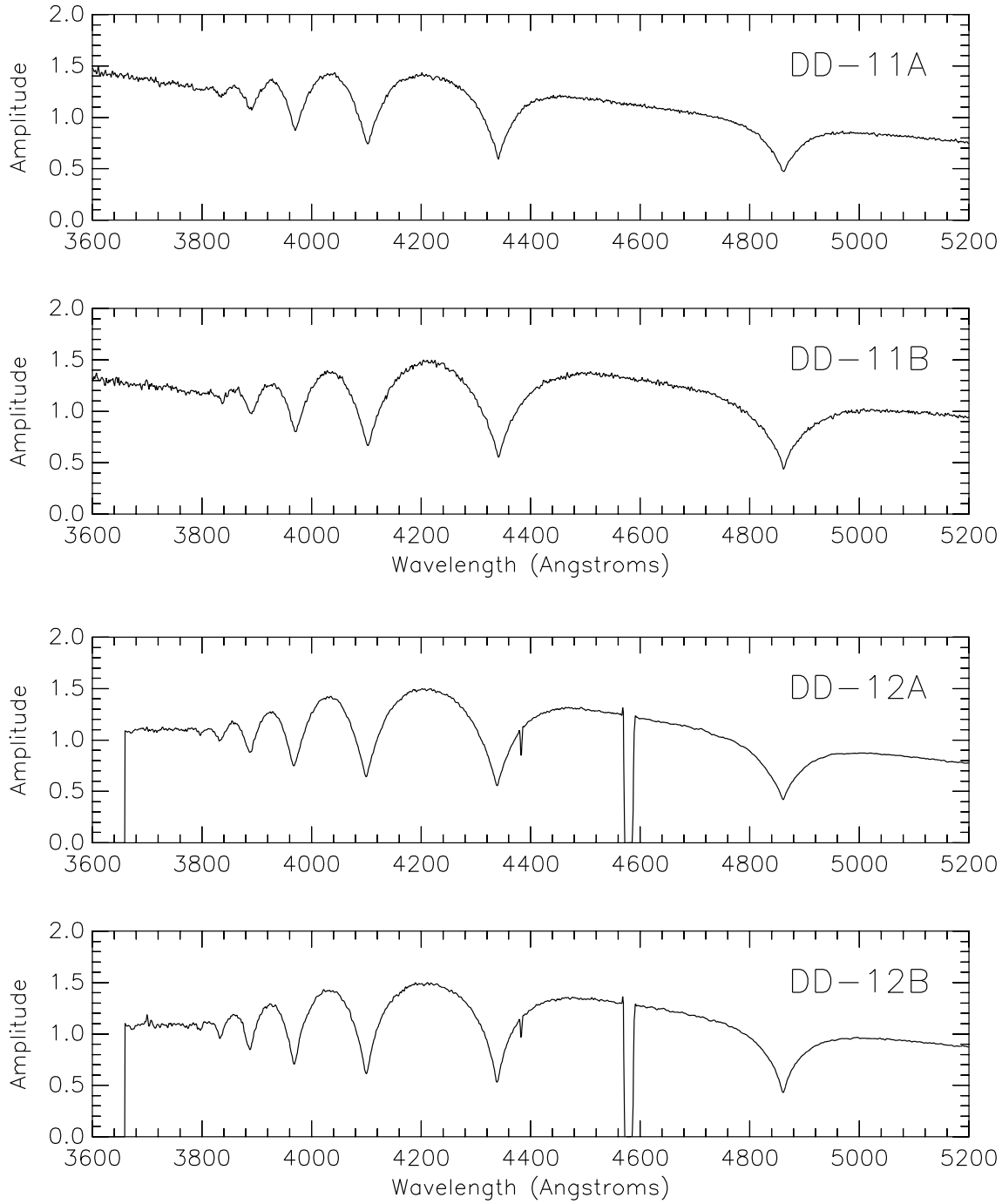


FIGURE 2.20: Flux Calibrated Spectrum - DD11 to DD12

2.4 Data modelling

The effective temperature and the surface gravity values for each WD were obtained using grids of pure-H synthetic spectra. These measurements are essential for determining the mass, the radius and the cooling age of the WD from evolutionary tracks. In the next sections I provide a brief guide to these models.

2.4.1 Atmospheric modelling

Atmospheric Models of DA white dwarfs

Atmospheric models allow the emergent spectrum of a WD star to be predicted as a function of effective temperature ($T_{\rm eff}$), surface gravity ($\log g$), and elemental composition. Codes for the construction of these models are publicly available (Hubeny and Lanz 1995, Werner 1986), as are pre-calculated tables containing solutions at select $T_{\rm eff}$, $\log g$, and elemental composition values. All WDs with $T_{\rm eff}$ >20000K modelled in this thesis utilised `TLUSTY` v200 (Hubeny 1988, Hubeny and Lanz 1995), where the spectral synthesis code `SYNSPEC` v48 (Hubeny and Lanz 2001) was used to predict the emergent spectra from these model atmosphere structures. The grids of models based on `TLUSTY` and `SYNSPEC` used in this thesis were generated by Paul Dobbie. For WDs with $T_{\rm eff}$ ≤20000K, I used a grid of synthetic spectra supplied by Detlev Koester. The version of the `TLUSTY` code in hand is not optimised for objects with ($T_{\rm eff} \lesssim 12000 - 14000K$) since it was originally written to deal with purely radiative atmospheres.

A variety of simplifying assumptions are adopted in the computation of model atmospheres and synthetic spectra (Auer and Mihalas 1969, Hubeny 1988). The stellar atmosphere is generally represented as a series of plane parallel layers, where the observable portion, the photosphere, represents only a fraction of the radius of the WD, for example $\frac{R_{atm}}{R} \approx 0.0011$ (Kubat 1995). Moreover, the photosphere is taken to be horizontally homogeneous (or uniform in x and y) with variations only in depth, z, reducing the problem to solving the physical and structural equations in a single dimension.

As these objects are generally stable over long periods of time, it is assumed that an equilibrium exists between the gas+radiation pressure and the weight of the atmosphere due to the star's own gravity. The surface gravity is assumed to be constant throughout the layers of the atmosphere, where the pressure derivative, dp, is taken with respect to mass column (m, Equation 2.3 with units gcm^{-2}), and not the depth value, z, since the gas pressure is close to linear in m (Mihalas 1978). Both stellar characteristics suggest all energy flux radiated from the atmosphere must come from the interior of the star, with the amount entering any given layer equaling the amount exiting that layer, and delineates constancy of flux F with optical depth, τ, or a state of radiative equilibrium, Equation 2.4

$$\frac{dp}{dm} = g \tag{2.3}$$

$$\frac{dF}{d\tau} = 0 \tag{2.4}$$

The radiative energy from the boundary layer between the non-observable atmosphere and observable atmosphere interacts with ions and electrons, and is absorbed, emitted, and scattered frequently, as it approaches the surface of the WD star. The interaction between the radiation field and the constituent material of the atmosphere is expressed in the equation of radiative transfer (Equation 2.5), where I_ν is the intensity of the radiation field at frequency ν (or the amount of energy, dE, passing across a surface of area dS, in a direction $\mu = \cos\theta$ with respect to the surface normal, into a solid angle $d\omega$, in the frequency interval $d\nu$, per unit time at the frequency ν), η is the emission coefficient, and χ is the absorption co-efficient.

$$\mu\frac{dI_\nu}{dz} = -\chi(z,\nu)I(z,\mu,\nu) + \eta(z,\nu) \tag{2.5}$$

The absorption and emission coefficients represent the efficiency with which radiation at frequency ν can pass through the material of the stellar atmosphere, and both depend heavily on the election occupation numbers of the atomic energy levels of the material. The opacity to radiation and emission of radiation is described as arising from a combination of the continuum and spectral lines for the purposes of the models. The continuum opacity to radiation is produced by a combination of a) bound-free absorption where an electron, bound to a nucleus, is removed by a photon with energy greater than the ionization energy of the ion, b) free-free absorption where an electron absorbs radiation, while the electron briefly interacts with the electric field of an ion, and c) electron scattering, where radiation interacts with an unbound electron, and is scattered - or isotropically redistributed (Hubeny 1988, Mihalas 1978).

The observed radiation arises from a region at an optical depth $\tau \approx \frac{2}{3}$, where τ is defined by $d\tau = \chi dz$, running from the top of the photosphere ($\tau = 0$) towards the deepest layers ($\tau \to \infty$). Bound-free absorption from `HI` is an important source of opacity in a DA WD atmosphere. The cross sections for `HI` are calculated using the "hydrogenic formula" (Mihalas 1978). As these are proportional to ν^{-3}, with increasing frequency (ie. towards the blue end of the spectrum) light emerges from deeper (and therefore hotter) layers of the stellar atmosphere, resulting in a harder spectral shape of a DA WD as compared with a black body of the same effective temperature (Shipman 1976).

The processes of absorption and emission have a large effect on the temperature and structure of the atmosphere. In bound-free absorption, any energy excess of ionisation energy (versus the energy of the instigator photon) contributes to the kinetic energy of the electron, which, through elastic collisions, is redistributed amongst neighbouring particles, allowing this energy to be transferred to the thermal pool of semi-opaque gas. Line opacity leads to a warming of the deeper atmospheric layers compared to the upper atmospheric layers where the flux is screened from reaching (Mihalas 1978). However if flux is suppressed in one spectral region, it must be emitted in another.

In a Local Thermal Equilibrium (LTE) model calculation, which is generally a good approximation for the high density atmospheres of WDs, both the electron populations

of the levels of an ion, and the relative ionisation fractions are described by the Boltzmann and Saha distribution functions respectively. The temperature of particles is the same as the temperature of the approximately isotropic radiation field. However, departures from LTE creep in when the atmospheres are extremely hot ($T_{\rm eff} > 40000K$) or are of low surface gravity ($logg < 7.0$) and therefore are of lower density. When non-LTE effects become important, radiative rates begin to dominate over collisional rates and the radiation field departs strongly from an near isotropic black body. All WDs studied here have $T_{\rm eff} < 40000K$ and $logg > 7.5$ so I have considered only LTE models in this work.

Model Grid Generation

A grid of hot ($T_{\rm eff}$ >20000K) pure-H LTE model spectra was generated using `TLUSTY` v200 (Hubeny 1988, Hubeny and Lanz 1995) and `SYNSPEC` v48 (Hubeny and Lanz 2001). The calculations adopt a model H-atom which incorporates explicitly the eight lowest energy levels and represents levels $n = 9$ to 80 by a single superlevel. The dissolution of the high-lying levels was taken into account with the occupation formalism of (Hummer and Mihalas 1988). All calculations were carried out under the assumption of radiative equilibrium, including bound-free and free-free opacities of the H^- ion and incorporated a full treatment for the blanketing effects of HI lines and the Lyman $-\alpha$, $-\beta$, and $-\gamma$ satellite opacities as computed by N. Allard (Allard et al. 2004). During the calculation of the model structure, the lines of the Lyman and Balmer series were treated using an approximate Stark profile, but in the spectral synthesis step detailed profiles for the Balmer lines were calculated from the Stark broadening tables of Lemke (1997).

The `TLUSTY` grid extends from $18000K \leq T_{\rm eff} \leq 47500K$ and $7.25 \leq \log g \leq 8.75$. A model grid extending to lower effective temperatures ($6000K \leq T_{\rm eff} \leq 20000K$) was provided to us by Detlev Koester and covered surface gravities of $7.00 \leq \log g \leq 9.50$. I verified that the synthetic spectra from the two sources were comparable, at least in the overlap region $18000K \leq T_{\rm eff} \leq 20000K$ (See Appendix Section A.5.3). The Koester models were preferred upto $T_{\rm eff} \approx 20000K$ since they included a more physically realistic treatment of the high order Balmer lines (Tremblay et al. 2008).

Fitting the data to the model grid

The spectral fitting program `XSPEC` (version 11.3) (provided personally by Bryan Irby at NASA) was used to perform the comparison between my data and the synthetic spectra. This version was chosen as it most closely matched the version of `XSPEC` used in previous work (11.2), and could be compiled to operate on a 64-bit (x86_64) Linux system. Before utilising `XSPEC`, the individual lines in the model grids were cut-out and converted to fits format as were the lines in the observed data. This was accomplished using the Perl scripts outlined in the Appendix Section A.5.9.

`XSPEC` operates by folding the model spectrum through a predefined instrument response function (to account for the spectral resolution of the data). The result is then compared to the data using a χ^2 statistic, where $\chi^2 = \sum_{i=1}^{n} \{\frac{1}{\sigma_i^2}[y_i - y(x_i)]^2\}$, σ_i^2 is the uncertainty in the flux of each data point (e.g. 1σ), y_i is the observed flux, $y(x_i)$

is the model predicted flux for the wavelength x_i and n is the total number of spectral data points.

The best fit model representation of the data is found by calculating the χ^2 statistic for different models, and stepping in small increments through the model grid, linearly interpolating between calculated grid points. Care must be taken by the user to ensure a true global minimum is found, and not simply a local minima.

After finding the global minimum in χ^2, confidence limits are placed on the free parameters ($\log g$, $T_{\rm eff}$, line scaling, and wavelength shifts). This is performed by applying another iterative procedure, whereby the parameter in question is stepped further and futher away from its optimum value and each time finding a new minimum χ^2, until the difference between this and the original "best fit" χ^2 value, $\delta\chi^2$, corresponds to the 1σ (68%) confidence level (for the particular number of free parameters). All errors (limits) derived in this way are formal 1σ (68%) confidence bounds only and do not factor in systematic issues with the models or the data.

Simultaneous fitting of the hydrogen Balmer lines has been performed as in Bergeron et al. (1992). The H-Balmer lines, β, γ, δ, ϵ and 8, were cut out of the optical spectrum of each WD, where limits are chosen such that the majority of the feature is included, but regions of severe overlap are excluded (in particular for the higher order balmer lines). The synthetic model profiles were then simultaneously fitted to the lines, allowing for variation of $T_{\rm eff}$, $\log g$, and wavelength shift, and utilising independent model normalisations for each spectral region.

In this project I use $\log g$ error of 0.07 and $T_{\rm eff}$ error of 2.3% to compensate for uncertainties in the line profiles used in the synthetic spectral calculations (which are much greater than the model fit errors calculated by `XSPEC`). Napiwotzki et al. (1999) found that for $T_{\rm eff} < 30000K$, the mean scatter in temperature and surface gravity for their sample was $\sigma(T_{\rm eff}) = 2.3\%$ and $\sigma(\log g) = 0.07$ dex respectively. See also Section 1.4.4.

The comparatively weaker (yet formally still high field) DAH WDs that were unable to be modelled externally using magnetic models (DD02A and DD4A), were remodelled in `XSPEC` with my non-magnetic model grid, ignoring their line centres to prevent an overestimate of their masses. This was required as for these magnetic WDs, the peaks of their balmer line decrements are contaminated as product of their magnetic fields. The modelling accuracy using this method of ignoring the centres of the balmer lines was verified using the hot companion, DD-04B. The values of its output parameters T_{eff} and $\log g$ when modelled using the same method (ignoring the balmer line centres) were calculated to be within the error limits of their original derivation (including balmer line centres), ie $T_{eff} = 27545.2 \pm 633.5$, $\log g = 8.38 \pm 0.07$ (ignoring centres) versus $T_{eff} = 28066.6 \pm 645.5$, $\log g = 8.45 \pm 0.07$ (original).

Figures 2.22, 2.23, 2.24, and 2.25 display "Christmas Tree" graphs of my spectra with models overlayed. Note DD-02B and DD-08A are missing from all plots as they have been modeled externally using magnetic models.

Magnetic WD Atmospheric Modelling

Although several of my WD components appeared to be high field magnetic white dwarfs (HFMWDs; DD-02A, DD-02B, DD-04A, DD-08A - see Results Chapter), not all of these systems were actually modelled using synthetic spectra for magnetic atmospheres. Those systems which have been compared to magnetic models to date are the objects DD-02B and DD-08A which have $B > 50$MG. These models were calculated with a radiative transfer code for magnetised, high gravity atmospheres (see Jordan 1992; 1993) and this part of the analysis was performed by Baybars Kulebi (e.g Klebi et al. 2009).

Fine flux calibration (or "flux re-calibration") was performed before magnetic models were fit to DD-02B and DD-08A, so as to improve the accuracy of the fitting. This was achieved by dividing the spectra of their alternate components (DD-02A and DD-08B) with their best fit (non-magnetic) spectroscopic models, and fitting a curve (polynomial function) to the result. The flux calibrated spectra of DD-02B and DD-08A was then multiplied by this fitted curve. The software used to perform this flux re-calibration is outlined in the Appendix Section A.5.6.

2.4.2 Evolutionary modelling

Once $logg$ and $T_{\rm eff}$ values had been determined for a WD using the atmospheric models, I used evolutionary models from the Montreal Group (Fontaine et al. 2001, Holberg and Bergeron 2006) to determine the cooling age, the mass and the absolute magnitude values. These models are available from http://www.astro.umontreal.ca/ bergeron/-CoolingModels where a description of them is also provided. I show examples of the theoretical cooling sequences of two WDs of different masses in Figure 2.21. The code used to perform these calculations is referenced in Appendix Section A.5.2.

The evolutionary sequence fitting requires apparent magnitude data for the components, and this is taken from the SDSS. It should be noted here that interstellar reddening is assumed to be negligible. This is a reasonable assumption since these are in general high galactic latitude systems within ∼200pc. In any case, these objects share very similar lines of sight so reddening levels towards the components of a given system are likely to be comparable.

For temperatures $< 30000K$ the theoretical mass-radius relations are based upon the CO (carbon-oxygen) core, thick H-layer evolutionary models of Fontaine et al. (2001) but with $q_H = 10^{-4}$ of Bergeron et al. (2001).

Colour calculations are described in Holberg and Bergeron (2006), which is an extension of the work by Bergeron et al. (1995b).

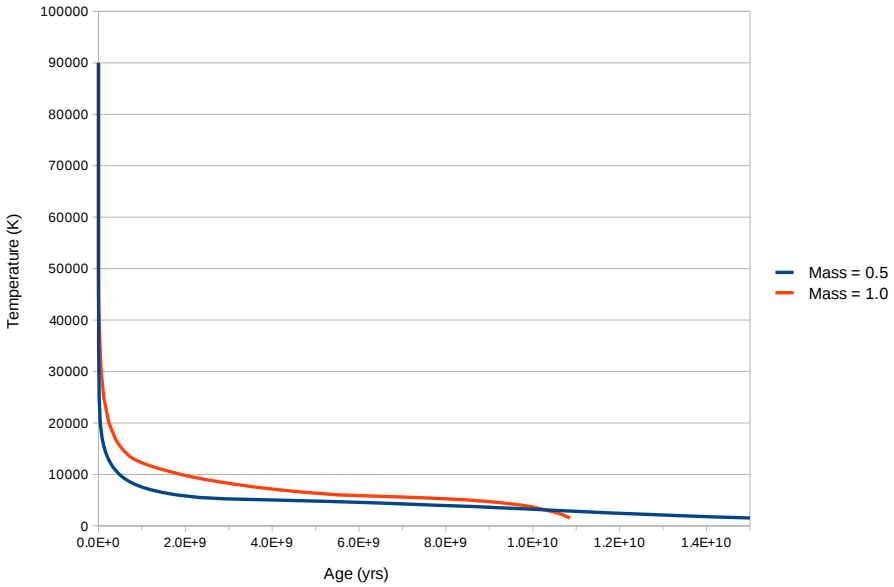


FIGURE 2.21: Bergeron Theoretical WD Cooling Tracts - Temperature versus Age for a given mass. The cooling tracts for both a low mass ($0.5M_{\odot}$) and a high mass ($1.0M_{\odot}$ WD are shown.

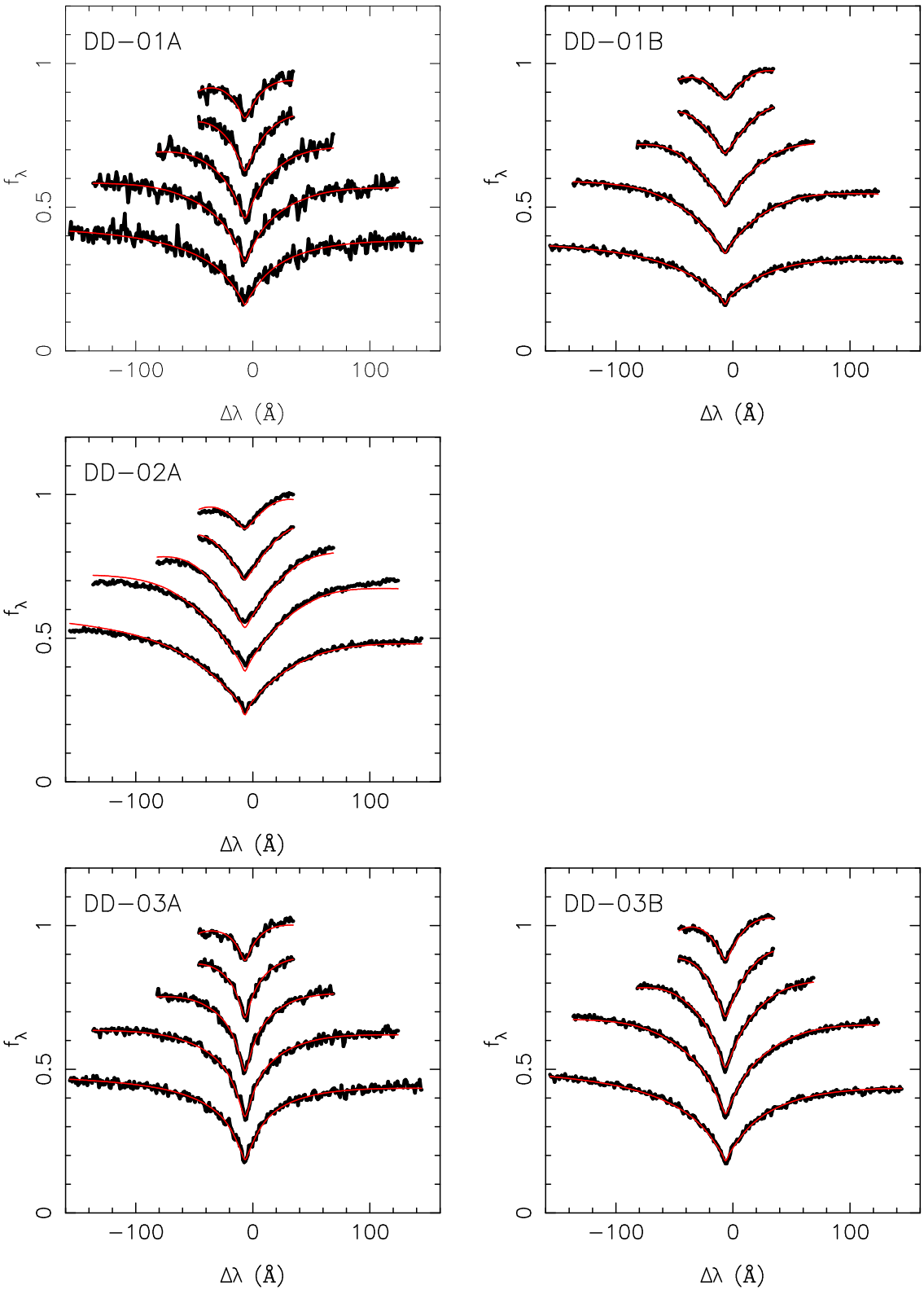


Figure 2.22: "Christmas Tree" graphs of my spectra with models overlayed (DD-01 to DD-03). Note DD-02B and DD-08A are missing (as they have been modeled externally using magnetic models).

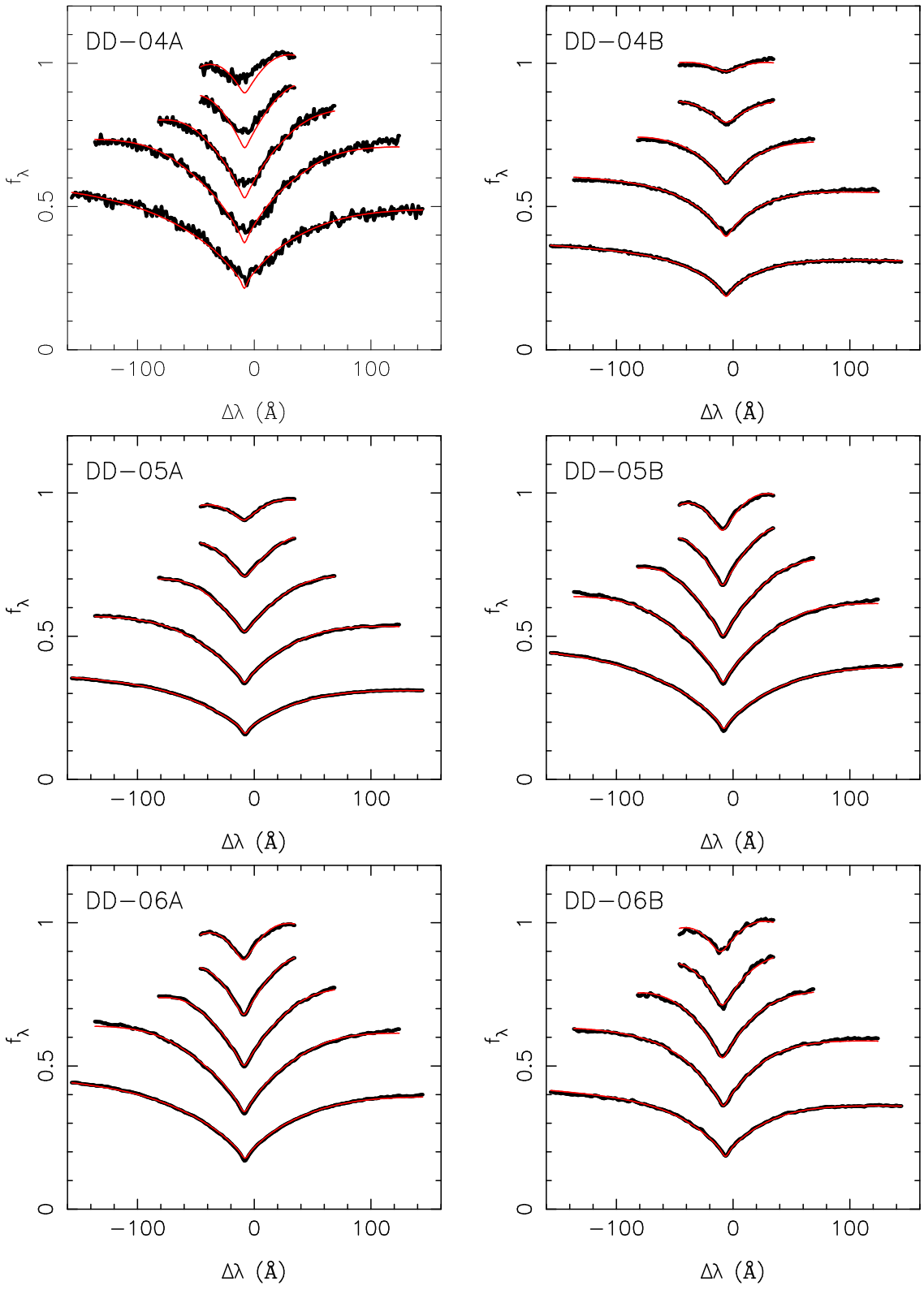


FIGURE 2.23: "Christmas Tree" graphs of my spectra with models overlayed (DD-04 to DD-06).

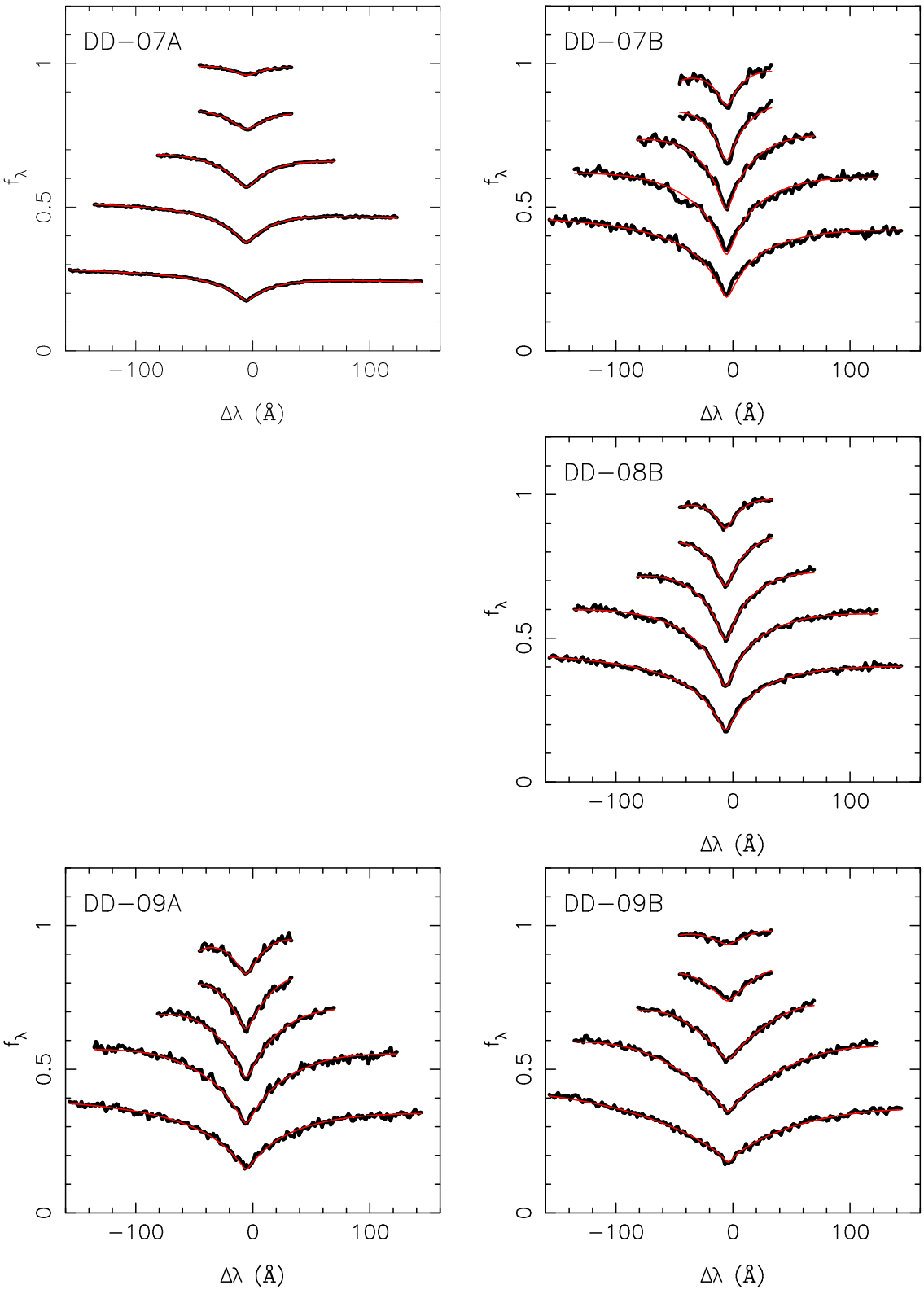


Figure 2.24: "Christmas Tree" graphs of my spectra with models overlayed (DD-07 to DD-09). Note DD-02B and DD-08A are missing (as they have been modeled externally using magnetic models).

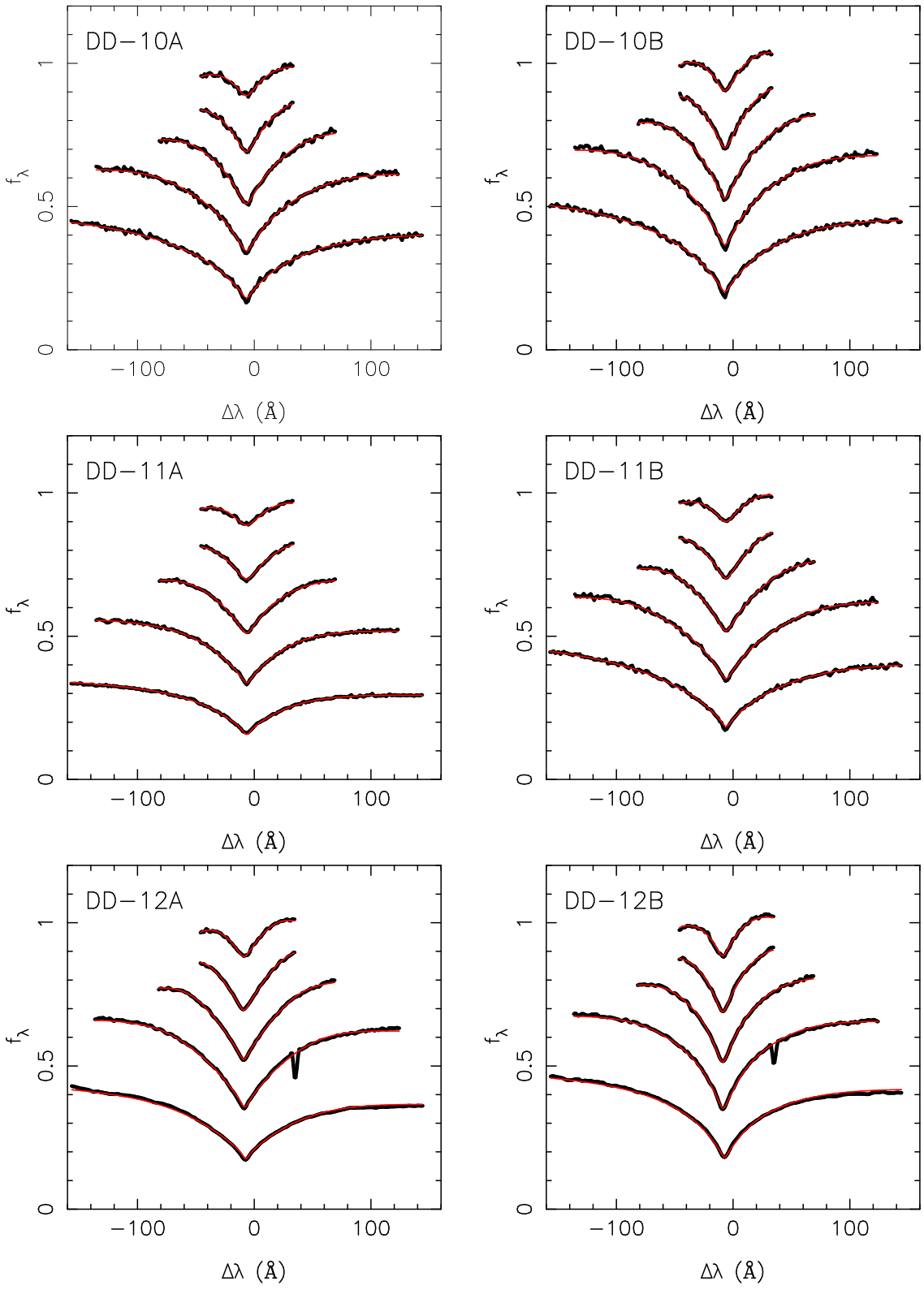


FIGURE 2.25: "Christmas Tree" graphs of my spectra with models overlayed (DD-10 to DD-12).

2.4.3 Proper motion verification of physical association

While I have estimated the probability of chance alignment for pairs of objects within 5″, 10″, 20″, and 30″ to be 5%, 10%, 25%, and 40% respectively, in my sample of 53 candidate systems this still translates to a non-negligible fraction of these being random alignments. Therefore for objects which were followed-up spectroscopically, I performed proper motion measurements to obtain a strong independent confirmation (or otherwise) on a case by case basis that system components were physically associated. These were performed using an implementation of the method outlined in Casewell et al. (2007) and Moraux et al. (2001). The full procedure is outlined in the Appendix Section A.3. It is dependent upon the software code $pmx - t.f$ but can be summarised as follows-:

For each candidate pairing I determined the X,Y positions within the frames of surrounding objects of comparable or greater brightness within 4.0′. The detection of these objects was performed in the `SExtractor` under `GAIA` and provided on average > 20 suitable reference sources. Next, I cross matched on RA and DEC the lists of reference star positions using the `STARLINK TOPCAT` software. Subsequently, I employed routines in the `STARLINK SLALIB` library to construct six co-efficient linear transforms ($x_1 = Ax_2 + By_2 + C, y_1 = Dx_2 + Ey_2 + F$) between the two images, epoch 1, POSS-I blue and epoch 2, SDSS g, of the putative systems, where $>3\sigma$ outliers were iteratively clipped from the fits (1σ was calculated by taking $1.48\times$ the median of the absolute residuals of the reference stars). The proper motions, in pixels, were then determined by taking the differences between the observed and predicted locations of candidates in the 2nd epoch imaging. These were converted into milli-arcseconds per year in right ascension and declination using the world co-ordinate systems of the 1st epoch datasets and dividing by the time baseline between the two observations.

As WDs are intrinsically faint and therefore typically nearby, it is likely that they will also lie quite far away from the average displacement ($\sim$0,$\sim$0) of the generally more distant reference stars within the field. If the WD components of any given BOCP are associated, then it is expected that they will have the same relative proper motions within the measurement uncertainties. These errors have been estimated by combining quadratically the `GAIA` object detection fit errors for both epochs (SDSS and POSS-I) and the RMS residue of the reference field star correlation as calculated by $pmx - t.f$. The proper motion error calculation procedure has been included in the Appendix Section A.3.1, along with the xy to RA-Dec calibration procedure in Section A.3.3.

Figures 2.26 and 2.27 display proper motions vector point diagrams for the components of my candidate wide DD systems. Additionally, the proper motions are listed in Table 2.8.

I found the proper motion measurements within the SDSS database to be unreliable for these systems. It is suspected that the low angular separation of the components has led to unreliable determinations of their centroids in the photographic data. Indeed, those systems with the highest angular separation have SDSS proper motion measurements with the highest level of correspondence to my own proper motion measurements, i.e. DD-03 and DD-04 at 11.16” and 9.14”, respectively (although it is

worth noting that DD-07 at 18.91" does not show a clear correspondence with its SDSS proper motion measurements).

The differences in the proper motions between the components of some systems are substantially larger than their estimated 1σ statistical errors. While it should be expected, for the case of normally distributed errors, that with a sample of twelve pairings that are genuinely physically associated, $\sim$4 will lie beyond 1σ, the proper motions of the components of two systems are in discord at >4 sigma (i.e. DD-05 and DD-08). However, as the components of these pairs each have proper motions that are measured as >50 mas yr^{-1} and lie in the same general region of the vector point diagram, I suspect that these discrepancies are not indicative of these being visual binaries but are instead due to systematic uncertainties in the `GAIA SExtractor` object centroiding calculations. `SExtractor` was designed for low density, high Galactic latitude fields and has difficulty in determining the centroids of objects in pairs with very low angular separations (ie. blended). This is particularly relevant when working with the rather low resolution POSS-I blue photographic data. This suspected systematic error in the centroid calculation due to the blending of sources could be reduced using a more sophisticated modelling of the point spread function, but this was not attempted here. Instead, I recalculated the proper motions of the components of DD-08 using the SDSS r band image and the POSS-I red plate in which these sources are fainter but slightly better resolved. In this case, the proper motion estimates are much closer to each other (to within or near the extent of their calculated uncertainties), supporting my suspicions that there are systematic errors in my first proper motion measurements for these two systems. Indeed, it is apparent from a visual inspection that for DD-05, in the POSS-I blue frame the centroid calculations result in a very large $\sim$3 pixel deviation between components (which corresponds exactly to their calculated difference in proper motion).

With astrometry for these twelve systems in hand, two methods were then employed to estimate the likelihood of their components exhibiting common proper motions by chance. In a first approach, the probability of a BOCP exhibiting the observed proper motion without being associated was calculated based upon the probability of a generated circular field containing the BOCP proper motions (including their errors) lying within a generated circular area containing both the BOCP and the astrometric reference stars. This basic statistical calculation is implemented under the assumption that the BOCP components are most likely WDs, and therefore are likely to have higher proper motions than all of the observed targets within the field, making it realistic to assume that they could lie anywhere in the field (not just within the main field distribution of astrometric reference stars) regardless of their association. The code used in this method has been referenced in the Appendix Section A.3.5.

In an alternative approach, the probability of chance common proper motion alignment was calculated by counting the number of M stars at approximately the same distance as the DD system within $\approx 5'$ which would have been flagged as having a common proper motion with the DD system - i,e., have proper motions within an artificial two dimensional range bounded by the proper motions (plus proper motion errors) of each WD component. The proper motions of $\approx$ 150 nearest Red Dwarfs ($< 3^\circ$, $> 5'$) at the same distance as the DD system were obtained using an SDSS SQL

query. The distance of the M stars were calculated using an empirical low mass star colour-magnitude relationship discussed by Fuchs et al. (2009) and shown in Equation 2.6, using photometric data from the SDSS. This method should indicate that solar reflex has not significantly influenced/biased these proper motion values. The code used to perform this method has been referenced in the Appendix Section A.3.4.

$$M_r = 4.0 + 11.86(r-i) - 10.74(r-i)^2 + 5.99(r-i)^3 - 1.20(r-i)^4 \quad (2.6)$$

In a variant on the second approach, the astrometry of spectroscopically identified WDs from the SDSS lying within 10° of the double-degenerate pair is examined to determine what proportion of these would have been flagged as having a common proper motion. The code used to perform this method is also referenced in the Appendix Section A.3.4.

The probabilities of chance proper motion alignment were then combined with the probabilities of chance spatial alignment derived from by the Monte-Carlo like simulations. These results are presented in Table 2.9. All seem to indicate a low probability of a chance spatial+astrometric alignment. Note that the probability of having a chance alignment in my sample based on combining these statistics is calculated as 0.0089 ($\sim$1%), as per Equation 2.2.

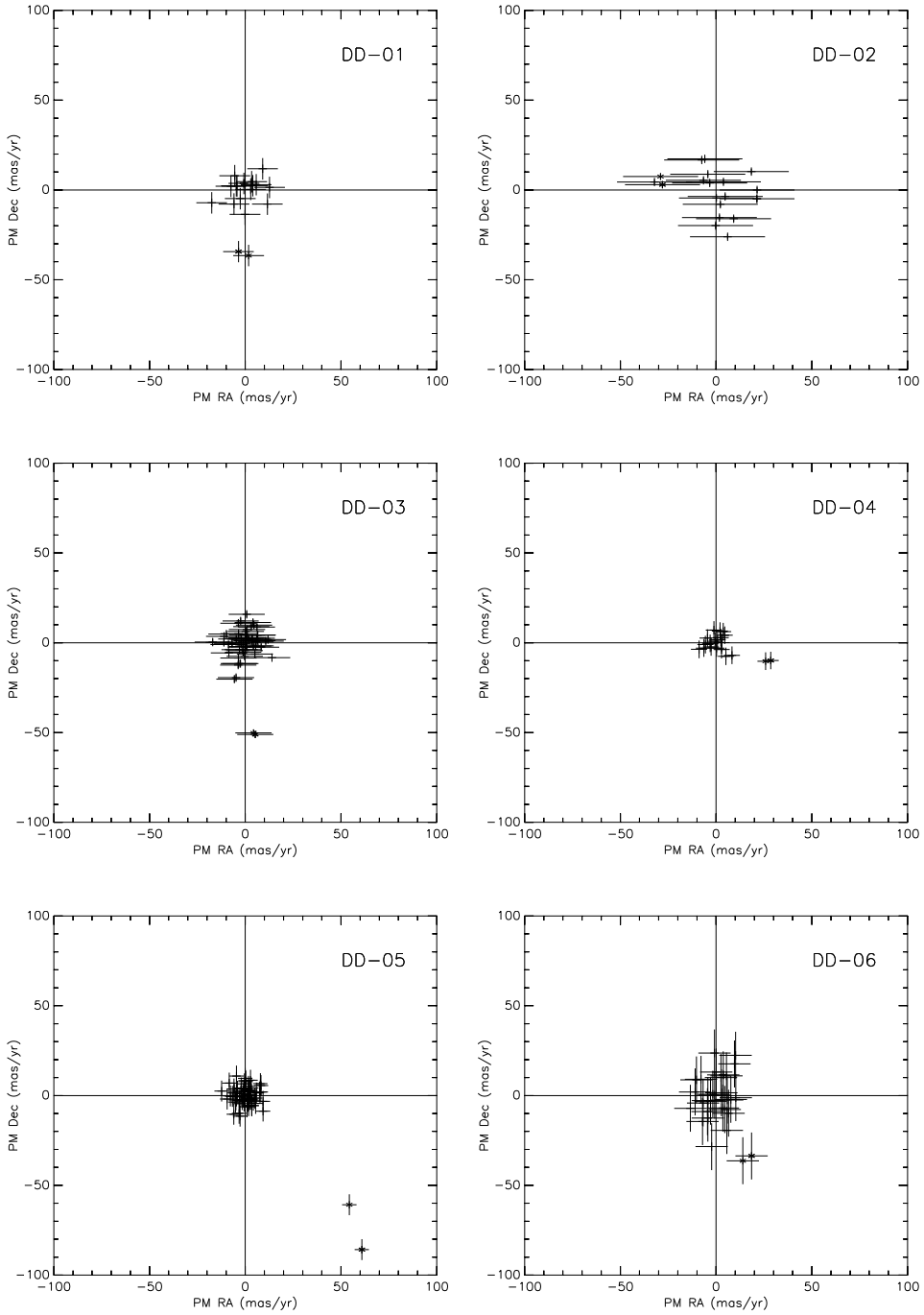


FIGURE 2.26: Proper motion diagrams for the 12 blue object close pairs for which I observed spectroscopic follow-up data (DD01 to DD06). The DD system components are marked as stars.

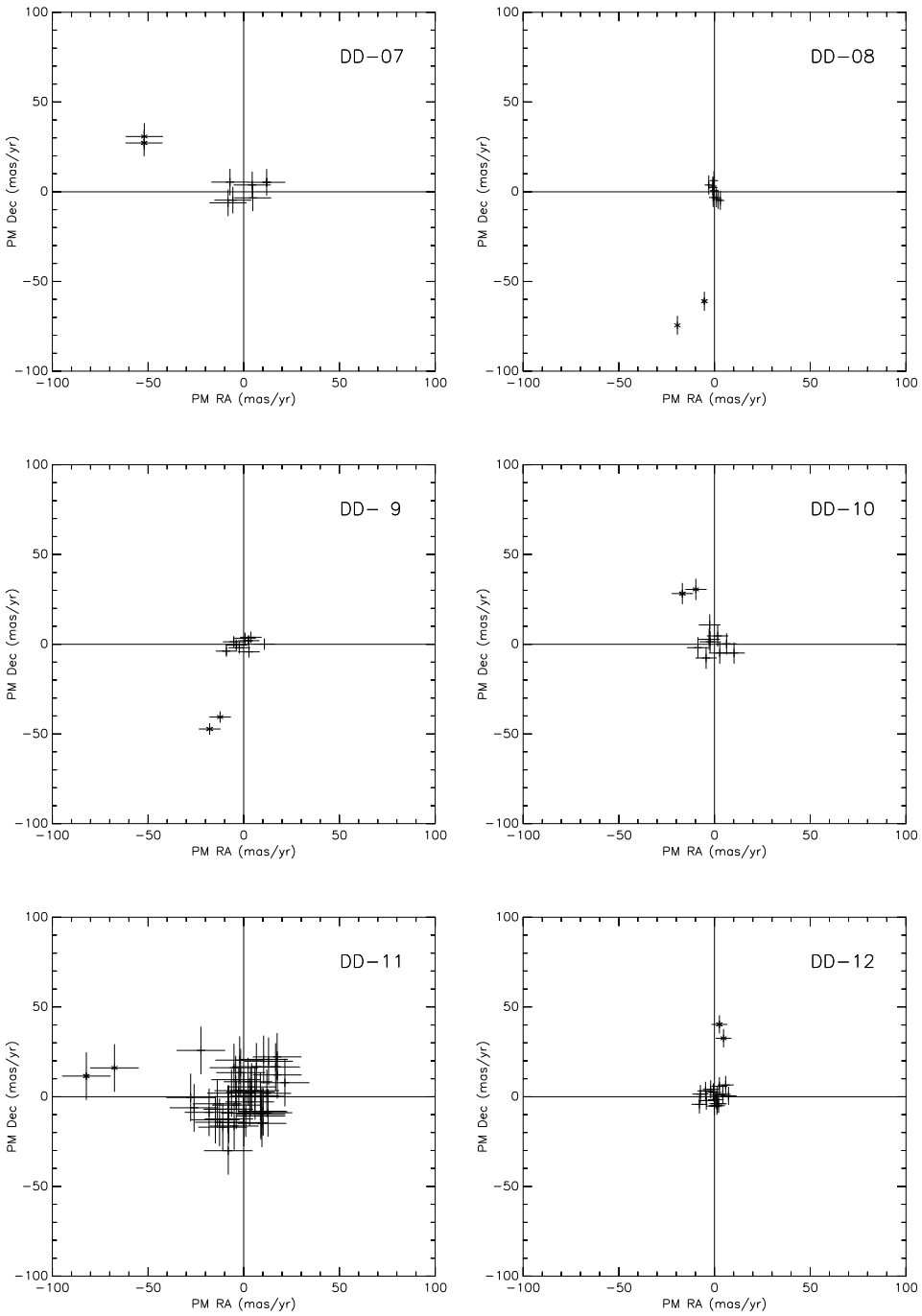


Figure 2.27: Proper motion diagrams for the 12 blue object close pairs for which I observed spectroscopic follow-up data (DD07 to DD12). The DD system components are marked as stars.

Table 2.8: Double-Degenerate Data - Proper Motions

WD	PM RA (mas/yr)	PM Dec (mas/yr)	δ PM RA (mas/yr)	δ PM Dec (mas/yr)	Avg PM RA nearby Red Dwarfs	Avg PM Dec nearby Red Dwarfs	Prob of PM Chance Align (PDD)	Prob of PM Chance Align (RBB)	SDSS PM RA (mas/yr)	SDSS PM Dec (mas/yr)	SDSS δ PM RA (mas/yr)	SDSS δ PM Dec (mas/yr)
DD-01A	1.74	-36.55	8.02	5.96	-4.38	6.86	0	0.02	18.25	-39.65	3.54	3.54
DD-01B	-3.56	-34.34	8.01	6	0	0	0	0	-2.98	-38.47	3.14	3.14
DD-02A	-28.01	2.91	19.59	1.54	-6.23	-1.64	0.02	0.01	-73.94	66.73	3.74	3.74
DD-02B	-29.01	7.51	19.59	1.61	0	0	0	0	0	0	0	0
DD-03A	4.29	-50.2	9.46	1.82	-2.76	-3.6	0	0	-1.79	-51.17	3.41	3.41
DD-03B	5.22	-51.13	9.45	1.7	0	0	0	0	0.32	-50.52	3.13	3.13
DD-04A	25.84	-10.3	4.21	4.91	2.75	-7.28	0.04	0.01	28.09	-14.32	3.26	3.26
DD-04B	28.54	-9.89	4.22	4.86	0	0	0	0	28.57	-16.32	2.75	2.75
DD-05A	60.85	-85.85	3.76	5.82	0.88	-1.71	0	0.07	0	0	0	0
DD-05B	54.35	-60.86	3.75	5.82	0	0	0	0	23.9	-5.01	2.44	2.44
DD-06A	18.56	-33.66	8.44	13.08	4.25	-4.62	0.03	0.02	0	0	0	0
DD-06B	13.92	-36.33	9.04	13.5	0	0	0	0	0	0	0	0
DD-07A	-51.97	30.75	9.7	7.42	6.2	-5.57	0	0	-30.91	-23.31	4.48	4.48
DD-07B	-52.09	27.16	9.68	7.4	0	0	0	0	-24.79	-23.63	5.39	5.39
DD-08A	-19.41	-74.43	0.7	5.29	-5.02	-4.53	0	0.06	-50.89	-142.99	3.88	3.88
DD-08B	-5.27	-61.05	0.88	5.28	0	0	0	0	11.46	-5.05	2.68	2.68
DD-09A	-17.81	-47.27	5.65	3.17	-8.88	-7.47	0	0.02	-22.42	-26.25	2.89	2.89
DD-09B	-12.34	-40.57	5.68	3.2	0	0	0	0	-19.23	-77.07	2.87	2.87
DD-10A	-9.76	30.56	5.6	5.94	-12.21	-3.7	0	0.04	-37.72	34.74	3.99	3.99
DD-10B	-16.79	28.26	5.67	5.93	0	0	0	0	-32.05	24.88	4.66	4.66
DD-11A	-82.2	11.45	12.71	13.26	-6.28	-4.4	0	0.04	-84.01	23.29	2.32	2.32
DD-11B	-67.5	16	12.76	13.31	0	0	0	0	0	0	0	0
DD-12A	4.7	32.6	4.15	5.07	5.58	-9.62	0	0.04	0	0	0	0
DD-12B	2.56	40.26	4.14	5.07	0	0	0	0	0	0	0	0

Table 2.9: Double-Degenerate Data - Probability of Chance Alignment

WD	Prob of MC Chance Align	Prob of PM Chance Align	Prob of Chance Align
DD-01	0.01	0.01	0.0001
DD-02	0.03	0.02	0.0006
DD-03	0.09	0.01	0.0009
DD-04	0.02	0.04	0.0007
DD-05	0.01	0.01	0.0001
DD-06	0.03	0.03	0.0012
DD-07	0.74	0.01	0.0074
DD-08	0.04	0.01	0.0004
DD-09	0.03	0.01	0.0003
DD-10	0.02	0.01	0.0002
DD-11	0.01	0.01	0.0001
DD-12	0.04	0.01	0.0004

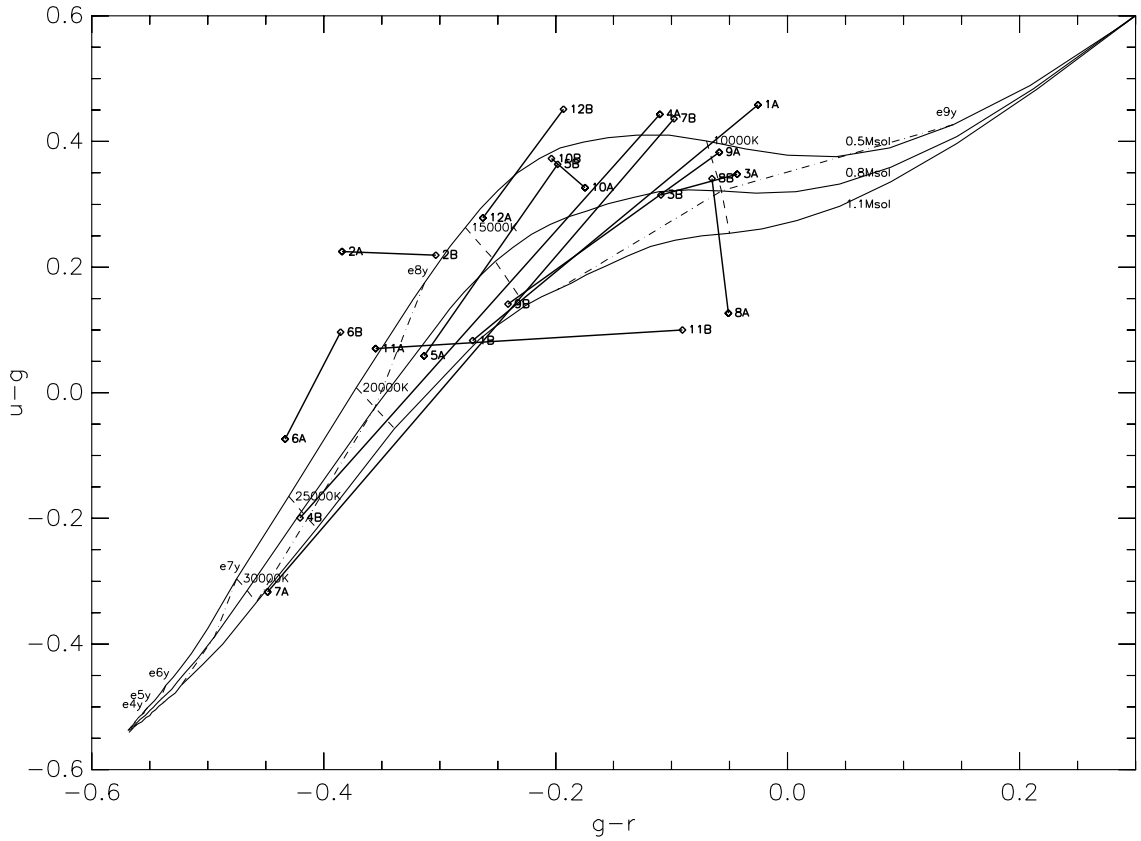


Figure 2.28: Colour-Colour for my 12 DD candidates (u-g vs g-r). Bergeron cooling tracts are shown with specific ages ($t_{cool} = 1 \times 10e^4 yrs - t_{cool} = 1 \times 10e^9 yrs$) and temperatures ($T_{\rm eff} = 30000K - T_{\rm eff} = 5000K$) marked.

"Further: Wish has for its object the End rather, but Moral Choice the means to the End"

from Aristotle's Nicomachean Ethics

3 Results

3.1 Properties of the components of the wide double-degenerate systems.

The masses, the cooling times and the absolute magnitudes for the WDs presented here were in general obtained by comparing my spectroscopic effective temperature ($T_{\rm eff}$) and surface gravity ($\log g$) estimates to grids of synthetic photometry (Holberg and Bergeron 2006) and evolutionary models (Fontaine et al. 2001). Custom written IDL routines (see Appendix), were used to perform bi-cubic spline interpolation between points in these grids. As these binaries consisted of WDs with different spectral characteristics, I had to follow a number of different approaches to determine their component properties. I outline each of these in turn below. My results are shown in Table 3.1.

3.1.1 Hot DA + hot DA systems

Here I define a hot DA as a H-rich object with no obvious sign of a magnetic field (i.e. no Zeeman splitting or broadening apparent in the line cores) and with an effective temperature, $T_{\rm eff} > 12000K$. For systems containing two such WDs, the component parameters were calculated completely independently. Absolute magnitudes, masses, cooling ages and distances were calculated by interpolating within the DA evolutionary grids to their measured spectroscopic temperatures and surface gravities. For each system (component), the uncertainties were calculated for the output parameters following the method outlined in Section 3.1.6. The systems which I found to contain two hot DA WDs are DD-05, DD-06, DD-10 and DD-11.

3.1.2 Cool DA + hot DA systems

Here I define a cool DA as a H-rich object with no obvious sign of a magnetic field (i.e. no Zeeman splitting or broadening apparent in the line cores) and with an effective temperature, $T_{\rm eff} \leq 12000K$. Spectroscopic mass determinations are known to be systematically larger in this regime than at higher effective temperatures where they agree well with those derived from gravitational redshifts (e.g. Bergeron et al. 1995a). This trend is most likely due to shortcomings in the treatment of convection within the model atmosphere calculations (Koester 2010, Tremblay et al. 2011). Hence for systems containing both a cool DA and a hot DA, I first determined the mass, the cooling time and the absolute magnitude of the latter component, as outlined above in Section 3.1.1. I then derived the distance modulus for the system from the apparent and the absolute g magnitudes of the hot DA. Subsequently, the parameters of the cool DA were calculated using the spectroscopic effective temperature estimate, the system distance and the apparent g magnitude of this component. While the same underlying evolutionary model grid was used in this case, the input parameters used here were the effective temperature, the system distance and the apparent g magnitude (as opposed to effective temperature and surface gravity). For each component, the uncertainties on the derived parameters are calculated following the method outlined in Section 3.1.6. The systems I found to contain a hot DA and a cool DA are DD-01, DD-07, DD-09 and DD-12.

3.1.3 Cool DA + cool DA systems

I have defined what I mean by a cool DA and outlined the issue with spectroscopic mass estimates for these objects in Section 3.1.2 above. Therefore, after following the prescription outlined in Section 3.1.1 to obtain a preliminary estimate of the masses of the cool components in these systems, I apply a (downward) correction of $\mathrm{d}M = -0.17 \pm 0.04 \mathrm{M}_{\odot}$, derived from the systems in my sample containing both a hot and a cool DA, to account for the spectroscopic overestimate. Based on my revised mass estimates, the parameters for each cool DA are recalculated (e.g. surface gravity and cooling time) using the same underlying evolutionary model grid but where the input parameters in these cases are effective temperature and mass. The uncertainties are calculated for all output parameters following the method outlined in section 3.1.6. These are larger than for systems containing at least one hot DA. The larger uncertainties in mass result in higher errors in the derived cooling times. DD-03 is the only system in my sample where both components are cool DA WDs.

3.1.4 HFMWD (DAH) + hot DA systems

The spectra of high field magnetic WDs (HFMWDs) are generally characterised by absorption lines which are very strongly broadened and shifted in wavelength and/or show Zeeman splitting (Schmidt et al. 2003). As detailed Stark line profile calculations do not yet exist for the high field regime, it is not possible to accurately determine the surface gravities and hence masses of these objects using the 'standard' spectroscopic

approach. Therefore, for my two systems which were found to contain a DAH and a hot DA (DD-02 and DD-04) I have followed the method described in Section 3.1.2, where the effective temperature of the DAH was obtained from the modelling undertaken by Baybars Kulebi. The larger errors in the effective temperature determinations for these DAHs (relative to a cool DA) propogate as greater uncertainties in the estimates of their masses.

3.1.5 HFMWD (DAH) + cool DA systems

For systems containing a cool DA and a DAH I first followed the procedure outlined in Section 3.1.3 to derive preliminary parameters of the former component. Next, I estimated the distance to the system using the apparent g magnitude and the absolute g magnitude predicted on the basis of the downward revised mass estimate for the DA. Subsequently, the paramaters of the DAH were determined as for those in the systems described in Section 3.1.4. The errors in the mass of the cool DAs and therefore the estimated system distances lead to greater errors in the masses determined for these DAHs relative to those paired with hot DAs. Only one of the systems in my sample, DD-08, was found to contain a cool DA and a HFMWD.

3.1.6 Estimation of the uncertainties in derived parameters

The uncertainties in the derived parameters of the WDs (distances, masses, cooling times, progenitor masses, progenitor lifetimes and system ages) were estimated using custom written IDL routines. After I had calculated the most likely value for each of these "derived" parameters, I generated 25000 realisations of the WD pairing, allowing for normally distributed errors in the "measured" parameters of these objects (effective temperatures, surface gravities, observed magnitudes, IFMR linear coefficients). Following Napiwotzki et al. (1999), for effective temperatures and surface gravities I assumed uncertainties of at least 2.3% and 0.07dex respectively, while for the SDSS photometry I assumed a minimum uncertainty of at least 0.02 magnitudes (Abazajian et al. 2009). The errors on the linear coefficients were obtained from the relevant papers. Each of these realisations was propagated through the interpolation routines written to determine the most likely values to build up distributions for these "derived" parameters. After sorting the arrays holding the 25000 estimates of these parameters, I selected the values in these distributions above and below which 16% of these data lay, which represent, approximately, 1σ upper and lower bounds.

3.2 The double-degenerate systems on a case-by-case basis

My estimates of the masses, the cooling times, the absolute magnitudes and the distances for the components of the twelve systems studied here are shown in Table 3.1. In determining the distances to my objects, I chose to ignore interstellar reddening since these targets are generally at relatively high galatic latitudes and are fairly close

to the Sun (typically $\lesssim$200pc). Moreover, as these pairings have very small angular separations, they share similar lines of sight so any low level extinction towards the components of a particular system is likely to be comparable and result in a small systematic overestimate of the binary's distance.

I have plotted the distances to the components of each system in Figure 3.2. Figures 3.3 and 3.4 show the matching of distance modulii for DD systems with at least one cool component. Figures 3.5 and 3.6 show the matching of distance modulii for DD systems with two cool components or a HFMWD system. Figure 3.7 shows the distances of the components within systems comprising only hot components.

I have used the components of these systems to probe three recent (independent) determinations of the form of the IFMR which are based largely on WD members of young and intermediate age open clusters ie. τ<1Gyr, ie. (e.g Dobbie et al. 2006), (Kalirai et al. 2008) and (Williams et al. 2009). In each of these cases the IFMR has been modelled using a simple linear function, with co-efficeints of a_0=0.289±0.051 a_1=0.133±0.015, a_0=0.394±0.025 a_1=0.109±0.007 and a_0=0.339±0.015 a_1=0.129±0.004, respectively.

I have used these relations to calculate the initial (progenitor) mass of each degenerate and then appealed to the solar metallicity stellar evolutionary models of Girardi et al. (2000) to determine their stellar lifetimes. By combining these with the WD cooling times I have obtained two estimates of the total age of each binary. If the published IFMRs are a good representation of the transformation of these objects from stars to WDs then it could be expected that for each system these two age estimates are consistent with one and other. However, if discord is found between the age estimates, this would suggest that these IFMRs do not reflect well the past evolution of the objects in question and for these cases I have proposed some possible reasons for this. These calculations have been performed using the codes listed in Appendix Section A.6.1, and my data spreadsheet referenced in Appendix Section A.7.1. I will now discuss the results of this process for each system on a case-by-case basis.

3.2.1 DD-01

Optical spectroscopy of the components of this binary indicates both to be hydrogen rich WDs and reveals no evidence for the presence of magnetic fields. Component A is a cool DA and component B is a hot DA thus it has been analysed following the method I outlined in 3.1.2. These objects are separated on sky by $6.78 \pm 0.1''$ and have consistent proper motions in RA,Dec of 1.7±8,−36.6±8 mas yr^{-1} and −3.6±8,−34.3±8 mas yr^{-1}, respectively. I have estimated, based on the observed properties, that there is less than a 1 in 10000 probability that these objects merely form a visual binary. I have determined DD-01A and DD-01B to have effective temperatures of $T_{\rm eff} = 10648 \pm 245K$ and $T_{\rm eff} = 19199 \pm 442K$, respectively, and derived their masses and cooling times to be M=$0.51^{+0.09}_{-0.05}$M$_\odot$, τ_{cool}=419^{+79}_{-38}Myr and M=$0.61^{+0.04}_{-0.04}$M$_\odot$, τ_{cool}=76^{+16}_{-13}Myr, respectively. Subsequently, from the characteristics of the hot DA, I have estimated the distance modulus of this system to be $m_g - M_g = 7.06 \pm 0.12$.

Adopting the recent Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al.

(2009) determinations of the IFMR, in turn, I estimate the progenitor mass of component A to be $M_{\rm init}$=$1.66^{+1.04}_{-0.51}$M$_{\odot}$, $1.07^{+0.93}_{-0.48}$M$_{\odot}$ or $1.33^{+0.72}_{-0.38}$M$_{\odot}$, respectively. The corresponding stellar lifetimes as predicted by the solar metallicity model grid of Girardi et al. (2000) are τ=2149^{+4089}_{-1584}Myr, 9756^{+5604}_{-9569}Myr or 4313^{+7997}_{-3119}Myr, resulting in total system age estimates of τ=2568^{+4108}_{-1410}Myr, 10170^{+5610}_{-9136}Myr or 4731^{+8019}_{-3033}Myr, respectively. Similarly, for component B I calculate initial masses of $M_{\rm init}$=$2.38^{+0.6}_{-0.52}$M$_{\odot}$, $1.94^{+0.46}_{-0.43}$M$_{\odot}$ or $2.06^{+0.34}_{-0.33}$M$_{\odot}$, leading to estimates of the total system age from this component of τ=989^{+732}_{-417}Myr, 1627^{+1382}_{-650}Myr or 1450^{+529}_{-476}Myr, for Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) respectively.

I find that my system age determinations from the two components are not formally consistent within their error bounds. This suggests that the simple models I adopt for mapping the masses of stars, from the main sequence to the WD stage, may not be a particularly good description of reality, at least in the case of this particular system. It is possible that none of the three IFMRs adopted here accurately reflects the mass loss history of one or both of these stars. The relation may not be very well represented in this initial mass regime by the simple linear fits which have been largely defined by studies of open cluster WDs, the majority of which are from stars of higher initial masses (ie. $M_{\rm init}\gtrsim$2-3M$_{\odot}$) or possibly there is substantial scatter in the relation here. Alternatively, perhaps as a consequence of having a close companion, component A has endured greater massloss than is predicted by my models. Studies of stellar multiplicity have revealed that at least 10% of stars are members of triple or higher order systems (Abt and Levy 1976, Raghavan et al. 2010). For reasons of dynamical stability, these systems are frequently hierarchically structured with triples often consisting of a body in a relatively wide orbit around a much closer pairing (Harrington 1972). DD-01A and DD-01B perhaps trace what was the wider orbit of a putative triple system, with component A possibly having been, or perhaps still being, part of a tighter pairing. A $M_{\rm init}\sim$2.5M$_{\odot}$ star that experiences Roche Lobe overflow while it ascends the thermally pulsing AGB can lead to the formation of a CO WD with a mass which is $\lesssim$0.15M$_{\odot}$ less than that from a single object of this mass (e.g. Iben and Tutukov 1985).

3.2.2 DD-02

The components of this system are separated on sky by 5.05±0.1″ and display comparable proper motions in RA,Dec of −28±19.6,2.9±19.6 mas yr^{-1} and −29±19.6,7.5±19.6 mas yr^{-1} for A and B, respectively. I estimate that there is less than a 1 in 1500 probability that these objects form a visual binary. My spectroscopy reveals the components of this system to be a hot DA (A) and a high field magnetic hydrogen rich (B) WD, so it has been analysed following the method I outlined in 3.1.4. I have measured the effective temperatures of DD-02A and DD-02B to be $T_{\rm eff} = 17622 \pm 405K$ and $T_{\rm eff} = 18000 \pm 1000K$, respectively. I have estimated the mass and the cooling time of DD-02A to be $0.69^{+0.04}_{-0.04}$M$_{\odot}$, and 157^{+25}_{-22}Myr, and the distance modulus of the system to be $m_g - M_g = 5.81 \pm 0.12$. Subsequently, I have derived the mass and the cooling time of DD-02B to be $1.03^{+0.04}_{-0.05}$M$_{\odot}$, and 354^{+50}_{-45}Myr. Given the distance of this system and the observed angular separation of the components, I find that DD-02A and DD-02B are separated by at least a$\sim$750AU. Even if it is assumed that the original orbital

separation of this binary was substantially smaller, the Roche lobes of the components have likely always been much larger than the dimensions of an asymptotic giant branch (AGB) star (r$\lesssim$4-5AU; Iben and Livio 1993).

In terms of mass and cooling time, component A is comparable to several of the Hyades WDs (e.g. WD 0352+098, WD 0421+162, Claver et al. 2001), whereas component B (M=1.01$\pm$0.05M$_{\odot}$) is similar in mass to the Pleiades WD LB 1497 (M=$1.02^{+0.04}_{-0.05}$M$_{\odot}$, $\tau_{\rm cool}\sim$50Myr, Bergeron et al. 1992, Dobbie et al. 2006, Wegner et al. 1991). Notably, LB 1497 is descended from a star with a lifetime shorter than the current age of this cluster (τ=125$\pm$8Myr, Stauffer et al. 1998) ie. $M_{\rm init}\gtrsim$4.8M$_{\odot}$. The parameters of the DA argue that this system is likely to have a similar total age to the Hyades (τ=625$\pm$50Myr, Perryman et al. 1998). Using the stellar lifetimes predicted by solar metallicity models of Girardi et al. (2000) and my determinations of the progenitor mass of component A ($M_{\rm init}$=$3.05^{+0.66}_{-0.57}$M$_{\odot}$, $2.76^{+0.5}_{-0.47}$M$_{\odot}$ or $2.76^{+0.37}_{-0.36}$M$_{\odot}$), I calculate the age of the binary to be τ=611^{+357}_{-181}Myr, 766^{+391}_{-219}Myr or 767^{+261}_{-166}Myr. The parameters of the latter object, which from my model of stellar evolution is the progeny of a star of $M_{\rm init}$=$5.55^{+0.88}_{-0.77}$M$_{\odot}$, $5.8^{+0.63}_{-0.62}$M$_{\odot}$ or $5.33^{+0.4}_{-0.44}$M$_{\odot}$, lead to estimates of the total age of the binary of τ=448^{+66}_{-46}Myr, 438^{+54}_{-41}Myr or 457^{+52}_{-40}Myr for Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) respectively.

I find that the estimates from the two components are only formally consistent within their quoted (1σ) error bounds when the oldest, least well constrained of the three approximations to the IFMR (Dobbie et al. 2006) is adopted. While the slight discord between some of the age estimates for this system alone do not constitute a significant result, it is notable for being similar in case to other HFMWDs, RE J0317-853 and EG 59, with that derived from the magnetic component being somewhat lower ($\sim$160-310Myr).

3.2.3 DD-03

My optical spectroscopy for the stars in this system indicates both to be hydrogen rich WDs. There is no evidence for the presence of magnetic fields. These two cool DAs are separated on sky by $11.16 \pm 0.1''$ and have "common" proper motions in RA,Dec of 4.3 ± 9.5,-50.2 ± 9.5 mas yr^{-1} and 5.2 ± 9.5,-51.1 ± 9.5 mas yr^{-1} for A and B, respectively. I estimate, based on these observed properties, that there is less than a 1 in 1000 likelihood that these stars are not physically associated. This system has been analysed following the method I outlined in 3.1.3.

I determine DD-03A and DD-03B to have effective temperatures of $T_{\rm eff} = 10008 \pm 230K$ and $T_{\rm eff} = 11222 \pm 258K$, respectively. I estimate their masses and cooling times to be M =$0.72^{+0.05}_{-0.05}$M$_{\odot}$, τ_{cool} =795^{+116}_{-101}Myr and M =$0.6^{+0.05}_{-0.05}$M$_{\odot}$, τ_{cool} =440^{+65}_{-53}Myr, respectively and find the distance modulii of the two components to be fully consistent with one and other (A: $m_g - M_g = 6.32 \pm 0.16$, B: $m_g - M_g = 6.36 \pm 0.14$).

Adopting the recent Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) determinations of the IFMR, in turn, I estimate the progenitor mass of component A to be $M_{\rm init}$=$3.22^{+0.71}_{-0.62}$M$_{\odot}$, $2.97^{+0.57}_{-0.54}$M$_{\odot}$ or $2.93^{+0.43}_{-0.42}$M$_{\odot}$, respectively. The corresponding stellar lifetimes as predicted by the solar metallicity model grid of Girardi et al. (2000) are τ=388^{+330}_{-164}Myr, 492^{+374}_{-194}Myr or 509^{+276}_{-167}Myr for Dobbie et al. (2006),

Kalirai et al. (2008) and Williams et al. (2009) respectively, resulting in total system age estimates of τ=1182$^{+296}_{-127}$Myr, 1287$^{+311}_{-116}$Myr or 1304$^{+201}_{-82}$Myr, respectively. Similarly, for component B I calculate initial masses of $M_{\rm init}$=2.32$^{+0.66}_{-0.58}$M$_{\odot}$, 1.87$^{+0.56}_{-0.53}$M$_{\odot}$ or 2.01$^{+0.45}_{-0.42}$M$_{\odot}$, leading to estimates of the total system age from this component of τ=1411$^{+887}_{-442}$Myr, 2073$^{+2435}_{-732}$Myr or 1900$^{+966}_{-566}$Myr, respectively. I find my estimates of the binary age from these two WDs are consistent with each other, indicating that the three IFMRs I adopt here may offer a satisfactory description of the massloss history of the components of this particular system.

3.2.4 DD-04

The components of this system are separated on sky by $9.14 \pm 0.1''$ and display similar but small proper motions in RA,Dec of 25.8 ± 4.2,-10.3 ± 4.2 mas yr^{-1} and 28.5 ± 4.2,-9.9 ± 4.2 mas yr^{-1} for A and B respectively. I estimate, based on these observed properties, that there is less than a 1 in 1000 probability that these objects are not physically related. Optical spectroscopy of this pairing reveals DD-04 to consist of two hydrogen rich WDs, one a high field magnetic (DD-04A) and the other non-magnetic (DD-04B). I have estimated the effective temperatures of DD-04B to be $T_{\rm eff} = 28067 \pm 646K$. I have also estimated the effective temperature (and surface gravity) of DD-04A to be $T_{\rm eff} = 16018 \pm 368K$, by excluding the central Zeeman split portions of the Balmer line cores from the spectral fitting process. As this system consists of a (comparatively) lower field DAH (DD-04A) and a hot DA (DD-04B), it has been analysed following the method I outlined in 3.1.1.

I have determined the mass and the cooling time of DD-04A to be 0.71$^{+0.04}_{-0.04}$M$_{\odot}$, and 310$^{+42}_{-36}$Myr, and the mass and the cooling time of DD-04B to be 0.91$^{+0.04}_{-0.04}$M$_{\odot}$, and 54$^{+14}_{-12}$Myr. I have calculated the distance modulii of DD-04A and DD-04B to be $m_g - M_g = 6.99 \pm 0.12$ and $m_g - M_g = 5.74 \pm 0.14$, respectively and I find them to be substantially different (1.26 magnitudes, $\sim 6\sigma$). Even if DD-04B was itself a multi system, which can result in objects appearing up to 0.75 magnitude closer, there would still be a discrepancy of 0.5 magnitudes. Thus, I am forced to conclude that these objects do not constitute a physical system.

3.2.5 DD-05

The components of DD-05 are separated on sky by $6.56 \pm 0.1''$ and display proper motions in RA,Dec of 60.9 ± 3.8,-85.9 ± 3.8 mas yr^{-1} and 54.3 ± 3.7,-60.9 ± 3.7 mas yr^{-1} for A and B respectively, which I consider to be in accord (see Section 2.4.3). I estimate, based on these observed properties, that there is less than a 1 in 8000 probability that these objects are not physically associated. My spectroscopy reveals this system to consist of two hot DAs. There is no evidence for the presence of magnetic fields so this system has been analysed following the method I outlined in Section 3.1.1.

I have estimated DD-05A and DD-05B to have effective temperatures of $T_{\rm eff} = 19020 \pm 438K$ and $T_{\rm eff} = 13950 \pm 321K$ and have determined their masses and cooling times to be 0.85$^{+0.04}_{-0.04}$M$_{\odot}$, 180$^{+29}_{-25}$Myr and 0.65$^{+0.04}_{-0.04}$M$_{\odot}$, 273$^{+36}_{-31}$Myr, respectively. I find the distance modulii of these two objects to be consistent since the error bounds

on my measurements overlap at the $\sim 1.5\sigma$ level (A: $m_g - M_g = 4.30 \pm 0.13$, B: $m_g - M_g = 4.55 \pm 0.12$). Given my distance estimate and the observed angular separation of the components, I determine that DD-05A and DD-05B are separated by at least a$\sim$475AU and thus unlikely to have ever exchanged mass as a result of Roche Lobe overflow (Girardi et al. 2000) . However, based on my estimates of the progenitor masses of these WDs (A: $M_{\rm init}$=$4.21^{+0.76}_{-0.65}$M$_\odot$, $4.17^{+0.56}_{-0.53}$M$_\odot$ or $3.95^{+0.39}_{-0.38}$M$_\odot$ and B: $M_{\rm init}$=$2.73^{+0.63}_{-0.55}$M$_\odot$, $2.36^{+0.49}_{-0.46}$M$_\odot$ or $2.42^{+0.36}_{-0.35}$M$_\odot$) and their corresponding predicted stellar lifetimes τ=187^{+106}_{-64}Myr, 191^{+85}_{-53}Myr or 221^{+70}_{-49}Myr and B: τ=630^{+540}_{-284}Myr, 926^{+666}_{-374}Myr or 867^{+486}_{-275}Myr I find the total system ages derived from these two components to be discordant, A: τ=367^{+101}_{-55}Myr, 371^{+72}_{-37}Myr or 401^{+54}_{-29}Myr and B: τ=903^{+527}_{-266}Myr, 1199^{+648}_{-350}Myr or 1139^{+462}_{-246}Myr (for Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) respectively).

Despite both these WDs having masses which are comparable to those of the objects seen in the young/intermediate age open clusters which dominate current estimates of the form the IFMR, this system does not fit well with the simple assumptions I have made about stellar evolution. It is possible that the level of scatter in the IFMR has been somewhat underestimated by the open cluster studies, with DD-05B lying well below the general pattern (see Section 3.2.13). Alternatively, I note that LB 5893, which is a member of Praesepe, sits well above the trend defined by the other nine WD members of the cluster (e.g Dobbie et al. 2006). It has been proposed that this object is the progeny of a blue straggler (Casewell et al. 2009), a type of star whose evolution appears to have been retarded. It is conceivable that DD-05A has similar origins and thus appears substantially younger than the binary age as estimated from DD-05B.

Further higher resolution spectroscopy of this pair of WDs would be useful. At effective temperatures comparable to those of these objects the narrow, non-LTE core of the Hα line is rather prominent in absorption, permitting it's wavelength displacement to be determined to a few km s^{-1} with data of moderate resolution (R$\sim$6000). This would allow measurements of their relative radial velocities so that the small possibility that they are not associated can be ruled out and/or to search for evidence of velocity variation which would betray the presence of a close companion object.

3.2.6 DD-06

The components of this system are separated on sky by $2.04 \pm 0.1''$ and have proper motions that are consistent with one and other (ie. RA=18.6 ± 8.4, Dec=-33.7 ± 8.4 mas yr^{-1} and RA=13.9 ± 9, Dec=-36.3 ± 9 mas yr^{-1} for A and B, respectively). I estimate, based on these observed properties, that there is less than a 1 in 800 probability that these objects are merely a visual binary. My optical spectroscopy reveals both objects to be hot, hydrogen rich WDs, with neither displaying any evidence for the presence of magnetic fields. I have analysed the components of this system following the method I outlined in 3.1.1.

I have estimated DD-06A and DD-06B to have effective temperatures of $T_{\rm eff} = 24788 \pm 570K$ and $T_{\rm eff} = 18502 \pm 426K$ respectively. I have determined the mass and the cooling time of DD-06A to be $0.67^{+0.04}_{-0.04}$M$_\odot$, and 27^{+8}_{-5}Myr, and the mass and the cooling time of DD-06B to be $0.66^{+0.04}_{-0.04}$M$_\odot$, and 108^{+20}_{-17}Myr. I find the distance

modulii of DD-06A and DD-06B ($m_g - M_g = 6.49 \pm 0.12$ and $m_g - M_g = 6.41 \pm 0.15$, respectively) to be fully consistent. Given the system distance and the observed angular separation of the components, I find that DD-06A and DD-06B are separated by at least a$\sim$390AU. Thus it is unlikely that the Roche lobes of the progenitors of these WDs too have even been smaller than the dimensions of an AGB star (Iben and Livio 1993).

Based on the recent Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) determinations of the IFMR, I estimate the progenitor mass of component A to be $M_{\rm init}$=$2.85^{+0.63}_{-0.55}$M$_{\odot}$, $2.52^{+0.48}_{-0.45}$M$_{\odot}$ or $2.55^{+0.35}_{-0.33}$M$_{\odot}$, respectively. The corresponding stellar lifetimes as predicted by the solar metallicity model grid of Girardi et al. (2000) are τ=554^{+444}_{-241}Myr, 786^{+587}_{-307}Myr or 756^{+351}_{-231}Myr (for Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) respectively), resulting in total system age estimates of τ=581^{+442}_{-238}Myr, 813^{+583}_{-301}Myr or 784^{+348}_{-225}Myr, respectively. Similarly, for component B I calculate initial masses of $M_{\rm init}$=$2.75^{+0.63}_{-0.54}$M$_{\odot}$, $2.4^{+0.48}_{-0.45}$M$_{\odot}$ or $2.45^{+0.35}_{-0.34}$M$_{\odot}$), leading to estimates of the total system age from this component of τ=721^{+502}_{-265}Myr, 1002^{+641}_{-338}Myr or 950^{+434}_{-245}Myr, respectively. The agreement I find between these binary age determinations indicate that DD-06A and DD-06B have characteristic which are consistent with current estimates of the form of the IFMR.

3.2.7 DD-07

The components of this system are separated on sky by $18.91 \pm 0.1''$ and have proper motions in RA,Dec of -52 ± 9.7,30.8 ± 9.7 mas yr^{-1} and -52.1 ± 9.7,27.2 ± 9.7 mas yr^{-1} for A and B, respectively. I estimate, based on these observed properties, that there is approximately a 1 in 135 probability that these objects are not physically associated. My follow-up spectroscopy has revealed both stars to be hydrogen rich WDs but no evidence for the presence of magnetic fields. I have estimated DD-07A and DD-07B to have effective temperatures of $T_{\rm eff} = 33444 \pm 769K$ and $T_{\rm eff} = 10393 \pm 239K$ respectively. Consequently, this system has been analysed following the method I outlined in 3.1.2.

I have determined the mass and the cooling time of DD-07A to be $0.65^{+0.04}_{-0.04}$M$_{\odot}$, and 6^{+3}_{0}Myr, and the distance modulus of the system to be $m_g - M_g = 6.17 \pm 0.13$. Subsequently I have determined the mass and the cooling time of DD-07B to be $0.6^{+0.08}_{-0.05}$M$_{\odot}$, and 540^{+104}_{-54}Myr. I estimate the progenitor mass of component A to be $M_{\rm init}$=$2.74^{+0.61}_{-0.53}$M$_{\odot}$, $2.38^{+0.47}_{-0.43}$M$_{\odot}$ or $2.43^{+0.33}_{-0.32}$M$_{\odot}$, using the IFMRs of Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009), respectively and similarly, for component B I calculate initial masses of $M_{\rm init}$=$2.31^{+0.95}_{-0.58}$M$_{\odot}$, $1.86^{+0.88}_{-0.55}$M$_{\odot}$ or $2^{+0.69}_{-0.44}$M$_{\odot}$). The corresponding stellar lifetimes for component A as predicted by the solar metallicity model grid of Girardi et al. (2000) are τ=623^{+511}_{-275}Myr, 913^{+628}_{-354}Myr or 857^{+419}_{-253}Myr, resulting in total system age estimates of τ=630^{+632}_{-268}Myr, 919^{+706}_{-346}Myr or 863^{+511}_{-249}Myr, respectively. From component B I estimate the total system age to be τ=1520^{+902}_{-454}Myr, 2188^{+2633}_{-935}Myr or 2012^{+1128}_{-717}Myr, respectively.

I find my system age estimates from these two components to be discordant at $\sim 3\sigma$. This argues that the simple models I have adopted for mapping the masses of stars, from the main sequence to the WD stage, are not an appropriate description of the evolutionary history of this particular system. As I have suggested for DD-01A, it could

be that as consequence of having a close companion, component B has suffered heavier massloss than predicted by the three IFMRs. However, in terms of their masses, these WDs lie at and beyond the lower end of the range of the IFMR which has been mapped using the members of open clusters. Thus an alternative explanation could be that a simple linear extrapolation of the form the IFMR observed at higher masses into this regime does not adequately represent reality. These data are consistent with a relation which is substantially flatter (ie. $a_1 \rightarrow 0$) in this mass regime.

3.2.8 DD-08

The components of this system are separated on sky by $4.35 \pm 0.1''$ and display proper motions in RA,Dec of -19.4 ± 0.7,-74.4 ± 0.7 mas yr^{-1} and -5.3 ± 0.9,-61.1 ± 0.9 mas yr^{-1} for A and B, respectively. I estimate on grounds of these observed properties, that there is less than a 1 in $\sim$2500 probability that these objects are a mere visual binary. My follow-up optical spectroscopy reveals this system to contain a high field magnetic DA (DD-08A) and a cool DA (DD-08B) so it has been analysed following the method I outlined in 3.1.5.

I have estimated components A and B to have effective temperatures of $T_{\rm eff} = 9500 \pm 500K$ and $T_{\rm eff} = 10492 \pm 241K$, respectively. I have determined the mass and the cooling time of DD-08B to be $0.8^{+0.05}_{-0.05}$M$_\odot$ and 849^{+114}_{-106}Myr, and the distance modulus of the system to be $m_g - M_g = 5.89 \pm 0.16$. Subsequently I have derived the mass and the cooling time of DD-08A to be $0.61^{+0.14}_{-0.09}$M$_\odot$, and 715^{+183}_{-91}Myr. Following again my procedure above, using the three different IFMRs, I estimate the progenitor mass for DD-08B as $M_{\rm init}$=$3.81^{+0.76}_{-0.65}$M$_\odot$, $3.69^{+0.6}_{-0.56}$M$_\odot$ or $3.54^{+0.44}_{-0.42}$M$_\odot$ and calculate the total age of this binary to be τ=1092^{+179}_{-92}Myr, 1115^{+146}_{-62}Myr or 1146^{+109}_{-48}Myr (for Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) respectively). From the parameters of the DAH component, which, from my modelling, is expected to be the progeny of a star of $M_{\rm init}$=$2.44^{+1.17}_{-0.78}$M$_\odot$, $2.02^{+1.31}_{-0.85}$M$_\odot$ or $2.13^{+1.08}_{-0.7}$M$_\odot$, I infer a total system age of τ=1562^{+1159}_{-405}Myr, 2158^{+2986}_{-899}Myr or 1967^{+1819}_{-637}Myr (again, for Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009) respectively). These two sets of estimates are not formally consistent within their 1σ errors for any of the three adopted IFMRs, although, admittedly the discrepancy is not overwhelming, statistically. However, the variance suggested here is in the opposite sense to that observed for DD-02, RE J0317-853 and EG 59 with DD-08A appearing "too old" for its mass or, alternatively, too "low in mass" for its cooling time. It could be expected, on the basis of my simple model, that since the WDs in this system have similar cooling times, they should have comparable masses, having descended from stars with similar initial masses. Component A appears to have experienced greater massloss than would be expected for a single star. As was suggested for DD-01B, this could be a consequence of having a close companion, if this system was a primordial triple. A $M_{\rm init}\sim$3.5M$_\odot$ star that experiences Roche Lobe overflow around the time of central helium ignition can lead to the formation of a CO white dwarf with a mass towards the lower end of the range estimated for this HFMWD (e.g. Iben and Tutukov 1985). However, I note that there is an observed lack of HFMWDs with close detached companions (Liebert et al. 2005b). Nevertheless, the intermediate temperature of the non-magnetic component

in this system, coupled with its comparatively high mass suggests that this HFMWD is linked to an early-type star, $M_{\rm init}$>2.1M$_{\odot}$.

3.2.9 DD-09

The proper motions I determine for the components of this system are consistent with one and other, RA,Dec = -17.8 ± 5.6,-47.3 ± 5.6 mas yr^{-1} and -12.3 ± 5.7,-40.6 ± 5.7 mas yr^{-1} for A and B respectively. These sources are separated on sky by $5.18\pm0.1''$. On grounds of these observed properties, I estimate that there is less than a 1 in 3500 probability that these objects are not physically associated. My optical spectroscopy of these stars indicates both to be hydrogen rich WDs and reveals no evidence for the presence of magnetic fields. I have estimated DD-09A and DD-09B to have effective temperatures of $T_{\rm eff} = 11543 \pm 266K$ and $T_{\rm eff} = 16037 \pm 369K$ respectively. As this system consists of a cool DA (DD-09A) and a normal DA (DD-09B) WD, it has been analysed following the method I outlined in 3.1.2.

I have determined the mass and the cooling time of DD-09B to be $1.17^{+0.03}_{-0.03}$M$_{\odot}$, and 893^{+111}_{-136}Myr, and the distance modulus of the system to be $m_g - M_g = 5.56 \pm 0.15$.

Close scrutiny of DD-09A reveals that it is unlike the cooler components in the three other cool DA + hot DA systems in my sample, where the spectroscopic mass determinations are larger by $\sim$0.17M$_{\odot}$ than those calculated by assuming that the objects in each pairing have a common distance (the binary method). I find the spectroscopic mass of DD-09A is smaller by $\sim$0.20M$_{\odot}$ than that determined by the binary method. As the results of several studies of field WDs also point to spectroscopic determinations overestimating the true mass at $T_{\rm eff} < 12000$K, I am led to conclude that my mass estimate for this star might be in error. Therefore, I have also explored the parameters of this system by applying the spectroscopic mass offset as calculated for cool components ($\sim$0.17M$_{\odot}$). In this case, the mass of component A is set to $0.54^{+0.05}_{-0.05}$M$_{\odot}$ (with a corresponding cooling time of 360^{+49}_{-42}Myr). I find determinations of the system age from this object of τ=1950^{+2328}_{-792}Myr, 4356^{+8794}_{-2485}Myr or 2894^{+3649}_{-1015}Myr. These are not consistent with the ages estimates from component B. The revised distance modulii of component A is $m_g - M_g = 6.47 \pm 0.14$ and also appears to be discrepant with respect to that of component B.

It can be seen in Figures 3.3 and 3.4 that the cooler component of DD-09 (DD-09A) actually has a higher apparent distance modulus than its hot companion before mass correction (distance matching) - where as the opposite is the case for all other Cool+Hot Systems. This discrepancy can also be seen in Figure 3.1, in which my effective spectroscopic mass offsets (to align cool component distances with their respective hot components) are compared to that of Tremblay et al. (2011) across temperature.

Thus the obvious conclusion on this basis is that these objects do not form a physical binary. However, their proximity to each other on sky and relatively large and similar proper motions seem to point rather strongly to a physical association. It is interesting to note that the disparity between the distance modulii of these two objects is within 1σ of $\sim$0.75 magnitudes and in the sense that the distances would be in accord if DD-09B, which appears to be unusually massive, actually consists of a spatially unresolved pairing of WDs with similar masses and effective temperatures.

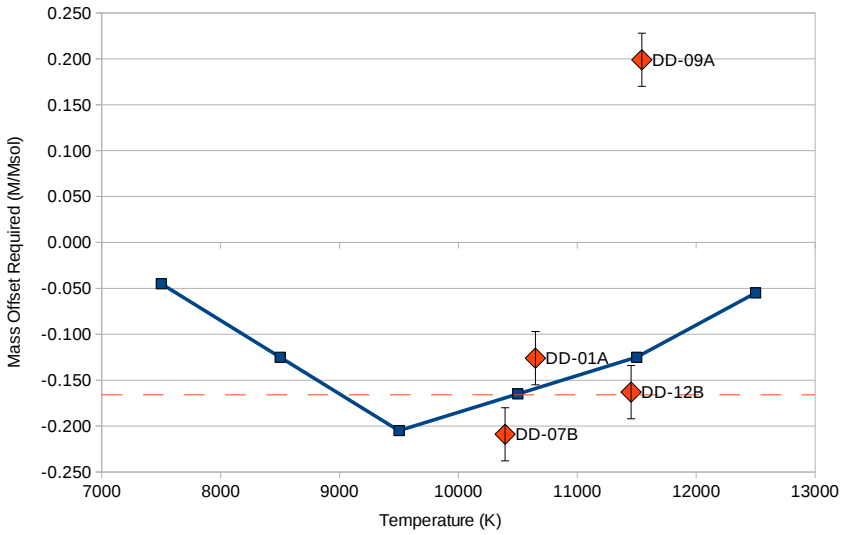


FIGURE 3.1: A plot in which my effective spectroscopic mass offsets (to align cool component distances with their hot components) are compared to that of Tremblay et al. (2011) across temperature. The blue curve is that of Tremblay et al. (2011), while the red data points are calculated from my Double Degenerate survey. The error in my derived mass offset is displayed with error bars, while my derived mass offset itself is displayed as a dashed red line about $\mathrm{d}M = -0.17 \pm 0.04 \mathrm{M}_{\odot}$.

This is an intriguing system and clearly one which warrants further detailed follow-up study e.g. as proposed for DD-05, higher resolution spectroscopy extending over the Hα line to permit measurements of the radial velocities of these objects. For the rest of my analysis, I assume that this pair of objects is not associated.

3.2.10 DD-10

The components of this system are separated on sky by $5.93 \pm 0.1''$ and have proper motions in RA,Dec of -9.8 ± 5.6,30.6 ± 5.6 mas yr^{-1} and -16.8 ± 5.7,28.3 ± 5.7 mas yr^{-1} for A and B respectively. I estimate, based on these observed properties, that there is less than a 1 in 5000 probability that these objects are only a visual binary. From my follow-up spectroscopy I find these stars to both be hydrogen rich WDs, with neither displaying any evidence for the presence of magnetic fields. I have estimated DD-10A and DD-10B to have effective temperatures of $T_{\mathrm{eff}} = 13072 \pm 301K$ and

$T_{\rm eff} = 12758 \pm 293K$ respectively. As this system consists of a normal DA (DD-10A) and a normal DA (DD-10B) WD, it has been analysed following the method I outlined in 3.1.1.

I have determined the masses and the cooling times of DD-10A and DD-10B to be $0.88^{+0.04}_{-0.05}$M$_\odot$, 574^{+82}_{-70}Myr and $0.76^{+0.05}_{-0.04}$M$_\odot$, 450^{+62}_{-52}Myr, respectively. Subsequently, I have calculated the distance modulii of DD-10A and DD-10B to be $m_g - M_g = 5.66 \pm 0.12$ and $m_g - M_g = 5.59 \pm 0.12$, respectively, which I note are in accord with one and other. Based on the IFMRs of Dobbie et al. (2006), Kalirai et al. (2008) and Williams et al. (2009), I have derived the progenitor masses of components A and B to be $M_{\rm init}$=$4.43^{+0.79}_{-0.67}$M$_\odot$, $4.45^{+0.57}_{-0.54}$M$_\odot$ or $4.18^{+0.4}_{-0.39}$M$_\odot$ and $M_{\rm init}$=$3.51^{+0.7}_{-0.6}$M$_\odot$, $3.31^{+0.53}_{-0.5}$M$_\odot$ or $3.23^{+0.39}_{-0.37}$M$_\odot$, respectively. These lead to estimates of the total system age from component A of τ=737^{+106}_{-61}Myr, 736^{+79}_{-43}Myr or 763^{+65}_{-36}Myr and from component B of τ=756^{+206}_{-99}Myr, 808^{+190}_{-83}Myr or 836^{+130}_{-62}Myr, respectively.

I find that the two binary age determinations are in excellent agreement for all three of the adopted IFMRs. I note that these two WDs have masses which are typical of those studied in young/intermediate age open clusters and that largely dominate current estimates of the form the relation. The components of DD-10 appear to lend support to the results derived from the open cluster investigations.

3.2.11 DD-11

Components A and B in this pairing have consistent proper motions in RA,Dec of -82.2 ± 12.7,11.4 ± 12.7 mas yr^{-1} and -67.5 ± 12.8,16 ± 12.8 mas yr^{-1}, respectively. They are separated on sky by only $1.32 \pm 0.1''$. Based on these observed properties, I estimate that there is only a 1 in 10500 probability that these objects are not physically associated. Optical spectroscopy of these stars indicates both to be hydrogen rich WDs and in neither case reveals evidence for the presence of magnetic fields. I have measured DD-11A and DD-11B to have effective temperatures of $T_{\rm eff} = 20000 \pm 460K$ and $T_{\rm eff} = 14037 \pm 323K$ respectively. As this system consists of two hot DAs, it has been analysed following the method I outlined in 3.1.1.

I have estimated the masses and the cooling times of DD-11A and DD-11B to be $0.74^{+0.04}_{-0.04}$M$_\odot$, 107^{+20}_{-18}Myr and $0.89^{+0.04}_{-0.05}$M$_\odot$, 489^{+70}_{-60}Myr, respectively and I find their distance modulii to be consistent (A:$m_g - M_g = 5.37 \pm 0.12$ and B: $m_g - M_g = 5.61 \pm 0.27$). From this distance determination and the observed angular separation of these sources, I calculate that the minimum separation is a$\sim$175AU. Thus even allowing for orbital expansion by a factor 5-10 as a result of the loss of primordial mass from the system, it is unlikely that either of these objects has exchanged gas as the result of Roche Lobe overflow. I derive progenitor masses for components A and B, based on the IFMRs used in previous sections, of $M_{\rm init}$=$3.36^{+0.69}_{-0.58}$M$_\odot$, $3.14^{+0.52}_{-0.49}$M$_\odot$ or $3.08^{+0.38}_{-0.37}$M$_\odot$ and $M_{\rm init}$=$4.54^{+0.79}_{-0.67}$M$_\odot$, $4.58^{+0.58}_{-0.54}$M$_\odot$ or $4.29^{+0.39}_{-0.39}$M$_\odot$, respectively. These lead to estimates of the total system age from component A of τ=450^{+244}_{-127}Myr, 524^{+251}_{-131}Myr or 548^{+183}_{-108}Myr and from component B of τ=643^{+93}_{-54}Myr, 640^{+69}_{-38}Myr or 666^{+56}_{-31}Myr, respectively.

I find that the binary age determinations from these two components are fully consistent with each other for all models of the IFMR considered. This argues that for DD-11 these can be a good representation of the massloss history of the component

stars. As was the case for DD-10, this result is not entirely surprising since these two WDs also have masses which are typical of those studied in young/intermediate age open clusters and that largely dominate current estimates of the form the IFMR. Indeed, I note that the estimates of system age, from model to model are all very similar, highlighting this point.

3.2.12 DD-12

My follow-up spectroscopy for these stars has revealed both to be hydrogen rich WDs. I also find no evidence for the presence of magnetic fields in these datasets. The components of this system are separated on sky by $4.29 \pm 0.1''$ and have proper motions in RA,Dec of 4.7 ± 4.2,32.6 ± 4.2 mas yr^{-1} and 2.6 ± 4.1,40.3 ± 4.1 mas yr^{-1} for A and B respectively, which I consider to be consistent (see Section 2.4.3). I estimate, based on these properties, that there is less than a 1 in 2500 probability that these objects are merely a visual binary. From an analysis of the spectra I determine the effective temperatures of component A and component B to be $T_{\rm eff} = 15834 \pm 364K$ and $T_{\rm eff} = 11453 \pm 263K$ respectively. As this system consists of a hot DA (DD-12A) and a cool DA (DD-12B), it has been analysed following the method I outlined in 3.1.2.

I have estimated the mass and the cooling time of DD-12A to be $0.63^{+0.04}_{-0.04}$M$_{\odot}$, and 172^{+26}_{-22}Myr, and the distance modulus of the system to be $m_g - M_g = 5.30 \pm 0.12$. Subsequently I have calculated the mass and the cooling time of DD-12B to be $0.63^{+0.12}_{-0.04}$M$_{\odot}$, and 449^{+144}_{-41}Myr. Using the three IFMRs discussed in the Sections above, I have determined the progenitor masses of component A and component B to be $M_{\rm init}$=$2.54^{+0.6}_{-0.54}$M$_{\odot}$, $2.13^{+0.47}_{-0.45}$M$_{\odot}$ or $2.23^{+0.35}_{-0.34}$M$_{\odot}$ and $M_{\rm init}$=$2.57^{+1.08}_{-0.54}$M$_{\odot}$, $2.17^{+1.23}_{-0.47}$M$_{\odot}$ or $2.26^{+0.99}_{-0.37}$M$_{\odot}$, respectively. These lead to estimates of the total system age from component A of τ=942^{+691}_{-339}Myr, 1429^{+807}_{-524}Myr or 1270^{+496}_{-344}Myr and from component B of τ=1190^{+655}_{-325}Myr, 1629^{+807}_{-655}Myr or 1494^{+531}_{-480}Myr, respectively. I find general agreement between these binary age determinations, with only the ages stemming from the use of the oldest IFMR not being formally consistent within the 1σ error bounds. I am led to conclude that the observed properties of DD-12A and DD-12B are consistent with current estimates of the form of the IFMR.

3.2.13 My systems overall and understanding of the IFMR

The derived parameters of half of the pairs which I find to be physically associated are consistent with recent estimates of the IFMR that are based largely on open cluster WDs. However, I find that three of my systems, DD-01, DD-05 and DD-07, do not conform to predictions based on these same estimates of the IFMR. I note that the WDs in these three binaries all have $M \lesssim 0.65$M$_{\odot}$. While open clusters now provide more than 50 data points for the IFMR which span a broad range of initial masses (1.5-6M$_{\odot}$), they are largely concentrated within the range M$_{\rm init}$=2.8-5M$_{\odot}$. These data suggest that the bulk of stars follow relatively closely a monotonic relation which can be adequately represented by a simple linear function at least down to M$_{\rm init}$$\approx$2.5M$_{\odot}$ (e.g. Williams et al. 2009).

That the three discordant systems contain WDs with $M \lesssim 0.65\mathrm{M}_{\odot}$ might be taken as an indication that the IFMR may not be very well represented at lower initial/final masses by these linear functions. Theoretical considerations suggest that the relation is likely to become flatter at $\mathrm{M}_{\mathrm{init}} \approx$2-2.5$\mathrm{M}_{\odot}$ than it is at higher masses (Weidemann 2000). Nevertheless, three of the systems which have parameters that appear to be consistent with the cluster dominated IFMRs also contain at least one WD with $M \lesssim 0.65\mathrm{M}_{\odot}$, although the masses of the components of DD-06 and DD-12 are very similar to one and other and could be expected to, in my approach here, be little affected by erroneous assumptions about the relations form.

A small number of points in the open clusters are known to deviate from the general trend delineated by the bulk of the data, including LB 5893, LB 390 and half a dozen WDs towards the intermediate age population NGC 2099 (Kalirai et al. 2008). This latter dataset may not be reliable since the spectroscopy for these WDs was of rather low signal-to-noise towards the blue end of the optical where the gravity sensitive high order Balmer lines reside. Nevertheless, a recent investigation of the IFMR based on wide stellar + WD binaries, which probes relatively lower masses, has found some evidence of greater scatter in the IFMR than indicated by the cluster studies (Catalan et al. 2008). It seems plausible that the level of scatter in the IFMR maybe mass dependent. Dobbie et al. (2009) argue that the IFMR at $M_{\mathrm{init}}>2.5\mathrm{M}_{\odot}$, where third dredge-up is expected to be rather efficient at preventing further growth of the degenerate core during the AGB, resembles the M_{init} versus core mass at the time of the first thermal-pulse relation. At lower initial masses third dredge-up is expected to be less efficient with the core growing significantly during this phase. Thus any factor that can substantially influence the rate of mass loss during the AGB (e.g. the presence of a close companion), when a star attains its maximum radius, could impact the final mass of the resulting WD.

Alternatively, it could be that at least some of the open cluster studies have been slightly biased against finding WDs of lower mass that form from recently deceased stars. Generally, "young" degenerates are identified in colour-magnitude diagrams by their blue colours which often cleanly separate them from the bulk of field stars. However, as WDs of lower mass have larger radii and cool more quickly, they will generally be redder in colour and reside closer to the field star population. Since the 8m class telescope time required to obtain spectra of these objects is at a premium, this part of colour-magnitude space is less likely to be targeted for follow-up observations.

In summary, my analysis of these twelve systems suggests that recent estimates of the form of the IFMR provide a fair description of the transformation of intermediate mass stars into WDs ($M_{\mathrm{init}}>2.5\mathrm{M}_{\odot}$), but it is likely that there is more scatter in the relation than assumed from the results of recent open cluster based studies, at least at lower initial/final masses (ie. $M_{\mathrm{init}}<2.5\mathrm{M}_{\odot}$).

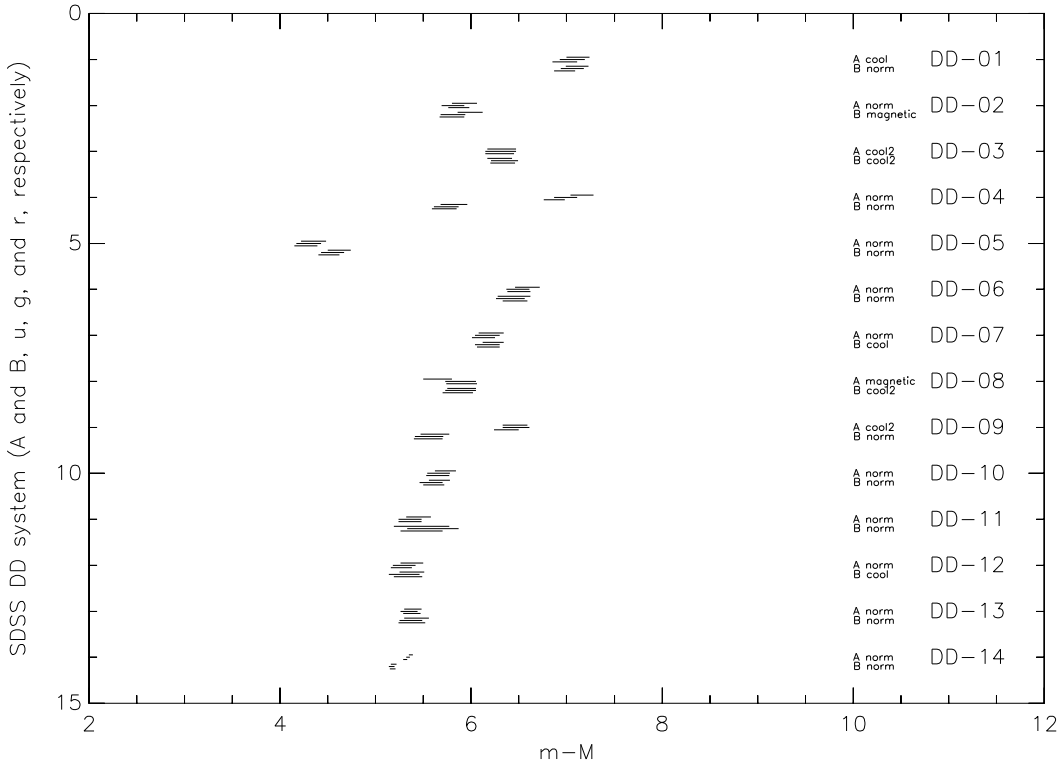


FIGURE 3.2: Distance Modulii (m - M) for all DD systems (u, g, r), with cool component mass corrections / distance matching

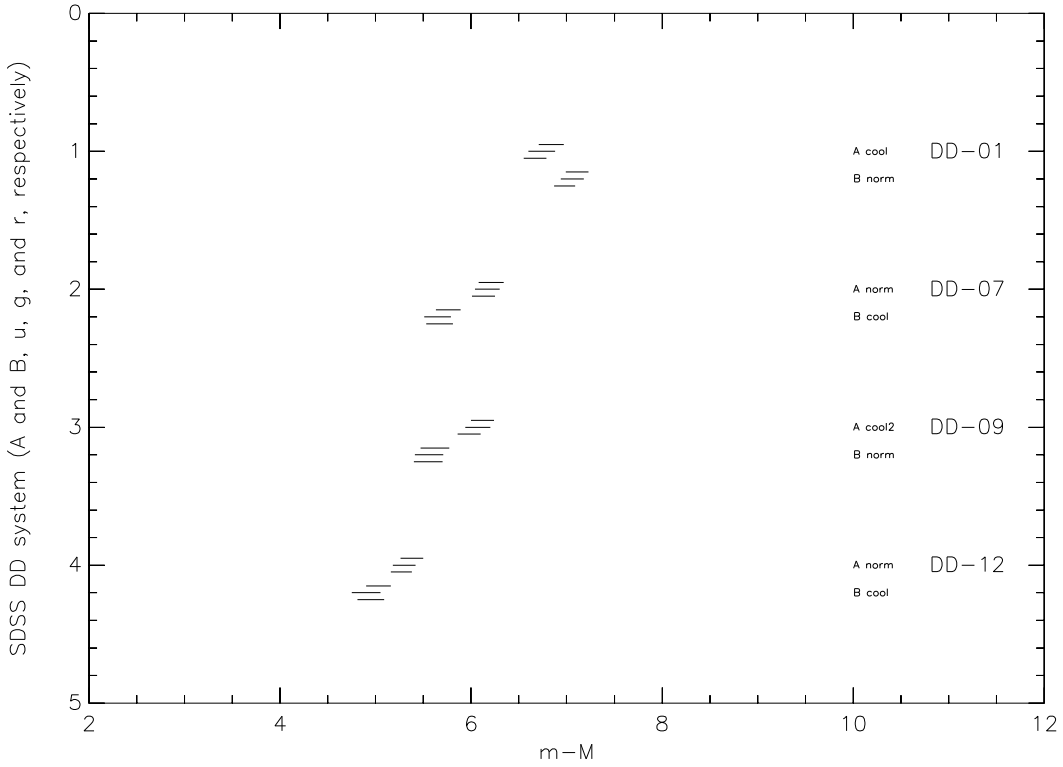


FIGURE 3.3: Distance Modulii (m - M) for DD systems with at least one cool component (u, g, r), without cool component mass corrections / distance matching

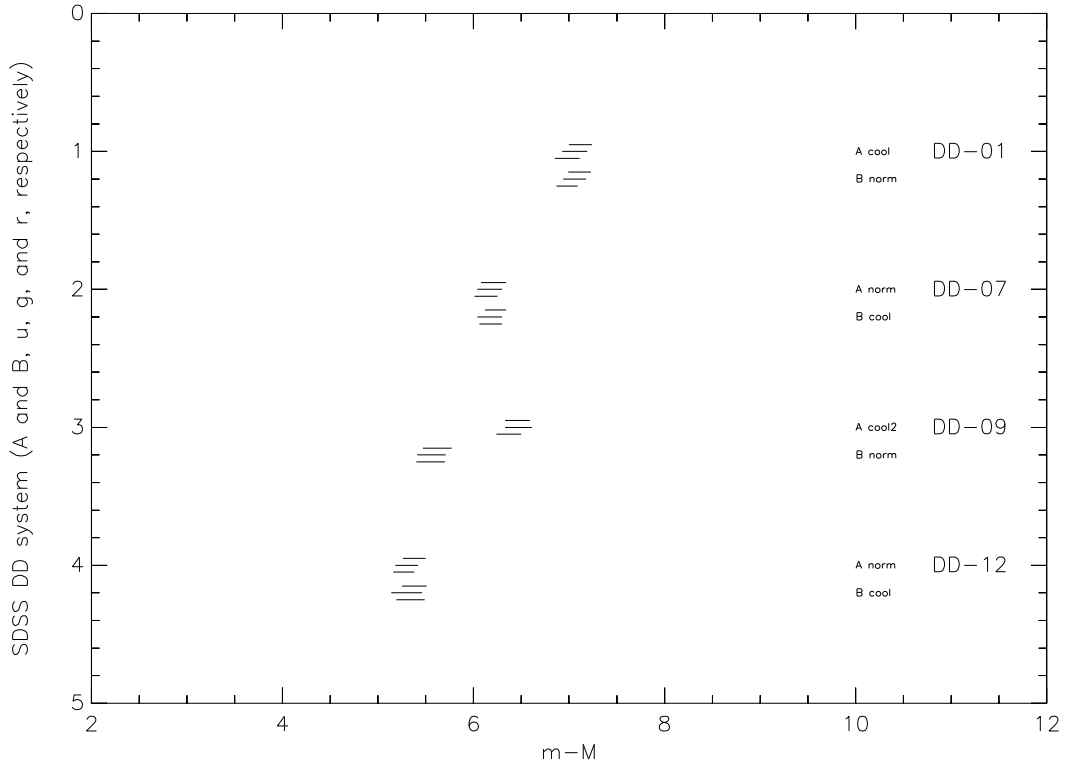


Figure 3.4: Distance Modulii (m - M) for DD systems with at least one cool component (u, g, r), with cool component mass corrections / distance matching

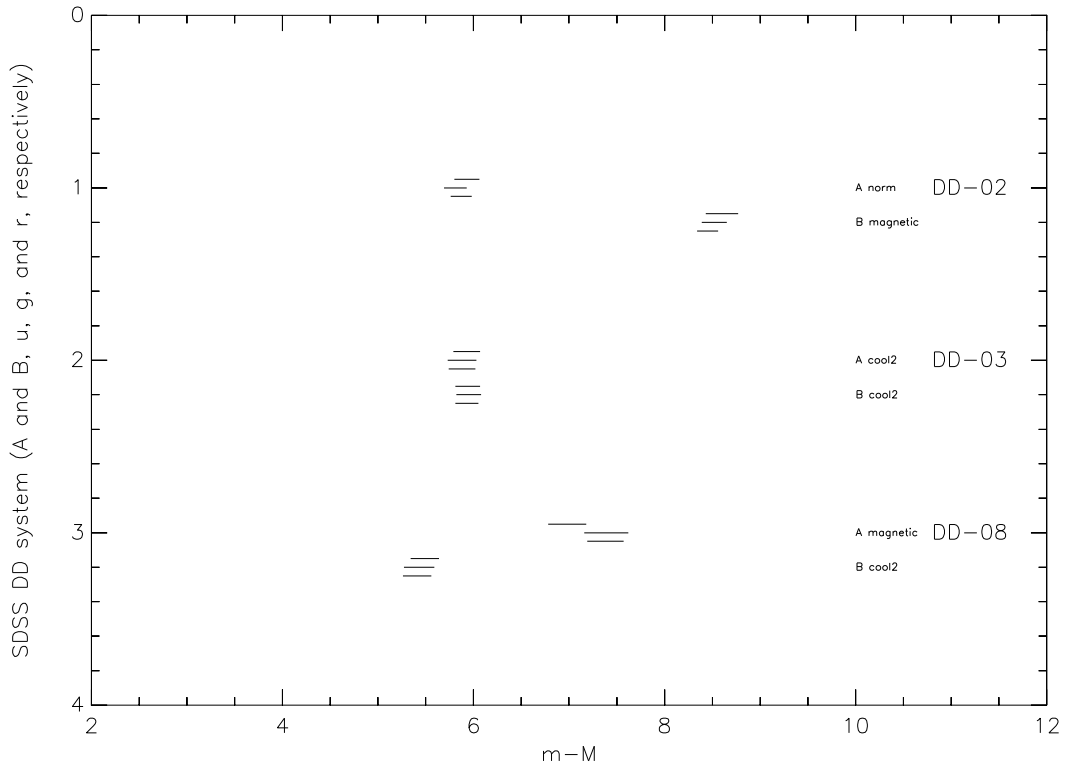


Figure 3.5: Distance Modulii (m - M) for DD systems with two cool components or a HFMWD system (u, g, r), without cool component mass corrections / distance matching

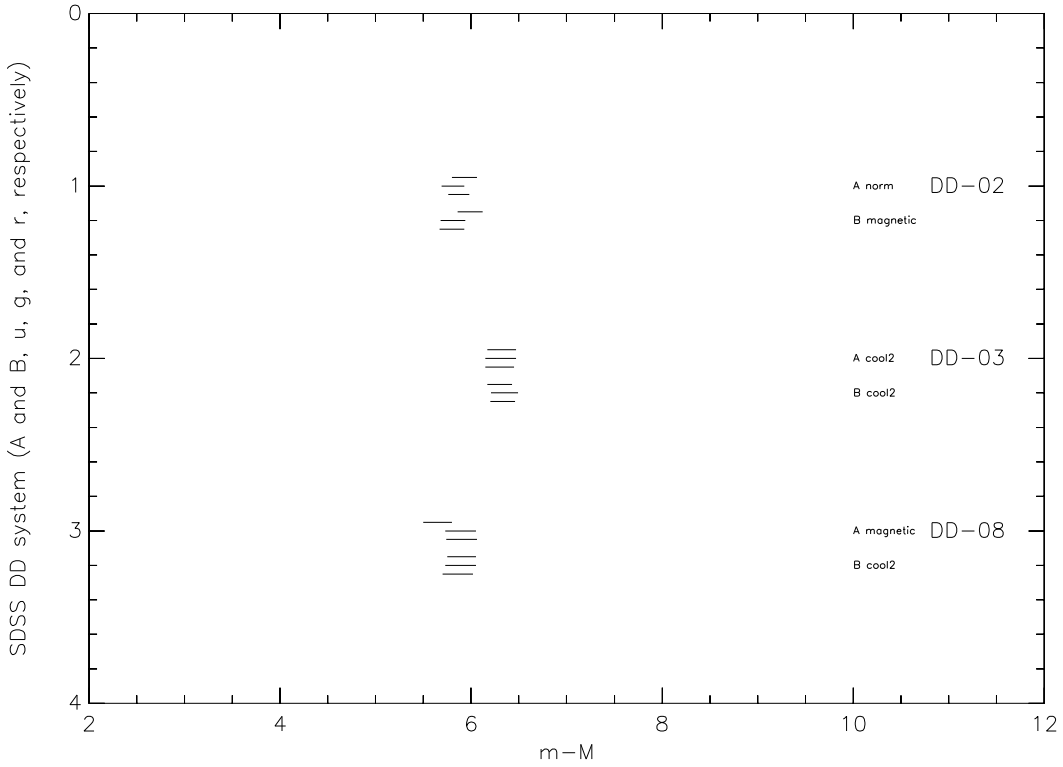


FIGURE 3.6: Distance Modulii (m - M) for DD systems with two cool components or a HFMWD system (u, g, r), with cool component mass corrections / distance matching

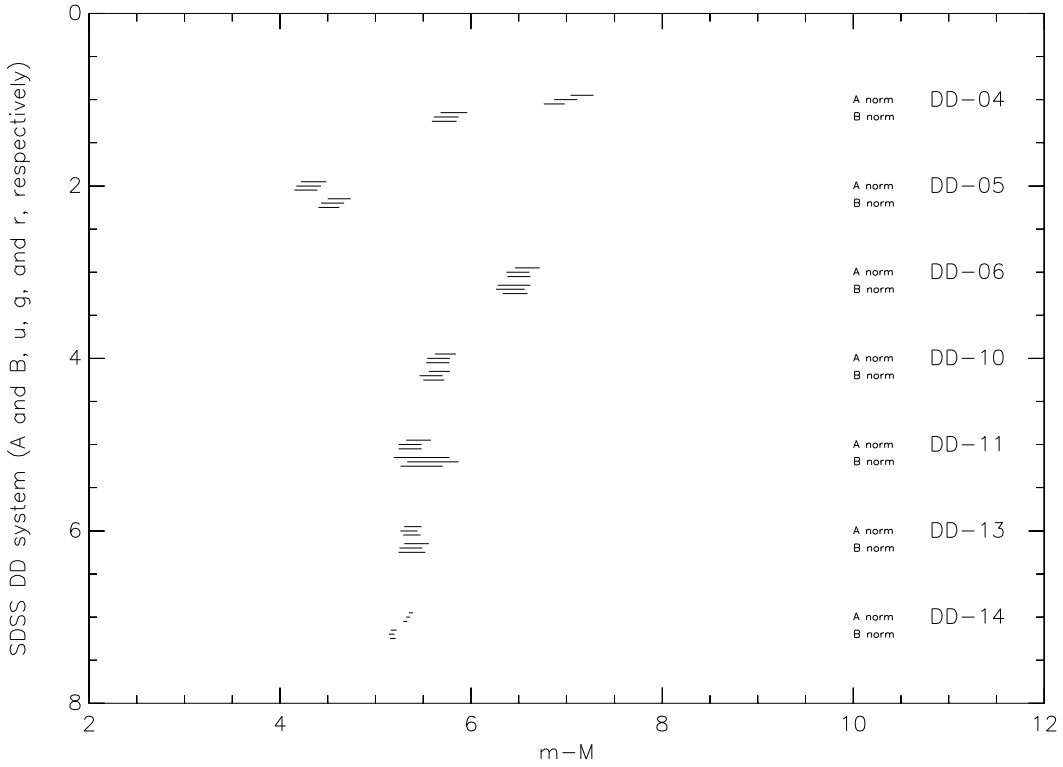


FIGURE 3.7: Distance Modulii (m - M) for DD systems with hot components (u, g, r)

Table 3.1: Double Degenerate Data - Results

WD	$\log g$	$\log g$ err	T_{eff} (K)	T_{eff} (K) err	τ_c (Myr)	τ_c (Myr) err	M_{final} ($M_\odot$)	M_{final} ($M_\odot$) err	abs u	abs g	abs r	abs u err	abs g err	abs r err	dist	dist err	tang vel (km/s)	tang vel (km/s) err
DD-01A	7.90	0.10	10648	245	419	59	0.510	0.068	12.23	11.83	11.94	0.11	0.13	0.13	7.06	0.12	44.8	9.1
DD-01B	7.97	0.07	19199	442	76	14	0.605	0.039	10.68	10.65	11.00	0.12	0.12	0.11	7.06	0.12	42.3	9.3
DD-02A	8.13	0.07	17285	398	157	23	0.695	0.042	11.21	11.1	11.42	0.12	0.12	0.11	5.81	0.12	19.4	11.7
DD-02B	8.67	0.09	18000	1000	354	48	1.027	0.047	11.99	11.95	12.26	0.13	0.13	0.13	5.81	0.13	20.6	11.6
DD-03A	8.19	0.08	10008	230	795	108	0.717	0.051	12.84	12.5	12.56	0.15	0.16	0.15	6.32	0.16	43.8	4.7
DD-03B	7.99	0.09	11222	258	440	59	0.598	0.052	12.18	11.81	11.94	0.13	0.14	0.13	6.36	0.14	45.5	5.1
DD-04A	8.16	0.07	13987	322	310	39	0.707	0.044	11.86	11.58	11.81	0.12	0.12	0.11	6.99	0.12	33.0	5.5
DD-04B	8.45	0.07	28067	646	54	13	0.911	0.044	10.34	10.62	11.06	0.14	0.13	0.13	5.74	0.14	20.2	2.8
DD-05A	8.37	0.07	19020	438	180	27	0.849	0.045	11.31	11.30	11.64	0.13	0.13	0.12	4.30	0.13	36.1	0.2
DD-05B	8.07	0.07	13950	321	273	34	0.652	0.042	11.75	11.46	11.69	0.12	0.12	0.11	4.55	0.12	31.5	0.8
DD-06A	8.06	0.07	24788	570	27	7	0.668	0.040	10.10	10.27	10.69	0.13	0.12	0.12	6.49	0.12	36.2	11.8
DD-06B	8.06	0.07	18502	426	108	19	0.655	0.041	10.91	10.85	11.19	0.12	0.12	0.11	6.41	0.15	35.3	11.4
DD-07A	8.00	0.07	33444	769	6	1	0.653	0.038	9.11	9.47	9.96	0.13	0.12	0.12	6.17	0.13	49.1	6.6
DD-07B	7.98	0.10	10393	239	540	79	0.597	0.069	12.41	12.03	12.12	0.11	0.12	0.12	6.17	0.12	47.7	6.8
DD-08A	8.03	0.18	9500	500	715	137	0.614	0.113	12.81	12.45	12.49	0.15	0.16	0.16	5.89	0.16	54.8	0.4
DD-08B	8.31	0.08	10492	241	849	110	0.796	0.052	12.84	12.51	12.60	0.15	0.16	0.15	5.89	0.16	43.7	0.4
DD-09A	7.89	0.09	11543	266	360	45	0.542	0.049	11.99	11.60	11.75	0.13	0.14	0.13	6.47	0.14	47.1	2.6
DD-09B	8.94	0.07	16037	369	893	123	1.173	0.032	12.80	12.72	12.97	0.15	0.15	0.15	5.56	0.15	26.1	2.0
DD-10A	8.43	0.07	13072	301	574	76	0.879	0.045	12.40	12.14	12.33	0.11	0.12	0.12	5.66	0.12	20.6	3.8
DD-10B	8.24	0.07	12758	293	450	57	0.755	0.045	12.19	11.90	12.08	0.11	0.12	0.11	5.59	0.12	20.5	3.8
DD-11A	8.19	0.07	20000	460	107	19	0.737	0.044	10.89	10.91	11.27	0.13	0.12	0.12	5.37	0.12	46.6	7.2
DD-11B	8.45	0.07	14037	323	489	65	0.893	0.045	12.27	12.05	12.26	0.12	0.13	0.12	5.61	0.27	43.5	5.2
DD-12A	8.02	0.07	15834	364	172	24	0.626	0.041	11.30	11.11	11.40	0.12	0.12	0.11	5.30	0.12	17.9	2.4
DD-12B	8.03	0.10	11453	263	449	93	0.631	0.083	12.18	11.81	11.96	0.12	0.14	0.14	5.30	0.12	21.9	2.2

3.3 IFMR probing

Systems with one or more magnetic components, which are less accurately modelled, have been isolated in the analysis of system age conformance. DD-04 and DD-09, which result in noticable distance age discrepancies have also been isolated in this analysis.

See Table 3.2 for a list of the published IFMR forms used for this comparison (including their linear gradient m and y intersection c parameters). This table also contains a summary of the DD system age conformance resultant from their respective forms, as described in the following paragraphs.

This table displays statistics calculated from a monte carlo generated distribution of 25000 hypothetical WDs with parameters selected within their error limits, as discussed in Section 3.1.6. These results rely upon a numerical (rather than analytical) propagation of their errors in derivation of their initial (main sequence progenitor) age/masses and therefore system age estimates. The code used to perform this method is described in Appendix Section A.6.2.

Firstly, this table displays the number of DD systems determined to have components with a common system age; [1] using an independent system age comparison, and [2] using an interdependent/covariant system age difference derivation. This section of the table was generated using my DD Data spreadsheet referenced in Appendix Section A.7.1. Secondarily, this table displays the median system age difference, averaged across all systems, and the number of DD systems best satisfied by the published IFMR (i.e. having the lowest system age difference with the given published IFMR).

Figures 3.12 and 3.13 show plots of the system age difference distribution for each DD system, based on the generated monte carlo distribution of 25000 hypothetical WDs where parameters are selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), and dashed (Williams et al. 2009). Figures 3.15, 3.16, and 3.17 each show plots of the system age difference distributions for all DD systems based on a particular published IFMR (Dobbie et al. (2006), Kalirai et al. (2008), and Williams et al. (2009) respectively). Plots of the systems ages are shown in Figures 3.9, 3.10, and 3.11, each assuming a unique form of the IFMR published in the literature.

For reference, plots of the published IFMRs used in this analysis are given in Figure 3.8.

TABLE 3.2: Published forms of the IFMR (linear gradient m and y intersection c parameters). It displays statistics calculated from a monte carlo generated distribution of 25000 hypothetical WDs with parameters selected within their error limits. Firstly it displays the number of DD systems determined to have components with a common system age; [1] using an independent system age comparison, and [2] using the interdependent/covariant system age difference derivation - from Figures 3.9, 3.10, and 3.11. Secondarily, it displays the median system age difference, averaged across all systems, and the number of DD systems best satisfied by the published IFMR (i.e. having the lowest system age difference with the given published IFMR).

IFMR	m	m err	c	c err	DD t_{sys} common ([1])	DD t_{sys} common [2]	average t_{sys} diff	DD t_{sys} diff low
Dobbie et al. (2006)	0.133	0.01	0.289	0.051	11	5	438.72821	7
Kalirai et al. (2008)	0.109	0.007	0.394	0.025	9	6	971.28286	7
Williams et al. (2009)	0.129	0.004	0.339	0.015	8	5	734.30429	0

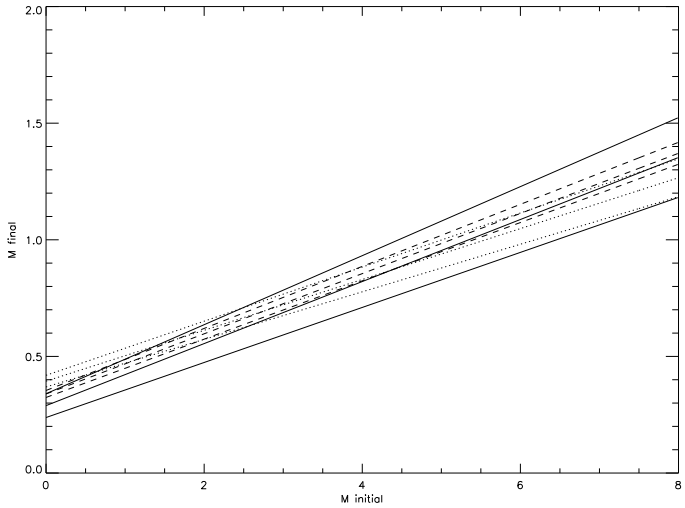


FIGURE 3.8: Plots of all published IFMRs used in this analysis; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009).

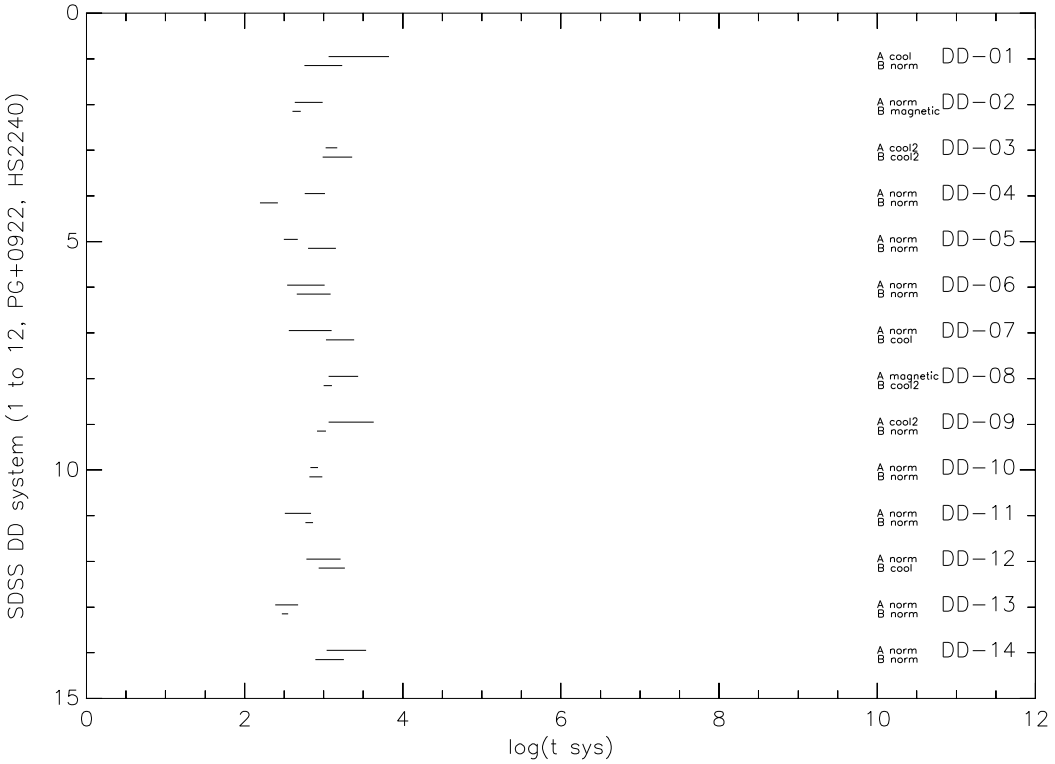


FIGURE 3.9: Calculated System Age for SDSS DD systems, using known IFMR from Dobbie et al. (2006).

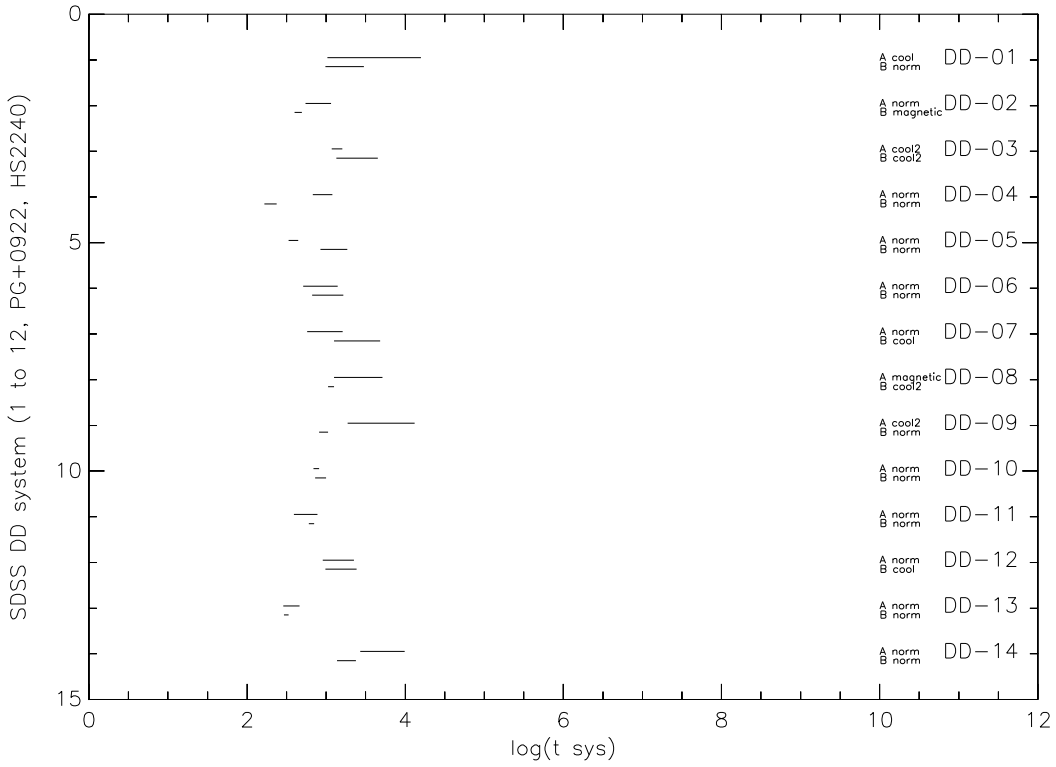


FIGURE 3.10: Calculated System Age for SDSS DD systems, using known IFMR from Kalirai et al. (2008).

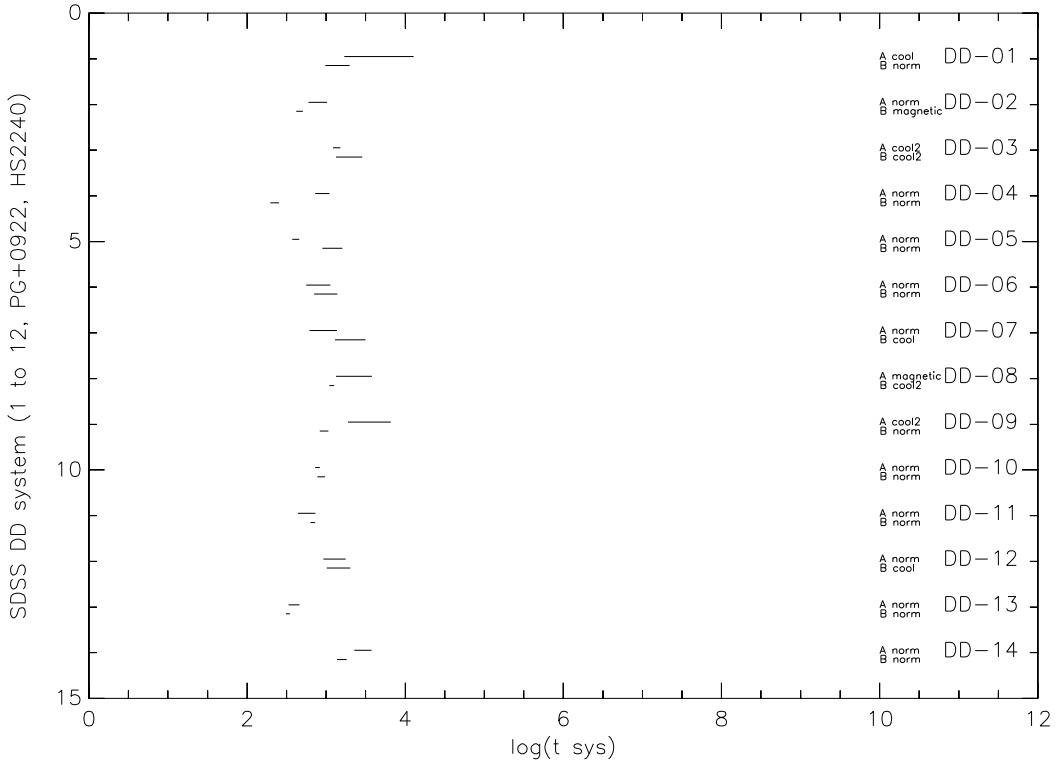


FIGURE 3.11: Calculated System Age for SDSS DD systems, using known IFMR from Williams et al. (2009).

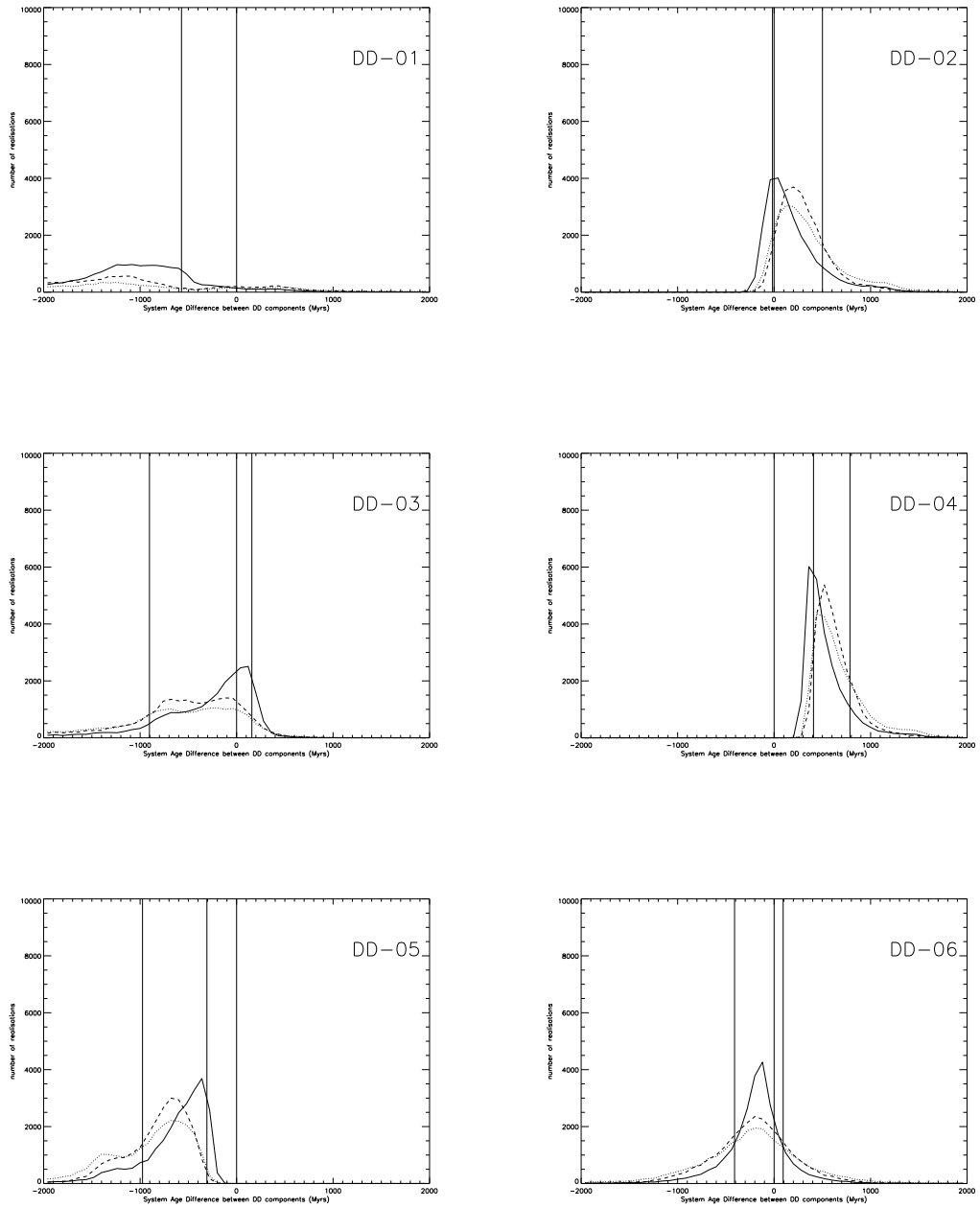


FIGURE 3.12: Plots of the system age difference distribution for DD-01 to DD-06, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009).

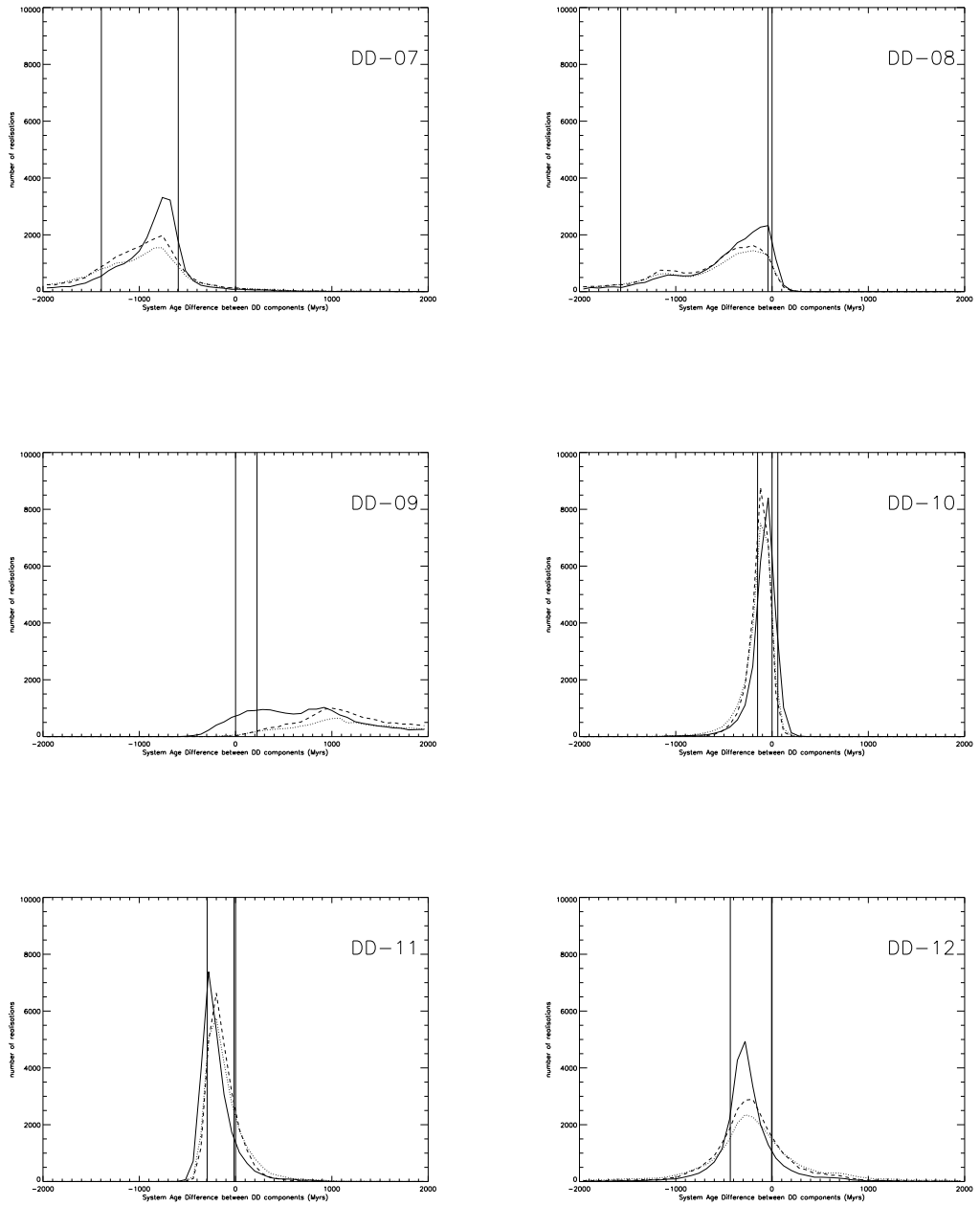


FIGURE 3.13: Plots of the system age difference distribution for DD-07 to DD-12, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009).

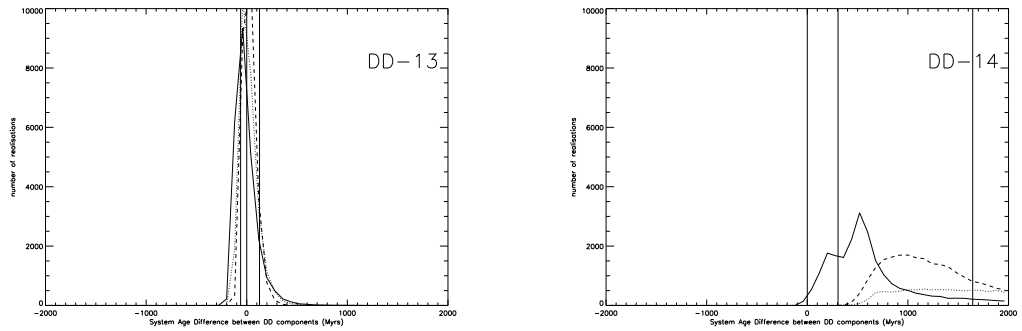


FIGURE 3.14: Plots of the system age difference distribution for external WDs PG0922+162 (DD-13) and HS2240+125A (DD-14) respectively, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For each plot a number of published IFMR forms are assumed; solid (Dobbie et al. 2006), dotted (Kalirai et al. 2008), dashed (Williams et al. 2009).

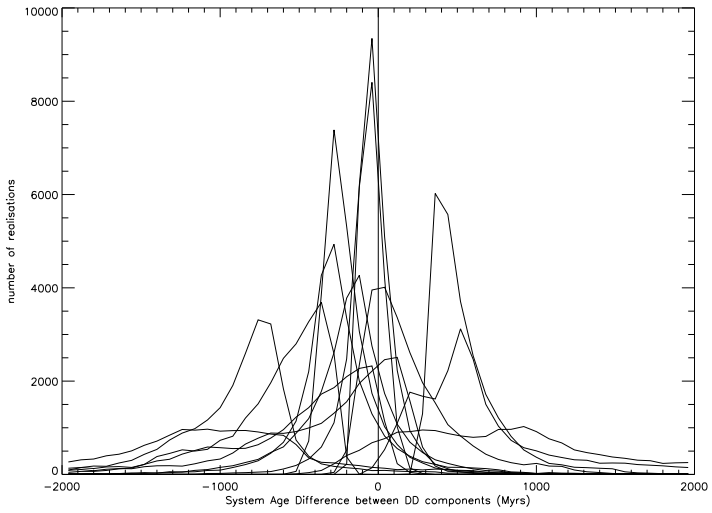


FIGURE 3.15: Plots of the system age difference distributions for all DD systems, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For this plot the following published IFMR form has been assumed; Dobbie et al. (2006).

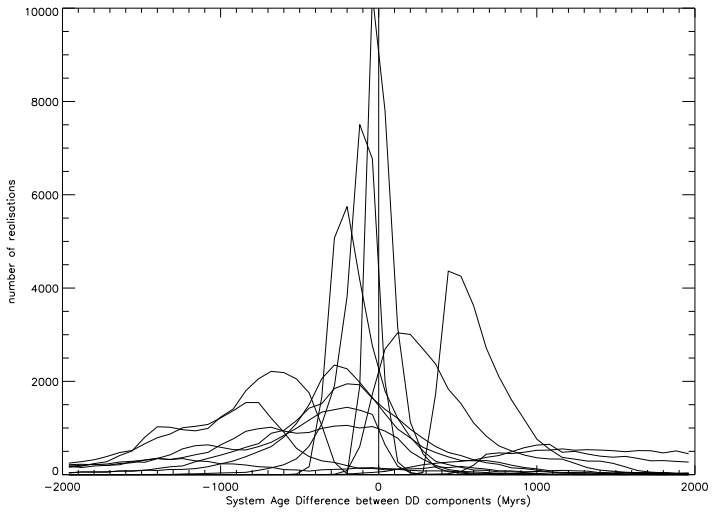


FIGURE 3.16: Plots of the system age difference distributions for all DD systems, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For this plot the following published IFMR form has been assumed; Kalirai et al. (2008).

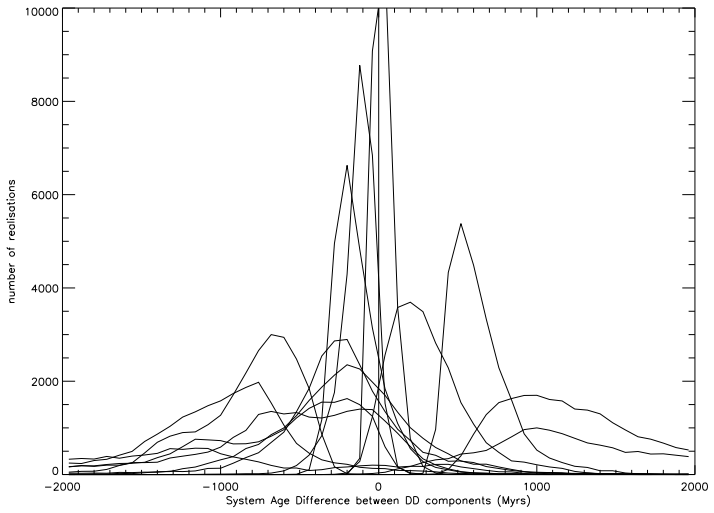


FIGURE 3.17: Plots of the system age difference distributions for all DD systems, based upon a generated monte carlo distribution of 25000 hypothetical WDs with parameters selected within their error limits. For this plot the following published IFMR form has been assumed; Williams et al. (2009).

Figures 3.18 and 3.19 show the WD component initial-final mass according to Dobbie et al. (2006) (A/B), for each DD system respectively.

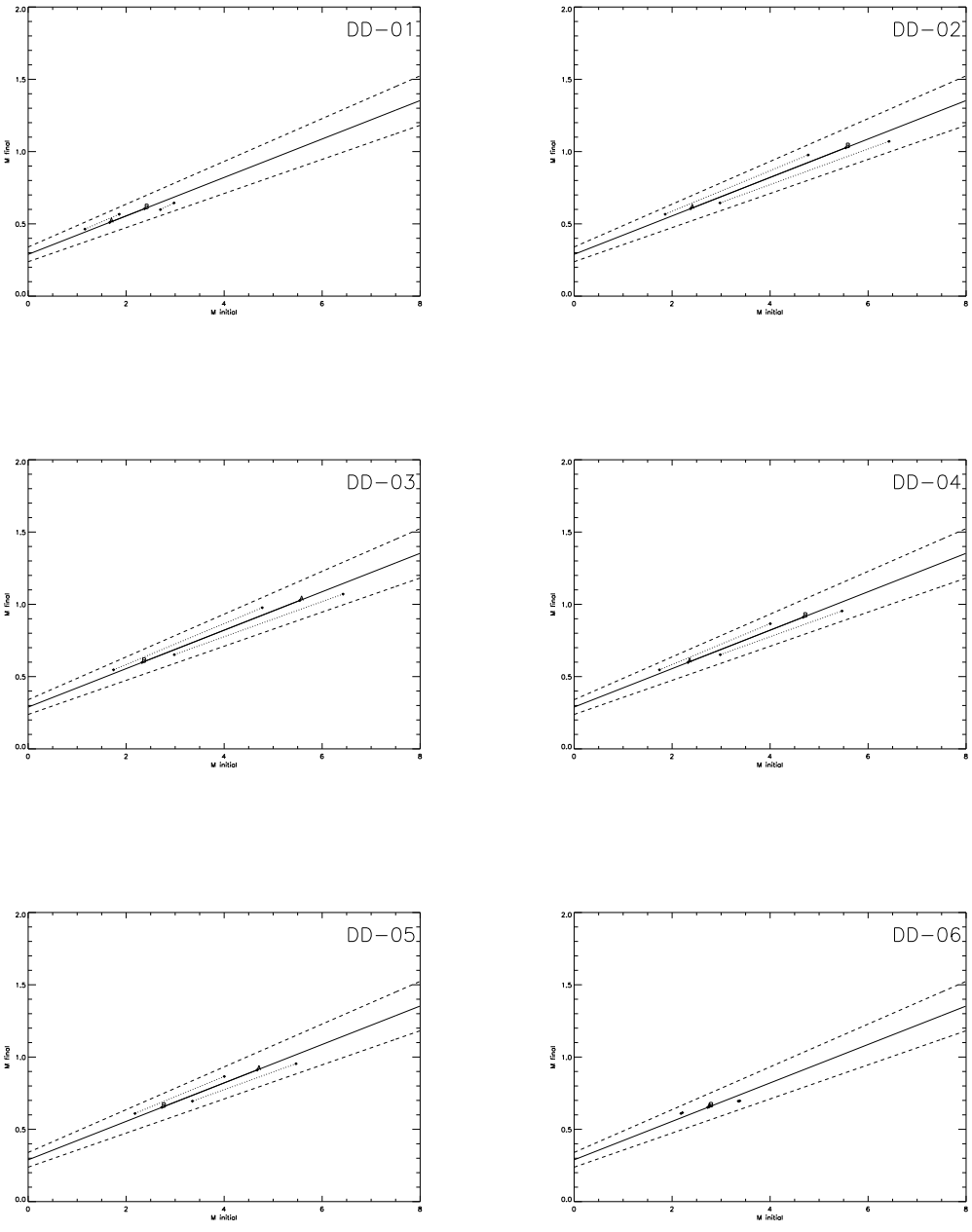


FIGURE 3.18: Plots of the initial-final mass for each component WD (of DD-01 through to DD-06) according to the (Dobbie et al. 2006) IFMR (A/B).

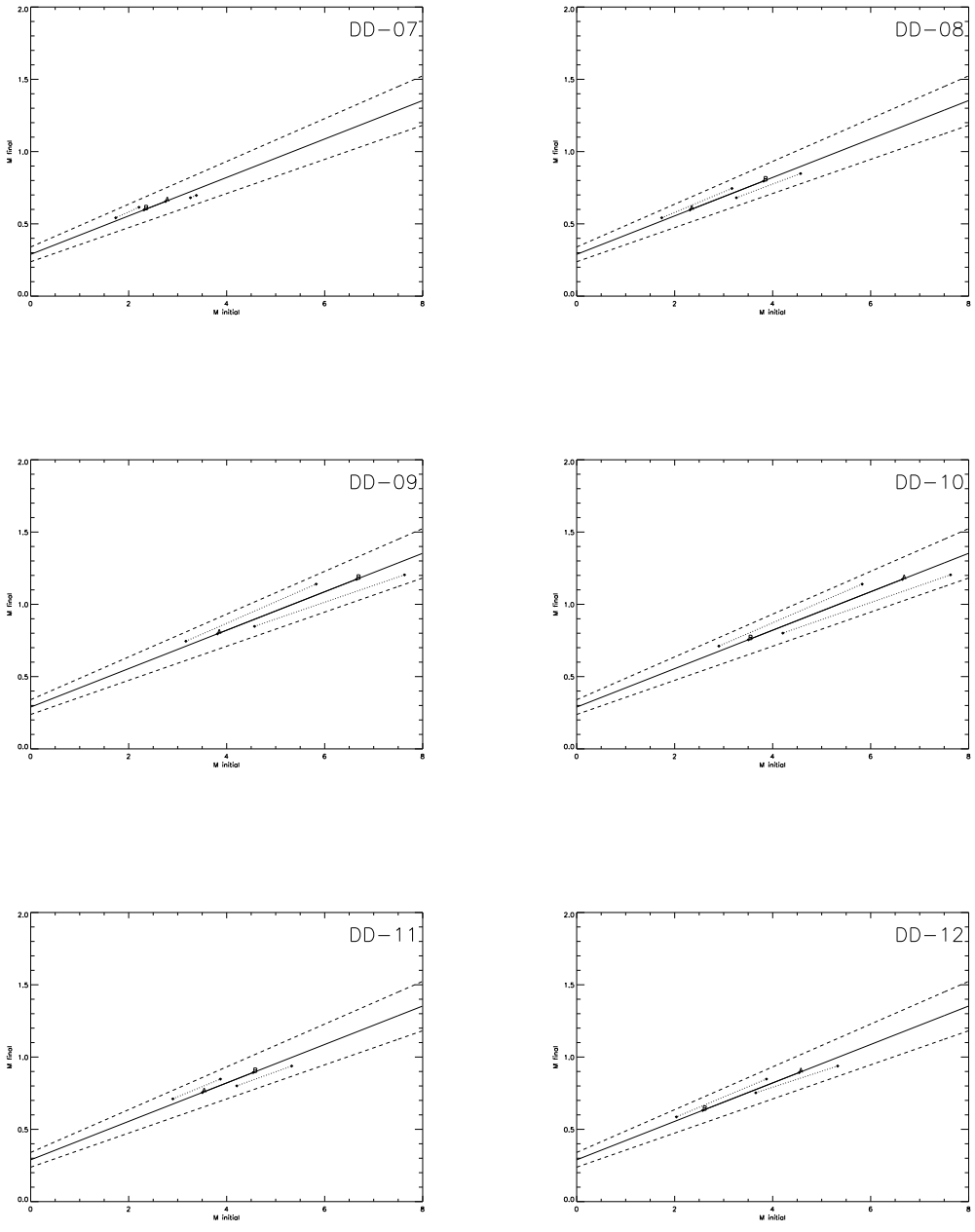


FIGURE 3.19: Plots of the initial-final mass for each component WD (of DD-07 through to DD-12) according to the (Dobbie et al. 2006) IFMR (A/B).

3.4 A preliminary investigation of the double degenerate mass distribution

The field WD mass distribution has been determined several times during the last few decades. These studies have applied corrections to account for the mass dependence of the volume being probed in a magnitude limited survey. If these corrections are not applied, mass distributions assembled from magnitude limited samples tend to over represent low mass WDs and under represent high mass WDs. The recent field WD mass distribution obtained by Kepler et al. (2007) using the SDSS DR4 and Liebert et al. (2005a) using the Palomar-Green (PG) Bright Quasar Survey both include $1/V_{\rm max}$ corrections, which are shown in Figures 3.20 and 3.21 respectively. The SDSS field WD mass distribution was compiled from 1733 WDs with $T_{\rm eff} > 12000K$, $g < 19$, while the PG based distribution was constructed from 347 WD stars with $T_{\rm eff} > 13000K$. These effective temperature limits are designed to negate the documented discrepancy in the reliability of the WD log g determinations at lower effective temperatures.

In generating my DD mass distribution, I have used all known spectroscopically studied DD systems that lie within the SDSS DR7 footprint and that pass my selection criteria (ie. separation, colour and magnitude limits). This includes binaries which were omitted from my follow-up work because they had already been observed spectroscopically and have had their parameters determined by other investigators ie. PG 0922 + 162A+B from the Palomar-Green Bright Quasar Survey (Finley and Koester 1997) and HS 2240 + 125A+B from the Hamburg Quasar Survey (Jordan et al. 1998a). Details on these additional DD systems are shown in Table 3.3. My distribution contains the 24 WDs from the double degenerate pairing which after detailed analysis I believe to be physically associated (ie, excluding DD-04 and DD-09), and has been binned at $0.1M_{\odot}$ intervals. This fits best with the uncertainties in my individual mass calculations, provides a number of reasonably well populated bins and at the same time provides fair resolution in mass. I have made no attempt to model the impact of WD mass on my sampling volume (ie. I have not applied a $1/V_{\rm max}$ correction) so if anything, high mass WDs should be under represented in my preliminary mass distribution.

I have plotted my mass distribution for WDs in wide double-degenerate systems in Figure 3.22. I have overplotted in this figure, the raw SDSS field WD mass distribution of Kepler et al. (2007), the $1/V_{\rm max}$ corrected SDSS WD mass distribution of the same investigators and also the volume corrected field WD mass distribution of Liebert et al. (2005a). A glance at this figure suggests that my sample is skewed towards higher WD masses than the field studies.

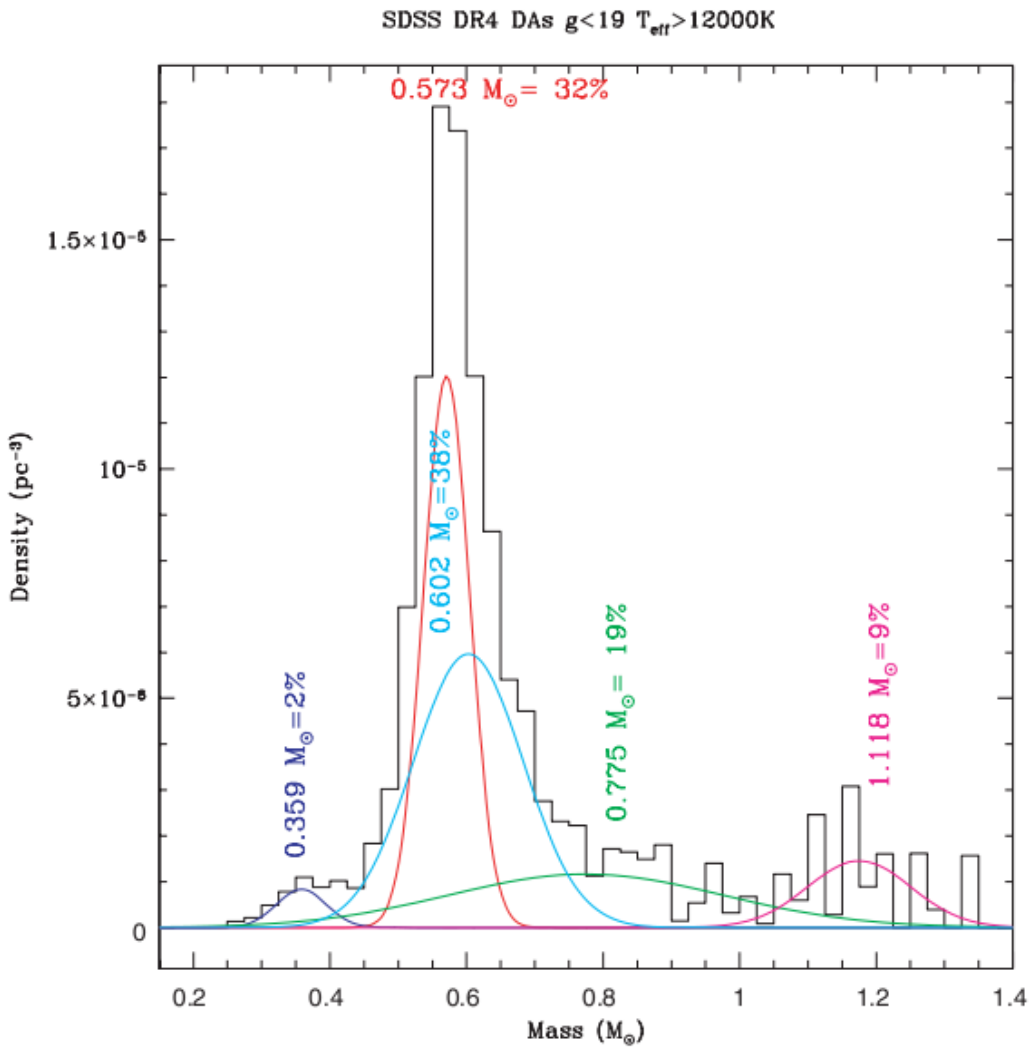


FIGURE 3.20: Kepler SDSS V_{max} Corrected Mass Distribution (Kepler et al. 2007)

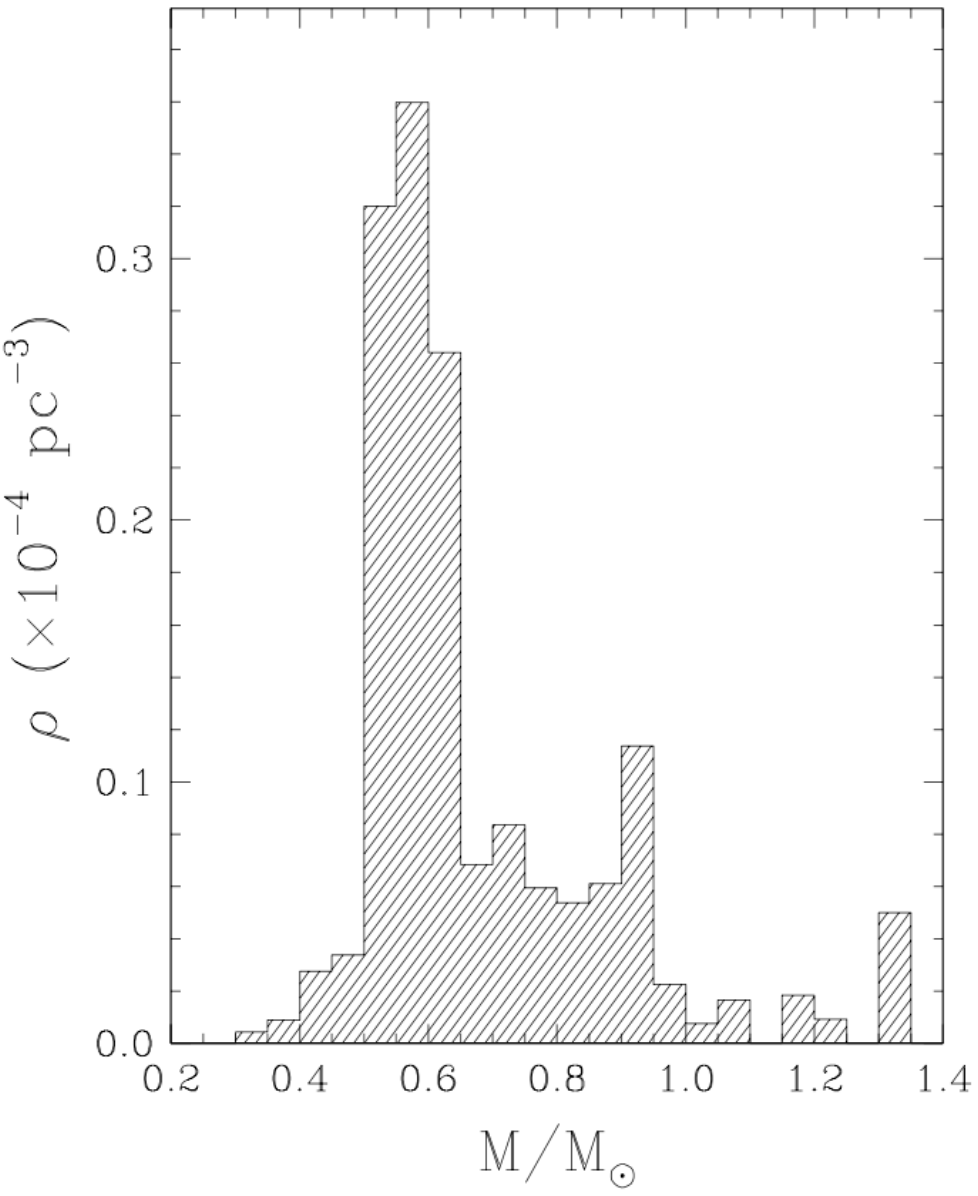


FIGURE 3.21: Liebert PG Survey V_{max} Corrected Mass Distribution (Liebert et al. 2005a)

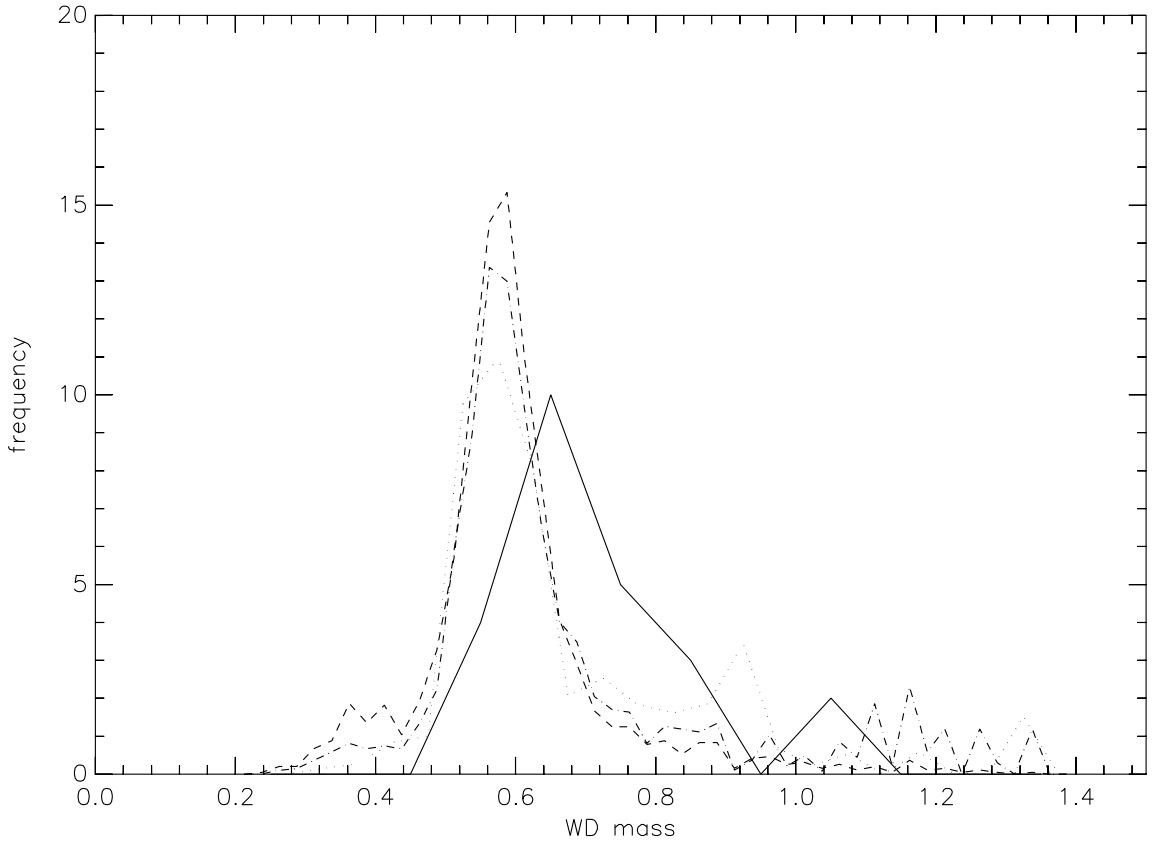


FIGURE 3.22: DD mass distribution (Solid) versus the PG Survey Vmax corrected mass distribution (Dotted), the Kepler SDSS DR4 WD mass distribution (Kepler et al. 2007) (Dashed), and the Kepler SDSS DR4 WD Vmax corrected mass distribution (Kepler et al. 2007) (Dash Dot)

TABLE 3.3: Additional (third party) Double Degenerate systems added to my mass distribution

WD	$\log g$	$\log g$ err	T_{eff} (K)	T_{eff} (K) err	τ_c (Myr)	τ_c (Myr) err	M_{final} ($M_\odot$)	M_{final} ($M_\odot$) err	abs u	abs g	abs r	abs u err	abs g err	abs r err	dist	dist err
PG0922+162A	8.27	0.05	22740	230	78	10	0.792	0.031	10.66	10.78	11.17	0.08	0.09	0.08	5.35	0.09
PG0922+162B	8.78	0.05	22130	220	248	24	1.094	0.026	11.62	11.76	12.13	0.1	0.1	0.1	5.38	0.12
HS2240+125A	7.86	0.01	15636	9	146	1	0.536	0.004	11.11	10.89	11.19	0.01	0.01	0.01	5.33	0.02
HS2240+125B	7.99	0.01	13935	112	244	7	0.604	0.007	11.64	11.33	11.57	0.03	0.03	0.02	5.17	0.03

3.4.1 Comparing the DD mass distribution to the field WD mass distribution

To calculate the likelihood that the mass distribution I obtain from the DD systems and the field WD mass distribution are drawn from the same underlying population, I have performed Kolmogorov-Smirnov (K-S) tests. In order to perform these comparisons I first took the respective field WD mass distributions (Kepler SDSS Survey WD mass distribution, Kepler SDSS Survey $V_{\rm max}$ corrected WD mass distribution and Liebert PG Survey $V_{\rm max}$ corrected WD mass distribution) and created a model of them using combinations of gaussians (or in the case of the SDSS distributions I lifted the gaussian model parameters directly from the Kepler paper). I then formed a set of "hypothetical" mass values representing WDs within the distribution using the number of WDs used to create the distribution originally (1733, 1733, and 347 respectively). I then compared this set of data with the observed WD masses from my survey using a K-S test routine written in IDL. This code is provided in the Appendix (Section A.6.4). I calculated the probabilities of the DD and the field WD mass distributions being the same as P$\sim 2.11E-6$, 0.00068, and 0.00371 for the Kepler SDSS Survey WD mass distribution, Kepler SDSS Survey $V_{\rm max}$ corrected WD mass distribution, and Liebert PG Survey $V_{\rm max}$ corrected WD mass distribution respectively. Thus considering that I have not applied $1/{\rm V_{max}}$ corrections, I conclude that the masses of my WDs in wide DD systems are generally higher than those of field WDs.

In my sample, one component (DD-09B) has a very high mass of $1.17M_{\odot}$, and 6 components have $M > 0.82M_{\odot}$. 25% of my components have high masses whereas only 5% of the Gianninas et al. (2011) white dwarfs have similar high masses.

It is also evident that all of my high mass components also have $T_{eff} > 13000K$. The 9 cool components in the sample ($T_{eff} < 13000K$) have an average mass of $0.64M_{\odot}$, whereas the 15 hot components ($T_{eff} > 13000K$) have an average mass of $0.78M_{\odot}$. Hotter white dwarfs should however have lower mass than cooler white dwarfs (Kleinman et al. 2004). Gianninas et al. (2011) has found that hot white dwarfs ($T_{eff} > 13000K$) have an average mass of $0.661M_{\odot}$, with a dispersion of $0.160M_{\odot}$, where as as cool white dwarfs ($T_{eff} < 13000K$) have an average mass of $0.638M_{\odot}$, with a dispersion of $0.145M_{\odot}$.

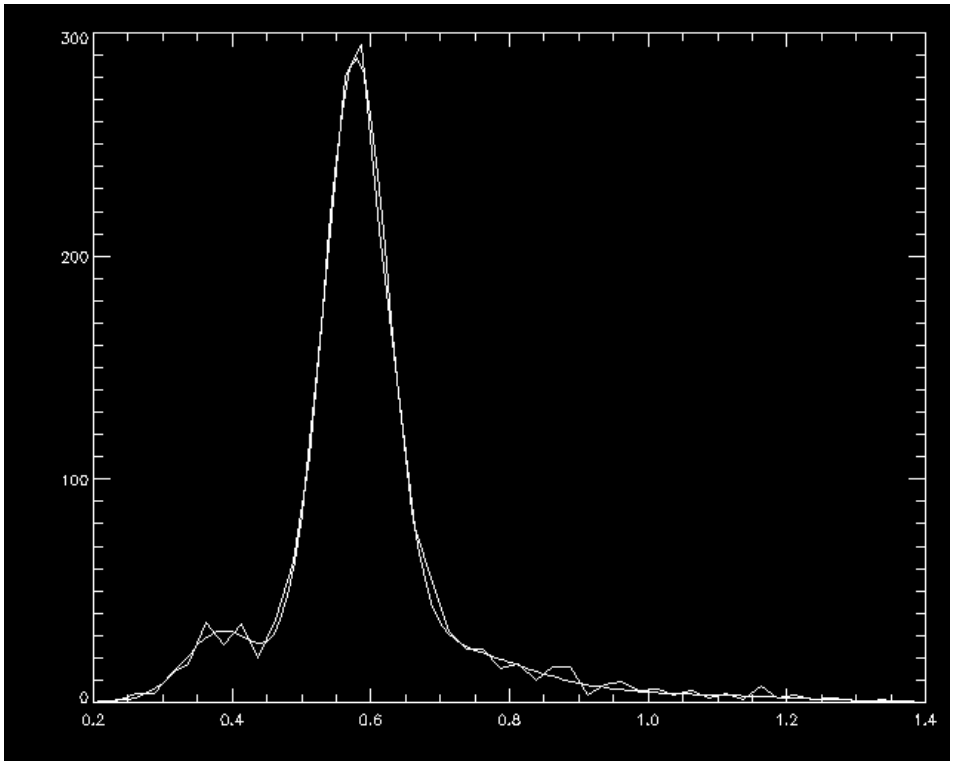


FIGURE 3.23: Kepler SDSS Survey Mass Distribution Modelled With Gaussians

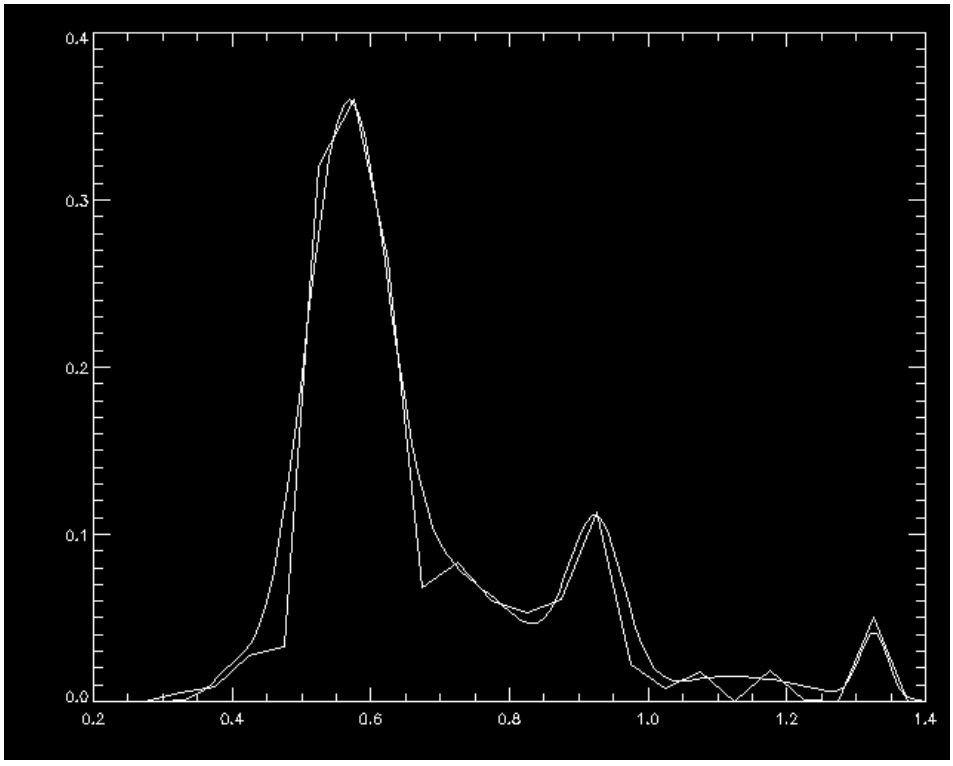


FIGURE 3.24: PG Survey $V_{\max}$ Corrected Mass Distribution Modelled With Gaussians

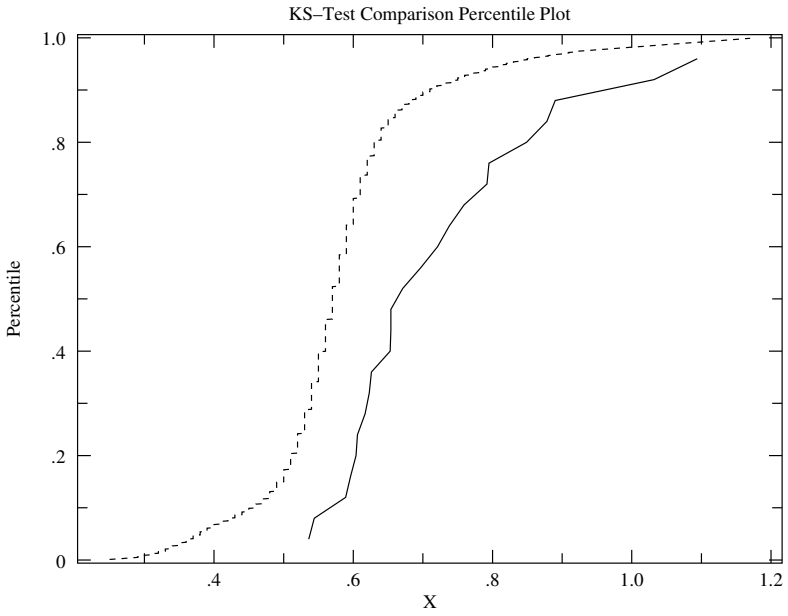


FIGURE 3.25: KS Test Method 2 - Comparison of DD masses with masses derived from Kepler theoretical SDSS WD mass distribution modelled with Gaussians

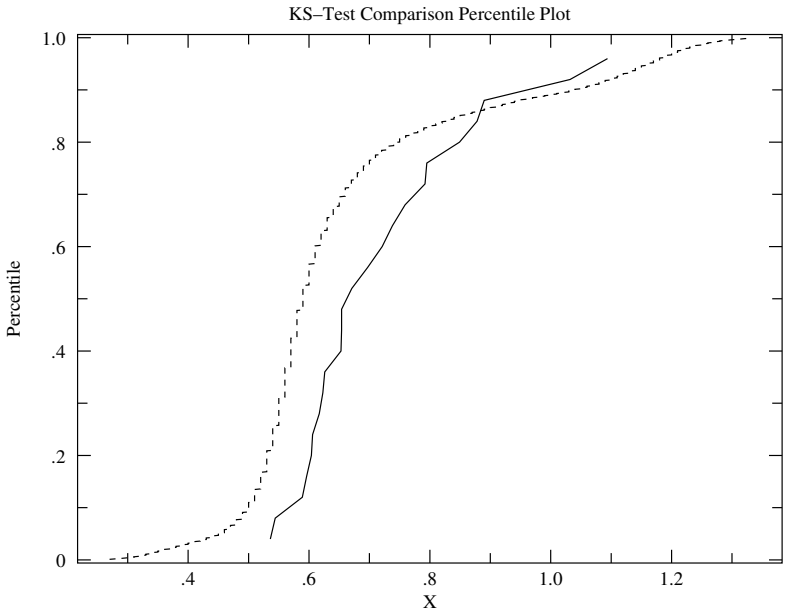


FIGURE 3.26: KS Test Method 6 - Comparison of DD masses with masses derived from Kepler theoretical $V_{\max}$ corrected SDSS WD mass distribution modelled with Gaussians

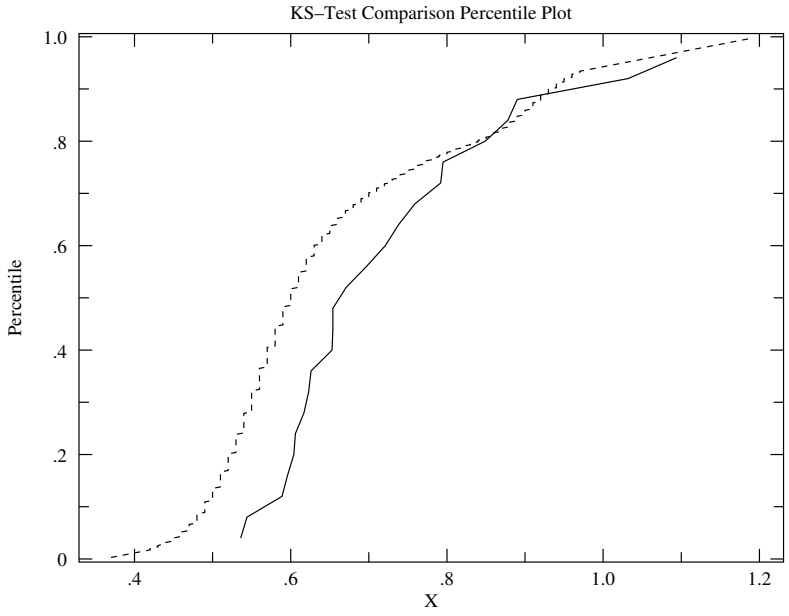


FIGURE 3.27: KS Test Method 5 - Comparison of DD masses with masses derived from Liebert theoretical PG Survey $V_{\max}$ corrected mass distribution modelled with Gaussians

3.4.2 Why might the DD mass distribution be skewed to higher mass?

There are a number of possible reasons as to why my preliminary mass distribution for the components of wide DD systems may be skewed to higher mass than that of the field WD population. First, there are differences in the binary fraction of main sequence stars as a function of mass. Stars of earlier spectral types are observed to occur more often in binary systems than those of later spectral type (e.g Sterzik and Durisen 2004). For example, Kouwenhoven et al. (2005) have determined the binary fraction of B, A and F stars in the Upper Scorpius Association to be ∼60%, ∼40% and ∼20%, respectively. Thus since in general, higher mass stars lead to more massive WDs, the binary fraction of WDs may also shown this mass dependence. Second, it could be expected that for my systems there are proportionately fewer close primordial pairings ($\approx 10-20\%$ percent of my DD systems could be triple systems) than for a sample of apparently single field objects where the overall binary fraction is typically ∼40-50% (e.g Kouwenhoven et al. 2005). Therefore within the field population, a higher fraction of these systems may have interacted with a companion and experienced enhanced massloss at earlier evolutionary phases, which can lead to the formation of a WD of lower mass than that from a single star of corresponding initial mass. A third possible reason is that orbital separation likely increases due to the loss of mass from the system as stars become WDs. Since intermediate mass stars, which produce high mass WDs, lose a much higher proportion of their initial mass during their evolution than lower mass stars, primordial binaries with components of this mass likely lead to on average, WD pairs with larger final separations. For example, by applying Jean's theorem which states that the product of the total mass and the semi-major axis of the orbit remain constant (e.g. Iben 2000), I find that the transformation of a $M_i = 1.8M_\odot$ main sequence star into a $M_f = 0.6M_\odot$ WD would result in an orbital expansion factor of ~ 3, whereas a $M_i = 4.5M_\odot$ main sequence star evolving into a $M_f = 0.9M_\odot$ WD (based on the recent Casewell et al. (2009) IFMR), would lead to an expansion in the orbital separation by a factor of ∼5. Thus assuming a similar primordial distribution of orbital separations, I estimate that on an object by object basis a double degenerate system containing two $0.9M_\odot$ components is ∼3 times more likely to be unearthed by my target selection criteria than a binary containing two $0.6M_\odot$ components.

3.5 A preliminary investigation of the double degenerate magnetic WD distribution

I also found an unusually high percentage of magnetic stars. Gianninas et al. (2011) found 25 magnetic (or suspected magnetic) stars out of about 1300 stars studied (2%). In my sample, 4 out of 24 of my stars were found to be magnetic (DD-02A, DD-02B, DD-04A, and DD-08A), an order of magnitude higher.

"So it is plain that Science must mean the most accurate of all Knowledge; but if so, then the Scientific man must not merely know the deductions from the First Principles but be in possession of truth respecting the First Principles.."

from Aristotle's Nicomachean Ethics

4 Moving forward

4.1 Further optical spectroscopy

4.1.1 Non-magnetic systems

While I have secured long-slit optical spectroscopy for 12 systems, there are 25 further pairs with angular separations $<10''$ awaiting follow-up observations. A key goal of this further work is to identify systems with extreme mass ratios e.g. M_1=0.7M$_\odot$, M_2=1.3M$_\odot$. Unfortunately I have not yet confirmed a physical system in which one component has a mass M>1.1M$_\odot$, the theoretical dividing line between CO and ONe core WDs. A substantial population of these ultramassive (M$_{WD}\gtrsim$1.05M$_\odot$) WDs has been recognised in the field for well over a decade (e.g. Fleming et al. 1986, Marsh et al. 1997, Vennes et al. 1997) but their origins remain a matter of considerable debate. The results of population synthesis calculations suggest that the majority of these objects might be produced through the merging of the lower mass components of close binary systems, at least at the higher galactic latitudes typical of the Palomar Green and Sloan Digital Sky surveys (Liebert et al. 2005a). However, models of this merging process encounter difficulties in explaining the observed space density of nearby, hot, ultramassive white dwarfs unless they assume the formation rate of close binaries in the Galaxy is somewhat higher than expected and the viscosity of the merger interior is very low (Segretain et al. 1997). Liebert et al. (2005a) have noted that the spatial distribution of the 28 most massive white dwarfs detected in the extreme ultraviolet (EUV) sky surveys displays similarities with that of the younger stellar populations in the vicinity of Gould's Belt (with the caveat that the EUV source distribution is affected by the patchy opacity of the local ISM). The identification of these objects in wide double-degenerate systems would allow me to constrain their pre-WD lifetimes and examine whether these are consistent (or otherwise) with them having formed from single heavy-weight intermediate mass stars.

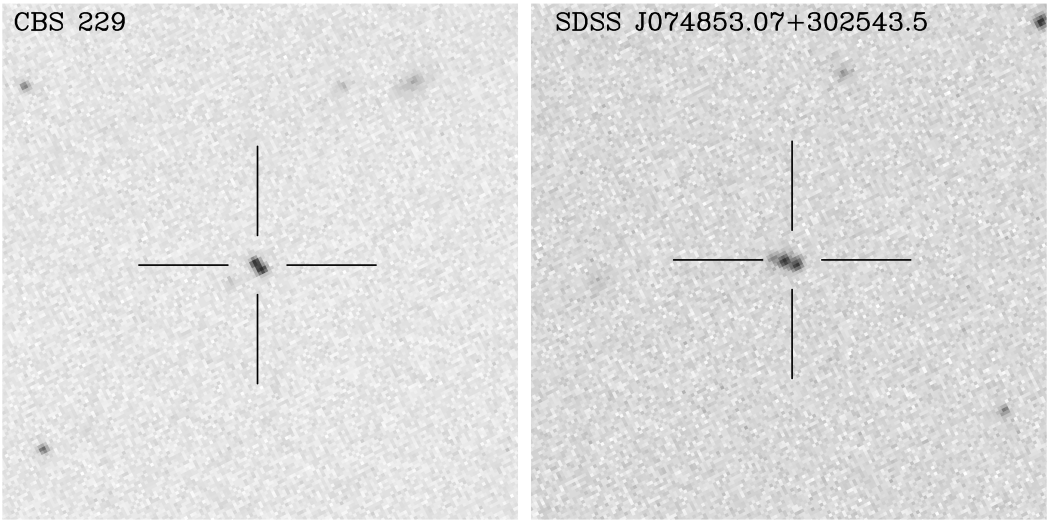


FIGURE 4.1: SDSS z band image of two candidate spatially resolved magnetic/non-magnetic double-degenerate systems, CBS 229 (Gianninas et al. 2009) and SDSS J074853.07+302543.5. Images are approximately 1'×1' with N at the top and E to the left.

An enlarged sample of spectroscopically studied wide double-degenerate systems would also be useful for searching for WDs with atmospheres dominated by elements other than hydrogen e.g. helium or carbon. There is a suggestion from open clusters that helium dominated atmospheres are extremely rare in young and intermediate age populations, $\tau \lesssim 1$Gyr (Kalirai et al. 2005b). WDs of this atmospheric composition are believed to form as a result of a late-helium shell flash during the final phases of the AGB, which consumes all hydrogen rich material. The possibility that this mechanism is mass dependent can be tested by studying DB WDs in wide binaries, where constraints on the system age are available from the companion star.

4.1.2 Systems containing HFMWDs

To help shed further light on the formation of HFMWDs and the impact of magnetic fields on stellar evolution, it is also desirable to enlarge the sample of these objects that are located in wide double-degenerate binaries (or in nearby open clusters) so that any trends in their cooling times (with respect to non-magnetic WDs) can be clearly identified. At least one of my candidate systems awaiting detailed spectroscopic follow-up,

SDSS J074853.07+302543.5 and SDSS J074852.95+302543, which have an angular separation of ∼1.6" (see Figure 4.1, right), seems likely to be a DAH+WD system, since the SDSS spectrum of the former shows clear evidence of Zeeman split Balmer lines. I also note that the previously known hot DA+DAH system CBS 229, which lies within the SDSS DR7 footprint, is probably a wide binary. It was identified as an unresolved pairing during the course of a spectroscopic survey of bright WDs ($V \leq 17.5$) drawn from the catalogue of McCook and Sion (1999). Gianninas et al. (2009) have performed a preliminary analysis of a composite spectrum of this system and find that the non-magnetic component has $T_{\rm eff} \approx 15000$K and log $g \approx 8.5$, corresponding to a mass of $M \approx 0.9 M_{\odot}$. The shape of their deblended spectrum of the DAH suggests that the two objects have similar effective temperatures. My examination of the SDSS imaging reveals that the components are resolved into two photocenters with a projected separation of ∼1.3" (see Figure 4.1, left). From the Gianninas et al. (2009) parameters for the DA and its magnitude from the SDSS z imaging (the band in which the objects are most clearly resolved), I provisionally estimate a distance to this binary of d∼140pc and a projected orbital separation of a∼180AU. Thus, despite both components having $T_{\rm eff} \gtrsim 9000$K, CBS 229 appears to be a wide magnetic/non-magnetic double-degenerate system which escaped detection by my survey. This is probably due to the u band magnitude measurement for the NE component which appears to be anomalous. Undoubtedly further magnetic/non-magnetic pairings remain to be identified within my existing sample of candidate DD systems.

Thus I find a number of scientific reasons for obtaining spectroscopy of more of my candidate DD systems. A number of the systems for which I have already secured spectroscopy were observed during "band 3" weather conditions, ie. poorer seeing and poorer sky transparency. A significant proportion of the remaining systems will require better observing conditions since they are fainter and/or of lower angular separation. The number of candidate wide DD systems available for follow-up will shortly increase when southern sky survey imaging datasets such as Skymapper and VST ATLAS become publicly available. As part of a preliminary investigation, I have performed a basic replication of the target selection process on the SDSS DR8 dataset. Due to time constraints, detailed error checking has not been performed as yet on this initial collection of additional DD candidate data. See Figure 4.2 for a list of these DR8 specific candidates.

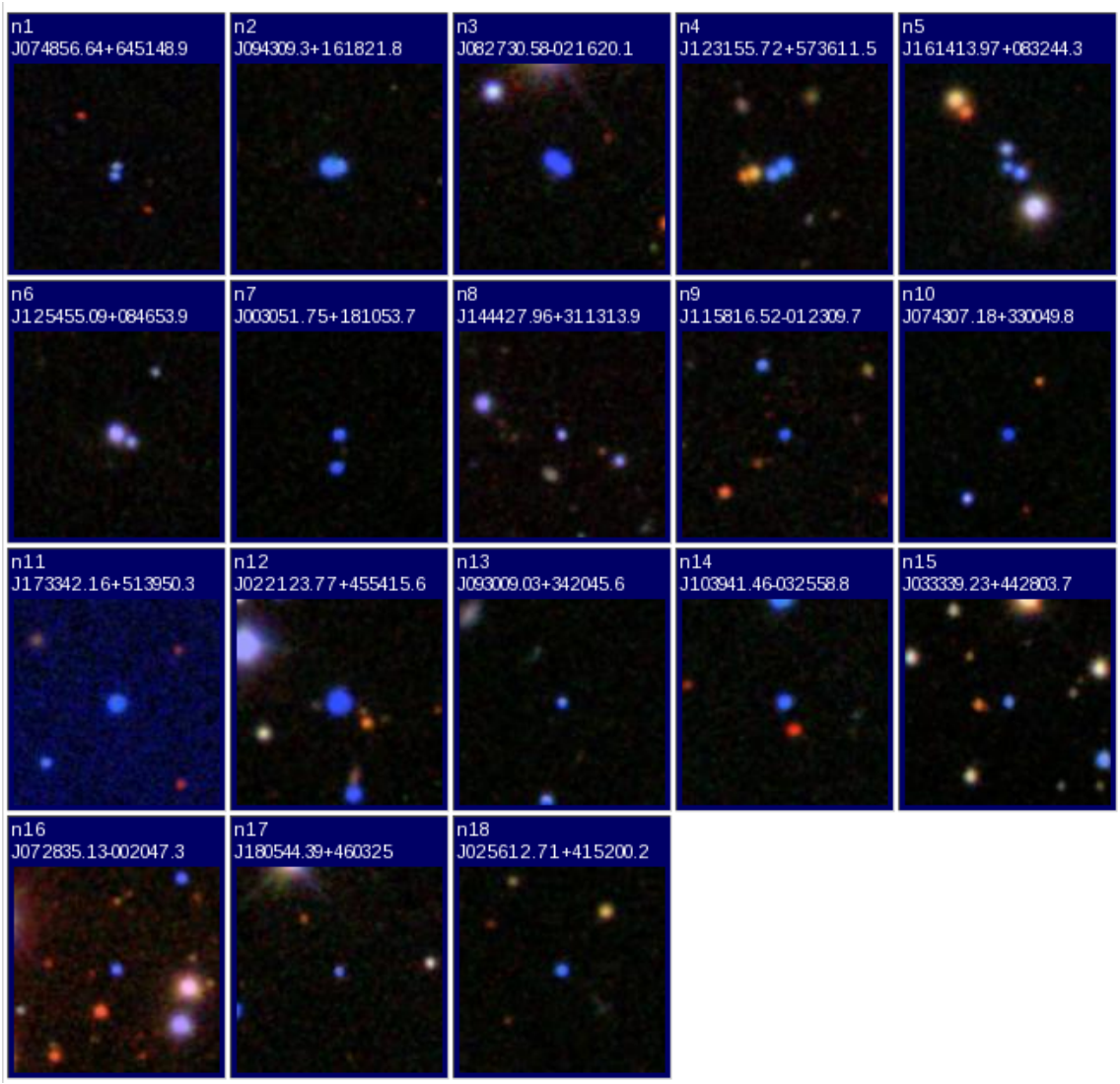


FIGURE 4.2: DR8 Only DD Candidates

Table 4.1: A table of my preliminary IFMR linear parameter derivations, [1] for all systems, and [2] excluding systems with an uncommon distance between components (DD-04 and DD-09)

IFMR	m	m err	c	c err
Dobbie et al. (2006)	0.133	0.015	0.289	0.051
DD IFMR derivation (all systems [1])	0.155	0.017	0.167	0.037
DD IFMR derivation (excluding [2])	0.159	0.018	0.149	0.03

4.2 Preliminary investigation of IFMR form derivation

In addition to probing the IFMR, where system ages were confirmed as consistent (or inconsistent) with established forms of the IFMR, a preliminary investigation was made into the ability to independently derive the form of the IFMR using DD systems. A basic constraint of the form of the IFMR was derived using my DD data, by equalising the system age of each system across a range of linear IFMR forms (m and c parameters).

Figure 4.3 shows the derivation of the form of the IFMR for each DD system individually, by selecting a linear IFMR form (m and c parameters) resulting in a common system age. These plots also include derivations taking into account the errors in the masses and cooling ages of the components, although they have not been formally treated at present (they are interdependent). A statistical derivation of the form of the IFMR is shown by averaging these parameters (with error bars showing min/max mass assumption extremes). This graph also shows the Dobbie et al. (2006) linear IFMR form for comparison (with cited errors).

Figure 4.4 shows another derivation of the form of the IFMR by selecting a linear IFMR form (m and c parameters) resulting in a common system age, excluding systems with an uncommon distance between components (DD-04 and DD-09).

Table 4.1 displays my preliminary IFMR linear parameter derivations, [1] for all systems, and [2] excluding systems with an uncommon distance between components (DD-04 and DD-09).

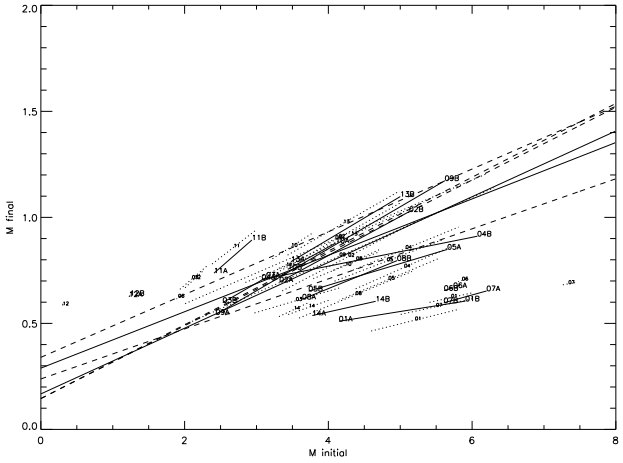


FIGURE 4.3: A plot of the derivation of the form of the IFMR for each DD system individually, by selecting a linear IFMR form (m and c parameters) resulting in a common system age. A statistical derivation of the form of the IFMR is shown by averaging these parameters (with error bars showing min/max mass/cooling age assumption extremes). This graph also shows the Dobbie et al. (2006) linear IFMR form for comparison (with cited errors).

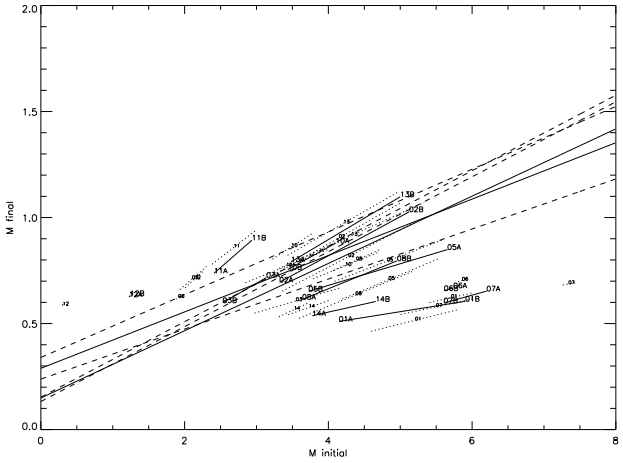


FIGURE 4.4: A second plot of the derivation of the form of the IFMR as in Figure 4.3 - excluding systems with an uncommon distance between components (DD-04 and DD-09).

4.3 Conclusion

In conclusion, I have performed a preliminary study of twelve candidate double-degenerate binaries. Blue Object Close Pairs (BOCPs) were identified from the Sloan Digital Sky Survey (SDSS) imaging data by applying selection criteria in software. Monte Carlo like simulations were performed to estimate the probability of BOCPs being mere chance alignments as a function of angular separation. Targets were prioritised for spectroscopic follow-up based upon weather, RA, magnitude and angular separation constraints, and astrometric analysis was performed on all these targets to determine if the components had (or did not have) consistent proper motions.

Spectroscopic data were obtained from a variety of telescopes, including the WHT, Gemini North, Gemini South, and the VLT ANTU, and reduced using IRAF scripts. I used existing WD model atmospheres, and grids of evolutionary models to analyse these data. The WD parameters of mass, $M_{\rm f}$, and cooling time, $t_{\rm cool}$, were used to probe the form of the stellar initial-final mass relation (IFMR). I have used existing estimates of the IFMR to confirm that binary components within the majority of my DD systems indicate a common system age. For those systems where these age estimates are inconsistent, allowing for uncertainties, I have suggested reasons for the disparity. I have concluded on the basis of distance estimates that the components in two out of the twelve systems are unlikely to be physically associated. For those systems which appear to be associated I have examined them in the context of recent estimates of the IFMR, finding that these provide a fair description of the transformation of intermediate mass stars into WDs ($M_{\rm init}$>2.5M$_{\odot}$), but that there is likely more scatter in the relation than assumed from the results of recent open cluster based studies, at least at lower initial/final masses (ie. $M_{\rm init}$<2.5M$_{\odot}$).

I have also generated a preliminary DD mass distribution containing all systems within my target selection with follow-up spectroscopic data (DD-01 through to DD-12, with PG0922+162 and HS2240+125 also), excluding systems unlikely to be physically associated based on my present data (DD-04 and DD-09). This distribution reveals a slight bias towards higher mass, even without a volume correction being applied to my sample. Although further analysis is required, a number of ideas have been presented to provide a possible explanation for this variation. I have also concluded that DD systems could be useful for studying ultramassive WDs, and may therefore help in establishing more firmly the lower limits on Type II supernovae progenitors.



FIGURE 4.5: La Palma - Cloud Valley



FIGURE 4.6: La Palma - Volcano



Figure 4.7: La Palma - INT



Figure 4.8: La Palma - From Top



FIGURE 4.9: La Palma - Sunset



FIGURE 4.10: La Palma - Forest Mountain



Figure 4.11: Cerro Paranal in Chile - Accomodation Paronama



Figure 4.12: Cerro Paranal in Chile - From Top



FIGURE 4.13: VLT Unit Telescope view 1

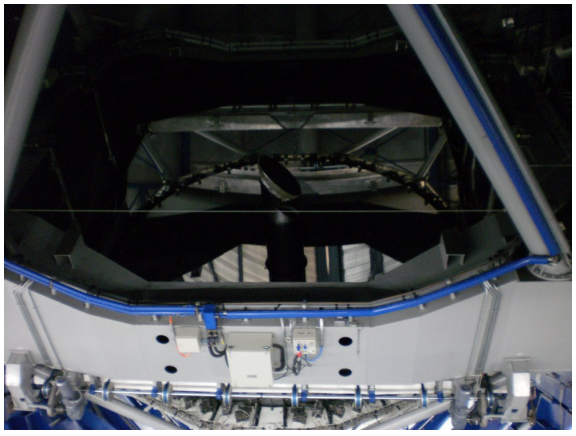


FIGURE 4.14: VLT Unit Telescope view 2

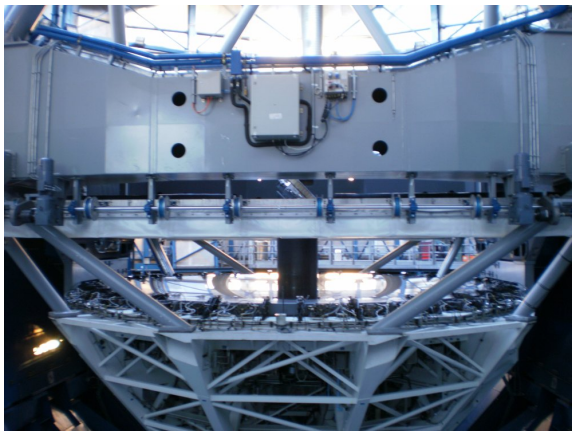


FIGURE 4.15: VLT Unit Telescope view 3



Figure 4.16: VLT Unit Telescope view 4

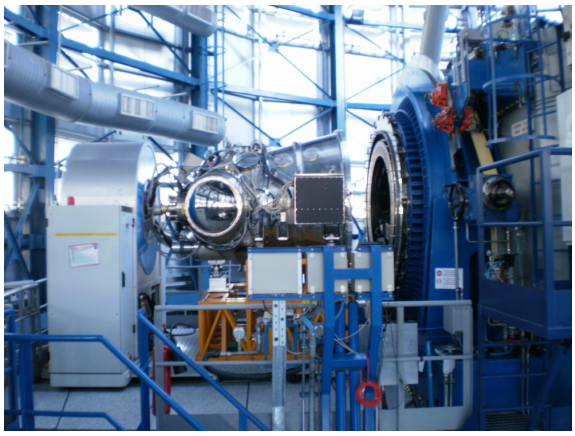


Figure 4.17: VLT Unit Telescope view 5

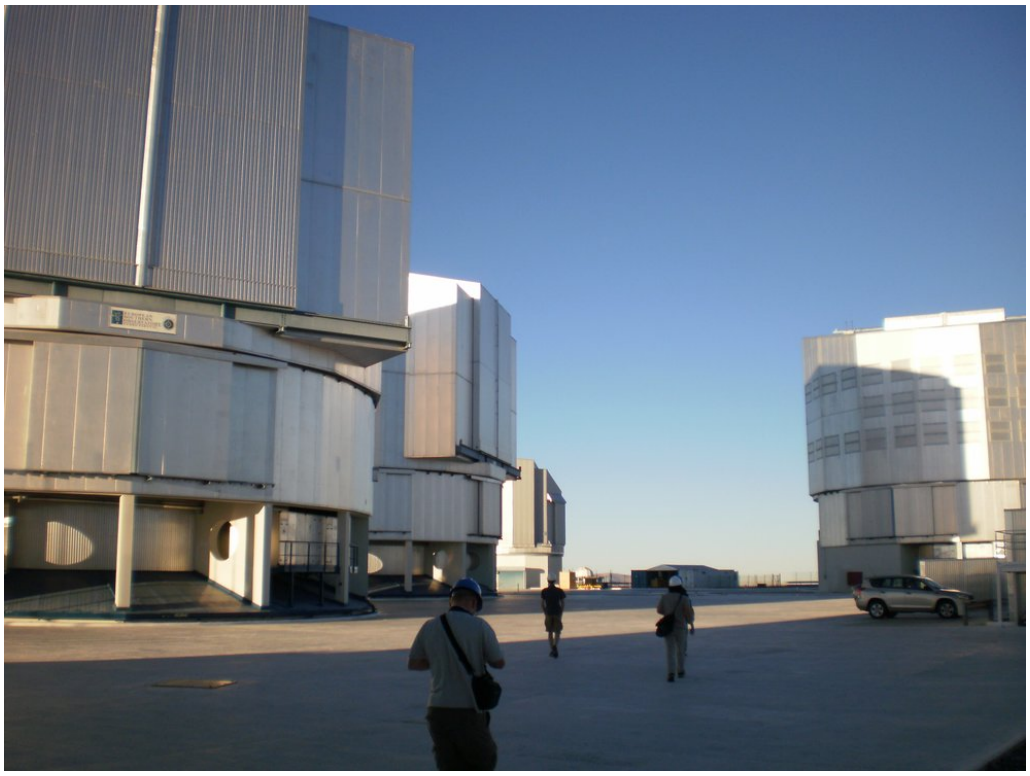


FIGURE 4.18: VLT Platform 1

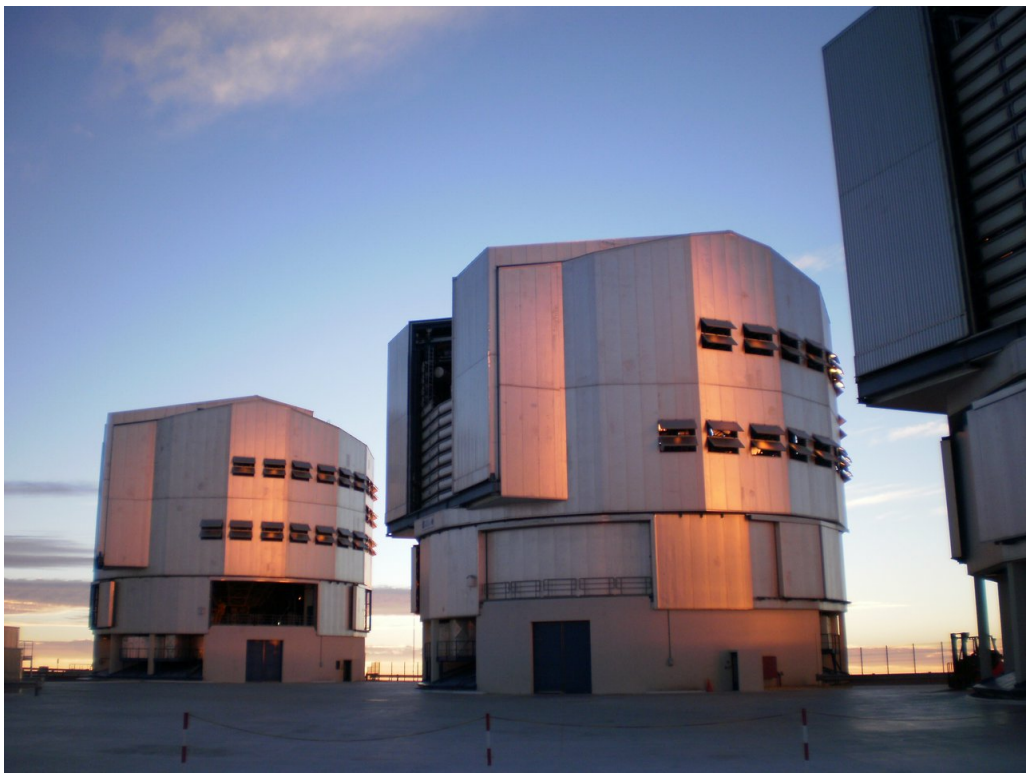


FIGURE 4.19: VLT Platform 2



Figure 4.20: VLT Platform 3

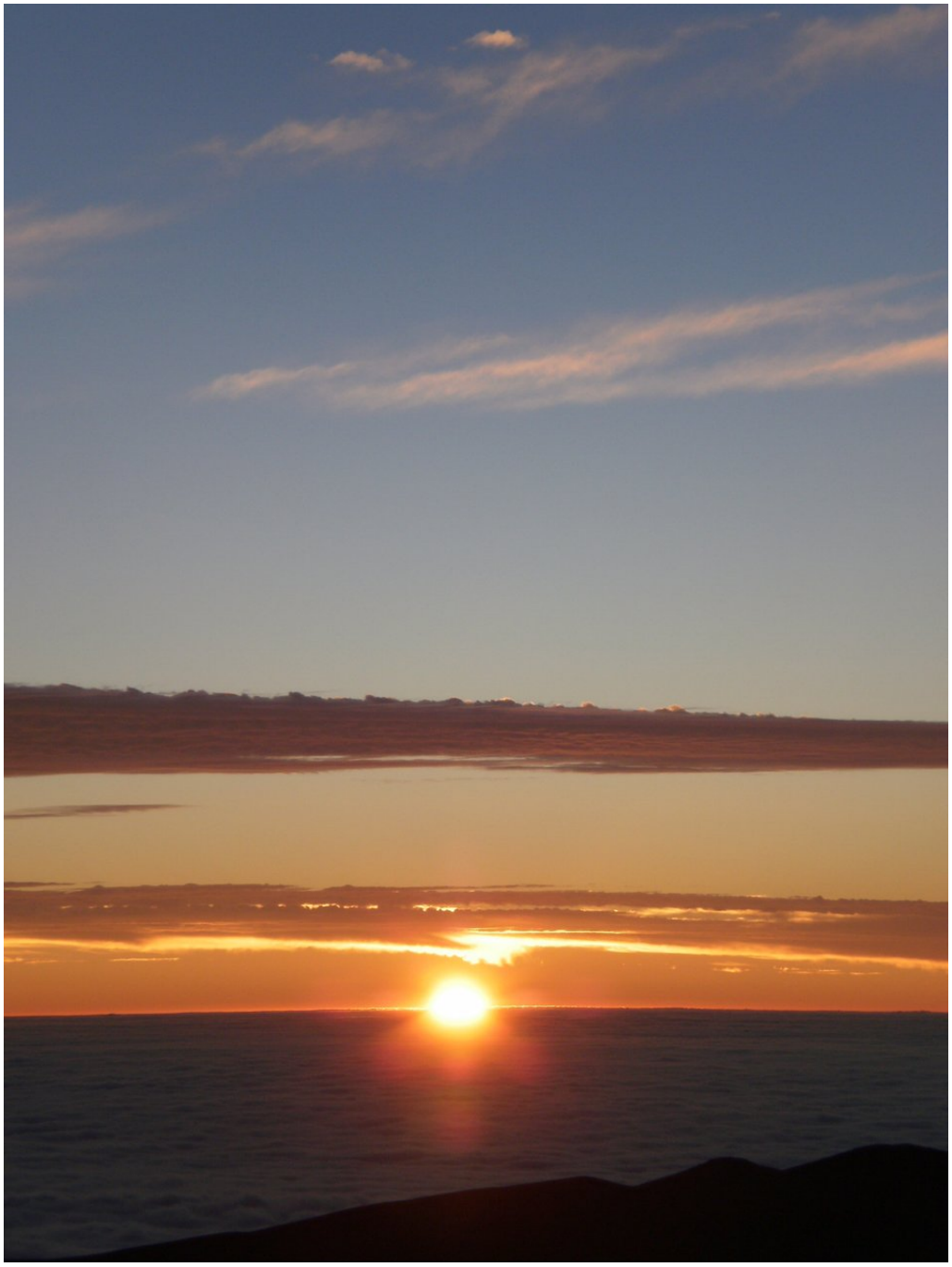


FIGURE 4.21: Cerro Paranal in Chile - Sunset

"people, for instance, who eat sweetmeats in the theatre do so most when the performance is indifferent.."

from Aristotle's Nicomachean Ethics

A Appendix

A.1 Overview

This appendix references over 12000 lines of code written specifically for this project. As such, only the headers of source files have been included in the Appendix. Additional routines have been written but have since been discarded and are not required for the replication of this project. This Appendix also references the names of routines not written specifically for this project, but used in this project. An example IDL routine has been attached at the end of the Appendix in Section A.9 for reference.

A.1.1 Standard Libraries

The IDL (/GDL) code written for this project requires the following standard libraries (or modifications thereof).

```
;    repchr.pro [NASA]
;    gettok.pro [NASA]
;    readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;    remchar.pro [NASA]
;    skip_lun.pro [required for GDL]
;    strsplit.pro [GDL library]
;    strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
```

A.2 Target Selection Code and Data

A.2.1 SDSS SQL Scripts for blue object list generation

The following SQL script may be used in conjunction with the SDSS Skyserver (http://cas.sdss.org/ to extract a list of blue objects (SDSSblueObjectsDR7detailed.txt) under the a set of magnitude/colour criteria;

resultsDR7detailed.sql

```
http://cas.sdss.org/astrodr7/en/tools/search/sql.asp

SELECT P.ObjID as SDSSID, P.ra as RIGHT_ASCENSION, P.dec as DECLINATION, P.b as GALATIC_LATITUDE, P.
    l as GALATIC_LONGITUDE, P.psfmag_u as u, P.psfmag_g as g, P.psfmag_r as r, P.clean as pok
FROM Star P
WHERE
   P.clean > 0
    and (P.psfmag_u-P.psfmag_g)<=0.5
    and (P.psfmag_u-P.psfmag_g)>=-0.7
    and (P.psfmag_g-P.psfmag_r)<=0.0
    and (P.psfmag_g-P.psfmag_r)>=-0.7
    and P.psfmag_g<=20
    and P.psfmag_r<=20
    and P.psfmag_i<=20.5
    and P.psfmag_g>=15.3
```

A.2.2 Count Close Pairs within a BO List Code

This program generates a list of blue object pairs within a given angular separation from SDSS blue object data

BOCPmonteCarloStats-01June11a.zip

CCPtest.pro

This code calculates the probability of a close pair existing within a set of BOs at a given angular separations.

```
; Title: Count Close Pairs
; File: CCPtest.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 9 March 09 6:00pm
; Summary:  Preconditions; Assumes a file SDSSblueObjects.csv, containing 4 columns of data and
    56587 rows (56587 stars within a total 11663 square degree field)
;           Column 1 = RIGHT_ASCENSION, Column 2 = DECLINATION, Column 3 = GALATIC_LATITUDE, Column
    4 = GALATIC_LONGITUDE
;       Postconditions; Estimates the number of close pairs within a maximum radius of
    maxDistanceForConsideredPairInArcDegrees (Eg 30 Arc Seconds)


;distances are given in arc seconds here

firstPairMaxDistance = 30
lastPairMaxDistance = 30
diffPairMaxDistance = 1

;firstPairMaxDistance = 0
;lastPairMaxDistance = 200
;diffPairMaxDistance = 10


;ORIGINAL;
fieldSquareDegrees = 1000L  ; or 1000   ;or 11663
fieldNumberOfStars = 4852L  ; or 4852   ;or 56587
numberOfSamples = 100L

;TEST HARNESS - load from file;
fieldSquareDegrees = 11663L         ; or 1000   ;or 11663
fieldNumberOfStars = 35324L         ; or 4852   ;or 56587
numberOfSamples = 1L

fieldxDegrees = SQRT(fieldSquareDegrees)    ; or 100
fieldyDegrees = SQRT(fieldSquareDegrees)    ; or 100

IDLrandomNumberMin = 0L
IDLrandomNumberMax = 1L
IDLrandomNumberVariation = (IDLrandomNumberMax - IDLrandomNumberMin)

for maxDistanceIndex=firstPairMaxDistance, lastPairMaxDistance, diffPairMaxDistance do begin

    maxDistanceForConsideredPairInArcSeconds = double(maxDistanceIndex)
    maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D
    totalNumberOfPairsFoundAcrossAllSamples = 0L

    for sampleIndex=1L, numberOfSamples, 1L do begin

        ;ORIGINAL;
        ;starXCoordinatesBase = RANDOMU(seed, fieldNumberOfStars, /DOUBLE)
```

```
;starYCoordinatesBase = RANDOMU(seed, fieldNumberOfStars, /DOUBLE)
;starXCoordinates = starXCoordinatesBase/IDLrandomNumberVariation*fieldxDegrees
;starYCoordinates = starYCoordinatesBase/IDLrandomNumberVariation*fieldyDegrees

;TEST HARNESS - load from file;

    numberOfLines = 35324
    numberOfColumns = 4
    data = DblArr(numberOfColumns,numberOfLines)
    NameOfFile = 'SDSSblueObjects.txt'
    openr, 5, NameOfFile
    readf, 5, data
    close,/all ; close all open files

    RIGHT_ASCENSION = data(0,*)
    DECLINATION = data(1,*)
    GALATIC_LATITUDE = data(2,*)
    GALATIC_LONGITUDE = data(3,*)

    starYCoordinates = DECLINATION ;GALATIC_LATITUDE
    starXCoordinates = RIGHT_ASCENSION ;GALATIC_LONGITUDE

    ;PRINT, [starXCoordinates, starYCoordinates]
    ;starYCoordinates = starYCoordinates+90.0
    ;plot, starXCoordinates, starYCoordinates, title='WD Candidates',xtitle='
        GALATIC_LONGITUDE (deg)',ytitle='GALATIC_LATITUDE (deg)',xrange=[0,360],yrange
        =[0,200],linestyle=5,psym=3, SYMSIZE=0.2

    ;plot, starXCoordinates, starYCoordinates, title='WD Candidates',xtitle='RIGHT_ASCENSION
        (deg)',ytitle='DECLINATION (deg)',xrange=[0,360],yrange=[-100,100],linestyle=5,
        psym=3, SYMSIZE=0.2


starXCoordinatesSortIndicies = sort(starXCoordinates)
starXCoordinatesSorted = starXCoordinates[starXCoordinatesSortIndicies]
starYCoordinatesSortIndicies = sort(starYCoordinates)
starYCoordinatesSorted = starYCoordinates[starYCoordinatesSortIndicies]

;stringmessage = "starXCoordinatesSorted ="
;PRINT, starXCoordinatesSorted

totalNumberOfPairsFound = 0L

for starIndex=1L, fieldNumberOfStars, 1L do begin

    numberOfPairsFound = 0L

    starIndexArray = starIndex-1

    centralPositionInStarXCoordinatesSorted = where(starXCoordinatesSortIndicies eq
        starIndexArray)
    centralValueInStarXCoordinatesSorted = starXCoordinatesSorted[
        centralPositionInStarXCoordinatesSorted]
        ;centralPositionInStarXCoordinates = starIndexArray
        ;or centralValueInStarXCoordinates = starXCoordinates[
            centralPositionInStarXCoordinates]
    centralPositionInStarYCoordinatesSorted = where(starYCoordinatesSortIndicies eq
        starIndexArray)
    centralValueInStarYCoordinatesSorted = starYCoordinatesSorted[
        centralPositionInStarYCoordinatesSorted]
        ;centralPositionInStarYCoordinates = starIndexArray
        ;or centralValueInStarYCoordinates = starYCoordinates[
            centralPositionInStarYCoordinates]



    ;stringmessage = "   centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted



        ;hunt in positive x direction
    consideredPairIndex = 1L
    currentMaxDistanceBetweenPairX = 0D

    while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
        begin

        if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) gt (
            fieldNumberOfStars-1) then begin
            currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
        endif else begin
```

```
currentPositionInStarXCoordinatesSorted =
     centralPositionInStarXCoordinatesSorted + consideredPairIndex
currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
     currentPositionInStarXCoordinatesSorted]
currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
     currentPositionInStarXCoordinatesSorted]
currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
currentValueInStarYCoordinates = starYCoordinates[
     currentPositionInStarYCoordinates]
currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentPositionInStarXCoordinatesSorted
;stringmessage = "currentValueInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarXCoordinatesSorted
;stringmessage = "currentPositionInStarXCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarXCoordinates
;stringmessage = "currentPositionInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinatesSorted

currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
     centralValueInStarXCoordinatesSorted)
currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
     centralValueInStarYCoordinatesSorted)
averageYDeclination = (currentValueInStarYCoordinatesSorted +
     centralValueInStarYCoordinatesSorted) / 2
currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
     currentMaxDistanceBetweenPairX
currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
      currentMaxDistanceBetweenPairYCorrected^2)

;stringmessage = "    currentMaxDistanceBetweenPairX ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairX
;stringmessage = "currentMaxDistanceBetweenPairY ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairY
;stringmessage = "currentMaxDistanceBetweenPair ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPair


if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
     then begin

    numberOfPairsFound = numberOfPairsFound + 1
    ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
         +9999

    ;stringmessage = "                              Pair Found A!"
    ;PRINT, stringmessage
    ;stringmessage = "StarIndex = "
    ;PRINT, stringmessage
    ;PRINT, starIndex
    ;stringmessage = "currentMaxDistanceBetweenPair = "
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair
    ;exit    ;TEMP

    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted

    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinatesSorted
    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarXCoordinatesSorted
```

```
            ;stringmessage = "currentPositionInStarXCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarXCoordinates
            ;stringmessage = "currentPositionInStarYCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarYCoordinates
            ;stringmessage = "currentValueInStarYCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinates
            ;stringmessage = "currentValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinatesSorted

            ;stringmessage = "    currentMaxDistanceBetweenPairX ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairX
            ;stringmessage = "currentMaxDistanceBetweenPairY ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairY
            ;stringmessage = "currentMaxDistanceBetweenPair ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair

        endif

        consideredPairIndex = consideredPairIndex + 1

    endelse


endwhile


    ;hunt in negative x direction
consideredPairIndex = -1
currentMaxDistanceBetweenPairX = 0
while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
     begin

    if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) lt 0 then begin
        currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
    endif else begin

        currentPositionInStarXCoordinatesSorted =
             centralPositionInStarXCoordinatesSorted + consideredPairIndex
        currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
        currentValueInStarYCoordinates = starYCoordinates[
             currentPositionInStarYCoordinates]
        currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

        ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinatesSorted
        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarXCoordinatesSorted
        ;stringmessage = "currentPositionInStarXCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinates
        ;stringmessage = "currentPositionInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinatesSorted

        currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
             centralValueInStarXCoordinatesSorted)
        currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
             centralValueInStarYCoordinatesSorted)
        averageYDeclination = (currentValueInStarYCoordinatesSorted +
             centralValueInStarYCoordinatesSorted) / 2
        currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
             currentMaxDistanceBetweenPairX
        currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

        currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
              currentMaxDistanceBetweenPairYCorrected^2)

        ;stringmessage = "    currentMaxDistanceBetweenPairX ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairX
        ;stringmessage = "currentMaxDistanceBetweenPairY ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairY
```

```
                ;stringmessage = "currentMaxDistanceBetweenPair ="
                ;PRINT, stringmessage
                ;PRINT, currentMaxDistanceBetweenPair

                if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
                    then begin

                    numberOfPairsFound = numberOfPairsFound + 1
                    ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
                        +9999

                    ;stringmessage = "                              Pair Found B!"
                    ;PRINT, stringmessage
                    ;stringmessage = "StarIndex = "
                    ;PRINT, stringmessage
                    ;PRINT, starIndex
                    ;stringmessage = "currentMaxDistanceBetweenPair = "
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPair
                    ;exit    ;TEMP

                    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralPositionInStarXCoordinatesSorted
                    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralValueInStarXCoordinatesSorted
                    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralPositionInStarYCoordinatesSorted
                    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralValueInStarYCoordinatesSorted

                    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarXCoordinatesSorted
                    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarXCoordinatesSorted
                    ;stringmessage = "currentPositionInStarXCoordinates ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarXCoordinates
                    ;stringmessage = "currentPositionInStarYCoordinates ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarYCoordinates
                    ;stringmessage = "currentValueInStarYCoordinates ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarYCoordinates
                    ;stringmessage = "currentValueInStarYCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarYCoordinatesSorted

                    ;stringmessage = "    currentMaxDistanceBetweenPairX ="
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPairX
                    ;stringmessage = "currentMaxDistanceBetweenPairY ="
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPairY
                    ;stringmessage = "currentMaxDistanceBetweenPair ="
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPair



                endif


                consideredPairIndex = consideredPairIndex - 1

            endelse


        endwhile

        totalNumberOfPairsFound = totalNumberOfPairsFound+numberOfPairsFound
    endfor

    stringmessage = "totalNumberOfPairsFound = "
    PRINT, stringmessage
    PRINT, (totalNumberOfPairsFound/2)

    totalNumberOfPairsFoundAcrossAllSamples = totalNumberOfPairsFoundAcrossAllSamples +
        totalNumberOfPairsFound
endfor

stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
PRINT, stringmessage
PRINT, maxDistanceForConsideredPairInArcSeconds
```

```
    stringmessage = "fieldNumberOfStars = "
    PRINT, stringmessage
    PRINT, fieldNumberOfStars

    stringmessage = "fieldSquareDegrees = "
    PRINT, stringmessage
    PRINT, fieldSquareDegrees

    averageNumberOfPairsFound = (DOUBLE(totalNumberOfPairsFoundAcrossAllSamples)/DOUBLE(
        numberOfSamples))/DOUBLE(2)
    stringmessage = "averageNumberOfPairsFound = "
    PRINT, stringmessage
    PRINT, averageNumberOfPairsFound
endfor



end
```

CCPtestsweep.pro

This code calculates the probability of a close pair existing within a set of BOs across a range of angular separations.

```
; Title: Count Close Pairs (across multiple separations)
; File: CCPtestsweep.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 10 March 09 12pm
; Summary:  Preconditions; Assumes a file SDSSblueObjects.csv, containing 4 columns of data and
    56587 rows (56587 stars within a total 11663 square degree field)
;           Column 1 = RIGHT_ASCENSION, Column 2 = DECLINATION, Column 3 = GALATIC_LATITUDE, Column
    4 = GALATIC_LONGITUDE
;       Postconditions; Estimates the number of close pairs within a maximum radius of
    maxDistanceForConsideredPairInArcDegrees (Eg 30 Arc Seconds)


;distances are given in arc seconds here

; firstPairMaxDistance = 60
; lastPairMaxDistance = 60
; diffPairMaxDistance = 1

firstPairMaxDistance = 1
lastPairMaxDistance = 200
diffPairMaxDistance = 1

;ORIGINAL;
;fieldSquareDegrees = 1000L ; or 1000   ;or 11663
;fieldNumberOfStars = 4852L ; or 4852   ;or 56587
;numberOfSamples = 100L

;TEST HARNESS - load from file;
fieldSquareDegrees = 11663L                                                    ; or 1000   ;or 11663
fieldNumberOfStars = 35324L         ;OLD2: 56490, OLD1: 56587   ;must match number of lines in file
    ; or 4852   ;or 56587
numberOfSamples = 1L

fieldxDegrees = SQRT(fieldSquareDegrees)    ; or 100
fieldyDegrees = SQRT(fieldSquareDegrees)    ; or 100

IDLrandomNumberMin = 0L
IDLrandomNumberMax = 1L
IDLrandomNumberVariation = (IDLrandomNumberMax - IDLrandomNumberMin)

for maxDistanceIndex=firstPairMaxDistance, lastPairMaxDistance, diffPairMaxDistance do begin

    maxDistanceForConsideredPairInArcSeconds = double(maxDistanceIndex)
    maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D
    totalNumberOfPairsFoundAcrossAllSamples = 0L

    for sampleIndex=1L, numberOfSamples, 1L do begin

        ;ORIGINAL;
        ;starXCoordinatesBase = RANDOMU(seed, fieldNumberOfStars, /DOUBLE)
        ;starYCoordinatesBase = RANDOMU(seed, fieldNumberOfStars, /DOUBLE)
        ;starXCoordinates = starXCoordinatesBase/IDLrandomNumberVariation*fieldxDegrees
        ;starYCoordinates = starYCoordinatesBase/IDLrandomNumberVariation*fieldyDegrees

        ;TEST HARNESS - load from file;

            numberOfLines = fieldNumberOfStars
            numberOfColumns = 4
            data = DblArr(numberOfColumns,numberOfLines)
            NameOfFile = 'BOCPLSDSSBOconfirmedNotQuasarsAndNotBadDataSmall.txt' ;OLD2:
                BOCPLSDSSBOconfirmedNotBadObjectsDataSmall.txt, OLD1: SDSSblueObjects.txt
```

```
    openr, 5, NameOfFile
    readf, 5, data
    close,/all ; close all open files

    RIGHT_ASCENSION = data(0,*)
    DECLINATION = data(1,*)
    GALATIC_LATITUDE = data(2,*)
    GALATIC_LONGITUDE = data(3,*)

    starYCoordinates = DECLINATION ;GALATIC_LATITUDE
    starXCoordinates = RIGHT_ASCENSION ;GALATIC_LONGITUDE

    ;PRINT, [starXCoordinates, starYCoordinates]
    ;starYCoordinates = starYCoordinates+90.0
    ;plot, starXCoordinates, starYCoordinates, title='WD Candidates',xtitle='
        GALATIC_LONGITUDE (deg)',ytitle='GALATIC_LATITUDE (deg)',xrange=[0,360],yrange
        =[0,200],linestyle=5,psym=3, SYMSIZE=0.2

    ;plot, starXCoordinates, starYCoordinates, title='WD Candidates',xtitle='RIGHT_ASCENSION
         (deg)',ytitle='DECLINATION (deg)',xrange=[0,360],yrange=[-100,100],linestyle=5,
        psym=3, SYMSIZE=0.2


starXCoordinatesSortIndicies = sort(starXCoordinates)
starXCoordinatesSorted = starXCoordinates[starXCoordinatesSortIndicies]
starYCoordinatesSortIndicies = sort(starYCoordinates)
starYCoordinatesSorted = starYCoordinates[starYCoordinatesSortIndicies]

;stringmessage = "starXCoordinatesSorted ="
;PRINT, starXCoordinatesSorted

totalNumberOfPairsFound = 0L

for starIndex=1L, fieldNumberOfStars, 1L do begin

    numberOfPairsFound = 0L

    starIndexArray = starIndex-1

    centralPositionInStarXCoordinatesSorted = where(starXCoordinatesSortIndicies eq
        starIndexArray)
    centralValueInStarXCoordinatesSorted = starXCoordinatesSorted[
        centralPositionInStarXCoordinatesSorted]
        ;centralPositionInStarXCoordinates = starIndexArray
        ;or centralValueInStarXCoordinates = starXCoordinates[
            centralPositionInStarXCoordinates]
    centralPositionInStarYCoordinatesSorted = where(starYCoordinatesSortIndicies eq
        starIndexArray)
    centralValueInStarYCoordinatesSorted = starYCoordinatesSorted[
        centralPositionInStarYCoordinatesSorted]
        ;centralPositionInStarYCoordinates = starIndexArray
        ;or centralValueInStarYCoordinates = starYCoordinates[
            centralPositionInStarYCoordinates]



    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted



        ;hunt in positive x direction
    consideredPairIndex = 1L
    currentMaxDistanceBetweenPairX = 0D

    while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
        begin

        if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) gt (
            fieldNumberOfStars-1) then begin
            currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
        endif else begin

            currentPositionInStarXCoordinatesSorted =
                centralPositionInStarXCoordinatesSorted + consideredPairIndex
            currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
                currentPositionInStarXCoordinatesSorted]
            currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
                currentPositionInStarXCoordinatesSorted]
            currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
            currentValueInStarYCoordinates = starYCoordinates[
                currentPositionInStarYCoordinates]
```

```
currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentPositionInStarXCoordinatesSorted
;stringmessage = "currentValueInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarXCoordinatesSorted
;stringmessage = "currentPositionInStarXCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarXCoordinates
;stringmessage = "currentPositionInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinatesSorted

currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
    centralValueInStarXCoordinatesSorted)
currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
    centralValueInStarYCoordinatesSorted)
;averageYDeclination = currentValueInStarYCoordinatesSorted +
    centralValueInStarYCoordinatesSorted) / 2
averageYDeclination = centralValueInStarYCoordinatesSorted
currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
    currentMaxDistanceBetweenPairX
currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
     currentMaxDistanceBetweenPairYCorrected^2)

;stringmessage = "    currentMaxDistanceBetweenPairX ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairX
;stringmessage = "currentMaxDistanceBetweenPairY ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairY
;stringmessage = "currentMaxDistanceBetweenPair ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPair


if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
    then begin

    numberOfPairsFound = numberOfPairsFound + 1
    ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
        +9999

    ;stringmessage = "                              Pair Found A!"
    ;PRINT, stringmessage
    ;stringmessage = "StarIndex = "
    ;PRINT, stringmessage
    ;PRINT, starIndex
    ;stringmessage = "currentMaxDistanceBetweenPair = "
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair
    ;exit    ;TEMP

    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted

    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinatesSorted
    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarXCoordinatesSorted
    ;stringmessage = "currentPositionInStarXCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinates
    ;stringmessage = "currentPositionInStarYCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarYCoordinates
    ;stringmessage = "currentValueInStarYCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarYCoordinates
```

```
            ;stringmessage = "currentValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinatesSorted

            ;stringmessage = "    currentMaxDistanceBetweenPairX ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairX
            ;stringmessage = "currentMaxDistanceBetweenPairY ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairY
            ;stringmessage = "currentMaxDistanceBetweenPair ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair

        endif

        consideredPairIndex = consideredPairIndex + 1

    endelse


endwhile


    ;hunt in negative x direction
consideredPairIndex = -1
currentMaxDistanceBetweenPairX = 0
while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
     begin

    if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) lt 0 then begin
        currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
    endif else begin

        currentPositionInStarXCoordinatesSorted =
             centralPositionInStarXCoordinatesSorted + consideredPairIndex
        currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
        currentValueInStarYCoordinates = starYCoordinates[
             currentPositionInStarYCoordinates]
        currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

        ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinatesSorted
        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarXCoordinatesSorted
        ;stringmessage = "currentPositionInStarXCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinates
        ;stringmessage = "currentPositionInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinatesSorted

        currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
             centralValueInStarXCoordinatesSorted)
        currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
             centralValueInStarYCoordinatesSorted)
        ;averageYDeclination = (currentValueInStarYCoordinatesSorted +
             centralValueInStarYCoordinatesSorted) / 2
        averageYDeclination = centralValueInStarYCoordinatesSorted
        currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
             currentMaxDistanceBetweenPairX
        currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

        currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
              currentMaxDistanceBetweenPairYCorrected^2)

        ;stringmessage = "    currentMaxDistanceBetweenPairX ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairX
        ;stringmessage = "currentMaxDistanceBetweenPairY ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairY
        ;stringmessage = "currentMaxDistanceBetweenPair ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPair

        if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
             then begin

            numberOfPairsFound = numberOfPairsFound + 1
```

```
                    ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
                        +9999

                    ;stringmessage = "                                Pair Found B!"
                    ;PRINT, stringmessage
                    ;stringmessage = "StarIndex = "
                    ;PRINT, stringmessage
                    ;PRINT, starIndex
                    ;stringmessage = "currentMaxDistanceBetweenPair = "
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPair
                    ;exit    ;TEMP

                    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralPositionInStarXCoordinatesSorted
                    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralValueInStarXCoordinatesSorted
                    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralPositionInStarYCoordinatesSorted
                    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, centralValueInStarYCoordinatesSorted

                    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarXCoordinatesSorted
                    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarXCoordinatesSorted
                    ;stringmessage = "currentPositionInStarXCoordinates ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarXCoordinates
                    ;stringmessage = "currentPositionInStarYCoordinates ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarYCoordinates
                    ;stringmessage = "currentValueInStarYCoordinates ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarYCoordinates
                    ;stringmessage = "currentValueInStarYCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarYCoordinatesSorted

                    ;stringmessage = "    currentMaxDistanceBetweenPairX ="
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPairX
                    ;stringmessage = "currentMaxDistanceBetweenPairY ="
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPairY
                    ;stringmessage = "currentMaxDistanceBetweenPair ="
                    ;PRINT, stringmessage
                    ;PRINT, currentMaxDistanceBetweenPair



                endif


                consideredPairIndex = consideredPairIndex - 1

            endelse


        endwhile

        totalNumberOfPairsFound = totalNumberOfPairsFound+numberOfPairsFound
    endfor

    ;stringmessage = "totalNumberOfPairsFound = "
    ;PRINT, stringmessage
    ;PRINT, (totalNumberOfPairsFound/2)

    totalNumberOfPairsFoundAcrossAllSamples = totalNumberOfPairsFoundAcrossAllSamples +
        totalNumberOfPairsFound
endfor

;stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
;PRINT, stringmessage
;PRINT, maxDistanceForConsideredPairInArcSeconds

;stringmessage = "fieldNumberOfStars = "
;PRINT, stringmessage
;PRINT, fieldNumberOfStars

;stringmessage = "fieldSquareDegrees = "
;PRINT, stringmessage
;PRINT, fieldSquareDegrees
```

```
    averageNumberOfPairsFound = (DOUBLE(totalNumberOfPairsFoundAcrossAllSamples)/DOUBLE(
        numberOfSamples))/DOUBLE(2)
    ;stringmessage = "averageNumberOfPairsFound = "
    ;PRINT, stringmessage
    PRINT, averageNumberOfPairsFound
endfor



end
```

A.2.3 Count Close Pairs within a Random Distribution Code

Forms a random distribution of stars (monte-carlo simulation), and calculates the number of close pairs found at a given separation (note it is a parabolic distribution; x^2).

CCP.pro

This code calculates the probability of a close pair existing within a random distribution at a given angular separations.

```
; Title: Count Close Pairs
; File: CCP.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 9 March 09 10pm
; Summary:  Preconditions; Assumes a random distribution of stars within a field of
    fieldSquareDegrees (Eg 1000 sqr deg) with fieldNumberOfStars (Eg 4852).
;       Postconditions; Estimates the number of close pairs within a maximum radius of
    maxDistanceForConsideredPairInArcDegrees (Eg 30 Arc Seconds)

;distances are given in arc seconds here

firstPairMaxDistance = 60
lastPairMaxDistance = 60
diffPairMaxDistance = 1

; firstPairMaxDistance = 1
; lastPairMaxDistance = 200
; diffPairMaxDistance = 1

fieldSquareDegrees = 1000           ; or 1000    ;or 11663
fieldxDegrees = SQRT(fieldSquareDegrees)     ; or 100
fieldyDegrees = SQRT(fieldSquareDegrees)     ; or 100
fieldNumberOfStars = 4852           ; or 4852    ;or 56587

IDLrandomNumberMin = 0
IDLrandomNumberMax = 1
IDLrandomNumberVariation = (IDLrandomNumberMax - IDLrandomNumberMin)     ; 1

numberOfSamples = 100
totalNumberOfPairsFoundAcrossAllSamples = 0

for maxDistanceIndex=firstPairMaxDistance, lastPairMaxDistance, diffPairMaxDistance do begin

    maxDistanceForConsideredPairInArcSeconds = double(maxDistanceIndex)
    maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D
    totalNumberOfPairsFoundAcrossAllSamples = 0L

    for sampleIndex=1L, numberOfSamples, 1 do begin

        starXCoordinatesBase = ABS(RANDOMU(seed, fieldNumberOfStars, /DOUBLE))
        starYCoordinatesBase = ABS(RANDOMU(seed, fieldNumberOfStars, /DOUBLE))
        starXCoordinates = starXCoordinatesBase/IDLrandomNumberVariation*fieldxDegrees
        starYCoordinates = starYCoordinatesBase/IDLrandomNumberVariation*fieldyDegrees
        starXCoordinatesSortIndicies = sort(starXCoordinates)
        starXCoordinatesSorted = starXCoordinates[starXCoordinatesSortIndicies]
        starYCoordinatesSortIndicies = sort(starYCoordinates)
        starYCoordinatesSorted = starYCoordinates[starYCoordinatesSortIndicies]

        ;stringmessage = "starXCoordinatesSorted ="
        ;PRINT, starXCoordinatesSorted

        totalNumberOfPairsFound = 0
        for starIndex=1, fieldNumberOfStars, 1 do begin

            numberOfPairsFound = 0

            currentMaxDistanceBetweenPair = 0
```

```
starIndexArray = starIndex-1

centralPositionInStarXCoordinatesSorted = where(starXCoordinatesSortIndicies eq
     starIndexArray)
centralValueInStarXCoordinatesSorted = starXCoordinatesSorted[
     centralPositionInStarXCoordinatesSorted]
    ;centralPositionInStarXCoordinates = starIndexArray
    ;or centralValueInStarXCoordinates = starXCoordinates[
         centralPositionInStarXCoordinates]
centralPositionInStarYCoordinatesSorted = where(starYCoordinatesSortIndicies eq
     starIndexArray)
centralValueInStarYCoordinatesSorted = starYCoordinatesSorted[
     centralPositionInStarYCoordinatesSorted]
    ;centralPositionInStarYCoordinates = starIndexArray
    ;or centralValueInStarYCoordinates = starYCoordinates[
         centralPositionInStarYCoordinates]

;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, centralPositionInStarXCoordinatesSorted
;stringmessage = "centralValueInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, centralValueInStarXCoordinatesSorted
;stringmessage = "centralPositionInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, centralPositionInStarYCoordinatesSorted
;stringmessage = "centralValueInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, centralValueInStarYCoordinatesSorted

    ;hunt in positive x direction
consideredPairIndex = 1
currentMaxDistanceBetweenPairX = 0D

while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
     begin

    if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) gt (
         fieldNumberOfStars-1) then begin
        currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
    endif else begin

        currentPositionInStarXCoordinatesSorted =
             centralPositionInStarXCoordinatesSorted + consideredPairIndex
        currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
        currentValueInStarYCoordinates = starYCoordinates[
             currentPositionInStarYCoordinates]
        currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

        ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinatesSorted
        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarXCoordinatesSorted
        ;stringmessage = "currentPositionInStarXCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinates
        ;stringmessage = "currentPositionInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinatesSorted

        currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
             centralValueInStarXCoordinatesSorted)
        currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
             centralValueInStarYCoordinatesSorted)
        currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairX^2 +
             currentMaxDistanceBetweenPairY^2)

        ;stringmessage = "    currentMaxDistanceBetweenPairX ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairX
        ;stringmessage = "currentMaxDistanceBetweenPairY ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairY
        ;stringmessage = "currentMaxDistanceBetweenPair ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPair
```

```
        if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
             then begin

            numberOfPairsFound = numberOfPairsFound + 1

            ;stringmessage = "                          Pair Found A!"
            ;PRINT, stringmessage
            ;stringmessage = "StarIndex = "
            ;PRINT, stringmessage
            ;PRINT, starIndex
            ;stringmessage = "currentMaxDistanceBetweenPair = "
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair
            ;exit    ;TEMP

            ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralPositionInStarXCoordinatesSorted
            ;stringmessage = "centralValueInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralValueInStarXCoordinatesSorted
            ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralPositionInStarYCoordinatesSorted
            ;stringmessage = "centralValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralValueInStarYCoordinatesSorted

            ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarXCoordinatesSorted
            ;stringmessage = "currentValueInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarXCoordinatesSorted
            ;stringmessage = "currentPositionInStarXCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarXCoordinates
            ;stringmessage = "currentPositionInStarYCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarYCoordinates
            ;stringmessage = "currentValueInStarYCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinates
            ;stringmessage = "currentValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinatesSorted

            ;stringmessage = "    currentMaxDistanceBetweenPairX ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairX
            ;stringmessage = "currentMaxDistanceBetweenPairY ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairY
            ;stringmessage = "currentMaxDistanceBetweenPair ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair

        endif

        consideredPairIndex = consideredPairIndex + 1

    endelse


endwhile


    ;hunt in negative x direction
consideredPairIndex = -1
currentMaxDistanceBetweenPairX = 0
while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
     begin

    if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) lt 0 then begin
        currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
    endif else begin

        currentPositionInStarXCoordinatesSorted =
             centralPositionInStarXCoordinatesSorted + consideredPairIndex
        currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
        currentValueInStarYCoordinates = starYCoordinates[
             currentPositionInStarYCoordinates]
        currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

        ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinatesSorted
        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
```

```
;PRINT, stringmessage
;PRINT, currentValueInStarXCoordinatesSorted
;stringmessage = "currentPositionInStarXCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarXCoordinates
;stringmessage = "currentPositionInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinatesSorted

currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
    centralValueInStarXCoordinatesSorted)
currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
    centralValueInStarYCoordinatesSorted)
currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairX^2 +
    currentMaxDistanceBetweenPairY^2)

;stringmessage = "    currentMaxDistanceBetweenPairX ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairX
;stringmessage = "currentMaxDistanceBetweenPairY ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairY
;stringmessage = "currentMaxDistanceBetweenPair ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPair

if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
    then begin

    numberOfPairsFound = numberOfPairsFound + 1

    ;stringmessage = "                              Pair Found B!"
    ;PRINT, stringmessage
    ;stringmessage = "StarIndex = "
    ;PRINT, stringmessage
    ;PRINT, starIndex
    ;stringmessage = "currentMaxDistanceBetweenPair = "
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair
    ;exit    ;TEMP

    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted

    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinatesSorted
    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarXCoordinatesSorted
    ;stringmessage = "currentPositionInStarXCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinates
    ;stringmessage = "currentPositionInStarYCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarYCoordinates
    ;stringmessage = "currentValueInStarYCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarYCoordinates
    ;stringmessage = "currentValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarYCoordinatesSorted

    ;stringmessage = "    currentMaxDistanceBetweenPairX ="
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPairX
    ;stringmessage = "currentMaxDistanceBetweenPairY ="
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPairY
    ;stringmessage = "currentMaxDistanceBetweenPair ="
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair
```

```
                    endif


                    consideredPairIndex = consideredPairIndex - 1

                endelse


            endwhile

            totalNumberOfPairsFound = totalNumberOfPairsFound+numberOfPairsFound
        endfor

        ;stringmessage = "totalNumberOfPairsFound = "
        ;PRINT, stringmessage
        ;PRINT, (totalNumberOfPairsFound/2)

        totalNumberOfPairsFoundAcrossAllSamples = totalNumberOfPairsFoundAcrossAllSamples +
            totalNumberOfPairsFound
    endfor

    stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
    PRINT, stringmessage
    PRINT, maxDistanceForConsideredPairInArcSeconds

    stringmessage = "fieldNumberOfStars = "
    PRINT, stringmessage
    PRINT, fieldNumberOfStars

    stringmessage = "fieldSquareDegrees = "
    PRINT, stringmessage
    PRINT, fieldSquareDegrees

    averageNumberOfPairsFound = (DOUBLE(totalNumberOfPairsFoundAcrossAllSamples)/DOUBLE(
        numberOfSamples))/DOUBLE(2)
    stringmessage = "averageNumberOfPairsFound = "
    PRINT, stringmessage
    PRINT, averageNumberOfPairsFound

endfor




end
```

CCPsweep.pro

This code calculates the probability of a close pair existing within a random distribution across a range of angular separations.

```
; Title: Count Close Pairs (across multiple separations)
; File: CCPsweep.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 10 March 09 12pm (updated 12 April 11 7pm)
; Summary:  Preconditions; Assumes a random distribution of stars within a field of
    fieldSquareDegrees (Eg 1000 sqr deg) with fieldNumberOfStars (Eg 4852).
;       Postconditions; Estimates the number of close pairs within a maximum radius of
    maxDistanceForConsideredPairInArcDegrees (Eg 30 Arc Seconds)

;distances are given in arc seconds here

; firstPairMaxDistance = 60
; lastPairMaxDistance = 60
; diffPairMaxDistance = 1

firstPairMaxDistance = 1
lastPairMaxDistance = 200
diffPairMaxDistance = 1

fieldSquareDegrees = 1000           ; or 1000    ;or 11663
fieldxDegrees = SQRT(fieldSquareDegrees)     ; or 100
fieldyDegrees = SQRT(fieldSquareDegrees)     ; or 100
fieldNumberOfStars = long((1000.0/11663.0)*35324)           ;OLD:    long((1000.0/11663.0)*56587)   or
     4852           ; or 4852    ;or 56587
;PRINT, "fieldNumberOfStars = ", fieldNumberOfStars

IDLrandomNumberMin = 0
IDLrandomNumberMax = 1
IDLrandomNumberVariation = (IDLrandomNumberMax - IDLrandomNumberMin)     ; 1
```

```
numberOfSamples = 100
totalNumberOfPairsFoundAcrossAllSamples = 0

for maxDistanceIndex=firstPairMaxDistance, lastPairMaxDistance, diffPairMaxDistance do begin

    maxDistanceForConsideredPairInArcSeconds = double(maxDistanceIndex)
    maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D
    totalNumberOfPairsFoundAcrossAllSamples = 0L

    for sampleIndex=1L, numberOfSamples, 1 do begin

        starXCoordinatesBase = ABS(RANDOMU(seed, fieldNumberOfStars, /DOUBLE))
        starYCoordinatesBase = ABS(RANDOMU(seed, fieldNumberOfStars, /DOUBLE))
        starXCoordinates = starXCoordinatesBase/IDLrandomNumberVariation*fieldxDegrees
        starYCoordinates = starYCoordinatesBase/IDLrandomNumberVariation*fieldyDegrees
        starXCoordinatesSortIndicies = sort(starXCoordinates)
        starXCoordinatesSorted = starXCoordinates[starXCoordinatesSortIndicies]
        starYCoordinatesSortIndicies = sort(starYCoordinates)
        starYCoordinatesSorted = starYCoordinates[starYCoordinatesSortIndicies]

        ;stringmessage = "starXCoordinatesSorted ="
        ;PRINT, starXCoordinatesSorted

        totalNumberOfPairsFound = 0
        for starIndex=1, fieldNumberOfStars, 1 do begin

            numberOfPairsFound = 0

            currentMaxDistanceBetweenPair = 0

            starIndexArray = starIndex-1

            centralPositionInStarXCoordinatesSorted = where(starXCoordinatesSortIndicies eq
                starIndexArray)
            centralValueInStarXCoordinatesSorted = starXCoordinatesSorted[
                centralPositionInStarXCoordinatesSorted]
                ;centralPositionInStarXCoordinates = starIndexArray
                ;or centralValueInStarXCoordinates = starXCoordinates[
                    centralPositionInStarXCoordinates]
            centralPositionInStarYCoordinatesSorted = where(starYCoordinatesSortIndicies eq
                starIndexArray)
            centralValueInStarYCoordinatesSorted = starYCoordinatesSorted[
                centralPositionInStarYCoordinatesSorted]
                ;centralPositionInStarYCoordinates = starIndexArray
                ;or centralValueInStarYCoordinates = starYCoordinates[
                    centralPositionInStarYCoordinates]

            ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralPositionInStarXCoordinatesSorted
            ;stringmessage = "centralValueInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralValueInStarXCoordinatesSorted
            ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralPositionInStarYCoordinatesSorted
            ;stringmessage = "centralValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralValueInStarYCoordinatesSorted

                ;hunt in positive x direction
            consideredPairIndex = 1
            currentMaxDistanceBetweenPairX = 0D

            while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
                begin

                if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) gt (
                    fieldNumberOfStars-1) then begin
                    currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
                endif else begin

                    currentPositionInStarXCoordinatesSorted =
                        centralPositionInStarXCoordinatesSorted + consideredPairIndex
                    currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
                        currentPositionInStarXCoordinatesSorted]
                    currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
                        currentPositionInStarXCoordinatesSorted]
                    currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
                    currentValueInStarYCoordinates = starYCoordinates[
                        currentPositionInStarYCoordinates]
                    currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

                    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentPositionInStarXCoordinatesSorted
                    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
                    ;PRINT, stringmessage
                    ;PRINT, currentValueInStarXCoordinatesSorted
                    ;stringmessage = "currentPositionInStarXCoordinates ="
                    ;PRINT, stringmessage
```

```
;PRINT, currentPositionInStarXCoordinates
;stringmessage = "currentPositionInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinatesSorted

currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
    centralValueInStarXCoordinatesSorted)
currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
    centralValueInStarYCoordinatesSorted)
currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairX^2 +
    currentMaxDistanceBetweenPairY^2)

;stringmessage = "    currentMaxDistanceBetweenPairX ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairX
;stringmessage = "currentMaxDistanceBetweenPairY ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairY
;stringmessage = "currentMaxDistanceBetweenPair ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPair


if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
    then begin

    numberOfPairsFound = numberOfPairsFound + 1

    ;stringmessage = "                          Pair Found A!"
    ;PRINT, stringmessage
    ;stringmessage = "StarIndex = "
    ;PRINT, stringmessage
    ;PRINT, starIndex
    ;stringmessage = "currentMaxDistanceBetweenPair = "
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair
    ;exit    ;TEMP

    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted

    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinatesSorted
    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarXCoordinatesSorted
    ;stringmessage = "currentPositionInStarXCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinates
    ;stringmessage = "currentPositionInStarYCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarYCoordinates
    ;stringmessage = "currentValueInStarYCoordinates ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarYCoordinates
    ;stringmessage = "currentValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarYCoordinatesSorted

    ;stringmessage = "    currentMaxDistanceBetweenPairX ="
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPairX
    ;stringmessage = "currentMaxDistanceBetweenPairY ="
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPairY
    ;stringmessage = "currentMaxDistanceBetweenPair ="
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair

endif

consideredPairIndex = consideredPairIndex + 1

endelse
```

```
endwhile


    ;hunt in negative x direction
consideredPairIndex = -1
currentMaxDistanceBetweenPairX = 0
while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
     begin

    if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) lt 0 then begin
        currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
    endif else begin

        currentPositionInStarXCoordinatesSorted =
             centralPositionInStarXCoordinatesSorted + consideredPairIndex
        currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
        currentValueInStarYCoordinates = starYCoordinates[
             currentPositionInStarYCoordinates]
        currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

        ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinatesSorted
        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarXCoordinatesSorted
        ;stringmessage = "currentPositionInStarXCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinates
        ;stringmessage = "currentPositionInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinatesSorted

        currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
             centralValueInStarXCoordinatesSorted)
        currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
             centralValueInStarYCoordinatesSorted)
        currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairX^2 +
             currentMaxDistanceBetweenPairY^2)

        ;stringmessage = "    currentMaxDistanceBetweenPairX ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairX
        ;stringmessage = "currentMaxDistanceBetweenPairY ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairY
        ;stringmessage = "currentMaxDistanceBetweenPair ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPair

        if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
             then begin

            numberOfPairsFound = numberOfPairsFound + 1

            ;stringmessage = "                              Pair Found B!"
            ;PRINT, stringmessage
            ;stringmessage = "StarIndex = "
            ;PRINT, stringmessage
            ;PRINT, starIndex
            ;stringmessage = "currentMaxDistanceBetweenPair = "
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair
            ;exit    ;TEMP

            ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralPositionInStarXCoordinatesSorted
            ;stringmessage = "centralValueInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralValueInStarXCoordinatesSorted
            ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralPositionInStarYCoordinatesSorted
            ;stringmessage = "centralValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, centralValueInStarYCoordinatesSorted

            ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
```

```
                        ;PRINT, currentPositionInStarXCoordinatesSorted
                        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
                        ;PRINT, stringmessage
                        ;PRINT, currentValueInStarXCoordinatesSorted
                        ;stringmessage = "currentPositionInStarXCoordinates ="
                        ;PRINT, stringmessage
                        ;PRINT, currentPositionInStarXCoordinates
                        ;stringmessage = "currentPositionInStarYCoordinates ="
                        ;PRINT, stringmessage
                        ;PRINT, currentPositionInStarYCoordinates
                        ;stringmessage = "currentValueInStarYCoordinates ="
                        ;PRINT, stringmessage
                        ;PRINT, currentValueInStarYCoordinates
                        ;stringmessage = "currentValueInStarYCoordinatesSorted ="
                        ;PRINT, stringmessage
                        ;PRINT, currentValueInStarYCoordinatesSorted

                        ;stringmessage = "    currentMaxDistanceBetweenPairX ="
                        ;PRINT, stringmessage
                        ;PRINT, currentMaxDistanceBetweenPairX
                        ;stringmessage = "currentMaxDistanceBetweenPairY ="
                        ;PRINT, stringmessage
                        ;PRINT, currentMaxDistanceBetweenPairY
                        ;stringmessage = "currentMaxDistanceBetweenPair ="
                        ;PRINT, stringmessage
                        ;PRINT, currentMaxDistanceBetweenPair



                    endif


                    consideredPairIndex = consideredPairIndex - 1

                endelse


            endwhile

            totalNumberOfPairsFound = totalNumberOfPairsFound+numberOfPairsFound
        endfor

        ;stringmessage = "totalNumberOfPairsFound = "
        ;PRINT, stringmessage
        ;PRINT, (totalNumberOfPairsFound/2)

        totalNumberOfPairsFoundAcrossAllSamples = totalNumberOfPairsFoundAcrossAllSamples +
            totalNumberOfPairsFound
    endfor

    ;stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
    ;PRINT, stringmessage
    ;PRINT, maxDistanceForConsideredPairInArcSeconds

    ;stringmessage = "fieldNumberOfStars = "
    ;PRINT, stringmessage
    ;PRINT, fieldNumberOfStars

    ;stringmessage = "fieldSquareDegrees = "
    ;PRINT, stringmessage
    ;PRINT, fieldSquareDegrees

    averageNumberOfPairsFound = (DOUBLE(totalNumberOfPairsFoundAcrossAllSamples)/DOUBLE(
        numberOfSamples))/DOUBLE(2)
    ;stringmessage = "averageNumberOfPairsFound = "
    ;PRINT, stringmessage
    PRINT, averageNumberOfPairsFound

endfor




end
```

CCPtestsweepAndCCPsweepResults-01June11a.ods

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	**Data**	**(with bad o. removed up to 200as only)**			*Number BOs =*	35324	*Area (sq deg) :*	*11663*		*Normalisation =*	*1*	*Offset =*	*60*						
2		CCPsweep.pr	CCPtestsweep			“showing diff 2”			“showing diff 1”			“showing diff 3”		“showing diff 4”		“showing diff 5”			
3	Separation (ar	Number of pai	Number Of BC	Number of pairs Theoretical (Com	Analytical Estimate - Stuve, 1827	Separation (ar	number of BO	number Of BO	Separation (ar	number of BO pairs DR	number Of BO	Separation (ar	Difference bet	Separation (ar	probability BOCP is a real binar	Separation (ar	probability BOCP is a real binar	Number Of BC	Number of pai
4	1	0	2	=B4/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F4/60/60)^2/(2*$H$1)	=A4	=D4	=C4	=A4	=D4+$M$1	=C4	=A4	=C4-D4	=A4	=((C4-D4)/C4)*100	=A4			
5	2	0.01	5	=B5/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F5/60/60)^2/(2*$H$1)	=A5	=D5	=C5	=A5	=D5+$M$1	=C5	=A5	=C5-D5	=A5	=((C5-D5)/C5)*100	=A5			
6	3	0	15	=B6/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F6/60/60)^2/(2*$H$1)	=A6	=D6	=C6	=A6	=D6+$M$1	=C6	=A6	=C6-D6	=A6	=((C6-D6)/C6)*100	=A6			
7	4	0.01	16	=B7/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F7/60/60)^2/(2*$H$1)	=A7	=D7	=C7	=A7	=D7+$M$1	=C7	=A7	=C7-D7	=A7	=((C7-D7)/C7)*100	=A7			
8	5	0.04	21	=B8/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F8/60/60)^2/(2*$H$1)	=A8	=D8	=C8	=A8	=D8+$M$1	=C8	=A8	=C8-D8	=A8	=((C8-D8)/C8)*100	=A8	=((R8-S8)/R8)*100	=C8-C3	=D8-D3
9	6	0.05	25.5	=B9/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F9/60/60)^2/(2*$H$1)	=A9	=D9	=C9	=A9	=D9+$M$1	=C9	=A9	=C9-D9	=A9	=((C9-D9)/C9)*100	=A9	=((R9-S9)/R9)*100	=C9-C4	=D9-D4
10	7	0.06	30	=B10/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F10/60/60)^2/(2*$H$1)	=A10	=D10	=C10	=A10	=D10+$M$1	=C10	=A10	=C10-D10	=A10	=((C10-D10)/C10)*100	=A10	=((R10-S10)/R10)*100	=C10-C5	=D10-D5
11	8	0.11	32	=B11/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F11/60/60)^2/(2*$H$1)	=A11	=D11	=C11	=A11	=D11+$M$1	=C11	=A11	=C11-D11	=A11	=((C11-D11)/C11)*100	=A11	=((R11-S11)/R11)*100	=C11-C6	=D11-D6
12	9	0.04	33	=B12/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F12/60/60)^2/(2*$H$1)	=A12	=D12	=C12	=A12	=D12+$M$1	=C12	=A12	=C12-D12	=A12	=((C12-D12)/C12)*100	=A12	=((R12-S12)/R12)*100	=C12-C7	=D12-D7
13	10	0.08	35.5	=B13/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F13/60/60)^2/(2*$H$1)	=A13	=D13	=C13	=A13	=D13+$M$1	=C13	=A13	=C13-D13	=A13	=((C13-D13)/C13)*100	=A13	=((R13-S13)/R13)*100	=C13-C8	=D13-D8
14	11	0.11	37	=B14/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F14/60/60)^2/(2*$H$1)	=A14	=D14	=C14	=A14	=D14+$M$1	=C14	=A14	=C14-D14	=A14	=((C14-D14)/C14)*100	=A14	=((R14-S14)/R14)*100	=C14-C9	=D14-D9
15	12	0.17	39	=B15/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F15/60/60)^2/(2*$H$1)	=A15	=D15	=C15	=A15	=D15+$M$1	=C15	=A15	=C15-D15	=A15	=((C15-D15)/C15)*100	=A15	=((R15-S15)/R15)*100	=C15-C10	=D15-D10
16	13	0.16	40.5	=B16/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F16/60/60)^2/(2*$H$1)	=A16	=D16	=C16	=A16	=D16+$M$1	=C16	=A16	=C16-D16	=A16	=((C16-D16)/C16)*100	=A16	=((R16-S16)/R16)*100	=C16-C11	=D16-D11
17	14	0.21	40.5	=B17/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F17/60/60)^2/(2*$H$1)	=A17	=D17	=C17	=A17	=D17+$M$1	=C17	=A17	=C17-D17	=A17	=((C17-D17)/C17)*100	=A17	=((R17-S17)/R17)*100	=C17-C12	=D17-D12
18	15	0.32	41.5	=B18/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F18/60/60)^2/(2*$H$1)	=A18	=D18	=C18	=A18	=D18+$M$1	=C18	=A18	=C18-D18	=A18	=((C18-D18)/C18)*100	=A18	=((R18-S18)/R18)*100	=C18-C13	=D18-D13
19	16	0.45	43	=B19/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F19/60/60)^2/(2*$H$1)	=A19	=D19	=C19	=A19	=D19+$M$1	=C19	=A19	=C19-D19	=A19	=((C19-D19)/C19)*100	=A19	=((R19-S19)/R19)*100	=C19-C14	=D19-D14
20	17	0.36	45	=B20/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F20/60/60)^2/(2*$H$1)	=A20	=D20	=C20	=A20	=D20+$M$1	=C20	=A20	=C20-D20	=A20	=((C20-D20)/C20)*100	=A20	=((R20-S20)/R20)*100	=C20-C15	=D20-D15
21	18	0.43	45	=B21/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F21/60/60)^2/(2*$H$1)	=A21	=D21	=C21	=A21	=D21+$M$1	=C21	=A21	=C21-D21	=A21	=((C21-D21)/C21)*100	=A21	=((R21-S21)/R21)*100	=C21-C16	=D21-D16
22	19	0.4	47	=B22/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F22/60/60)^2/(2*$H$1)	=A22	=D22	=C22	=A22	=D22+$M$1	=C22	=A22	=C22-D22	=A22	=((C22-D22)/C22)*100	=A22	=((R22-S22)/R22)*100	=C22-C17	=D22-D17
23	20	0.39	48	=B23/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F23/60/60)^2/(2*$H$1)	=A23	=D23	=C23	=A23	=D23+$M$1	=C23	=A23	=C23-D23	=A23	=((C23-D23)/C23)*100	=A23	=((R23-S23)/R23)*100	=C23-C18	=D23-D18
24	21	0.51	48	=B24/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F24/60/60)^2/(2*$H$1)	=A24	=D24	=C24	=A24	=D24+$M$1	=C24	=A24	=C24-D24	=A24	=((C24-D24)/C24)*100	=A24	=((R24-S24)/R24)*100	=C24-C19	=D24-D19
25	22	0.47	48	=B25/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F25/60/60)^2/(2*$H$1)	=A25	=D25	=C25	=A25	=D25+$M$1	=C25	=A25	=C25-D25	=A25	=((C25-D25)/C25)*100	=A25	=((R25-S25)/R25)*100	=C25-C20	=D25-D20
26	23	0.59	48	=B26/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F26/60/60)^2/(2*$H$1)	=A26	=D26	=C26	=A26	=D26+$M$1	=C26	=A26	=C26-D26	=A26	=((C26-D26)/C26)*100	=A26	=((R26-S26)/R26)*100	=C26-C21	=D26-D21
27	24	0.56	48	=B27/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F27/60/60)^2/(2*$H$1)	=A27	=D27	=C27	=A27	=D27+$M$1	=C27	=A27	=C27-D27	=A27	=((C27-D27)/C27)*100	=A27	=((R27-S27)/R27)*100	=C27-C22	=D27-D22
28	25	0.58	48	=B28/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F28/60/60)^2/(2*$H$1)	=A28	=D28	=C28	=A28	=D28+$M$1	=C28	=A28	=C28-D28	=A28	=((C28-D28)/C28)*100	=A28	=((R28-S28)/R28)*100	=C28-C23	=D28-D23
29	26	0.52	51	=B29/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F29/60/60)^2/(2*$H$1)	=A29	=D29	=C29	=A29	=D29+$M$1	=C29	=A29	=C29-D29	=A29	=((C29-D29)/C29)*100	=A29	=((R29-S29)/R29)*100	=C29-C24	=D29-D24
30	27	0.96	54	=B30/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F30/60/60)^2/(2*$H$1)	=A30	=D30	=C30	=A30	=D30+$M$1	=C30	=A30	=C30-D30	=A30	=((C30-D30)/C30)*100	=A30	=((R30-S30)/R30)*100	=C30-C25	=D30-D25
31	28	0.76	54.5	=B31/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F31/60/60)^2/(2*$H$1)	=A31	=D31	=C31	=A31	=D31+$M$1	=C31	=A31	=C31-D31	=A31	=((C31-D31)/C31)*100	=A31	=((R31-S31)/R31)*100	=C31-C26	=D31-D26
32	29	0.97	55	=B32/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F32/60/60)^2/(2*$H$1)	=A32	=D32	=C32	=A32	=D32+$M$1	=C32	=A32	=C32-D32	=A32	=((C32-D32)/C32)*100	=A32	=((R32-S32)/R32)*100	=C32-C27	=D32-D27
33	30	0.98	55	=B33/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F33/60/60)^2/(2*$H$1)	=A33	=D33	=C33	=A33	=D33+$M$1	=C33	=A33	=C33-D33	=A33	=((C33-D33)/C33)*100	=A33	=((R33-S33)/R33)*100	=C33-C28	=D33-D28
34	31	1.17	56	=B34/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F34/60/60)^2/(2*$H$1)	=A34	=D34	=C34	=A34	=D34+$M$1	=C34	=A34	=C34-D34	=A34	=((C34-D34)/C34)*100	=A34	=((R34-S34)/R34)*100	=C34-C29	=D34-D29
35	32	1.1	56	=B35/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F35/60/60)^2/(2*$H$1)	=A35	=D35	=C35	=A35	=D35+$M$1	=C35	=A35	=C35-D35	=A35	=((C35-D35)/C35)*100	=A35	=((R35-S35)/R35)*100	=C35-C30	=D35-D30
36	33	1.18	58	=B36/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F36/60/60)^2/(2*$H$1)	=A36	=D36	=C36	=A36	=D36+$M$1	=C36	=A36	=C36-D36	=A36	=((C36-D36)/C36)*100	=A36	=((R36-S36)/R36)*100	=C36-C31	=D36-D31
37	34	1.28	59	=B37/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F37/60/60)^2/(2*$H$1)	=A37	=D37	=C37	=A37	=D37+$M$1	=C37	=A37	=C37-D37	=A37	=((C37-D37)/C37)*100	=A37	=((R37-S37)/R37)*100	=C37-C32	=D37-D32
38	35	1.44	61	=B38/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F38/60/60)^2/(2*$H$1)	=A38	=D38	=C38	=A38	=D38+$M$1	=C38	=A38	=C38-D38	=A38	=((C38-D38)/C38)*100	=A38	=((R38-S38)/R38)*100	=C38-C33	=D38-D33
39	36	1.38	61	=B39/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F39/60/60)^2/(2*$H$1)	=A39	=D39	=C39	=A39	=D39+$M$1	=C39	=A39	=C39-D39	=A39	=((C39-D39)/C39)*100	=A39	=((R39-S39)/R39)*100	=C39-C34	=D39-D34
40	37	1.35	61	=B40/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F40/60/60)^2/(2*$H$1)	=A40	=D40	=C40	=A40	=D40+$M$1	=C40	=A40	=C40-D40	=A40	=((C40-D40)/C40)*100	=A40	=((R40-S40)/R40)*100	=C40-C35	=D40-D35
41	38	1.78	61	=B41/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F41/60/60)^2/(2*$H$1)	=A41	=D41	=C41	=A41	=D41+$M$1	=C41	=A41	=C41-D41	=A41	=((C41-D41)/C41)*100	=A41	=((R41-S41)/R41)*100	=C41-C36	=D41-D36
42	39	1.46	63	=B42/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F42/60/60)^2/(2*$H$1)	=A42	=D42	=C42	=A42	=D42+$M$1	=C42	=A42	=C42-D42	=A42	=((C42-D42)/C42)*100	=A42	=((R42-S42)/R42)*100	=C42-C37	=D42-D37
43	40	1.7	63	=B43/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F43/60/60)^2/(2*$H$1)	=A43	=D43	=C43	=A43	=D43+$M$1	=C43	=A43	=C43-D43	=A43	=((C43-D43)/C43)*100	=A43	=((R43-S43)/R43)*100	=C43-C38	=D43-D38
44	41	1.88	64	=B44/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F44/60/60)^2/(2*$H$1)	=A44	=D44	=C44	=A44	=D44+$M$1	=C44	=A44	=C44-D44	=A44	=((C44-D44)/C44)*100	=A44	=((R44-S44)/R44)*100	=C44-C39	=D44-D39
45	42	2.28	65.5	=B45/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F45/60/60)^2/(2*$H$1)	=A45	=D45	=C45	=A45	=D45+$M$1	=C45	=A45	=C45-D45	=A45	=((C45-D45)/C45)*100	=A45	=((R45-S45)/R45)*100	=C45-C40	=D45-D40
46	43	2.22	67.5	=B46/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F46/60/60)^2/(2*$H$1)	=A46	=D46	=C46	=A46	=D46+$M$1	=C46	=A46	=C46-D46	=A46	=((C46-D46)/C46)*100	=A46	=((R46-S46)/R46)*100	=C46-C41	=D46-D41
47	44	2.08	67.5	=B47/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F47/60/60)^2/(2*$H$1)	=A47	=D47	=C47	=A47	=D47+$M$1	=C47	=A47	=C47-D47	=A47	=((C47-D47)/C47)*100	=A47	=((R47-S47)/R47)*100	=C47-C42	=D47-D42
48	45	2.12	70	=B48/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F48/60/60)^2/(2*$H$1)	=A48	=D48	=C48	=A48	=D48+$M$1	=C48	=A48	=C48-D48	=A48	=((C48-D48)/C48)*100	=A48	=((R48-S48)/R48)*100	=C48-C43	=D48-D43
49	46	2.27	70	=B49/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F49/60/60)^2/(2*$H$1)	=A49	=D49	=C49	=A49	=D49+$M$1	=C49	=A49	=C49-D49	=A49	=((C49-D49)/C49)*100	=A49	=((R49-S49)/R49)*100	=C49-C44	=D49-D44
50	47	2.23	70	=B50/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F50/60/60)^2/(2*$H$1)	=A50	=D50	=C50	=A50	=D50+$M$1	=C50	=A50	=C50-D50	=A50	=((C50-D50)/C50)*100	=A50	=((R50-S50)/R50)*100	=C50-C45	=D50-D45
51	48	2.68	70	=B51/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F51/60/60)^2/(2*$H$1)	=A51	=D51	=C51	=A51	=D51+$M$1	=C51	=A51	=C51-D51	=A51	=((C51-D51)/C51)*100	=A51	=((R51-S51)/R51)*100	=C51-C46	=D51-D46
52	49	2.69	72	=B52/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F52/60/60)^2/(2*$H$1)	=A52	=D52	=C52	=A52	=D52+$M$1	=C52	=A52	=C52-D52	=A52	=((C52-D52)/C52)*100	=A52	=((R52-S52)/R52)*100	=C52-C47	=D52-D47
53	50	2.88	73	=B53/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F53/60/60)^2/(2*$H$1)	=A53	=D53	=C53	=A53	=D53+$M$1	=C53	=A53	=C53-D53	=A53	=((C53-D53)/C53)*100	=A53	=((R53-S53)/R53)*100	=C53-C48	=D53-D48
54	51	3.08	76	=B54/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F54/60/60)^2/(2*$H$1)	=A54	=D54	=C54	=A54	=D54+$M$1	=C54	=A54	=C54-D54	=A54	=((C54-D54)/C54)*100	=A54	=((R54-S54)/R54)*100	=C54-C49	=D54-D49
55	52	3.02	78	=B55/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F55/60/60)^2/(2*$H$1)	=A55	=D55	=C55	=A55	=D55+$M$1	=C55	=A55	=C55-D55	=A55	=((C55-D55)/C55)*100	=A55	=((R55-S55)/R55)*100	=C55-C50	=D55-D50
56	53	3.05	79	=B56/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F56/60/60)^2/(2*$H$1)	=A56	=D56	=C56	=A56	=D56+$M$1	=C56	=A56	=C56-D56	=A56	=((C56-D56)/C56)*100	=A56	=((R56-S56)/R56)*100	=C56-C51	=D56-D51
57	54	3.45	81	=B57/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F57/60/60)^2/(2*$H$1)	=A57	=D57	=C57	=A57	=D57+$M$1	=C57	=A57	=C57-D57	=A57	=((C57-D57)/C57)*100	=A57	=((R57-S57)/R57)*100	=C57-C52	=D57-D52
58	55	3.32	82	=B58/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F58/60/60)^2/(2*$H$1)	=A58	=D58	=C58	=A58	=D58+$M$1	=C58	=A58	=C58-D58	=A58	=((C58-D58)/C58)*100	=A58	=((R58-S58)/R58)*100	=C58-C53	=D58-D53
59	56	3.86	84.5	=B59/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F59/60/60)^2/(2*$H$1)	=A59	=D59	=C59	=A59	=D59+$M$1	=C59	=A59	=C59-D59	=A59	=((C59-D59)/C59)*100	=A59	=((R59-S59)/R59)*100	=C59-C54	=D59-D54
60	57	3.57	85.5	=B60/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F60/60/60)^2/(2*$H$1)	=A60	=D60	=C60	=A60	=D60+$M$1	=C60	=A60	=C60-D60	=A60	=((C60-D60)/C60)*100	=A60	=((R60-S60)/R60)*100	=C60-C55	=D60-D55
61	58	3.66	85.5	=B61/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F61/60/60)^2/(2*$H$1)	=A61	=D61	=C61	=A61	=D61+$M$1	=C61	=A61	=C61-D61	=A61	=((C61-D61)/C61)*100	=A61	=((R61-S61)/R61)*100	=C61-C56	=D61-D56
62	59	3.54	88	=B62/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F62/60/60)^2/(2*$H$1)	=A62	=D62	=C62	=A62	=D62+$M$1	=C62	=A62	=C62-D62	=A62	=((C62-D62)/C62)*100	=A62	=((R62-S62)/R62)*100	=C62-C57	=D62-D57
63	60	3.81	91.5	=B63/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F63/60/60)^2/(2*$H$1)	=A63	=D63	=C63	=A63	=D63+$M$1	=C63	=A63	=C63-D63	=A63	=((C63-D63)/C63)*100	=A63	=((R63-S63)/R63)*100	=C63-C58	=D63-D58
64	61	4.17	91.5	=B64/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F64/60/60)^2/(2*$H$1)	=A64	=D64	=C64	=A64	=D64+$M$1	=C64	=A64	=C64-D64	=A64	=((C64-D64)/C64)*100	=A64	=((R64-S64)/R64)*100	=C64-C59	=D64-D59
65	62	4.22	92.5	=B65/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F65/60/60)^2/(2*$H$1)	=A65	=D65	=C65	=A65	=D65+$M$1	=C65	=A65	=C65-D65	=A65	=((C65-D65)/C65)*100	=A65	=((R65-S65)/R65)*100	=C65-C60	=D65-D60
66	63	4.62	92.5	=B66/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F66/60/60)^2/(2*$H$1)	=A66	=D66	=C66	=A66	=D66+$M$1	=C66	=A66	=C66-D66	=A66	=((C66-D66)/C66)*100	=A66	=((R66-S66)/R66)*100	=C66-C61	=D66-D61
67	64	4.82	93	=B67/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F67/60/60)^2/(2*$H$1)	=A67	=D67	=C67	=A67	=D67+$M$1	=C67	=A67	=C67-D67	=A67	=((C67-D67)/C67)*100	=A67	=((R67-S67)/R67)*100	=C67-C62	=D67-D62
68	65	4.74	93	=B68/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F68/60/60)^2/(2*$H$1)	=A68	=D68	=C68	=A68	=D68+$M$1	=C68	=A68	=C68-D68	=A68	=((C68-D68)/C68)*100	=A68	=((R68-S68)/R68)*100	=C68-C63	=D68-D63
69	66	4.74	95.5	=B69/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F69/60/60)^2/(2*$H$1)	=A69	=D69	=C69	=A69	=D69+$M$1	=C69	=A69	=C69-D69	=A69	=((C69-D69)/C69)*100	=A69	=((R69-S69)/R69)*100	=C69-C64	=D69-D64
70	67	5.19	99.5	=B70/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F70/60/60)^2/(2*$H$1)	=A70	=D70	=C70	=A70	=D70+$M$1	=C70	=A70	=C70-D70	=A70	=((C70-D70)/C70)*100	=A70	=((R70-S70)/R70)*100	=C70-C65	=D70-D65
71	68	4.93	100.5	=B71/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F71/60/60)^2/(2*$H$1)	=A71	=D71	=C71	=A71	=D71+$M$1	=C71	=A71	=C71-D71	=A71	=((C71-D71)/C71)*100	=A71	=((R71-S71)/R71)*100	=C71-C66	=D71-D66
72	69	5.04	102	=B72/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F72/60/60)^2/(2*$H$1)	=A72	=D72	=C72	=A72	=D72+$M$1	=C72	=A72	=C72-D72	=A72	=((C72-D72)/C72)*100	=A72	=((R72-S72)/R72)*100	=C72-C67	=D72-D67
73	70	5.76	103.5	=B73/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F73/60/60)^2/(2*$H$1)	=A73	=D73	=C73	=A73	=D73+$M$1	=C73	=A73	=C73-D73	=A73	=((C73-D73)/C73)*100	=A73	=((R73-S73)/R73)*100	=C73-C68	=D73-D68
74	71	5.6	105.5	=B74/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F74/60/60)^2/(2*$H$1)	=A74	=D74	=C74	=A74	=D74+$M$1	=C74	=A74	=C74-D74	=A74	=((C74-D74)/C74)*100	=A74	=((R74-S74)/R74)*100	=C74-C69	=D74-D69
75	72	5.87	109	=B75/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F75/60/60)^2/(2*$H$1)	=A75	=D75	=C75	=A75	=D75+$M$1	=C75	=A75	=C75-D75	=A75	=((C75-D75)/C75)*100	=A75	=((R75-S75)/R75)*100	=C75-C70	=D75-D70
76	73	5.58	109.5	=B76/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F76/60/60)^2/(2*$H$1)	=A76	=D76	=C76	=A76	=D76+$M$1	=C76	=A76	=C76-D76	=A76	=((C76-D76)/C76)*100	=A76	=((R76-S76)/R76)*100	=C76-C71	=D76-D71
77	74	6.26	113.5	=B77/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F77/60/60)^2/(2*$H$1)	=A77	=D77	=C77	=A77	=D77+$M$1	=C77	=A77	=C77-D77	=A77	=((C77-D77)/C77)*100	=A77	=((R77-S77)/R77)*100	=C77-C72	=D77-D72
78	75	6.17	117	=B78/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F78/60/60)^2/(2*$H$1)	=A78	=D78	=C78	=A78	=D78+$M$1	=C78	=A78	=C78-D78	=A78	=((C78-D78)/C78)*100	=A78	=((R78-S78)/R78)*100	=C78-C73	=D78-D73
79	76	6.3	122.5	=B79/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F79/60/60)^2/(2*$H$1)	=A79	=D79	=C79	=A79	=D79+$M$1	=C79	=A79	=C79-D79	=A79	=((C79-D79)/C79)*100	=A79	=((R79-S79)/R79)*100	=C79-C74	=D79-D74
80	77	6.02	122.5	=B80/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F80/60/60)^2/(2*$H$1)	=A80	=D80	=C80	=A80	=D80+$M$1	=C80	=A80	=C80-D80	=A80	=((C80-D80)/C80)*100	=A80	=((R80-S80)/R80)*100	=C80-C75	=D80-D75
81	78	6.83	123.5	=B81/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F81/60/60)^2/(2*$H$1)	=A81	=D81	=C81	=A81	=D81+$M$1	=C81	=A81	=C81-D81	=A81	=((C81-D81)/C81)*100	=A81	=((R81-S81)/R81)*100	=C81-C76	=D81-D76
82	79	7.09	126	=B82/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F82/60/60)^2/(2*$H$1)	=A82	=D82	=C82	=A82	=D82+$M$1	=C82	=A82	=C82-D82	=A82	=((C82-D82)/C82)*100	=A82	=((R82-S82)/R82)*100	=C82-C77	=D82-D77
83	80	7.34	128	=B83/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F83/60/60)^2/(2*$H$1)	=A83	=D83	=C83	=A83	=D83+$M$1	=C83	=A83	=C83-D83	=A83	=((C83-D83)/C83)*100	=A83	=((R83-S83)/R83)*100	=C83-C78	=D83-D78
84	81	7.25	128	=B84/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F84/60/60)^2/(2*$H$1)	=A84	=D84	=C84	=A84	=D84+$M$1	=C84	=A84	=C84-D84	=A84	=((C84-D84)/C84)*100	=A84	=((R84-S84)/R84)*100	=C84-C79	=D84-D79
85	82	7	130	=B85/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F85/60/60)^2/(2*$H$1)	=A85	=D85	=C85	=A85	=D85+$M$1	=C85	=A85	=C85-D85	=A85	=((C85-D85)/C85)*100	=A85	=((R85-S85)/R85)*100	=C85-C80	=D85-D80
86	83	8.01	134	=B86/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F86/60/60)^2/(2*$H$1)	=A86	=D86	=C86	=A86	=D86+$M$1	=C86	=A86	=C86-D86	=A86	=((C86-D86)/C86)*100	=A86	=((R86-S86)/R86)*100	=C86-C81	=D86-D81
87	84	8.24	137	=B87/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F87/60/60)^2/(2*$H$1)	=A87	=D87	=C87	=A87	=D87+$M$1	=C87	=A87	=C87-D87	=A87	=((C87-D87)/C87)*100	=A87	=((R87-S87)/R87)*100	=C87-C82	=D87-D82
88	85	8.07	139.5	=B88/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F88/60/60)^2/(2*$H$1)	=A88	=D88	=C88	=A88	=D88+$M$1	=C88	=A88	=C88-D88	=A88	=((C88-D88)/C88)*100	=A88	=((R88-S88)/R88)*100	=C88-C83	=D88-D83
89	86	8.61	141.5	=B89/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F89/60/60)^2/(2*$H$1)	=A89	=D89	=C89	=A89	=D89+$M$1	=C89	=A89	=C89-D89	=A89	=((C89-D89)/C89)*100	=A89	=((R89-S89)/R89)*100	=C89-C84	=D89-D84
90	87	7.94	143.5	=B90/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F90/60/60)^2/(2*$H$1)	=A90	=D90	=C90	=A90	=D90+$M$1	=C90	=A90	=C90-D90	=A90	=((C90-D90)/C90)*100	=A90	=((R90-S90)/R90)*100	=C90-C85	=D90-D85
91	88	8.63	145.5	=B91/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F91/60/60)^2/(2*$H$1)	=A91	=D91	=C91	=A91	=D91+$M$1	=C91	=A91	=C91-D91	=A91	=((C91-D91)/C91)*100	=A91	=((R91-S91)/R91)*100	=C91-C86	=D91-D86
92	89	8.69	149.5	=B92/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F92/60/60)^2/(2*$H$1)	=A92	=D92	=C92	=A92	=D92+$M$1	=C92	=A92	=C92-D92	=A92	=((C92-D92)/C92)*100	=A92	=((R92-S92)/R92)*100	=C92-C87	=D92-D87
93	90	8.64	154	=B93/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F93/60/60)^2/(2*$H$1)	=A93	=D93	=C93	=A93	=D93+$M$1	=C93	=A93	=C93-D93	=A93	=((C93-D93)/C93)*100	=A93	=((R93-S93)/R93)*100	=C93-C88	=D93-D88
94	91	8.71	155	=B94/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F94/60/60)^2/(2*$H$1)	=A94	=D94	=C94	=A94	=D94+$M$1	=C94	=A94	=C94-D94	=A94	=((C94-D94)/C94)*100	=A94	=((R94-S94)/R94)*100	=C94-C89	=D94-D89
95	92	9	157	=B95/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F95/60/60)^2/(2*$H$1)	=A95	=D95	=C95	=A95	=D95+$M$1	=C95	=A95	=C95-D95	=A95	=((C95-D95)/C95)*100	=A95	=((R95-S95)/R95)*100	=C95-C90	=D95-D90
96	93	9.57	160	=B96/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F96/60/60)^2/(2*$H$1)	=A96	=D96	=C96	=A96	=D96+$M$1	=C96	=A96	=C96-D96	=A96	=((C96-D96)/C96)*100	=A96	=((R96-S96)/R96)*100	=C96-C91	=D96-D91
97	94	10.24	161	=B97/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F97/60/60)^2/(2*$H$1)	=A97	=D97	=C97	=A97	=D97+$M$1	=C97	=A97	=C97-D97	=A97	=((C97-D97)/C97)*100	=A97	=((R97-S97)/R97)*100	=C97-C92	=D97-D92
98	95	10.1	165	=B98/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F98/60/60)^2/(2*$H$1)	=A98	=D98	=C98	=A98	=D98+$M$1	=C98	=A98	=C98-D98	=A98	=((C98-D98)/C98)*100	=A98	=((R98-S98)/R98)*100	=C98-C93	=D98-D93
99	96	9.91	166	=B99/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F99/60/60)^2/(2*$H$1)	=A99	=D99	=C99	=A99	=D99+$M$1	=C99	=A99	=C99-D99	=A99	=((C99-D99)/C99)*100	=A99	=((R99-S99)/R99)*100	=C99-C94	=D99-D94
100	97	10.52	169	=B100/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F100/60/60)^2/(2*$H$1)	=A100	=D100	=C100	=A100	=D100+$M$1	=C100	=A100	=C100-D100	=A100	=((C100-D100)/C100)*100	=A100	=((R100-S100)/R100)*100	=C100-C95	=D100-D95
101	98	10.43	172	=B101/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F101/60/60)^2/(2*$H$1)	=A101	=D101	=C101	=A101	=D101+$M$1	=C101	=A101	=C101-D101	=A101	=((C101-D101)/C101)*100	=A101	=((R101-S101)/R101)*100	=C101-C96	=D101-D96
102	99	10.92	174	=B102/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F102/60/60)^2/(2*$H$1)	=A102	=D102	=C102	=A102	=D102+$M$1	=C102	=A102	=C102-D102	=A102	=((C102-D102)/C102)*100	=A102	=((R102-S102)/R102)*100	=C102-C97	=D102-D97
103	100	11.59	178.5	=B103/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F103/60/60)^2/(2*$H$1)	=A103	=D103	=C103	=A103	=D103+$M$1	=C103	=A103	=C103-D103	=A103	=((C103-D103)/C103)*100	=A103	=((R103-S103)/R103)*100	=C103-C98	=D103-D98
104	101	11.39	181.5	=B104/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F104/60/60)^2/(2*$H$1)	=A104	=D104	=C104	=A104	=D104+$M$1	=C104	=A104	=C104-D104	=A104	=((C104-D104)/C104)*100	=A104	=((R104-S104)/R104)*100	=C104-C99	=D104-D99
105	102	11.26	185.5	=B105/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F105/60/60)^2/(2*$H$1)	=A105	=D105	=C105	=A105	=D105+$M$1	=C105	=A105	=C105-D105	=A105	=((C105-D105)/C105)*100	=A105	=((R105-S105)/R105)*100	=C105-C100	=D105-D100
106	103	12.28	188	=B106/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F106/60/60)^2/(2*$H$1)	=A106	=D106	=C106	=A106	=D106+$M$1	=C106	=A106	=C106-D106	=A106	=((C106-D106)/C106)*100	=A106	=((R106-S106)/R106)*100	=C106-C101	=D106-D101
107	104	11.94	188	=B107/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F107/60/60)^2/(2*$H$1)	=A107	=D107	=C107	=A107	=D107+$M$1	=C107	=A107	=C107-D107	=A107	=((C107-D107)/C107)*100	=A107	=((R107-S107)/R107)*100	=C107-C102	=D107-D102
108	105	12.49	189	=B108/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F108/60/60)^2/(2*$H$1)	=A108	=D108	=C108	=A108	=D108+$M$1	=C108	=A108	=C108-D108	=A108	=((C108-D108)/C108)*100	=A108	=((R108-S108)/R108)*100	=C108-C103	=D108-D103
109	106	12.03	190	=B109/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F109/60/60)^2/(2*$H$1)	=A109	=D109	=C109	=A109	=D109+$M$1	=C109	=A109	=C109-D109	=A109	=((C109-D109)/C109)*100	=A109	=((R109-S109)/R109)*100	=C109-C104	=D109-D104
110	107	12.27	193	=B110/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F110/60/60)^2/(2*$H$1)	=A110	=D110	=C110	=A110	=D110+$M$1	=C110	=A110	=C110-D110	=A110	=((C110-D110)/C110)*100	=A110	=((R110-S110)/R110)*100	=C110-C105	=D110-D105
111	108	12.66	196.5	=B111/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F111/60/60)^2/(2*$H$1)	=A111	=D111	=C111	=A111	=D111+$M$1	=C111	=A111	=C111-D111	=A111	=((C111-D111)/C111)*100	=A111	=((R111-S111)/R111)*100	=C111-C106	=D111-D106
112	109	13.59	200.5	=B112/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F112/60/60)^2/(2*$H$1)	=A112	=D112	=C112	=A112	=D112+$M$1	=C112	=A112	=C112-D112	=A112	=((C112-D112)/C112)*100	=A112	=((R112-S112)/R112)*100	=C112-C107	=D112-D107
113	110	13.49	206.5	=B113/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F113/60/60)^2/(2*$H$1)	=A113	=D113	=C113	=A113	=D113+$M$1	=C113	=A113	=C113-D113	=A113	=((C113-D113)/C113)*100	=A113	=((R113-S113)/R113)*100	=C113-C108	=D113-D108
114	111	13.47	207.5	=B114/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F114/60/60)^2/(2*$H$1)	=A114	=D114	=C114	=A114	=D114+$M$1	=C114	=A114	=C114-D114	=A114	=((C114-D114)/C114)*100	=A114	=((R114-S114)/R114)*100	=C114-C109	=D114-D109
115	112	14.34	212	=B115/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F115/60/60)^2/(2*$H$1)	=A115	=D115	=C115	=A115	=D115+$M$1	=C115	=A115	=C115-D115	=A115	=((C115-D115)/C115)*100	=A115	=((R115-S115)/R115)*100	=C115-C110	=D115-D110
116	113	13.88	216.5	=B116/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F116/60/60)^2/(2*$H$1)	=A116	=D116	=C116	=A116	=D116+$M$1	=C116	=A116	=C116-D116	=A116	=((C116-D116)/C116)*100	=A116	=((R116-S116)/R116)*100	=C116-C111	=D116-D111
117	114	14.94	221.5	=B117/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F117/60/60)^2/(2*$H$1)	=A117	=D117	=C117	=A117	=D117+$M$1	=C117	=A117	=C117-D117	=A117	=((C117-D117)/C117)*100	=A117	=((R117-S117)/R117)*100	=C117-C112	=D117-D112
118	115	14.34	225.5	=B118/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F118/60/60)^2/(2*$H$1)	=A118	=D118	=C118	=A118	=D118+$M$1	=C118	=A118	=C118-D118	=A118	=((C118-D118)/C118)*100	=A118	=((R118-S118)/R118)*100	=C118-C113	=D118-D113
119	116	14.69	228	=B119/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F119/60/60)^2/(2*$H$1)	=A119	=D119	=C119	=A119	=D119+$M$1	=C119	=A119	=C119-D119	=A119	=((C119-D119)/C119)*100	=A119	=((R119-S119)/R119)*100	=C119-C114	=D119-D114
120	117	15.06	232.5	=B120/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F120/60/60)^2/(2*$H$1)	=A120	=D120	=C120	=A120	=D120+$M$1	=C120	=A120	=C120-D120	=A120	=((C120-D120)/C120)*100	=A120	=((R120-S120)/R120)*100	=C120-C115	=D120-D115
121	118	15.07	234.5	=B121/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F121/60/60)^2/(2*$H$1)	=A121	=D121	=C121	=A121	=D121+$M$1	=C121	=A121	=C121-D121	=A121	=((C121-D121)/C121)*100	=A121	=((R121-S121)/R121)*100	=C121-C116	=D121-D116
122	119	15.28	239	=B122/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F122/60/60)^2/(2*$H$1)	=A122	=D122	=C122	=A122	=D122+$M$1	=C122	=A122	=C122-D122	=A122	=((C122-D122)/C122)*100	=A122	=((R122-S122)/R122)*100	=C122-C117	=D122-D117
123	120	15.55	242.5	=B123/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F123/60/60)^2/(2*$H$1)	=A123	=D123	=C123	=A123	=D123+$M$1	=C123	=A123	=C123-D123	=A123	=((C123-D123)/C123)*100	=A123	=((R123-S123)/R123)*100	=C123-C118	=D123-D118
124	121	15.99	247.5	=B124/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F124/60/60)^2/(2*$H$1)	=A124	=D124	=C124	=A124	=D124+$M$1	=C124	=A124	=C124-D124	=A124	=((C124-D124)/C124)*100	=A124	=((R124-S124)/R124)*100	=C124-C119	=D124-D119
125	122	15.93	249	=B125/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F125/60/60)^2/(2*$H$1)	=A125	=D125	=C125	=A125	=D125+$M$1	=C125	=A125	=C125-D125	=A125	=((C125-D125)/C125)*100	=A125	=((R125-S125)/R125)*100	=C125-C120	=D125-D120
126	123	16.08	253	=B126/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F126/60/60)^2/(2*$H$1)	=A126	=D126	=C126	=A126	=D126+$M$1	=C126	=A126	=C126-D126	=A126	=((C126-D126)/C126)*100	=A126	=((R126-S126)/R126)*100	=C126-C121	=D126-D121
127	124	17.37	257.5	=B127/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F127/60/60)^2/(2*$H$1)	=A127	=D127	=C127	=A127	=D127+$M$1	=C127	=A127	=C127-D127	=A127	=((C127-D127)/C127)*100	=A127	=((R127-S127)/R127)*100	=C127-C122	=D127-D122
128	125	17.98	261.5	=B128/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F128/60/60)^2/(2*$H$1)	=A128	=D128	=C128	=A128	=D128+$M$1	=C128	=A128	=C128-D128	=A128	=((C128-D128)/C128)*100	=A128	=((R128-S128)/R128)*100	=C128-C123	=D128-D123
129	126	17.56	266.5	=B129/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F129/60/60)^2/(2*$H$1)	=A129	=D129	=C129	=A129	=D129+$M$1	=C129	=A129	=C129-D129	=A129	=((C129-D129)/C129)*100	=A129	=((R129-S129)/R129)*100	=C129-C124	=D129-D124
130	127	17.9	271.5	=B130/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F130/60/60)^2/(2*$H$1)	=A130	=D130	=C130	=A130	=D130+$M$1	=C130	=A130	=C130-D130	=A130	=((C130-D130)/C130)*100	=A130	=((R130-S130)/R130)*100	=C130-C125	=D130-D125
131	128	17.77	274.5	=B131/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F131/60/60)^2/(2*$H$1)	=A131	=D131	=C131	=A131	=D131+$M$1	=C131	=A131	=C131-D131	=A131	=((C131-D131)/C131)*100	=A131	=((R131-S131)/R131)*100	=C131-C126	=D131-D126
132	129	18.78	275	=B132/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F132/60/60)^2/(2*$H$1)	=A132	=D132	=C132	=A132	=D132+$M$1	=C132	=A132	=C132-D132	=A132	=((C132-D132)/C132)*100	=A132	=((R132-S132)/R132)*100	=C132-C127	=D132-D127
133	130	18.98	278	=B133/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F133/60/60)^2/(2*$H$1)	=A133	=D133	=C133	=A133	=D133+$M$1	=C133	=A133	=C133-D133	=A133	=((C133-D133)/C133)*100	=A133	=((R133-S133)/R133)*100	=C133-C128	=D133-D128
134	131	18.9	282	=B134/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F134/60/60)^2/(2*$H$1)	=A134	=D134	=C134	=A134	=D134+$M$1	=C134	=A134	=C134-D134	=A134	=((C134-D134)/C134)*100	=A134	=((R134-S134)/R134)*100	=C134-C129	=D134-D129
135	132	19.17	283.5	=B135/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F135/60/60)^2/(2*$H$1)	=A135	=D135	=C135	=A135	=D135+$M$1	=C135	=A135	=C135-D135	=A135	=((C135-D135)/C135)*100	=A135	=((R135-S135)/R135)*100	=C135-C130	=D135-D130
136	133	19.26	287	=B136/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F136/60/60)^2/(2*$H$1)	=A136	=D136	=C136	=A136	=D136+$M$1	=C136	=A136	=C136-D136	=A136	=((C136-D136)/C136)*100	=A136	=((R136-S136)/R136)*100	=C136-C131	=D136-D131
137	134	20.16	290.5	=B137/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F137/60/60)^2/(2*$H$1)	=A137	=D137	=C137	=A137	=D137+$M$1	=C137	=A137	=C137-D137	=A137	=((C137-D137)/C137)*100	=A137	=((R137-S137)/R137)*100	=C137-C132	=D137-D132
138	135	19.78	293.5	=B138/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F138/60/60)^2/(2*$H$1)	=A138	=D138	=C138	=A138	=D138+$M$1	=C138	=A138	=C138-D138	=A138	=((C138-D138)/C138)*100	=A138	=((R138-S138)/R138)*100	=C138-C133	=D138-D133
139	136	20.68	294	=B139/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F139/60/60)^2/(2*$H$1)	=A139	=D139	=C139	=A139	=D139+$M$1	=C139	=A139	=C139-D139	=A139	=((C139-D139)/C139)*100	=A139	=((R139-S139)/R139)*100	=C139-C134	=D139-D134
140	137	20.83	304.5	=B140/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F140/60/60)^2/(2*$H$1)	=A140	=D140	=C140	=A140	=D140+$M$1	=C140	=A140	=C140-D140	=A140	=((C140-D140)/C140)*100	=A140	=((R140-S140)/R140)*100	=C140-C135	=D140-D135
141	138	20.71	307.5	=B141/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F141/60/60)^2/(2*$H$1)	=A141	=D141	=C141	=A141	=D141+$M$1	=C141	=A141	=C141-D141	=A141	=((C141-D141)/C141)*100	=A141	=((R141-S141)/R141)*100	=C141-C136	=D141-D136
142	139	21.58	314.5	=B142/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F142/60/60)^2/(2*$H$1)	=A142	=D142	=C142	=A142	=D142+$M$1	=C142	=A142	=C142-D142	=A142	=((C142-D142)/C142)*100	=A142	=((R142-S142)/R142)*100	=C142-C137	=D142-D137
143	140	22.53	316.5	=B143/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F143/60/60)^2/(2*$H$1)	=A143	=D143	=C143	=A143	=D143+$M$1	=C143	=A143	=C143-D143	=A143	=((C143-D143)/C143)*100	=A143	=((R143-S143)/R143)*100	=C143-C138	=D143-D138
144	141	21.63	322	=B144/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F144/60/60)^2/(2*$H$1)	=A144	=D144	=C144	=A144	=D144+$M$1	=C144	=A144	=C144-D144	=A144	=((C144-D144)/C144)*100	=A144	=((R144-S144)/R144)*100	=C144-C139	=D144-D139
145	142	22.36	327.5	=B145/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F145/60/60)^2/(2*$H$1)	=A145	=D145	=C145	=A145	=D145+$M$1	=C145	=A145	=C145-D145	=A145	=((C145-D145)/C145)*100	=A145	=((R145-S145)/R145)*100	=C145-C140	=D145-D140
146	143	22.48	333	=B146/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F146/60/60)^2/(2*$H$1)	=A146	=D146	=C146	=A146	=D146+$M$1	=C146	=A146	=C146-D146	=A146	=((C146-D146)/C146)*100	=A146	=((R146-S146)/R146)*100	=C146-C141	=D146-D141
147	144	23.24	336.5	=B147/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F147/60/60)^2/(2*$H$1)	=A147	=D147	=C147	=A147	=D147+$M$1	=C147	=A147	=C147-D147	=A147	=((C147-D147)/C147)*100	=A147	=((R147-S147)/R147)*100	=C147-C142	=D147-D142
148	145	23.06	341.5	=B148/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F148/60/60)^2/(2*$H$1)	=A148	=D148	=C148	=A148	=D148+$M$1	=C148	=A148	=C148-D148	=A148	=((C148-D148)/C148)*100	=A148	=((R148-S148)/R148)*100	=C148-C143	=D148-D143
149	146	24.13	342.5	=B149/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F149/60/60)^2/(2*$H$1)	=A149	=D149	=C149	=A149	=D149+$M$1	=C149	=A149	=C149-D149	=A149	=((C149-D149)/C149)*100	=A149	=((R149-S149)/R149)*100	=C149-C144	=D149-D144
150	147	23.91	344.5	=B150/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F150/60/60)^2/(2*$H$1)	=A150	=D150	=C150	=A150	=D150+$M$1	=C150	=A150	=C150-D150	=A150	=((C150-D150)/C150)*100	=A150	=((R150-S150)/R150)*100	=C150-C145	=D150-D145
151	148	24.41	347	=B151/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F151/60/60)^2/(2*$H$1)	=A151	=D151	=C151	=A151	=D151+$M$1	=C151	=A151	=C151-D151	=A151	=((C151-D151)/C151)*100	=A151	=((R151-S151)/R151)*100	=C151-C146	=D151-D146
152	149	24.52	355.5	=B152/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F152/60/60)^2/(2*$H$1)	=A152	=D152	=C152	=A152	=D152+$M$1	=C152	=A152	=C152-D152	=A152	=((C152-D152)/C152)*100	=A152	=((R152-S152)/R152)*100	=C152-C147	=D152-D147
153	150	25.19	359	=B153/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F153/60/60)^2/(2*$H$1)	=A153	=D153	=C153	=A153	=D153+$M$1	=C153	=A153	=C153-D153	=A153	=((C153-D153)/C153)*100	=A153	=((R153-S153)/R153)*100	=C153-C148	=D153-D148
154	151	23.94	361	=B154/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F154/60/60)^2/(2*$H$1)	=A154	=D154	=C154	=A154	=D154+$M$1	=C154	=A154	=C154-D154	=A154	=((C154-D154)/C154)*100	=A154	=((R154-S154)/R154)*100	=C154-C149	=D154-D149
155	152	24.5	365.5	=B155/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F155/60/60)^2/(2*$H$1)	=A155	=D155	=C155	=A155	=D155+$M$1	=C155	=A155	=C155-D155	=A155	=((C155-D155)/C155)*100	=A155	=((R155-S155)/R155)*100	=C155-C150	=D155-D150
156	153	25.54	374.5	=B156/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F156/60/60)^2/(2*$H$1)	=A156	=D156	=C156	=A156	=D156+$M$1	=C156	=A156	=C156-D156	=A156	=((C156-D156)/C156)*100	=A156	=((R156-S156)/R156)*100	=C156-C151	=D156-D151
157	154	26.28	379	=B157/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F157/60/60)^2/(2*$H$1)	=A157	=D157	=C157	=A157	=D157+$M$1	=C157	=A157	=C157-D157	=A157	=((C157-D157)/C157)*100	=A157	=((R157-S157)/R157)*100	=C157-C152	=D157-D152
158	155	25.97	383	=B158/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F158/60/60)^2/(2*$H$1)	=A158	=D158	=C158	=A158	=D158+$M$1	=C158	=A158	=C158-D158	=A158	=((C158-D158)/C158)*100	=A158	=((R158-S158)/R158)*100	=C158-C153	=D158-D153
159	156	26.37	389	=B159/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F159/60/60)^2/(2*$H$1)	=A159	=D159	=C159	=A159	=D159+$M$1	=C159	=A159	=C159-D159	=A159	=((C159-D159)/C159)*100	=A159	=((R159-S159)/R159)*100	=C159-C154	=D159-D154
160	157	26.25	392	=B160/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F160/60/60)^2/(2*$H$1)	=A160	=D160	=C160	=A160	=D160+$M$1	=C160	=A160	=C160-D160	=A160	=((C160-D160)/C160)*100	=A160	=((R160-S160)/R160)*100	=C160-C155	=D160-D155
161	158	28.1	394	=B161/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F161/60/60)^2/(2*$H$1)	=A161	=D161	=C161	=A161	=D161+$M$1	=C161	=A161	=C161-D161	=A161	=((C161-D161)/C161)*100	=A161	=((R161-S161)/R161)*100	=C161-C156	=D161-D156
162	159	27.97	400	=B162/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F162/60/60)^2/(2*$H$1)	=A162	=D162	=C162	=A162	=D162+$M$1	=C162	=A162	=C162-D162	=A162	=((C162-D162)/C162)*100	=A162	=((R162-S162)/R162)*100	=C162-C157	=D162-D157
163	160	28.89	406.5	=B163/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F163/60/60)^2/(2*$H$1)	=A163	=D163	=C163	=A163	=D163+$M$1	=C163	=A163	=C163-D163	=A163	=((C163-D163)/C163)*100	=A163	=((R163-S163)/R163)*100	=C163-C158	=D163-D158
164	161	28.43	413	=B164/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F164/60/60)^2/(2*$H$1)	=A164	=D164	=C164	=A164	=D164+$M$1	=C164	=A164	=C164-D164	=A164	=((C164-D164)/C164)*100	=A164	=((R164-S164)/R164)*100	=C164-C159	=D164-D159
165	162	28.89	416.5	=B165/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F165/60/60)^2/(2*$H$1)	=A165	=D165	=C165	=A165	=D165+$M$1	=C165	=A165	=C165-D165	=A165	=((C165-D165)/C165)*100	=A165	=((R165-S165)/R165)*100	=C165-C160	=D165-D160
166	163	29.15	421.5	=B166/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F166/60/60)^2/(2*$H$1)	=A166	=D166	=C166	=A166	=D166+$M$1	=C166	=A166	=C166-D166	=A166	=((C166-D166)/C166)*100	=A166	=((R166-S166)/R166)*100	=C166-C161	=D166-D161
167	164	29.19	431	=B167/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F167/60/60)^2/(2*$H$1)	=A167	=D167	=C167	=A167	=D167+$M$1	=C167	=A167	=C167-D167	=A167	=((C167-D167)/C167)*100	=A167	=((R167-S167)/R167)*100	=C167-C162	=D167-D162
168	165	29.95	436.5	=B168/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F168/60/60)^2/(2*$H$1)	=A168	=D168	=C168	=A168	=D168+$M$1	=C168	=A168	=C168-D168	=A168	=((C168-D168)/C168)*100	=A168	=((R168-S168)/R168)*100	=C168-C163	=D168-D163
169	166	31.01	440	=B169/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F169/60/60)^2/(2*$H$1)	=A169	=D169	=C169	=A169	=D169+$M$1	=C169	=A169	=C169-D169	=A169	=((C169-D169)/C169)*100	=A169	=((R169-S169)/R169)*100	=C169-C164	=D169-D164
170	167	32.6	446	=B170/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F170/60/60)^2/(2*$H$1)	=A170	=D170	=C170	=A170	=D170+$M$1	=C170	=A170	=C170-D170	=A170	=((C170-D170)/C170)*100	=A170	=((R170-S170)/R170)*100	=C170-C165	=D170-D165
171	168	32.28	452.5	=B171/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F171/60/60)^2/(2*$H$1)	=A171	=D171	=C171	=A171	=D171+$M$1	=C171	=A171	=C171-D171	=A171	=((C171-D171)/C171)*100	=A171	=((R171-S171)/R171)*100	=C171-C166	=D171-D166
172	169	30.3	459	=B172/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F172/60/60)^2/(2*$H$1)	=A172	=D172	=C172	=A172	=D172+$M$1	=C172	=A172	=C172-D172	=A172	=((C172-D172)/C172)*100	=A172	=((R172-S172)/R172)*100	=C172-C167	=D172-D167
173	170	32.43	466	=B173/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F173/60/60)^2/(2*$H$1)	=A173	=D173	=C173	=A173	=D173+$M$1	=C173	=A173	=C173-D173	=A173	=((C173-D173)/C173)*100	=A173	=((R173-S173)/R173)*100	=C173-C168	=D173-D168
174	171	31.7	473	=B174/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F174/60/60)^2/(2*$H$1)	=A174	=D174	=C174	=A174	=D174+$M$1	=C174	=A174	=C174-D174	=A174	=((C174-D174)/C174)*100	=A174	=((R174-S174)/R174)*100	=C174-C169	=D174-D169
175	172	32.42	477.5	=B175/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F175/60/60)^2/(2*$H$1)	=A175	=D175	=C175	=A175	=D175+$M$1	=C175	=A175	=C175-D175	=A175	=((C175-D175)/C175)*100	=A175	=((R175-S175)/R175)*100	=C175-C170	=D175-D170
176	173	33.26	484.5	=B176/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F176/60/60)^2/(2*$H$1)	=A176	=D176	=C176	=A176	=D176+$M$1	=C176	=A176	=C176-D176	=A176	=((C176-D176)/C176)*100	=A176	=((R176-S176)/R176)*100	=C176-C171	=D176-D171
177	174	32.67	486.5	=B177/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F177/60/60)^2/(2*$H$1)	=A177	=D177	=C177	=A177	=D177+$M$1	=C177	=A177	=C177-D177	=A177	=((C177-D177)/C177)*100	=A177	=((R177-S177)/R177)*100	=C177-C172	=D177-D172
178	175	33.23	489.5	=B178/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F178/60/60)^2/(2*$H$1)	=A178	=D178	=C178	=A178	=D178+$M$1	=C178	=A178	=C178-D178	=A178	=((C178-D178)/C178)*100	=A178	=((R178-S178)/R178)*100	=C178-C173	=D178-D173
179	176	35.67	493.5	=B179/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F179/60/60)^2/(2*$H$1)	=A179	=D179	=C179	=A179	=D179+$M$1	=C179	=A179	=C179-D179	=A179	=((C179-D179)/C179)*100	=A179	=((R179-S179)/R179)*100	=C179-C174	=D179-D174
180	177	34.86	494.5	=B180/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F180/60/60)^2/(2*$H$1)	=A180	=D180	=C180	=A180	=D180+$M$1	=C180	=A180	=C180-D180	=A180	=((C180-D180)/C180)*100	=A180	=((R180-S180)/R180)*100	=C180-C175	=D180-D175
181	178	35.42	501	=B181/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F181/60/60)^2/(2*$H$1)	=A181	=D181	=C181	=A181	=D181+$M$1	=C181	=A181	=C181-D181	=A181	=((C181-D181)/C181)*100	=A181	=((R181-S181)/R181)*100	=C181-C176	=D181-D176
182	179	35.31	508.5	=B182/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F182/60/60)^2/(2*$H$1)	=A182	=D182	=C182	=A182	=D182+$M$1	=C182	=A182	=C182-D182	=A182	=((C182-D182)/C182)*100	=A182	=((R182-S182)/R182)*100	=C182-C177	=D182-D177
183	180	34.94	516	=B183/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F183/60/60)^2/(2*$H$1)	=A183	=D183	=C183	=A183	=D183+$M$1	=C183	=A183	=C183-D183	=A183	=((C183-D183)/C183)*100	=A183	=((R183-S183)/R183)*100	=C183-C178	=D183-D178
184	181	37.44	520	=B184/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F184/60/60)^2/(2*$H$1)	=A184	=D184	=C184	=A184	=D184+$M$1	=C184	=A184	=C184-D184	=A184	=((C184-D184)/C184)*100	=A184	=((R184-S184)/R184)*100	=C184-C179	=D184-D179
185	182	35.29	526.5	=B185/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F185/60/60)^2/(2*$H$1)	=A185	=D185	=C185	=A185	=D185+$M$1	=C185	=A185	=C185-D185	=A185	=((C185-D185)/C185)*100	=A185	=((R185-S185)/R185)*100	=C185-C180	=D185-D180
186	183	35.92	530.5	=B186/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F186/60/60)^2/(2*$H$1)	=A186	=D186	=C186	=A186	=D186+$M$1	=C186	=A186	=C186-D186	=A186	=((C186-D186)/C186)*100	=A186	=((R186-S186)/R186)*100	=C186-C181	=D186-D181
187	184	37.24	538.5	=B187/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F187/60/60)^2/(2*$H$1)	=A187	=D187	=C187	=A187	=D187+$M$1	=C187	=A187	=C187-D187	=A187	=((C187-D187)/C187)*100	=A187	=((R187-S187)/R187)*100	=C187-C182	=D187-D182
188	185	38.34	545.5	=B188/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F188/60/60)^2/(2*$H$1)	=A188	=D188	=C188	=A188	=D188+$M$1	=C188	=A188	=C188-D188	=A188	=((C188-D188)/C188)*100	=A188	=((R188-S188)/R188)*100	=C188-C183	=D188-D183
189	186	37.89	552	=B189/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F189/60/60)^2/(2*$H$1)	=A189	=D189	=C189	=A189	=D189+$M$1	=C189	=A189	=C189-D189	=A189	=((C189-D189)/C189)*100	=A189	=((R189-S189)/R189)*100	=C189-C184	=D189-D184
190	187	39.57	559	=B190/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F190/60/60)^2/(2*$H$1)	=A190	=D190	=C190	=A190	=D190+$M$1	=C190	=A190	=C190-D190	=A190	=((C190-D190)/C190)*100	=A190	=((R190-S190)/R190)*100	=C190-C185	=D190-D185
191	188	39.55	564	=B191/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F191/60/60)^2/(2*$H$1)	=A191	=D191	=C191	=A191	=D191+$M$1	=C191	=A191	=C191-D191	=A191	=((C191-D191)/C191)*100	=A191	=((R191-S191)/R191)*100	=C191-C186	=D191-D186
192	189	39.51	564.5	=B192/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F192/60/60)^2/(2*$H$1)	=A192	=D192	=C192	=A192	=D192+$M$1	=C192	=A192	=C192-D192	=A192	=((C192-D192)/C192)*100	=A192	=((R192-S192)/R192)*100	=C192-C187	=D192-D187
193	190	40.43	567.5	=B193/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F193/60/60)^2/(2*$H$1)	=A193	=D193	=C193	=A193	=D193+$M$1	=C193	=A193	=C193-D193	=A193	=((C193-D193)/C193)*100	=A193	=((R193-S193)/R193)*100	=C193-C188	=D193-D188
194	191	40.16	575	=B194/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F194/60/60)^2/(2*$H$1)	=A194	=D194	=C194	=A194	=D194+$M$1	=C194	=A194	=C194-D194	=A194	=((C194-D194)/C194)*100	=A194	=((R194-S194)/R194)*100	=C194-C189	=D194-D189
195	192	40.84	585.5	=B195/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F195/60/60)^2/(2*$H$1)	=A195	=D195	=C195	=A195	=D195+$M$1	=C195	=A195	=C195-D195	=A195	=((C195-D195)/C195)*100	=A195	=((R195-S195)/R195)*100	=C195-C190	=D195-D190
196	193	41.64	588.5	=B196/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F196/60/60)^2/(2*$H$1)	=A196	=D196	=C196	=A196	=D196+$M$1	=C196	=A196	=C196-D196	=A196	=((C196-D196)/C196)*100	=A196	=((R196-S196)/R196)*100	=C196-C191	=D196-D191
197	194	41.58	592.5	=B197/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F197/60/60)^2/(2*$H$1)	=A197	=D197	=C197	=A197	=D197+$M$1	=C197	=A197	=C197-D197	=A197	=((C197-D197)/C197)*100	=A197	=((R197-S197)/R197)*100	=C197-C192	=D197-D192
198	195	42.42	602.5	=B198/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F198/60/60)^2/(2*$H$1)	=A198	=D198	=C198	=A198	=D198+$M$1	=C198	=A198	=C198-D198	=A198	=((C198-D198)/C198)*100	=A198	=((R198-S198)/R198)*100	=C198-C193	=D198-D193
199	196	42.71	610	=B199/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F199/60/60)^2/(2*$H$1)	=A199	=D199	=C199	=A199	=D199+$M$1	=C199	=A199	=C199-D199	=A199	=((C199-D199)/C199)*100	=A199	=((R199-S199)/R199)*100	=C199-C194	=D199-D194
200	197	44.08	616	=B200/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F200/60/60)^2/(2*$H$1)	=A200	=D200	=C200	=A200	=D200+$M$1	=C200	=A200	=C200-D200	=A200	=((C200-D200)/C200)*100	=A200	=((R200-S200)/R200)*100	=C200-C195	=D200-D195
201	198	43.85	622	=B201/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F201/60/60)^2/(2*$H$1)	=A201	=D201	=C201	=A201	=D201+$M$1	=C201	=A201	=C201-D201	=A201	=((C201-D201)/C201)*100	=A201	=((R201-S201)/R201)*100	=C201-C196	=D201-D196
202	199	43.06	628	=B202/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F202/60/60)^2/(2*$H$1)	=A202	=D202	=C202	=A202	=D202+$M$1	=C202	=A202	=C202-D202	=A202	=((C202-D202)/C202)*100	=A202	=((R202-S202)/R202)*100	=C202-C197	=D202-D197
203	200	43.67	631.5	=B203/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F203/60/60)^2/(2*$H$1)	=A203	=D203	=C203	=A203	=D203+$M$1	=C203	=A203	=C203-D203	=A203	=((C203-D203)/C203)*100	=A203	=((R203-S203)/R203)*100	=C203-C198	=D203-D198
204	201	45.75	640	=B204/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F204/60/60)^2/(2*$H$1)	=A204	=D204	=C204	=A204	=D204+$M$1	=C204	=A204	=C204-D204	=A204	=((C204-D204)/C204)*100	=A204	=((R204-S204)/R204)*100	=C204-C199	=D204-D199
205	202	45.12	645	=B205/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F205/60/60)^2/(2*$H$1)	=A205	=D205	=C205	=A205	=D205+$M$1	=C205	=A205	=C205-D205	=A205	=((C205-D205)/C205)*100	=A205	=((R205-S205)/R205)*100	=C205-C200	=D205-D200
206	203	44.92	654.5	=B206/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F206/60/60)^2/(2*$H$1)	=A206	=D206	=C206	=A206	=D206+$M$1	=C206	=A206	=C206-D206	=A206	=((C206-D206)/C206)*100	=A206	=((R206-S206)/R206)*100	=C206-C201	=D206-D201
207	204	46.33	659	=B207/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F207/60/60)^2/(2*$H$1)	=A207	=D207	=C207	=A207	=D207+$M$1	=C207	=A207	=C207-D207	=A207	=((C207-D207)/C207)*100	=A207	=((R207-S207)/R207)*100	=C207-C202	=D207-D202
208	205	47.28	667	=B208/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F208/60/60)^2/(2*$H$1)	=A208	=D208	=C208	=A208	=D208+$M$1	=C208	=A208	=C208-D208	=A208	=((C208-D208)/C208)*100	=A208	=((R208-S208)/R208)*100	=C208-C203	=D208-D203
209	206	47.36	674	=B209/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F209/60/60)^2/(2*$H$1)	=A209	=D209	=C209	=A209	=D209+$M$1	=C209	=A209	=C209-D209	=A209	=((C209-D209)/C209)*100	=A209	=((R209-S209)/R209)*100	=C209-C204	=D209-D204
210	207	47.19	679.5	=B210/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F210/60/60)^2/(2*$H$1)	=A210	=D210	=C210	=A210	=D210+$M$1	=C210	=A210	=C210-D210	=A210	=((C210-D210)/C210)*100	=A210	=((R210-S210)/R210)*100	=C210-C205	=D210-D205
211	208	48.56	687	=B211/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F211/60/60)^2/(2*$H$1)	=A211	=D211	=C211	=A211	=D211+$M$1	=C211	=A211	=C211-D211	=A211	=((C211-D211)/C211)*100	=A211	=((R211-S211)/R211)*100	=C211-C206	=D211-D206
212	209	48.22	694.5	=B212/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F212/60/60)^2/(2*$H$1)	=A212	=D212	=C212	=A212	=D212+$M$1	=C212	=A212	=C212-D212	=A212	=((C212-D212)/C212)*100	=A212	=((R212-S212)/R212)*100	=C212-C207	=D212-D207
213	210	50.34	705	=B213/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F213/60/60)^2/(2*$H$1)	=A213	=D213	=C213	=A213	=D213+$M$1	=C213	=A213	=C213-D213	=A213	=((C213-D213)/C213)*100	=A213	=((R213-S213)/R213)*100	=C213-C208	=D213-D208
214	211	49.16	714.5	=B214/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F214/60/60)^2/(2*$H$1)	=A214	=D214	=C214	=A214	=D214+$M$1	=C214	=A214	=C214-D214	=A214	=((C214-D214)/C214)*100	=A214	=((R214-S214)/R214)*100	=C214-C209	=D214-D209
215	212	49.67	719	=B215/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F215/60/60)^2/(2*$H$1)	=A215	=D215	=C215	=A215	=D215+$M$1	=C215	=A215	=C215-D215	=A215	=((C215-D215)/C215)*100	=A215	=((R215-S215)/R215)*100	=C215-C210	=D215-D210
216	213	50	725.5	=B216/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F216/60/60)^2/(2*$H$1)	=A216	=D216	=C216	=A216	=D216+$M$1	=C216	=A216	=C216-D216	=A216	=((C216-D216)/C216)*100	=A216	=((R216-S216)/R216)*100	=C216-C211	=D216-D211
217	214	51.43	732.5	=B217/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F217/60/60)^2/(2*$H$1)	=A217	=D217	=C217	=A217	=D217+$M$1	=C217	=A217	=C217-D217	=A217	=((C217-D217)/C217)*100	=A217	=((R217-S217)/R217)*100	=C217-C212	=D217-D212
218	215	51.25	736.5	=B218/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F218/60/60)^2/(2*$H$1)	=A218	=D218	=C218	=A218	=D218+$M$1	=C218	=A218	=C218-D218	=A218	=((C218-D218)/C218)*100	=A218	=((R218-S218)/R218)*100	=C218-C213	=D218-D213
219	216	52.59	747.5	=B219/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F219/60/60)^2/(2*$H$1)	=A219	=D219	=C219	=A219	=D219+$M$1	=C219	=A219	=C219-D219	=A219	=((C219-D219)/C219)*100	=A219	=((R219-S219)/R219)*100	=C219-C214	=D219-D214

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
220	217	52.38	753.5	=B220/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F220/60/60)^2/(2*$H$1)	=A220	=D220	=C220	=A220	=D220+$M$1	=C220	=A220	=C220-D220	=A220	=((C220-D220)/C220)*100	=A220	=((R220-S220)/R220)*100	=C220-C215	=D220-D215
221	218	52.1	758.5	=B221/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F221/60/60)^2/(2*$H$1)	=A221	=D221	=C221	=A221	=D221+$M$1	=C221	=A221	=C221-D221	=A221	=((C221-D221)/C221)*100	=A221	=((R221-S221)/R221)*100	=C221-C216	=D221-D216
222	219	53.75	765	=B222/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F222/60/60)^2/(2*$H$1)	=A222	=D222	=C222	=A222	=D222+$M$1	=C222	=A222	=C222-D222	=A222	=((C222-D222)/C222)*100	=A222	=((R222-S222)/R222)*100	=C222-C217	=D222-D217
223	220	53.87	772	=B223/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F223/60/60)^2/(2*$H$1)	=A223	=D223	=C223	=A223	=D223+$M$1	=C223	=A223	=C223-D223	=A223	=((C223-D223)/C223)*100	=A223	=((R223-S223)/R223)*100	=C223-C218	=D223-D218
224	221	54.19	780.5	=B224/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F224/60/60)^2/(2*$H$1)	=A224	=D224	=C224	=A224	=D224+$M$1	=C224	=A224	=C224-D224	=A224	=((C224-D224)/C224)*100	=A224	=((R224-S224)/R224)*100	=C224-C219	=D224-D219
225	222	55.48	787.5	=B225/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F225/60/60)^2/(2*$H$1)	=A225	=D225	=C225	=A225	=D225+$M$1	=C225	=A225	=C225-D225	=A225	=((C225-D225)/C225)*100	=A225	=((R225-S225)/R225)*100	=C225-C220	=D225-D220
226	223	54.83	794	=B226/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F226/60/60)^2/(2*$H$1)	=A226	=D226	=C226	=A226	=D226+$M$1	=C226	=A226	=C226-D226	=A226	=((C226-D226)/C226)*100	=A226	=((R226-S226)/R226)*100	=C226-C221	=D226-D221
227	224	54.73	804	=B227/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F227/60/60)^2/(2*$H$1)	=A227	=D227	=C227	=A227	=D227+$M$1	=C227	=A227	=C227-D227	=A227	=((C227-D227)/C227)*100	=A227	=((R227-S227)/R227)*100	=C227-C222	=D227-D222
228	225	57.14	810	=B228/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F228/60/60)^2/(2*$H$1)	=A228	=D228	=C228	=A228	=D228+$M$1	=C228	=A228	=C228-D228	=A228	=((C228-D228)/C228)*100	=A228	=((R228-S228)/R228)*100	=C228-C223	=D228-D223
229	226	57.3	821.5	=B229/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F229/60/60)^2/(2*$H$1)	=A229	=D229	=C229	=A229	=D229+$M$1	=C229	=A229	=C229-D229	=A229	=((C229-D229)/C229)*100	=A229	=((R229-S229)/R229)*100	=C229-C224	=D229-D224
230	227	57.17	825	=B230/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F230/60/60)^2/(2*$H$1)	=A230	=D230	=C230	=A230	=D230+$M$1	=C230	=A230	=C230-D230	=A230	=((C230-D230)/C230)*100	=A230	=((R230-S230)/R230)*100	=C230-C225	=D230-D225
231	228	57.44	829.5	=B231/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F231/60/60)^2/(2*$H$1)	=A231	=D231	=C231	=A231	=D231+$M$1	=C231	=A231	=C231-D231	=A231	=((C231-D231)/C231)*100	=A231	=((R231-S231)/R231)*100	=C231-C226	=D231-D226
232	229	57.73	836	=B232/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F232/60/60)^2/(2*$H$1)	=A232	=D232	=C232	=A232	=D232+$M$1	=C232	=A232	=C232-D232	=A232	=((C232-D232)/C232)*100	=A232	=((R232-S232)/R232)*100	=C232-C227	=D232-D227
233	230	58.3	843.5	=B233/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F233/60/60)^2/(2*$H$1)	=A233	=D233	=C233	=A233	=D233+$M$1	=C233	=A233	=C233-D233	=A233	=((C233-D233)/C233)*100	=A233	=((R233-S233)/R233)*100	=C233-C228	=D233-D228
234	231	59.64	850.5	=B234/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F234/60/60)^2/(2*$H$1)	=A234	=D234	=C234	=A234	=D234+$M$1	=C234	=A234	=C234-D234	=A234	=((C234-D234)/C234)*100	=A234	=((R234-S234)/R234)*100	=C234-C229	=D234-D229
235	232	59.63	854.5	=B235/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F235/60/60)^2/(2*$H$1)	=A235	=D235	=C235	=A235	=D235+$M$1	=C235	=A235	=C235-D235	=A235	=((C235-D235)/C235)*100	=A235	=((R235-S235)/R235)*100	=C235-C230	=D235-D230
236	233	60.03	860.5	=B236/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F236/60/60)^2/(2*$H$1)	=A236	=D236	=C236	=A236	=D236+$M$1	=C236	=A236	=C236-D236	=A236	=((C236-D236)/C236)*100	=A236	=((R236-S236)/R236)*100	=C236-C231	=D236-D231
237	234	62.47	863.5	=B237/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F237/60/60)^2/(2*$H$1)	=A237	=D237	=C237	=A237	=D237+$M$1	=C237	=A237	=C237-D237	=A237	=((C237-D237)/C237)*100	=A237	=((R237-S237)/R237)*100	=C237-C232	=D237-D232
238	235	60.7	871	=B238/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F238/60/60)^2/(2*$H$1)	=A238	=D238	=C238	=A238	=D238+$M$1	=C238	=A238	=C238-D238	=A238	=((C238-D238)/C238)*100	=A238	=((R238-S238)/R238)*100	=C238-C233	=D238-D233
239	236	61.37	878	=B239/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F239/60/60)^2/(2*$H$1)	=A239	=D239	=C239	=A239	=D239+$M$1	=C239	=A239	=C239-D239	=A239	=((C239-D239)/C239)*100	=A239	=((R239-S239)/R239)*100	=C239-C234	=D239-D234
240	237	61.6	885	=B240/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F240/60/60)^2/(2*$H$1)	=A240	=D240	=C240	=A240	=D240+$M$1	=C240	=A240	=C240-D240	=A240	=((C240-D240)/C240)*100	=A240	=((R240-S240)/R240)*100	=C240-C235	=D240-D235
241	238	62.5	890	=B241/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F241/60/60)^2/(2*$H$1)	=A241	=D241	=C241	=A241	=D241+$M$1	=C241	=A241	=C241-D241	=A241	=((C241-D241)/C241)*100	=A241	=((R241-S241)/R241)*100	=C241-C236	=D241-D236
242	239	64.83	895	=B242/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F242/60/60)^2/(2*$H$1)	=A242	=D242	=C242	=A242	=D242+$M$1	=C242	=A242	=C242-D242	=A242	=((C242-D242)/C242)*100	=A242	=((R242-S242)/R242)*100	=C242-C237	=D242-D237
243	240	63.82	901.5	=B243/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F243/60/60)^2/(2*$H$1)	=A243	=D243	=C243	=A243	=D243+$M$1	=C243	=A243	=C243-D243	=A243	=((C243-D243)/C243)*100	=A243	=((R243-S243)/R243)*100	=C243-C238	=D243-D238
244	241	63.83	907.5	=B244/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F244/60/60)^2/(2*$H$1)	=A244	=D244	=C244	=A244	=D244+$M$1	=C244	=A244	=C244-D244	=A244	=((C244-D244)/C244)*100	=A244	=((R244-S244)/R244)*100	=C244-C239	=D244-D239
245	242	64.52	914	=B245/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F245/60/60)^2/(2*$H$1)	=A245	=D245	=C245	=A245	=D245+$M$1	=C245	=A245	=C245-D245	=A245	=((C245-D245)/C245)*100	=A245	=((R245-S245)/R245)*100	=C245-C240	=D245-D240
246	243	65.63	922	=B246/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F246/60/60)^2/(2*$H$1)	=A246	=D246	=C246	=A246	=D246+$M$1	=C246	=A246	=C246-D246	=A246	=((C246-D246)/C246)*100	=A246	=((R246-S246)/R246)*100	=C246-C241	=D246-D241
247	244	66.3	931	=B247/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F247/60/60)^2/(2*$H$1)	=A247	=D247	=C247	=A247	=D247+$M$1	=C247	=A247	=C247-D247	=A247	=((C247-D247)/C247)*100	=A247	=((R247-S247)/R247)*100	=C247-C242	=D247-D242
248	245	66.52	940	=B248/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F248/60/60)^2/(2*$H$1)	=A248	=D248	=C248	=A248	=D248+$M$1	=C248	=A248	=C248-D248	=A248	=((C248-D248)/C248)*100	=A248	=((R248-S248)/R248)*100	=C248-C243	=D248-D243
249	246	67.23	945.5	=B249/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F249/60/60)^2/(2*$H$1)	=A249	=D249	=C249	=A249	=D249+$M$1	=C249	=A249	=C249-D249	=A249	=((C249-D249)/C249)*100	=A249	=((R249-S249)/R249)*100	=C249-C244	=D249-D244
250	247	67.11	951	=B250/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F250/60/60)^2/(2*$H$1)	=A250	=D250	=C250	=A250	=D250+$M$1	=C250	=A250	=C250-D250	=A250	=((C250-D250)/C250)*100	=A250	=((R250-S250)/R250)*100	=C250-C245	=D250-D245
251	248	67.89	961	=B251/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F251/60/60)^2/(2*$H$1)	=A251	=D251	=C251	=A251	=D251+$M$1	=C251	=A251	=C251-D251	=A251	=((C251-D251)/C251)*100	=A251	=((R251-S251)/R251)*100	=C251-C246	=D251-D246
252	249	67.9	969	=B252/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F252/60/60)^2/(2*$H$1)	=A252	=D252	=C252	=A252	=D252+$M$1	=C252	=A252	=C252-D252	=A252	=((C252-D252)/C252)*100	=A252	=((R252-S252)/R252)*100	=C252-C247	=D252-D247
253	250	69.66	973.5	=B253/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F253/60/60)^2/(2*$H$1)	=A253	=D253	=C253	=A253	=D253+$M$1	=C253	=A253	=C253-D253	=A253	=((C253-D253)/C253)*100	=A253	=((R253-S253)/R253)*100	=C253-C248	=D253-D248
254	251	69.76	979	=B254/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F254/60/60)^2/(2*$H$1)	=A254	=D254	=C254	=A254	=D254+$M$1	=C254	=A254	=C254-D254	=A254	=((C254-D254)/C254)*100	=A254	=((R254-S254)/R254)*100	=C254-C249	=D254-D249
255	252	68.67	990	=B255/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F255/60/60)^2/(2*$H$1)	=A255	=D255	=C255	=A255	=D255+$M$1	=C255	=A255	=C255-D255	=A255	=((C255-D255)/C255)*100	=A255	=((R255-S255)/R255)*100	=C255-C250	=D255-D250
256	253	70.59	994.5	=B256/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F256/60/60)^2/(2*$H$1)	=A256	=D256	=C256	=A256	=D256+$M$1	=C256	=A256	=C256-D256	=A256	=((C256-D256)/C256)*100	=A256	=((R256-S256)/R256)*100	=C256-C251	=D256-D251
257	254	70.9	1001	=B257/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F257/60/60)^2/(2*$H$1)	=A257	=D257	=C257	=A257	=D257+$M$1	=C257	=A257	=C257-D257	=A257	=((C257-D257)/C257)*100	=A257	=((R257-S257)/R257)*100	=C257-C252	=D257-D252
258	255	71.09	1004.5	=B258/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F258/60/60)^2/(2*$H$1)	=A258	=D258	=C258	=A258	=D258+$M$1	=C258	=A258	=C258-D258	=A258	=((C258-D258)/C258)*100	=A258	=((R258-S258)/R258)*100	=C258-C253	=D258-D253
259	256	73.18	1016.5	=B259/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F259/60/60)^2/(2*$H$1)	=A259	=D259	=C259	=A259	=D259+$M$1	=C259	=A259	=C259-D259	=A259	=((C259-D259)/C259)*100	=A259	=((R259-S259)/R259)*100	=C259-C254	=D259-D254
260	257	71.98	1021.5	=B260/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F260/60/60)^2/(2*$H$1)	=A260	=D260	=C260	=A260	=D260+$M$1	=C260	=A260	=C260-D260	=A260	=((C260-D260)/C260)*100	=A260	=((R260-S260)/R260)*100	=C260-C255	=D260-D255
261	258	73.51	1028	=B261/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F261/60/60)^2/(2*$H$1)	=A261	=D261	=C261	=A261	=D261+$M$1	=C261	=A261	=C261-D261	=A261	=((C261-D261)/C261)*100	=A261	=((R261-S261)/R261)*100	=C261-C256	=D261-D256
262	259	72.94	1037.5	=B262/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F262/60/60)^2/(2*$H$1)	=A262	=D262	=C262	=A262	=D262+$M$1	=C262	=A262	=C262-D262	=A262	=((C262-D262)/C262)*100	=A262	=((R262-S262)/R262)*100	=C262-C257	=D262-D257
263	260	74.16	1041.5	=B263/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F263/60/60)^2/(2*$H$1)	=A263	=D263	=C263	=A263	=D263+$M$1	=C263	=A263	=C263-D263	=A263	=((C263-D263)/C263)*100	=A263	=((R263-S263)/R263)*100	=C263-C258	=D263-D258
264	261	74.9	1053	=B264/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F264/60/60)^2/(2*$H$1)	=A264	=D264	=C264	=A264	=D264+$M$1	=C264	=A264	=C264-D264	=A264	=((C264-D264)/C264)*100	=A264	=((R264-S264)/R264)*100	=C264-C259	=D264-D259
265	262	75.52	1058.5	=B265/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F265/60/60)^2/(2*$H$1)	=A265	=D265	=C265	=A265	=D265+$M$1	=C265	=A265	=C265-D265	=A265	=((C265-D265)/C265)*100	=A265	=((R265-S265)/R265)*100	=C265-C260	=D265-D260
266	263	76.87	1065.5	=B266/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F266/60/60)^2/(2*$H$1)	=A266	=D266	=C266	=A266	=D266+$M$1	=C266	=A266	=C266-D266	=A266	=((C266-D266)/C266)*100	=A266	=((R266-S266)/R266)*100	=C266-C261	=D266-D261
267	264	76.67	1068.5	=B267/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F267/60/60)^2/(2*$H$1)	=A267	=D267	=C267	=A267	=D267+$M$1	=C267	=A267	=C267-D267	=A267	=((C267-D267)/C267)*100	=A267	=((R267-S267)/R267)*100	=C267-C262	=D267-D262
268	265	79.26	1079	=B268/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F268/60/60)^2/(2*$H$1)	=A268	=D268	=C268	=A268	=D268+$M$1	=C268	=A268	=C268-D268	=A268	=((C268-D268)/C268)*100	=A268	=((R268-S268)/R268)*100	=C268-C263	=D268-D263
269	266	78.22	1089.5	=B269/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F269/60/60)^2/(2*$H$1)	=A269	=D269	=C269	=A269	=D269+$M$1	=C269	=A269	=C269-D269	=A269	=((C269-D269)/C269)*100	=A269	=((R269-S269)/R269)*100	=C269-C264	=D269-D264
270	267	78.56	1100.5	=B270/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F270/60/60)^2/(2*$H$1)	=A270	=D270	=C270	=A270	=D270+$M$1	=C270	=A270	=C270-D270	=A270	=((C270-D270)/C270)*100	=A270	=((R270-S270)/R270)*100	=C270-C265	=D270-D265
271	268	78.68	1111.5	=B271/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F271/60/60)^2/(2*$H$1)	=A271	=D271	=C271	=A271	=D271+$M$1	=C271	=A271	=C271-D271	=A271	=((C271-D271)/C271)*100	=A271	=((R271-S271)/R271)*100	=C271-C266	=D271-D266
272	269	79.22	1123	=B272/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F272/60/60)^2/(2*$H$1)	=A272	=D272	=C272	=A272	=D272+$M$1	=C272	=A272	=C272-D272	=A272	=((C272-D272)/C272)*100	=A272	=((R272-S272)/R272)*100	=C272-C267	=D272-D267
273	270	80.08	1130.5	=B273/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F273/60/60)^2/(2*$H$1)	=A273	=D273	=C273	=A273	=D273+$M$1	=C273	=A273	=C273-D273	=A273	=((C273-D273)/C273)*100	=A273	=((R273-S273)/R273)*100	=C273-C268	=D273-D268
274	271	81.79	1138.5	=B274/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F274/60/60)^2/(2*$H$1)	=A274	=D274	=C274	=A274	=D274+$M$1	=C274	=A274	=C274-D274	=A274	=((C274-D274)/C274)*100	=A274	=((R274-S274)/R274)*100	=C274-C269	=D274-D269
275	272	80.33	1144.5	=B275/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F275/60/60)^2/(2*$H$1)	=A275	=D275	=C275	=A275	=D275+$M$1	=C275	=A275	=C275-D275	=A275	=((C275-D275)/C275)*100	=A275	=((R275-S275)/R275)*100	=C275-C270	=D275-D270
276	273	82.61	1148	=B276/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F276/60/60)^2/(2*$H$1)	=A276	=D276	=C276	=A276	=D276+$M$1	=C276	=A276	=C276-D276	=A276	=((C276-D276)/C276)*100	=A276	=((R276-S276)/R276)*100	=C276-C271	=D276-D271
277	274	84.28	1158	=B277/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F277/60/60)^2/(2*$H$1)	=A277	=D277	=C277	=A277	=D277+$M$1	=C277	=A277	=C277-D277	=A277	=((C277-D277)/C277)*100	=A277	=((R277-S277)/R277)*100	=C277-C272	=D277-D272
278	275	81.64	1164	=B278/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F278/60/60)^2/(2*$H$1)	=A278	=D278	=C278	=A278	=D278+$M$1	=C278	=A278	=C278-D278	=A278	=((C278-D278)/C278)*100	=A278	=((R278-S278)/R278)*100	=C278-C273	=D278-D273
279	276	84.42	1172	=B279/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F279/60/60)^2/(2*$H$1)	=A279	=D279	=C279	=A279	=D279+$M$1	=C279	=A279	=C279-D279	=A279	=((C279-D279)/C279)*100	=A279	=((R279-S279)/R279)*100	=C279-C274	=D279-D274
280	277	84.58	1184	=B280/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F280/60/60)^2/(2*$H$1)	=A280	=D280	=C280	=A280	=D280+$M$1	=C280	=A280	=C280-D280	=A280	=((C280-D280)/C280)*100	=A280	=((R280-S280)/R280)*100	=C280-C275	=D280-D275
281	278	86.45	1192.5	=B281/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F281/60/60)^2/(2*$H$1)	=A281	=D281	=C281	=A281	=D281+$M$1	=C281	=A281	=C281-D281	=A281	=((C281-D281)/C281)*100	=A281	=((R281-S281)/R281)*100	=C281-C276	=D281-D276
282	279	84.34	1198.5	=B282/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F282/60/60)^2/(2*$H$1)	=A282	=D282	=C282	=A282	=D282+$M$1	=C282	=A282	=C282-D282	=A282	=((C282-D282)/C282)*100	=A282	=((R282-S282)/R282)*100	=C282-C277	=D282-D277
283	280	87.99	1208	=B283/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F283/60/60)^2/(2*$H$1)	=A283	=D283	=C283	=A283	=D283+$M$1	=C283	=A283	=C283-D283	=A283	=((C283-D283)/C283)*100	=A283	=((R283-S283)/R283)*100	=C283-C278	=D283-D278
284	281	86.77	1217	=B284/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F284/60/60)^2/(2*$H$1)	=A284	=D284	=C284	=A284	=D284+$M$1	=C284	=A284	=C284-D284	=A284	=((C284-D284)/C284)*100	=A284	=((R284-S284)/R284)*100	=C284-C279	=D284-D279
285	282	86.68	1223	=B285/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F285/60/60)^2/(2*$H$1)	=A285	=D285	=C285	=A285	=D285+$M$1	=C285	=A285	=C285-D285	=A285	=((C285-D285)/C285)*100	=A285	=((R285-S285)/R285)*100	=C285-C280	=D285-D280
286	283	89.22	1227.5	=B286/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F286/60/60)^2/(2*$H$1)	=A286	=D286	=C286	=A286	=D286+$M$1	=C286	=A286	=C286-D286	=A286	=((C286-D286)/C286)*100	=A286	=((R286-S286)/R286)*100	=C286-C281	=D286-D281
287	284	88.47	1236.5	=B287/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F287/60/60)^2/(2*$H$1)	=A287	=D287	=C287	=A287	=D287+$M$1	=C287	=A287	=C287-D287	=A287	=((C287-D287)/C287)*100	=A287	=((R287-S287)/R287)*100	=C287-C282	=D287-D282
288	285	89.65	1244	=B288/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F288/60/60)^2/(2*$H$1)	=A288	=D288	=C288	=A288	=D288+$M$1	=C288	=A288	=C288-D288	=A288	=((C288-D288)/C288)*100	=A288	=((R288-S288)/R288)*100	=C288-C283	=D288-D283
289	286	90.79	1257.5	=B289/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F289/60/60)^2/(2*$H$1)	=A289	=D289	=C289	=A289	=D289+$M$1	=C289	=A289	=C289-D289	=A289	=((C289-D289)/C289)*100	=A289	=((R289-S289)/R289)*100	=C289-C284	=D289-D284
290	287	91.58	1271	=B290/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F290/60/60)^2/(2*$H$1)	=A290	=D290	=C290	=A290	=D290+$M$1	=C290	=A290	=C290-D290	=A290	=((C290-D290)/C290)*100	=A290	=((R290-S290)/R290)*100	=C290-C285	=D290-D285
291	288	91.2	1284	=B291/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F291/60/60)^2/(2*$H$1)	=A291	=D291	=C291	=A291	=D291+$M$1	=C291	=A291	=C291-D291	=A291	=((C291-D291)/C291)*100	=A291	=((R291-S291)/R291)*100	=C291-C286	=D291-D286
292	289	92.34	1295	=B292/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F292/60/60)^2/(2*$H$1)	=A292	=D292	=C292	=A292	=D292+$M$1	=C292	=A292	=C292-D292	=A292	=((C292-D292)/C292)*100	=A292	=((R292-S292)/R292)*100	=C292-C287	=D292-D287
293	290	93.78	1306	=B293/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F293/60/60)^2/(2*$H$1)	=A293	=D293	=C293	=A293	=D293+$M$1	=C293	=A293	=C293-D293	=A293	=((C293-D293)/C293)*100	=A293	=((R293-S293)/R293)*100	=C293-C288	=D293-D288
294	291	93.26	1315.5	=B294/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F294/60/60)^2/(2*$H$1)	=A294	=D294	=C294	=A294	=D294+$M$1	=C294	=A294	=C294-D294	=A294	=((C294-D294)/C294)*100	=A294	=((R294-S294)/R294)*100	=C294-C289	=D294-D289
295	292	93.62	1322	=B295/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F295/60/60)^2/(2*$H$1)	=A295	=D295	=C295	=A295	=D295+$M$1	=C295	=A295	=C295-D295	=A295	=((C295-D295)/C295)*100	=A295	=((R295-S295)/R295)*100	=C295-C290	=D295-D290
296	293	96.38	1329.5	=B296/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F296/60/60)^2/(2*$H$1)	=A296	=D296	=C296	=A296	=D296+$M$1	=C296	=A296	=C296-D296	=A296	=((C296-D296)/C296)*100	=A296	=((R296-S296)/R296)*100	=C296-C291	=D296-D291
297	294	95.86	1342	=B297/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F297/60/60)^2/(2*$H$1)	=A297	=D297	=C297	=A297	=D297+$M$1	=C297	=A297	=C297-D297	=A297	=((C297-D297)/C297)*100	=A297	=((R297-S297)/R297)*100	=C297-C292	=D297-D292
298	295	95.16	1345	=B298/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F298/60/60)^2/(2*$H$1)	=A298	=D298	=C298	=A298	=D298+$M$1	=C298	=A298	=C298-D298	=A298	=((C298-D298)/C298)*100	=A298	=((R298-S298)/R298)*100	=C298-C293	=D298-D293
299	296	97.72	1356	=B299/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F299/60/60)^2/(2*$H$1)	=A299	=D299	=C299	=A299	=D299+$M$1	=C299	=A299	=C299-D299	=A299	=((C299-D299)/C299)*100	=A299	=((R299-S299)/R299)*100	=C299-C294	=D299-D294
300	297	97.72	1362.5	=B300/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F300/60/60)^2/(2*$H$1)	=A300	=D300	=C300	=A300	=D300+$M$1	=C300	=A300	=C300-D300	=A300	=((C300-D300)/C300)*100	=A300	=((R300-S300)/R300)*100	=C300-C295	=D300-D295
301	298	98.62	1371.5	=B301/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F301/60/60)^2/(2*$H$1)	=A301	=D301	=C301	=A301	=D301+$M$1	=C301	=A301	=C301-D301	=A301	=((C301-D301)/C301)*100	=A301	=((R301-S301)/R301)*100	=C301-C296	=D301-D296
302	299	98.62	1381	=B302/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F302/60/60)^2/(2*$H$1)	=A302	=D302	=C302	=A302	=D302+$M$1	=C302	=A302	=C302-D302	=A302	=((C302-D302)/C302)*100	=A302	=((R302-S302)/R302)*100	=C302-C297	=D302-D297
303	300	98.32	1395	=B303/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F303/60/60)^2/(2*$H$1)	=A303	=D303	=C303	=A303	=D303+$M$1	=C303	=A303	=C303-D303	=A303	=((C303-D303)/C303)*100	=A303	=((R303-S303)/R303)*100	=C303-C298	=D303-D298
304	301	99.08	1408	=B304/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F304/60/60)^2/(2*$H$1)	=A304	=D304	=C304	=A304	=D304+$M$1	=C304	=A304	=C304-D304	=A304	=((C304-D304)/C304)*100	=A304	=((R304-S304)/R304)*100	=C304-C299	=D304-D299
305	302	99.94	1412	=B305/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F305/60/60)^2/(2*$H$1)	=A305	=D305	=C305	=A305	=D305+$M$1	=C305	=A305	=C305-D305	=A305	=((C305-D305)/C305)*100	=A305	=((R305-S305)/R305)*100	=C305-C300	=D305-D300
306	303	102.55	1418.5	=B306/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F306/60/60)^2/(2*$H$1)	=A306	=D306	=C306	=A306	=D306+$M$1	=C306	=A306	=C306-D306	=A306	=((C306-D306)/C306)*100	=A306	=((R306-S306)/R306)*100	=C306-C301	=D306-D301
307	304	100.81	1424.5	=B307/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F307/60/60)^2/(2*$H$1)	=A307	=D307	=C307	=A307	=D307+$M$1	=C307	=A307	=C307-D307	=A307	=((C307-D307)/C307)*100	=A307	=((R307-S307)/R307)*100	=C307-C302	=D307-D302
308	305	101.28	1433.5	=B308/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F308/60/60)^2/(2*$H$1)	=A308	=D308	=C308	=A308	=D308+$M$1	=C308	=A308	=C308-D308	=A308	=((C308-D308)/C308)*100	=A308	=((R308-S308)/R308)*100	=C308-C303	=D308-D303
309	306	101.84	1441.5	=B309/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F309/60/60)^2/(2*$H$1)	=A309	=D309	=C309	=A309	=D309+$M$1	=C309	=A309	=C309-D309	=A309	=((C309-D309)/C309)*100	=A309	=((R309-S309)/R309)*100	=C309-C304	=D309-D304
310	307	103.61	1452.5	=B310/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F310/60/60)^2/(2*$H$1)	=A310	=D310	=C310	=A310	=D310+$M$1	=C310	=A310	=C310-D310	=A310	=((C310-D310)/C310)*100	=A310	=((R310-S310)/R310)*100	=C310-C305	=D310-D305
311	308	105.24	1460	=B311/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F311/60/60)^2/(2*$H$1)	=A311	=D311	=C311	=A311	=D311+$M$1	=C311	=A311	=C311-D311	=A311	=((C311-D311)/C311)*100	=A311	=((R311-S311)/R311)*100	=C311-C306	=D311-D306
312	309	105.54	1465	=B312/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F312/60/60)^2/(2*$H$1)	=A312	=D312	=C312	=A312	=D312+$M$1	=C312	=A312	=C312-D312	=A312	=((C312-D312)/C312)*100	=A312	=((R312-S312)/R312)*100	=C312-C307	=D312-D307
313	310	105.99	1476.5	=B313/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F313/60/60)^2/(2*$H$1)	=A313	=D313	=C313	=A313	=D313+$M$1	=C313	=A313	=C313-D313	=A313	=((C313-D313)/C313)*100	=A313	=((R313-S313)/R313)*100	=C313-C308	=D313-D308
314	311	105.76	1481.5	=B314/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F314/60/60)^2/(2*$H$1)	=A314	=D314	=C314	=A314	=D314+$M$1	=C314	=A314	=C314-D314	=A314	=((C314-D314)/C314)*100	=A314	=((R314-S314)/R314)*100	=C314-C309	=D314-D309
315	312	108.47	1493	=B315/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F315/60/60)^2/(2*$H$1)	=A315	=D315	=C315	=A315	=D315+$M$1	=C315	=A315	=C315-D315	=A315	=((C315-D315)/C315)*100	=A315	=((R315-S315)/R315)*100	=C315-C310	=D315-D310
316	313	109.43	1502	=B316/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F316/60/60)^2/(2*$H$1)	=A316	=D316	=C316	=A316	=D316+$M$1	=C316	=A316	=C316-D316	=A316	=((C316-D316)/C316)*100	=A316	=((R316-S316)/R316)*100	=C316-C311	=D316-D311
317	314	108.22	1511.5	=B317/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F317/60/60)^2/(2*$H$1)	=A317	=D317	=C317	=A317	=D317+$M$1	=C317	=A317	=C317-D317	=A317	=((C317-D317)/C317)*100	=A317	=((R317-S317)/R317)*100	=C317-C312	=D317-D312
318	315	110.04	1521.5	=B318/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F318/60/60)^2/(2*$H$1)	=A318	=D318	=C318	=A318	=D318+$M$1	=C318	=A318	=C318-D318	=A318	=((C318-D318)/C318)*100	=A318	=((R318-S318)/R318)*100	=C318-C313	=D318-D313
319	316	111.04	1528.5	=B319/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F319/60/60)^2/(2*$H$1)	=A319	=D319	=C319	=A319	=D319+$M$1	=C319	=A319	=C319-D319	=A319	=((C319-D319)/C319)*100	=A319	=((R319-S319)/R319)*100	=C319-C314	=D319-D314
320	317	110.86	1536	=B320/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F320/60/60)^2/(2*$H$1)	=A320	=D320	=C320	=A320	=D320+$M$1	=C320	=A320	=C320-D320	=A320	=((C320-D320)/C320)*100	=A320	=((R320-S320)/R320)*100	=C320-C315	=D320-D315
321	318	112.61	1545.5	=B321/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F321/60/60)^2/(2*$H$1)	=A321	=D321	=C321	=A321	=D321+$M$1	=C321	=A321	=C321-D321	=A321	=((C321-D321)/C321)*100	=A321	=((R321-S321)/R321)*100	=C321-C316	=D321-D316
322	319	111.46	1555.5	=B322/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F322/60/60)^2/(2*$H$1)	=A322	=D322	=C322	=A322	=D322+$M$1	=C322	=A322	=C322-D322	=A322	=((C322-D322)/C322)*100	=A322	=((R322-S322)/R322)*100	=C322-C317	=D322-D317
323	320	113.69	1564.5	=B323/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F323/60/60)^2/(2*$H$1)	=A323	=D323	=C323	=A323	=D323+$M$1	=C323	=A323	=C323-D323	=A323	=((C323-D323)/C323)*100	=A323	=((R323-S323)/R323)*100	=C323-C318	=D323-D318
324	321	114.52	1578	=B324/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F324/60/60)^2/(2*$H$1)	=A324	=D324	=C324	=A324	=D324+$M$1	=C324	=A324	=C324-D324	=A324	=((C324-D324)/C324)*100	=A324	=((R324-S324)/R324)*100	=C324-C319	=D324-D319
325	322	113.42	1585	=B325/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F325/60/60)^2/(2*$H$1)	=A325	=D325	=C325	=A325	=D325+$M$1	=C325	=A325	=C325-D325	=A325	=((C325-D325)/C325)*100	=A325	=((R325-S325)/R325)*100	=C325-C320	=D325-D320
326	323	115.29	1595.5	=B326/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F326/60/60)^2/(2*$H$1)	=A326	=D326	=C326	=A326	=D326+$M$1	=C326	=A326	=C326-D326	=A326	=((C326-D326)/C326)*100	=A326	=((R326-S326)/R326)*100	=C326-C321	=D326-D321
327	324	115.57	1605.5	=B327/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F327/60/60)^2/(2*$H$1)	=A327	=D327	=C327	=A327	=D327+$M$1	=C327	=A327	=C327-D327	=A327	=((C327-D327)/C327)*100	=A327	=((R327-S327)/R327)*100	=C327-C322	=D327-D322
328	325	117.94	1612	=B328/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F328/60/60)^2/(2*$H$1)	=A328	=D328	=C328	=A328	=D328+$M$1	=C328	=A328	=C328-D328	=A328	=((C328-D328)/C328)*100	=A328	=((R328-S328)/R328)*100	=C328-C323	=D328-D323
329	326	117.88	1627	=B329/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F329/60/60)^2/(2*$H$1)	=A329	=D329	=C329	=A329	=D329+$M$1	=C329	=A329	=C329-D329	=A329	=((C329-D329)/C329)*100	=A329	=((R329-S329)/R329)*100	=C329-C324	=D329-D324
330	327	117.09	1633	=B330/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F330/60/60)^2/(2*$H$1)	=A330	=D330	=C330	=A330	=D330+$M$1	=C330	=A330	=C330-D330	=A330	=((C330-D330)/C330)*100	=A330	=((R330-S330)/R330)*100	=C330-C325	=D330-D325
331	328	116.44	1646.5	=B331/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F331/60/60)^2/(2*$H$1)	=A331	=D331	=C331	=A331	=D331+$M$1	=C331	=A331	=C331-D331	=A331	=((C331-D331)/C331)*100	=A331	=((R331-S331)/R331)*100	=C331-C326	=D331-D326
332	329	119.22	1661.5	=B332/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F332/60/60)^2/(2*$H$1)	=A332	=D332	=C332	=A332	=D332+$M$1	=C332	=A332	=C332-D332	=A332	=((C332-D332)/C332)*100	=A332	=((R332-S332)/R332)*100	=C332-C327	=D332-D327
333	330	120.69	1671	=B333/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F333/60/60)^2/(2*$H$1)	=A333	=D333	=C333	=A333	=D333+$M$1	=C333	=A333	=C333-D333	=A333	=((C333-D333)/C333)*100	=A333	=((R333-S333)/R333)*100	=C333-C328	=D333-D328
334	331	121.28	1683	=B334/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F334/60/60)^2/(2*$H$1)	=A334	=D334	=C334	=A334	=D334+$M$1	=C334	=A334	=C334-D334	=A334	=((C334-D334)/C334)*100	=A334	=((R334-S334)/R334)*100	=C334-C329	=D334-D329
335	332	122.11	1693	=B335/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F335/60/60)^2/(2*$H$1)	=A335	=D335	=C335	=A335	=D335+$M$1	=C335	=A335	=C335-D335	=A335	=((C335-D335)/C335)*100	=A335	=((R335-S335)/R335)*100	=C335-C330	=D335-D330
336	333	123	1698	=B336/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F336/60/60)^2/(2*$H$1)	=A336	=D336	=C336	=A336	=D336+$M$1	=C336	=A336	=C336-D336	=A336	=((C336-D336)/C336)*100	=A336	=((R336-S336)/R336)*100	=C336-C331	=D336-D331
337	334	123.94	1703.5	=B337/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F337/60/60)^2/(2*$H$1)	=A337	=D337	=C337	=A337	=D337+$M$1	=C337	=A337	=C337-D337	=A337	=((C337-D337)/C337)*100	=A337	=((R337-S337)/R337)*100	=C337-C332	=D337-D332
338	335	125.93	1711.5	=B338/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F338/60/60)^2/(2*$H$1)	=A338	=D338	=C338	=A338	=D338+$M$1	=C338	=A338	=C338-D338	=A338	=((C338-D338)/C338)*100	=A338	=((R338-S338)/R338)*100	=C338-C333	=D338-D333
339	336	125.82	1723	=B339/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F339/60/60)^2/(2*$H$1)	=A339	=D339	=C339	=A339	=D339+$M$1	=C339	=A339	=C339-D339	=A339	=((C339-D339)/C339)*100	=A339	=((R339-S339)/R339)*100	=C339-C334	=D339-D334
340	337	126.22	1734	=B340/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F340/60/60)^2/(2*$H$1)	=A340	=D340	=C340	=A340	=D340+$M$1	=C340	=A340	=C340-D340	=A340	=((C340-D340)/C340)*100	=A340	=((R340-S340)/R340)*100	=C340-C335	=D340-D335
341	338	125.89	1740.5	=B341/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F341/60/60)^2/(2*$H$1)	=A341	=D341	=C341	=A341	=D341+$M$1	=C341	=A341	=C341-D341	=A341	=((C341-D341)/C341)*100	=A341	=((R341-S341)/R341)*100	=C341-C336	=D341-D336
342	339	128.41	1747	=B342/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F342/60/60)^2/(2*$H$1)	=A342	=D342	=C342	=A342	=D342+$M$1	=C342	=A342	=C342-D342	=A342	=((C342-D342)/C342)*100	=A342	=((R342-S342)/R342)*100	=C342-C337	=D342-D337
343	340	127.04	1758.5	=B343/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F343/60/60)^2/(2*$H$1)	=A343	=D343	=C343	=A343	=D343+$M$1	=C343	=A343	=C343-D343	=A343	=((C343-D343)/C343)*100	=A343	=((R343-S343)/R343)*100	=C343-C338	=D343-D338
344	341	129	1765.5	=B344/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F344/60/60)^2/(2*$H$1)	=A344	=D344	=C344	=A344	=D344+$M$1	=C344	=A344	=C344-D344	=A344	=((C344-D344)/C344)*100	=A344	=((R344-S344)/R344)*100	=C344-C339	=D344-D339
345	342	130.77	1783.5	=B345/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F345/60/60)^2/(2*$H$1)	=A345	=D345	=C345	=A345	=D345+$M$1	=C345	=A345	=C345-D345	=A345	=((C345-D345)/C345)*100	=A345	=((R345-S345)/R345)*100	=C345-C340	=D345-D340
346	343	129.54	1791.5	=B346/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F346/60/60)^2/(2*$H$1)	=A346	=D346	=C346	=A346	=D346+$M$1	=C346	=A346	=C346-D346	=A346	=((C346-D346)/C346)*100	=A346	=((R346-S346)/R346)*100	=C346-C341	=D346-D341
347	344	131.7	1799.5	=B347/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F347/60/60)^2/(2*$H$1)	=A347	=D347	=C347	=A347	=D347+$M$1	=C347	=A347	=C347-D347	=A347	=((C347-D347)/C347)*100	=A347	=((R347-S347)/R347)*100	=C347-C342	=D347-D342
348	345	132.18	1807	=B348/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F348/60/60)^2/(2*$H$1)	=A348	=D348	=C348	=A348	=D348+$M$1	=C348	=A348	=C348-D348	=A348	=((C348-D348)/C348)*100	=A348	=((R348-S348)/R348)*100	=C348-C343	=D348-D343
349	346	133.44	1816.5	=B349/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F349/60/60)^2/(2*$H$1)	=A349	=D349	=C349	=A349	=D349+$M$1	=C349	=A349	=C349-D349	=A349	=((C349-D349)/C349)*100	=A349	=((R349-S349)/R349)*100	=C349-C344	=D349-D344
350	347	134.21	1825.5	=B350/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F350/60/60)^2/(2*$H$1)	=A350	=D350	=C350	=A350	=D350+$M$1	=C350	=A350	=C350-D350	=A350	=((C350-D350)/C350)*100	=A350	=((R350-S350)/R350)*100	=C350-C345	=D350-D345
351	348	135.42	1833.5	=B351/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F351/60/60)^2/(2*$H$1)	=A351	=D351	=C351	=A351	=D351+$M$1	=C351	=A351	=C351-D351	=A351	=((C351-D351)/C351)*100	=A351	=((R351-S351)/R351)*100	=C351-C346	=D351-D346
352	349	132.84	1842.5	=B352/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F352/60/60)^2/(2*$H$1)	=A352	=D352	=C352	=A352	=D352+$M$1	=C352	=A352	=C352-D352	=A352	=((C352-D352)/C352)*100	=A352	=((R352-S352)/R352)*100	=C352-C347	=D352-D347
353	350	137.17	1851.5	=B353/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F353/60/60)^2/(2*$H$1)	=A353	=D353	=C353	=A353	=D353+$M$1	=C353	=A353	=C353-D353	=A353	=((C353-D353)/C353)*100	=A353	=((R353-S353)/R353)*100	=C353-C348	=D353-D348
354	351	137.37	1869	=B354/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F354/60/60)^2/(2*$H$1)	=A354	=D354	=C354	=A354	=D354+$M$1	=C354	=A354	=C354-D354	=A354	=((C354-D354)/C354)*100	=A354	=((R354-S354)/R354)*100	=C354-C349	=D354-D349
355	352	137.85	1883.5	=B355/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F355/60/60)^2/(2*$H$1)	=A355	=D355	=C355	=A355	=D355+$M$1	=C355	=A355	=C355-D355	=A355	=((C355-D355)/C355)*100	=A355	=((R355-S355)/R355)*100	=C355-C350	=D355-D350
356	353	136.13	1895	=B356/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F356/60/60)^2/(2*$H$1)	=A356	=D356	=C356	=A356	=D356+$M$1	=C356	=A356	=C356-D356	=A356	=((C356-D356)/C356)*100	=A356	=((R356-S356)/R356)*100	=C356-C351	=D356-D351
357	354	139.68	1904.5	=B357/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F357/60/60)^2/(2*$H$1)	=A357	=D357	=C357	=A357	=D357+$M$1	=C357	=A357	=C357-D357	=A357	=((C357-D357)/C357)*100	=A357	=((R357-S357)/R357)*100	=C357-C352	=D357-D352
358	355	140.61	1917	=B358/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F358/60/60)^2/(2*$H$1)	=A358	=D358	=C358	=A358	=D358+$M$1	=C358	=A358	=C358-D358	=A358	=((C358-D358)/C358)*100	=A358	=((R358-S358)/R358)*100	=C358-C353	=D358-D353
359	356	141.44	1927	=B359/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F359/60/60)^2/(2*$H$1)	=A359	=D359	=C359	=A359	=D359+$M$1	=C359	=A359	=C359-D359	=A359	=((C359-D359)/C359)*100	=A359	=((R359-S359)/R359)*100	=C359-C354	=D359-D354
360	357	142.08	1937.5	=B360/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F360/60/60)^2/(2*$H$1)	=A360	=D360	=C360	=A360	=D360+$M$1	=C360	=A360	=C360-D360	=A360	=((C360-D360)/C360)*100	=A360	=((R360-S360)/R360)*100	=C360-C355	=D360-D355
361	358	140.33	1949	=B361/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F361/60/60)^2/(2*$H$1)	=A361	=D361	=C361	=A361	=D361+$M$1	=C361	=A361	=C361-D361	=A361	=((C361-D361)/C361)*100	=A361	=((R361-S361)/R361)*100	=C361-C356	=D361-D356
362	359	140.98	1959	=B362/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F362/60/60)^2/(2*$H$1)	=A362	=D362	=C362	=A362	=D362+$M$1	=C362	=A362	=C362-D362	=A362	=((C362-D362)/C362)*100	=A362	=((R362-S362)/R362)*100	=C362-C357	=D362-D357
363	360	143.78	1972.5	=B363/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F363/60/60)^2/(2*$H$1)	=A363	=D363	=C363	=A363	=D363+$M$1	=C363	=A363	=C363-D363	=A363	=((C363-D363)/C363)*100	=A363	=((R363-S363)/R363)*100	=C363-C358	=D363-D358
364	361	142.94	1983	=B364/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F364/60/60)^2/(2*$H$1)	=A364	=D364	=C364	=A364	=D364+$M$1	=C364	=A364	=C364-D364	=A364	=((C364-D364)/C364)*100	=A364	=((R364-S364)/R364)*100	=C364-C359	=D364-D359
365	362	145.89	1999.5	=B365/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F365/60/60)^2/(2*$H$1)	=A365	=D365	=C365	=A365	=D365+$M$1	=C365	=A365	=C365-D365	=A365	=((C365-D365)/C365)*100	=A365	=((R365-S365)/R365)*100	=C365-C360	=D365-D360
366	363	143.13	2010	=B366/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F366/60/60)^2/(2*$H$1)	=A366	=D366	=C366	=A366	=D366+$M$1	=C366	=A366	=C366-D366	=A366	=((C366-D366)/C366)*100	=A366	=((R366-S366)/R366)*100	=C366-C361	=D366-D361
367	364	143.75	2023	=B367/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F367/60/60)^2/(2*$H$1)	=A367	=D367	=C367	=A367	=D367+$M$1	=C367	=A367	=C367-D367	=A367	=((C367-D367)/C367)*100	=A367	=((R367-S367)/R367)*100	=C367-C362	=D367-D362
368	365	143.82	2033.5	=B368/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F368/60/60)^2/(2*$H$1)	=A368	=D368	=C368	=A368	=D368+$M$1	=C368	=A368	=C368-D368	=A368	=((C368-D368)/C368)*100	=A368	=((R368-S368)/R368)*100	=C368-C363	=D368-D363
369	366	147.49	2045	=B369/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F369/60/60)^2/(2*$H$1)	=A369	=D369	=C369	=A369	=D369+$M$1	=C369	=A369	=C369-D369	=A369	=((C369-D369)/C369)*100	=A369	=((R369-S369)/R369)*100	=C369-C364	=D369-D364
370	367	150.47	2057.5	=B370/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F370/60/60)^2/(2*$H$1)	=A370	=D370	=C370	=A370	=D370+$M$1	=C370	=A370	=C370-D370	=A370	=((C370-D370)/C370)*100	=A370	=((R370-S370)/R370)*100	=C370-C365	=D370-D365
371	368	149.58	2069.5	=B371/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F371/60/60)^2/(2*$H$1)	=A371	=D371	=C371	=A371	=D371+$M$1	=C371	=A371	=C371-D371	=A371	=((C371-D371)/C371)*100	=A371	=((R371-S371)/R371)*100	=C371-C366	=D371-D366
372	369	150.61	2083.5	=B372/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F372/60/60)^2/(2*$H$1)	=A372	=D372	=C372	=A372	=D372+$M$1	=C372	=A372	=C372-D372	=A372	=((C372-D372)/C372)*100	=A372	=((R372-S372)/R372)*100	=C372-C367	=D372-D367
373	370	152.36	2092.5	=B373/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F373/60/60)^2/(2*$H$1)	=A373	=D373	=C373	=A373	=D373+$M$1	=C373	=A373	=C373-D373	=A373	=((C373-D373)/C373)*100	=A373	=((R373-S373)/R373)*100	=C373-C368	=D373-D368
374	371	152.94	2102.5	=B374/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F374/60/60)^2/(2*$H$1)	=A374	=D374	=C374	=A374	=D374+$M$1	=C374	=A374	=C374-D374	=A374	=((C374-D374)/C374)*100	=A374	=((R374-S374)/R374)*100	=C374-C369	=D374-D369
375	372	154.07	2114.5	=B375/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F375/60/60)^2/(2*$H$1)	=A375	=D375	=C375	=A375	=D375+$M$1	=C375	=A375	=C375-D375	=A375	=((C375-D375)/C375)*100	=A375	=((R375-S375)/R375)*100	=C375-C370	=D375-D370
376	373	152.24	2127.5	=B376/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F376/60/60)^2/(2*$H$1)	=A376	=D376	=C376	=A376	=D376+$M$1	=C376	=A376	=C376-D376	=A376	=((C376-D376)/C376)*100	=A376	=((R376-S376)/R376)*100	=C376-C371	=D376-D371
377	374	153.39	2134.5	=B377/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F377/60/60)^2/(2*$H$1)	=A377	=D377	=C377	=A377	=D377+$M$1	=C377	=A377	=C377-D377	=A377	=((C377-D377)/C377)*100	=A377	=((R377-S377)/R377)*100	=C377-C372	=D377-D372
378	375	155.87	2145	=B378/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F378/60/60)^2/(2*$H$1)	=A378	=D378	=C378	=A378	=D378+$M$1	=C378	=A378	=C378-D378	=A378	=((C378-D378)/C378)*100	=A378	=((R378-S378)/R378)*100	=C378-C373	=D378-D373
379	376	156.11	2155.5	=B379/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F379/60/60)^2/(2*$H$1)	=A379	=D379	=C379	=A379	=D379+$M$1	=C379	=A379	=C379-D379	=A379	=((C379-D379)/C379)*100	=A379	=((R379-S379)/R379)*100	=C379-C374	=D379-D374
380	377	157.44	2166.5	=B380/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F380/60/60)^2/(2*$H$1)	=A380	=D380	=C380	=A380	=D380+$M$1	=C380	=A380	=C380-D380	=A380	=((C380-D380)/C380)*100	=A380	=((R380-S380)/R380)*100	=C380-C375	=D380-D375
381	378	158.12	2182	=B381/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F381/60/60)^2/(2*$H$1)	=A381	=D381	=C381	=A381	=D381+$M$1	=C381	=A381	=C381-D381	=A381	=((C381-D381)/C381)*100	=A381	=((R381-S381)/R381)*100	=C381-C376	=D381-D376
382	379	158.69	2201.5	=B382/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F382/60/60)^2/(2*$H$1)	=A382	=D382	=C382	=A382	=D382+$M$1	=C382	=A382	=C382-D382	=A382	=((C382-D382)/C382)*100	=A382	=((R382-S382)/R382)*100	=C382-C377	=D382-D377
383	380	159.49	2212	=B383/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F383/60/60)^2/(2*$H$1)	=A383	=D383	=C383	=A383	=D383+$M$1	=C383	=A383	=C383-D383	=A383	=((C383-D383)/C383)*100	=A383	=((R383-S383)/R383)*100	=C383-C378	=D383-D378
384	381	160.96	2225	=B384/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F384/60/60)^2/(2*$H$1)	=A384	=D384	=C384	=A384	=D384+$M$1	=C384	=A384	=C384-D384	=A384	=((C384-D384)/C384)*100	=A384	=((R384-S384)/R384)*100	=C384-C379	=D384-D379
385	382	162.65	2237.5	=B385/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F385/60/60)^2/(2*$H$1)	=A385	=D385	=C385	=A385	=D385+$M$1	=C385	=A385	=C385-D385	=A385	=((C385-D385)/C385)*100	=A385	=((R385-S385)/R385)*100	=C385-C380	=D385-D380
386	383	162.27	2246.5	=B386/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F386/60/60)^2/(2*$H$1)	=A386	=D386	=C386	=A386	=D386+$M$1	=C386	=A386	=C386-D386	=A386	=((C386-D386)/C386)*100	=A386	=((R386-S386)/R386)*100	=C386-C381	=D386-D381
387	384	164.47	2260	=B387/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F387/60/60)^2/(2*$H$1)	=A387	=D387	=C387	=A387	=D387+$M$1	=C387	=A387	=C387-D387	=A387	=((C387-D387)/C387)*100	=A387	=((R387-S387)/R387)*100	=C387-C382	=D387-D382
388	385	164.75	2278.5	=B388/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F388/60/60)^2/(2*$H$1)	=A388	=D388	=C388	=A388	=D388+$M$1	=C388	=A388	=C388-D388	=A388	=((C388-D388)/C388)*100	=A388	=((R388-S388)/R388)*100	=C388-C383	=D388-D383
389	386	165.16	2290	=B389/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F389/60/60)^2/(2*$H$1)	=A389	=D389	=C389	=A389	=D389+$M$1	=C389	=A389	=C389-D389	=A389	=((C389-D389)/C389)*100	=A389	=((R389-S389)/R389)*100	=C389-C384	=D389-D384
390	387	164.79	2305.5	=B390/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F390/60/60)^2/(2*$H$1)	=A390	=D390	=C390	=A390	=D390+$M$1	=C390	=A390	=C390-D390	=A390	=((C390-D390)/C390)*100	=A390	=((R390-S390)/R390)*100	=C390-C385	=D390-D385
391	388	168.25	2316.5	=B391/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F391/60/60)^2/(2*$H$1)	=A391	=D391	=C391	=A391	=D391+$M$1	=C391	=A391	=C391-D391	=A391	=((C391-D391)/C391)*100	=A391	=((R391-S391)/R391)*100	=C391-C386	=D391-D386
392	389	166.74	2332.5	=B392/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F392/60/60)^2/(2*$H$1)	=A392	=D392	=C392	=A392	=D392+$M$1	=C392	=A392	=C392-D392	=A392	=((C392-D392)/C392)*100	=A392	=((R392-S392)/R392)*100	=C392-C387	=D392-D387
393	390	169.82	2348	=B393/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F393/60/60)^2/(2*$H$1)	=A393	=D393	=C393	=A393	=D393+$M$1	=C393	=A393	=C393-D393	=A393	=((C393-D393)/C393)*100	=A393	=((R393-S393)/R393)*100	=C393-C388	=D393-D388
394	391	167.99	2360.5	=B394/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F394/60/60)^2/(2*$H$1)	=A394	=D394	=C394	=A394	=D394+$M$1	=C394	=A394	=C394-D394	=A394	=((C394-D394)/C394)*100	=A394	=((R394-S394)/R394)*100	=C394-C389	=D394-D389
395	392	169.8	2372.5	=B395/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F395/60/60)^2/(2*$H$1)	=A395	=D395	=C395	=A395	=D395+$M$1	=C395	=A395	=C395-D395	=A395	=((C395-D395)/C395)*100	=A395	=((R395-S395)/R395)*100	=C395-C390	=D395-D390
396	393	170.43	2380	=B396/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F396/60/60)^2/(2*$H$1)	=A396	=D396	=C396	=A396	=D396+$M$1	=C396	=A396	=C396-D396	=A396	=((C396-D396)/C396)*100	=A396	=((R396-S396)/R396)*100	=C396-C391	=D396-D391
397	394	173.87	2394.5	=B397/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F397/60/60)^2/(2*$H$1)	=A397	=D397	=C397	=A397	=D397+$M$1	=C397	=A397	=C397-D397	=A397	=((C397-D397)/C397)*100	=A397	=((R397-S397)/R397)*100	=C397-C392	=D397-D392
398	395	172.52	2406	=B398/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F398/60/60)^2/(2*$H$1)	=A398	=D398	=C398	=A398	=D398+$M$1	=C398	=A398	=C398-D398	=A398	=((C398-D398)/C398)*100	=A398	=((R398-S398)/R398)*100	=C398-C393	=D398-D393
399	396	174.11	2416.5	=B399/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F399/60/60)^2/(2*$H$1)	=A399	=D399	=C399	=A399	=D399+$M$1	=C399	=A399	=C399-D399	=A399	=((C399-D399)/C399)*100	=A399	=((R399-S399)/R399)*100	=C399-C394	=D399-D394
400	397	176.83	2427.5	=B400/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F400/60/60)^2/(2*$H$1)	=A400	=D400	=C400	=A400	=D400+$M$1	=C400	=A400	=C400-D400	=A400	=((C400-D400)/C400)*100	=A400	=((R400-S400)/R400)*100	=C400-C395	=D400-D395
401	398	172.42	2442	=B401/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F401/60/60)^2/(2*$H$1)	=A401	=D401	=C401	=A401	=D401+$M$1	=C401	=A401	=C401-D401	=A401	=((C401-D401)/C401)*100	=A401	=((R401-S401)/R401)*100	=C401-C396	=D401-D396
402	399	175.42	2454.5	=B402/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F402/60/60)^2/(2*$H$1)	=A402	=D402	=C402	=A402	=D402+$M$1	=C402	=A402	=C402-D402	=A402	=((C402-D402)/C402)*100	=A402	=((R402-S402)/R402)*100	=C402-C397	=D402-D397
403	400	177.36	2468.5	=B403/1000*$H$1*$K$1	=$F$1*($F$1-1)*PI()*(F403/60/60)^2/(2*$H$1)	=A403	=D403	=C403	=A403	=D403+$M$1	=C403	=A403	=C403-D403	=A403	=((C403-D403)/C403)*100	=A403	=((R403-S403)/R403)*100	=C403-C398	=D403-D398

A.2.4 Plot Clustering Code

Plots a list of Blue Objects in a galatic coordinate system

PBO19Mar09.pro

WD candidate clustering IDL Plot (DR7) - galatic coord - normalised.zip

```
; Title: Plot SDSS Blue Objects
; File: PBO.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 19 March 09 1pm
; Summary:  Preconditions; Assumes a file SDSSblueObjects.csv, containing 4 columns of data and
    56587 rows (56587 stars within a total 11663 square degree field)
;           Column 1 = RIGHT_ASCENSION, Column 2 = DECLINATION, Column 3 = GALATIC_LATITUDE, Column
    4 = GALATIC_LONGITUDE
;       Postconditions; Plots the Blue Objects

numberOfLines = 56587
numberOfColumns = 4
data = DblArr(numberOfColumns,numberOfLines)

NameOfFile = 'SDSSblueObjects.txt'
openr, 5, NameOfFile
readf, 5, data
close,/all ; close all open files

RIGHT_ASCENSION = data(0,*)
DECLINATION = data(1,*)
GALATIC_LATITUDE = data(2,*)
GALATIC_LONGITUDE = data(3,*)

; using Aitoff projection of Galatic coordinates;
; http://irsa.ipac.caltech.edu/IRASdocs/surveys/coordproj.html

GALATIC_LONGITUDE_normalised = GALATIC_LONGITUDE ; declaration for size only
GALATIC_LATITUDE_normalised = GALATIC_LATITUDE ; declaration for size only
for starIndex=0L, numberOfLines-1, 1L do begin

    ; scale:  2 pixels per degree in the map
    ; l, b:  Galactic coordinates of a given position
    ; l0:  Galactic longitude of the map cent
    ; RHO = arccos[cos(b) x cos({l - l0}/2)]
    ; THETA = arcsin[cos(b) x sin({l - l0}/2)/sin(RHO)]
    ; SAMPLE = -4 x scale x 180/pi x sin(RHO/2) x sin(THETA)
        ; LINE = ( )2 x scale x 180/pi x sin(RHO/2) x cos(THETA)

    scale = 1.0
    l = GALATIC_LONGITUDE[starIndex] /180.0*!PI
    b = GALATIC_LATITUDE[starIndex] /180.0*!PI
    lo = !PI
    ;l = RIGHT_ASCENSION[starIndex] /180.0*!PI  ;Equatorial Coordinates
    ;b = DECLINATION[starIndex] /180.0*!PI      ;Equatorial Coordinates
    ;lo = !PI                                  ;Equatorial Coordinates
    RHO = ACOS(COS(b) * COS((l - lo)/2.0))
    THETA = ASIN(COS(b) * SIN((l - lo)/2.0)/SIN(RHO))
    x = -4.0 * scale * 180.0/!PI * SIN(RHO/2.0) * SIN(THETA)

    if (GALATIC_LATITUDE[starIndex] gt 0.0) then begin
    ;if (DECLINATION[starIndex] gt 0.0) then begin  ;Equatorial Coordinates
        y = 2.0 * scale * 180.0/!PI * SIN(RHO/2.0) * COS(THETA)
    endif else begin
        y = -(2.0 * scale * 180.0/!PI * SIN(RHO/2.0) * COS(THETA))
    endelse

    GALATIC_LONGITUDE_normalised[starIndex] = x + 180.0
    GALATIC_LATITUDE_normalised[starIndex] = y
endfor
plot, GALATIC_LONGITUDE_normalised, GALATIC_LATITUDE_normalised, title='WD Candidates (Blue Objects
    from SDSS DR7 Survey) - Aitoff Projection of Galatic Coordinates',xtitle='
    GALATIC_LONGITUDE_normalised (deg)',ytitle='GALATIC_LATITUDE_normalised (deg)',xrange=[0,360],
    yrange=[-100,100],linestyle=5,psym=3, SYMSIZE=0.01

end
```

A.2.5 Create Blue Object Close Pair List (BOCPL)

This program generates a list of blue object pairs within a given angular separation.

BOCPL1createBlueObjectClosePairList.pro

```
; Title: Blue Object Close Pair List - create Blue Object Close Pair List.pro
; File: BOCPLcreateBlueObjectClosePairList.pro [old file name; CCPtestDetailed.pro]
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 8 April 09 3pm [updated 19 May 11a - DR7+DR8 compatible]
; Summary:
;
;   Postconditions;
;
;   This program generates a list of blue object pairs within a given angular separation. In the
     output list each close pair is listed twice, A with B and B with A. The list has been ordered
     in terms of pair distance, DISTANCE_BETWEEN_PAIR. The output files are of the following format;
;
;   BOCPLPart1DR8.txt [RIGHT_ASCENSION A, DECLINATION A, GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U
     A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR, RIGHT_ASCENSION B, DECLINATION B, GALATIC_LATITUDE B
     , GALATIC_LONGITUDE B, U B, G B, R B, POK B]
;   BOCPLPart2DR8.txt [ID A, ID B]
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   BOCPLSDSSblueObjectsDR7detailed.txt [ID, RIGHT_ASCENSION, DECLINATION, GALATIC_LATITUDE,
     GALATIC_LONGITUDE, ID, U, G, R, POK] for 56587 blue objects, tab delimited



;distances are given in arc seconds here

firstPairMaxDistance = 60
lastPairMaxDistance = 60
diffPairMaxDistance = 1

;ORIGINAL;
fieldSquareDegrees = 1000L  ; or 1000   ;or 11663
fieldNumberOfStars = 4852L  ; or 4852   ;or 56587
numberOfSamples = 100L

;TEST HARNESS - load from file;
fieldSquareDegrees = 11663L ;WRONG  ; or 1000   ;or 11663
fieldNumberOfStars = 36681L ;WRONG  ; or 4852   ;or 56587
numberOfSamples = 1L

fieldxDegrees = SQRT(fieldSquareDegrees)    ; or 100
fieldyDegrees = SQRT(fieldSquareDegrees)    ; or 100

IDLrandomNumberMin = 0L
IDLrandomNumberMax = 1L
IDLrandomNumberVariation = (IDLrandomNumberMax - IDLrandomNumberMin)

for maxDistanceIndex=firstPairMaxDistance, lastPairMaxDistance, diffPairMaxDistance do begin

    maxDistanceForConsideredPairInArcSeconds = double(maxDistanceIndex)
    maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D
    totalNumberOfPairsFoundAcrossAllSamples = 0L

    for sampleIndex=1L, numberOfSamples, 1L do begin

        ;ORIGINAL;
        ;starXCoordinatesBase = RANDOMU(seed, fieldNumberOfStars, /DOUBLE)
        ;starYCoordinatesBase = RANDOMU(seed, fieldNumberOfStars, /DOUBLE)
        ;starXCoordinates = starXCoordinatesBase/IDLrandomNumberVariation*fieldxDegrees
        ;starYCoordinates = starYCoordinatesBase/IDLrandomNumberVariation*fieldyDegrees

        ;TEST HARNESS - load from file;

            NameOfFile = 'BOCPLSDSSblueObjectsDR7detailed.txt'

                ;long precision is required for ID
            readcol64bitvars, NameOfFile, ID, RIGHT_ASCENSION, DECLINATION, GALATIC_LATITUDE,
                GALATIC_LONGITUDE, U, G, R, POK, FORMAT='G,D,D,D,D,D,D,D,D', SKIPLINE=1

            starID = transpose(ID)
            starYCoordinates = transpose(DECLINATION) ;GALATIC_LATITUDE
            starXCoordinates = transpose(RIGHT_ASCENSION) ;GALATIC_LONGITUDE

            ;PRINT, [starXCoordinates, starYCoordinates]
            ;starYCoordinates = starYCoordinates+90.0
            ;plot, starXCoordinates, starYCoordinates, title='WD Candidates', xtitle='
                GALATIC_LONGITUDE (deg)', ytitle='GALATIC_LATITUDE (deg)', xrange=[0,360], yrange
                =[0,200], linestyle=5,psym=3, SYMSIZE=0.2
```

```
    ;plot, starXCoordinates, starYCoordinates, title='WD Candidates', xtitle='RIGHT_ASCENSION
        (deg)', ytitle='DECLINATION (deg)', xrange=[0,360], yrange=[-100,100], linestyle=5,
        psym=3, SYMSIZE=0.2

starXCoordinatesSortIndicies = sort(starXCoordinates)
starXCoordinatesSorted = starXCoordinates[starXCoordinatesSortIndicies]
starYCoordinatesSortIndicies = sort(starYCoordinates)
starYCoordinatesSorted = starYCoordinates[starYCoordinatesSortIndicies]

maxNumberOfPairsExpected = 10000 ;300
starIDArrayForClosePairs1 = ulon64arr(maxNumberOfPairsExpected)
starIDArrayForClosePairs2 = ulon64arr(maxNumberOfPairsExpected)

starDistanceArrayForClosePairs = dblarr(maxNumberOfPairsExpected)
starDataArrayForClosePairs1 = dblarr(8, maxNumberOfPairsExpected)
starDataArrayForClosePairs2 = dblarr(8, maxNumberOfPairsExpected)
data = [transpose(RIGHT_ASCENSION), transpose(DECLINATION), transpose(GALATIC_LATITUDE),
    transpose(GALATIC_LONGITUDE), transpose(U), transpose(G), transpose(R), transpose(POK)]

;PRINT, "data = "
;PRINT, data

;stringmessage = "starXCoordinatesSorted ="
;PRINT, starXCoordinatesSorted

totalNumberOfPairsFound = 0L

for starIndex=1L, fieldNumberOfStars, 1L do begin

    numberOfPairsFound = 0L

    starIndexArray = starIndex-1

    centralPositionInStarXCoordinatesSorted = where(starXCoordinatesSortIndicies eq
        starIndexArray)
    centralValueInStarXCoordinatesSorted = starXCoordinatesSorted[
        centralPositionInStarXCoordinatesSorted]
        ;centralPositionInStarXCoordinates = starIndexArray
        ;or centralValueInStarXCoordinates = starXCoordinates[
            centralPositionInStarXCoordinates]
    centralPositionInStarYCoordinatesSorted = where(starYCoordinatesSortIndicies eq
        starIndexArray)
    centralValueInStarYCoordinatesSorted = starYCoordinatesSorted[
        centralPositionInStarYCoordinatesSorted]
        ;centralPositionInStarYCoordinates = starIndexArray
        ;or centralValueInStarYCoordinates = starYCoordinates[
            centralPositionInStarYCoordinates]



    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted



        ;hunt in positive x direction
    consideredPairIndex = 1L
    currentMaxDistanceBetweenPairX = 0D

    while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
        begin

        if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) gt (
            fieldNumberOfStars-1) then begin
            currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
        endif else begin

            currentPositionInStarXCoordinatesSorted =
                centralPositionInStarXCoordinatesSorted + consideredPairIndex
            currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
                currentPositionInStarXCoordinatesSorted]
            currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
                currentPositionInStarXCoordinatesSorted]
            currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
            currentValueInStarYCoordinates = starYCoordinates[
                currentPositionInStarYCoordinates]
            currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

            ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarXCoordinatesSorted
```

```
;stringmessage = "currentValueInStarXCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarXCoordinatesSorted
;stringmessage = "currentPositionInStarXCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarXCoordinates
;stringmessage = "currentPositionInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentPositionInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinates ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinates
;stringmessage = "currentValueInStarYCoordinatesSorted ="
;PRINT, stringmessage
;PRINT, currentValueInStarYCoordinatesSorted

currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
    centralValueInStarXCoordinatesSorted)
currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
    centralValueInStarYCoordinatesSorted)
;averageYDeclination = currentValueInStarYCoordinatesSorted +
    centralValueInStarYCoordinatesSorted) / 2
averageYDeclination = centralValueInStarYCoordinatesSorted
currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
    currentMaxDistanceBetweenPairX
currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
     currentMaxDistanceBetweenPairYCorrected^2)

;stringmessage = "    currentMaxDistanceBetweenPairX ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairX
;stringmessage = "currentMaxDistanceBetweenPairY ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPairY
;stringmessage = "currentMaxDistanceBetweenPair ="
;PRINT, stringmessage
;PRINT, currentMaxDistanceBetweenPair


if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
    then begin

    starIDArrayForClosePairs1[totalNumberOfPairsFound+numberOfPairsFound] =
        starID[starIndexArray]
    starIDArrayForClosePairs2[totalNumberOfPairsFound+numberOfPairsFound] =
        starID[currentPositionInStarXCoordinates]
    starDistanceArrayForClosePairs[totalNumberOfPairsFound+numberOfPairsFound] =
         currentMaxDistanceBetweenPair
    starDataArrayForClosePairs1[*, totalNumberOfPairsFound+numberOfPairsFound] =
         data[*, starIndexArray]
    starDataArrayForClosePairs2[*, totalNumberOfPairsFound+numberOfPairsFound] =
         data[*, currentPositionInStarXCoordinates]

    numberOfPairsFound = numberOfPairsFound + 1

    ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
        +9999

    ;stringmessage = "                              Pair Found A!"
    ;PRINT, stringmessage
    ;stringmessage = "StarIndex = "
    ;PRINT, stringmessage
    ;PRINT, starIndex
    ;stringmessage = "currentMaxDistanceBetweenPair = "
    ;PRINT, stringmessage
    ;PRINT, currentMaxDistanceBetweenPair
    ;exit    ;TEMP

    ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarXCoordinatesSorted
    ;stringmessage = "centralValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarXCoordinatesSorted
    ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralPositionInStarYCoordinatesSorted
    ;stringmessage = "centralValueInStarYCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, centralValueInStarYCoordinatesSorted

    ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentPositionInStarXCoordinatesSorted
    ;stringmessage = "currentValueInStarXCoordinatesSorted ="
    ;PRINT, stringmessage
    ;PRINT, currentValueInStarXCoordinatesSorted
    ;stringmessage = "currentPositionInStarXCoordinates ="
    ;PRINT, stringmessage
```

```
            ;PRINT, currentPositionInStarXCoordinates
            ;stringmessage = "currentPositionInStarYCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentPositionInStarYCoordinates
            ;stringmessage = "currentValueInStarYCoordinates ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinates
            ;stringmessage = "currentValueInStarYCoordinatesSorted ="
            ;PRINT, stringmessage
            ;PRINT, currentValueInStarYCoordinatesSorted

            ;stringmessage = "    currentMaxDistanceBetweenPairX ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairX
            ;stringmessage = "currentMaxDistanceBetweenPairY ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPairY
            ;stringmessage = "currentMaxDistanceBetweenPair ="
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair

        endif

        consideredPairIndex = consideredPairIndex + 1

    endelse


endwhile


    ;hunt in negative x direction
consideredPairIndex = -1
currentMaxDistanceBetweenPairX = 0
while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
     begin

    if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) lt 0 then begin
        currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
    endif else begin

        currentPositionInStarXCoordinatesSorted =
             centralPositionInStarXCoordinatesSorted + consideredPairIndex
        currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
             currentPositionInStarXCoordinatesSorted]
        currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
        currentValueInStarYCoordinates = starYCoordinates[
             currentPositionInStarYCoordinates]
        currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

        ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinatesSorted
        ;stringmessage = "currentValueInStarXCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarXCoordinatesSorted
        ;stringmessage = "currentPositionInStarXCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarXCoordinates
        ;stringmessage = "currentPositionInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentPositionInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinates ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinates
        ;stringmessage = "currentValueInStarYCoordinatesSorted ="
        ;PRINT, stringmessage
        ;PRINT, currentValueInStarYCoordinatesSorted

        currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
             centralValueInStarXCoordinatesSorted)
        currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
             centralValueInStarYCoordinatesSorted)
        ;averageYDeclination = (currentValueInStarYCoordinatesSorted +
             centralValueInStarYCoordinatesSorted) / 2
        averageYDeclination = centralValueInStarYCoordinatesSorted
        currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
             currentMaxDistanceBetweenPairX
        currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

        currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
              currentMaxDistanceBetweenPairYCorrected^2)

        ;stringmessage = "    currentMaxDistanceBetweenPairX ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairX
        ;stringmessage = "currentMaxDistanceBetweenPairY ="
        ;PRINT, stringmessage
        ;PRINT, currentMaxDistanceBetweenPairY
        ;stringmessage = "currentMaxDistanceBetweenPair ="
```

```
            ;PRINT, stringmessage
            ;PRINT, currentMaxDistanceBetweenPair

            if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
                then begin

                starIDArrayForClosePairs1[totalNumberOfPairsFound+numberOfPairsFound] =
                    starID[starIndexArray]
                starIDArrayForClosePairs2[totalNumberOfPairsFound+numberOfPairsFound] =
                    starID[currentPositionInStarXCoordinates]
                starDistanceArrayForClosePairs[totalNumberOfPairsFound+numberOfPairsFound] =
                    currentMaxDistanceBetweenPair
                starDataArrayForClosePairs1[*, totalNumberOfPairsFound+numberOfPairsFound] =
                    data[*, starIndexArray]
                starDataArrayForClosePairs2[*, totalNumberOfPairsFound+numberOfPairsFound] =
                    data[*, currentPositionInStarXCoordinates]

                numberOfPairsFound = numberOfPairsFound + 1
                ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
                    +9999

                ;stringmessage = "                                Pair Found B!"
                ;PRINT, stringmessage
                ;stringmessage = "StarIndex = "
                ;PRINT, stringmessage
                ;PRINT, starIndex
                ;stringmessage = "currentMaxDistanceBetweenPair = "
                ;PRINT, stringmessage
                ;PRINT, currentMaxDistanceBetweenPair
                ;exit    ;TEMP

                ;stringmessage = "    centralPositionInStarXCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, centralPositionInStarXCoordinatesSorted
                ;stringmessage = "centralValueInStarXCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, centralValueInStarXCoordinatesSorted
                ;stringmessage = "centralPositionInStarYCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, centralPositionInStarYCoordinatesSorted
                ;stringmessage = "centralValueInStarYCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, centralValueInStarYCoordinatesSorted

                ;stringmessage = "    currentPositionInStarXCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, currentPositionInStarXCoordinatesSorted
                ;stringmessage = "currentValueInStarXCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, currentValueInStarXCoordinatesSorted
                ;stringmessage = "currentPositionInStarXCoordinates ="
                ;PRINT, stringmessage
                ;PRINT, currentPositionInStarXCoordinates
                ;stringmessage = "currentPositionInStarYCoordinates ="
                ;PRINT, stringmessage
                ;PRINT, currentPositionInStarYCoordinates
                ;stringmessage = "currentValueInStarYCoordinates ="
                ;PRINT, stringmessage
                ;PRINT, currentValueInStarYCoordinates
                ;stringmessage = "currentValueInStarYCoordinatesSorted ="
                ;PRINT, stringmessage
                ;PRINT, currentValueInStarYCoordinatesSorted

                ;stringmessage = "    currentMaxDistanceBetweenPairX ="
                ;PRINT, stringmessage
                ;PRINT, currentMaxDistanceBetweenPairX
                ;stringmessage = "currentMaxDistanceBetweenPairY ="
                ;PRINT, stringmessage
                ;PRINT, currentMaxDistanceBetweenPairY
                ;stringmessage = "currentMaxDistanceBetweenPair ="
                ;PRINT, stringmessage
                ;PRINT, currentMaxDistanceBetweenPair



            endif


            consideredPairIndex = consideredPairIndex - 1

        endelse


    endwhile

    totalNumberOfPairsFound = totalNumberOfPairsFound+numberOfPairsFound
endfor

stringmessage = "totalNumberOfPairsFound = "
PRINT, stringmessage
```

```
        PRINT, (totalNumberOfPairsFound/2)

        totalNumberOfPairsFoundAcrossAllSamples = totalNumberOfPairsFoundAcrossAllSamples +
            totalNumberOfPairsFound
    endfor

    ;PRINT, starIDArrayForClosePairs
    starDistanceArrayForClosePairs = starDistanceArrayForClosePairs[0: (
        totalNumberOfPairsFoundAcrossAllSamples-1)] ;clip starDistanceArrayForClosePairs
    starDistanceArrayForClosePairsSortIndicies = sort(starDistanceArrayForClosePairs)
    starIDArrayForClosePairsSorted1 = starIDArrayForClosePairs1[
        starDistanceArrayForClosePairsSortIndicies]
    starIDArrayForClosePairsSorted2 = starIDArrayForClosePairs2[
        starDistanceArrayForClosePairsSortIndicies]
    starDistanceArrayForClosePairsSorted = starDistanceArrayForClosePairs[
        starDistanceArrayForClosePairsSortIndicies]
    starDataArrayForClosePairsSorted1 = starDataArrayForClosePairs1[*,
        starDistanceArrayForClosePairsSortIndicies]
    starDataArrayForClosePairsSorted2 = starDataArrayForClosePairs2[*,
        starDistanceArrayForClosePairsSortIndicies]


    starIDArrayForClosePairsSorted1 = transpose(starIDArrayForClosePairsSorted1)
    starIDArrayForClosePairsSorted2 = transpose(starIDArrayForClosePairsSorted2)
    starDistanceArrayForClosePairsSorted = transpose(starDistanceArrayForClosePairsSorted)


    finalData = [starDataArrayForClosePairsSorted1, starDistanceArrayForClosePairsSorted,
        starDataArrayForClosePairsSorted2]
    finalDataID = [starIDArrayForClosePairsSorted1, starIDArrayForClosePairsSorted2]

    PRINT, "finalData = "
    PRINT, finalData
    PRINT, "finalDataID = "
    PRINT, finalDataID

    outputFileName1='BOCPLPart1.txt'
    OPENW,1,outputFileName1
    PRINTF,1,finalData,FORMAT='(F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1
        X,F,1X,F,1X,F)'
    CLOSE,1
    outputFileName2='BOCPLPart2.txt'
    OPENW,1,outputFileName2
    PRINTF,1,finalDataID,FORMAT='(I19,1X,I19)'
    CLOSE,1


    ;stringmessage = "starDistanceArrayForClosePairs = "
    ;PRINT, stringmessage
    ;PRINT, starDistanceArrayForClosePairs

    ;stringmessage = "starDistanceArrayForClosePairsSorted = "
    ;PRINT, stringmessage
    ;PRINT, starDistanceArrayForClosePairsSorted

    ;stringmessage = "starIDArrayForClosePairs = "
    ;PRINT, stringmessage
    ;PRINT, starIDArrayForClosePairs

    ;stringmessage = "starIDArrayForClosePairsSorted = "
    ;PRINT, stringmessage
    ;PRINT, starIDArrayForClosePairsSorted


    stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
    PRINT, stringmessage
    PRINT, maxDistanceForConsideredPairInArcSeconds

    stringmessage = "fieldNumberOfStars = "
    PRINT, stringmessage
    PRINT, fieldNumberOfStars

    stringmessage = "fieldSquareDegrees = "
    PRINT, stringmessage
    PRINT, fieldSquareDegrees

    averageNumberOfPairsFound = (DOUBLE(totalNumberOfPairsFoundAcrossAllSamples)/DOUBLE(
        numberOfSamples))/DOUBLE(2)
    stringmessage = "averageNumberOfPairsFound = "
    PRINT, stringmessage
    PRINT, averageNumberOfPairsFound

endfor



end
```

A.2.6 Create BOCPL Subset using Is Quasar Data

This program generates a list of blue object pairs from a BOCPL which do not match with objects from a quasar database within a given angular separation.

See Section A.2.9

A.2.7 create BOCPL Subset using Visual Check Bad List

This program generates a list of blue object pairs from a BOCPL which do not match with objects from a visual check bad database within a given angular separation.

See Section A.2.9

A.2.8 create BO Subset Using Generic Ra and Dec Data

This program generates a subset of a BO list based upon a list of arbitrary Ra and Dec values.

BOCPL2createBOsubsetusingRaDecList-01June11a.zip

BOCPL2createBOsubsetusingRaDecList.pro

```
; Title: Blue Object Close Pair List - create BO Subset using Ra Dec List
; File: BOCPL2createBOsubsetusingRaDecList.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 21 Mar 11 5:30pm (updated 12 April 11 5:30pm)
; Summary:
;
;   Postconditions;
;
;   This program generates a list of blue object pairs from database 1 which DO NOT match with
    objects from database 2 within a given angular separation. The output files are of the
    following format;
;
;   BOCPLSDSSconfirmedMatches.txt [RIGHT_ASCENSION A, DECLINATION A, GALATIC_LATITUDE A,
    GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR, RIGHT_ASCENSION B,
    DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B]
;   BOCPLSDSSconfirmedMatches.txt [ID A, ID B]
;   BOCPLSDSSconfirmedNonMatches.txt [RIGHT_ASCENSION A, DECLINATION A, GALATIC_LATITUDE A,
    GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR, RIGHT_ASCENSION B,
    DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B]
;   BOCPLSDSSconfirmedNonMatches.txt [ID A, ID B]
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   default: BOCPLSDSSblueObjectsDR7detailed.txt [ID A, RIGHT_ASCENSION A, DECLINATION A,
    GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U A, G A, R A, POK A] for X blue objects, tab
    delimited
;   alternate: BOCPLSDSSBOconfirmedNotQuasarsData.txt [RIGHT_ASCENSION A, DECLINATION A,
    GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR,
    RIGHT_ASCENSION B, DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B
    ] for X blue objects Pairs, tab delimited
;               BOCPLSDSSBOconfirmedNotQuasarsIDs.txt  [ID A, ID B] for X blue objects Pairs, tab
    delimited
;   SDSSsubset.txt, Eg BOCPLVisualCheckBad.txt, BOCPLHarrisSDSSDR4WDDataSmall.txt,
    SDSSDR7QuasarCatalogDataSmall.txt [RIGHT_ASCENSION, DECLINATION] for z blue objects, space
    delimited


maxDistanceForMatchedStarInArcSeconds = 1.0
maxDistanceForMatchedStarInArcDegrees = maxDistanceForMatchedStarInArcSeconds/60D/60D

    ;long precision is required for ID
```

```
;default:
DB1NameOfFile = 'BOCPLSDSSblueObjectsDR7detailed.txt'
readcol64bitvars, DB1NameOfFile, ID_DB1, RIGHT_ASCENSION_DB1, DECLINATION_DB1, GALATIC_LATITUDE_DB1,
      GALATIC_LONGITUDE_DB1, U_DB1, G_DB1, R_DB1, POK_DB1, FORMAT='G,D,D,D,D,D,D,D,D', SKIPLINE=1

;alternate:
;DB1NameOfFilePart1 = 'BOCPLSDSSBOconfirmedNotQuasarsData.txt'        ;
     BOCPLSDSSBOconfirmedNotBadObjectsData
;DB1NameOfFilePart2 = 'BOCPLSDSSBOconfirmedNotQuasarsIDs.txt'         ;
     BOCPLSDSSBOconfirmedNotBadObjectsData
;readcol64bitvars, DB1NameOfFilePart1, RIGHT_ASCENSION_DB1, DECLINATION_DB1, GALATIC_LATITUDE_DB1,
     GALATIC_LONGITUDE_DB1, U_DB1, G_DB1, R_DB1, POK_DB1, FORMAT='D,D,D,D,D,D,D,D'
;readcol64bitvars, DB1NameOfFilePart2, ID_DB1, FORMAT='G'

DB2NameOfFile = 'SDSSsubset.txt'
readcol64bitvars, DB2NameOfFile, RIGHT_ASCENSION_DB2, DECLINATION_DB2, FORMAT='D,D', SKIPLINE=1

DB1starID = [transpose(ID_DB1)]

DB1fieldNumberOfStars = (size(RIGHT_ASCENSION_DB1))[1]
DB2fieldNumberOfStars = (size(RIGHT_ASCENSION_DB2))[1]

DB1AstarYCoordinates = transpose(DECLINATION_DB1)
DB1AstarXCoordinates = transpose(RIGHT_ASCENSION_DB1)
DB2starYCoordinates = transpose(DECLINATION_DB2)
DB2starXCoordinates = transpose(RIGHT_ASCENSION_DB2)

    PRINT, "DB1fieldNumberOfStars = ", DB1fieldNumberOfStars
    PRINT, "DB2fieldNumberOfStars = ", DB2fieldNumberOfStars


maxNumberOfMatchesExpected = 200000

DB1data = [transpose(RIGHT_ASCENSION_DB1), transpose(DECLINATION_DB1), transpose(
     GALATIC_LATITUDE_DB1), transpose(GALATIC_LONGITUDE_DB1), transpose(U_DB1), transpose(G_DB1),
     transpose(R_DB1), transpose(POK_DB1)]


starIDArrayForNonMatches = lon64arr(1, maxNumberOfMatchesExpected)
starDataArrayForNonMatches = dblarr(8, maxNumberOfMatchesExpected)

nonMatchIndex = 0L
numberOfMatchesFound = 0L

for DB1starIndex=0L, DB1fieldNumberOfStars-1L, 1L do begin

    PRINT, "DB1starIndex = ", DB1starIndex

        AfoundInDB2 = 0
    AindexInDB2 = 0
    for DB2starIndex=0L, DB2fieldNumberOfStars-1L, 1L do begin

        currentMaxDistanceBetweenMatchX = ABS(DB1AstarXCoordinates[DB1starIndex] -
             DB2starXCoordinates[DB2starIndex])
        currentMaxDistanceBetweenMatchY = ABS(DB1AstarYCoordinates[DB1starIndex] -
             DB2starYCoordinates[DB2starIndex])
        if (currentMaxDistanceBetweenMatchX lt maxDistanceForMatchedStarInArcDegrees) then begin

                if (currentMaxDistanceBetweenMatchY lt maxDistanceForMatchedStarInArcDegrees) then
                     begin
                 AfoundInDB2 = 1
                 AindexInDB2 = DB2starIndex

                 ;PRINT, "AfoundInDB2"
                 ;PRINT, DB2starXCoordinates[DB2starIndex]
                 ;PRINT, "DB2starIndex = ", DB2starIndex
                endif
        endif

    endfor



    if (AfoundInDB2 eq 1) then begin
        numberOfMatchesFound = numberOfMatchesFound +1
        ;PRINT, "new"
    endif else begin

        ;no objects found in pair

        indexOfSameStarID = where(starIDArrayForNonMatches eq DB1starID[DB1starIndex])

        if (indexOfSameStarID[0] eq -1) then begin
            ;do not add double ups
            starIDArrayForNonMatches[*, nonMatchIndex] = DB1starID[*, DB1starIndex]
            starDataArrayForNonMatches[*, nonMatchIndex] = DB1data[*, DB1starIndex]
            nonMatchIndex = nonMatchIndex + 1

        endif
    endelse

    ;PRINT, "DB1starIndex = ", DB1starIndex
```

```
endfor


totalNumberOfNonMatchesFound = nonMatchIndex

stringmessage = "totalNumberOfNonMatchesFound = "
PRINT, stringmessage
PRINT, totalNumberOfNonMatchesFound

PRINT, "numberOfMatchesFound = ", numberOfMatchesFound


starDataArrayForNonMatches = starDataArrayForNonMatches[*, 0: (totalNumberOfNonMatchesFound-1)]
     ;clip starDataArrayForNonMatches
starIDArrayForNonMatches = starIDArrayForNonMatches[0: (totalNumberOfNonMatchesFound-1)]     ;clip
     starDistanceArrayForNonMatches



finalData = [starDataArrayForNonMatches]
finalDataID = [starIDArrayForNonMatches]


     ;PRINT, "finalData = "
     ;PRINT, finalData
     ;PRINT, "finalDataID = "
     ;PRINT, finalDataID


     outputFileName1='BOCPLSDSSBOconfirmedNonMatchesData.txt'
     OPENW,1,outputFileName1
     PRINTF,1,finalData,FORMAT='(F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F)'
     CLOSE,1
     outputFileName2='BOCPLSDSSBOconfirmedNonMatchesID.txt'
     OPENW,1,outputFileName2
     PRINTF,1,finalDataID,FORMAT='(I18)'
     CLOSE,1



end
```

BOCPstatistics-02June11a.ods

all (with bad objects)

number without	out q	number	DR	DR coverage	DR total num	fraction	Fraction (DR corre	Fraction (DR	Fraction (DR	and have spe
	BO	36681	DR7	11663	357	1.0000	=H4/F4*$F$4	NA	NA	
	BO with spectra	9788	DR7	11663	357	=D5/$D$4	=H5/F5*$F$4	NA	NA	
35390	**BO with confirmed quasars**	=$D$4-B6	DR7	11663	357	=D6/$D$4	=H6/F6*$F$4	=I6/$H$5	=I6	
31331	**BO with confirmed WDs**	=$D$4-B7	DR4	6670	180	=D7/$D$4	=H7/F7*$F$4	=I7/$H$5	=J7	

60as (with bad objects)

number pairs	number without	out q	number pairs	DR	DR coverage	DR total num	fraction	Fraction (DR corre	Fraction (DR	Fraction (DR	and have spe
278		**BOCP**	=A12/2	DR7	11663	357	1.0000	=H12/F12*$F$12	NA	NA	
	97	**BOCP with spectra**	=$D$12-B13	DR7	11663	357	=D13/$D$12	=H13/F13*$F$12	NA	NA	
	124	**BOCP with confirmed quasars**	=$D$12-B14	DR7	11663	357	=D14/$D$12	=H14/F14*$F$12	=I14/$H$13	=I14	
	107	**BOCP with confirmed WDs**	=$D$12-B15	DR4	6670	180	=D15/$D$12	=H15/F15*$F$12	=I15/$H$13	=J15	

60as

number pairs	number without	out q	number pairs	DR	DR coverage	DR total num	fraction	Fraction (DR corre	Fraction (DR	Fraction (DR	and have spe
		BOCP no bad	98	DR7	11663	357	1.0000	=H19/F19*$F$4	NA	NA	
	69	**BOCP no bad with spectra**	=$D$19-B20	DR7	11663	357	=D20/$D$19	=H20/F20*$F$4	NA	NA	
	93	**BOCP no bad with confirmed quasars**	=$D$19-B21	DR7	11663	357	=D21/$D$19	=H21/F21*$F$4	=I21/$H$20	=I21	
	76	**BOCP no bad with confirmed WDs**	=$D$19-B22	DR4	6670	180	=D22/$D$19	=H22/F22*$F$19	=I22/$H$20	=J22	

30as

number pairs	number without	out q	number pairs	DR	DR coverage	DR total num	fraction	Fraction (DR corre	Fraction (DR	Fraction (DR	and have spe
		BOCP no bad	55	DR7	11663	357	1.0000	=H26/F26*$F$4	NA	NA	
	41	**BOCP no bad with spectra**	=$D$26-B27	DR7	11663	357	=D27/$D$26	=H27/F27*$F$4	NA	NA	
	55	**BOCP no bad with confirmed quasars**	=$D$26-B28	DR7	11663	357	=D28/$D$26	=H28/F28*$F$4	=I28/$H$27	=I28	
	42	**BOCP no bad with confirmed WDs**	=$D$26-B29	DR4	6670	180	=D29/$D$26	=H29/F29*$F$26	=I29/$H$27	=J29	

30as (> 1as)

number pairs	number without	out q	number pairs	DR	DR coverage	DR total num	fraction	Fraction (DR corre	Fraction (DR	Fraction (DR	and have spe
		BOCP no bad	53	DR7	11663	357	1.0000	=H33/F33*$F$4	NA	NA	
	40	**BOCP no bad with spectra**	=$D$33-B34	DR7	11663	357	=D34/$D$33	=H34/F34*$F$4	NA	NA	
	53	**BOCP no bad with confirmed quasars**	=$D$33-B35	DR7	11663	357	=D35/$D$33	=H35/F35*$F$4	=I35/$H$34	=I35	
	41	**BOCP no bad with confirmed WDs**	=$D$33-B36	DR4	6670	180	=D36/$D$33	=H36/F36*$F$33	=I36/$H$34	=J36	

A.2.9 create BOCP Subset Using Generic Ra and Dec Data

This program generates a subset of a BOCP list based upon a list of arbitrary Ra and Dec values.

BOCPL2createBOsubsetusingRaDecList-01June11a.zip

BOCPL2createBOCPsubsetusingRaDecList.pro

```
; Title: Blue Object Close Pair List - create BOCP Subset using Ra Dec List
; File: BOCPL2createBOCPsubsetusingRaDecList.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 10 Apr 11 11:50pm (updated 11 April 11 11:50pm)
; Summary:
;
;   Postconditions;
;
;   This program generates a list of blue object pairs from database 1 which DO NOT match with
     objects from database 2 within a given angular separation. The output files are of the
     following format;
;
;   BOCPLSDSSconfirmedMatches.txt [RIGHT_ASCENSION A, DECLINATION A, GALATIC_LATITUDE A,
     GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR, RIGHT_ASCENSION B,
     DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B]
;   BOCPLSDSSconfirmedMatches.txt [ID A, ID B]
;   BOCPLSDSSconfirmedNonMatches.txt [RIGHT_ASCENSION A, DECLINATION A, GALATIC_LATITUDE A,
     GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR, RIGHT_ASCENSION B,
     DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B]
;   BOCPLSDSSconfirmedNonMatches.txt [ID A, ID B]
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   BOCPLPart1.txt/BOCPLSDSSBOCPconfirmedNotBadData.txt [RIGHT_ASCENSION A, DECLINATION A,
     GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR,
     RIGHT_ASCENSION B, DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B
     ] for X blue objects Pairs, tab delimited
;   BOCPLPart2.txt/BOCPLSDSSBOCPconfirmedNotBadIDs.txt [ID A, ID B] for X blue objects Pairs, tab
     delimited
;   SDSSsubset.txt, Eg BOCPLVisualCheckBad.txt, BOCPLHarrisSDSSDR4WDDataSmall.txt,
     SDSSDR7QuasarCatalogDataSmall.txt [RIGHT_ASCENSION, DECLINATION] for z blue objects, space
     delimited
;
; Important;
;
;   if using this code to regenerate original Bad Object / Quasar eliminated BOCPs then must
     uncomment "if (indexOfSameStarID[0] eq -1) then begin"
;       Note that this original code halves the list (removes duplicates pairs), PLUS it removes
     approximately 50% of all triple systems from the list (due to a software limitation) - not 0%
     or 100% of them, which would be more generally expected

maxDistanceForConsideredPairInArcSeconds = double(60)
maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D

maxDistanceForMatchedStarInArcSeconds = double(1)
maxDistanceForMatchedStarInArcDegrees = maxDistanceForMatchedStarInArcSeconds/60D/60D

DB1NameOfFilePart1 = 'BOCPLPart1.txt'    ;default = BOCPLPart1.txt/BOCPLSDSSBOCPconfirmedNotBadData.
     txt/BOCPLSDSSBOCPconfirmedNotBadData.txt.30as
DB1NameOfFilePart2 = 'BOCPLPart2.txt'        ;default = BOCPLPart2.txt/
     BOCPLSDSSBOCPconfirmedNotBadIDs.txt/BOCPLSDSSBOCPconfirmedNotBadIDs.txt.30as
DB2NameOfFile = 'SDSSsubset.txt'

    ;long precision is required for ID
readcol64bitvars, DB1NameOfFilePart1, RIGHT_ASCENSION_DB1_A, DECLINATION_DB1_A,
     GALATIC_LATITUDE_DB1_A, GALATIC_LONGITUDE_DB1_A, U_DB1_A, G_DB1_A, R_DB1_A, POK_DB1_A,
     DISTANCE_BETWEEN_PAIR_DB1, RIGHT_ASCENSION_DB1_B, DECLINATION_DB1_B, GALATIC_LATITUDE_DB1_B,
     GALATIC_LONGITUDE_DB1_B, U_DB1_B, G_DB1_B, R_DB1_B, POK_DB1_B, FORMAT='D,D,D,D,D,D,D,D,D,D,D,
     D,D,D,D,D,D,D,D,D'
readcol64bitvars, DB1NameOfFilePart2, ID_DB1_A, ID_DB1_B, FORMAT='G,G'
readcol64bitvars, DB2NameOfFile, RIGHT_ASCENSION_DB2, DECLINATION_DB2, FORMAT='D,D', SKIPLINE=1

DB1starID = [transpose(ID_DB1_A), transpose(ID_DB1_B)]

DB1fieldNumberOfStars = (size(RIGHT_ASCENSION_DB1_A))[1]
DB2fieldNumberOfStars = (size(RIGHT_ASCENSION_DB2))[1]
```

```
DB1AstarYCoordinates = transpose(DECLINATION_DB1_A)
DB1AstarXCoordinates = transpose(RIGHT_ASCENSION_DB1_A)
DB1BstarYCoordinates = transpose(DECLINATION_DB1_B)
DB1BstarXCoordinates = transpose(RIGHT_ASCENSION_DB1_B)
DB2starYCoordinates = transpose(DECLINATION_DB2)
DB2starXCoordinates = transpose(RIGHT_ASCENSION_DB2)

    PRINT, "DB1fieldNumberOfStars = ", DB1fieldNumberOfStars
    PRINT, "DB2fieldNumberOfStars = ", DB2fieldNumberOfStars


maxNumberOfMatchesExpected = 400000

DB1data = [transpose(RIGHT_ASCENSION_DB1_A), transpose(DECLINATION_DB1_A), transpose(
    GALATIC_LATITUDE_DB1_A), transpose(GALATIC_LONGITUDE_DB1_A), transpose(U_DB1_A), transpose(
    G_DB1_A), transpose(R_DB1_A), transpose(POK_DB1_A), transpose(DISTANCE_BETWEEN_PAIR_DB1),
    transpose(RIGHT_ASCENSION_DB1_B), transpose(DECLINATION_DB1_B), transpose(
    GALATIC_LATITUDE_DB1_B), transpose(GALATIC_LONGITUDE_DB1_B), transpose(U_DB1_B), transpose(
    G_DB1_B), transpose(R_DB1_B), transpose(POK_DB1_B)]
DB1Distances = transpose(DISTANCE_BETWEEN_PAIR_DB1)


starIDArrayForNonMatches = lon64arr(2, maxNumberOfMatchesExpected)
starDistanceArrayForNonMatches = dblarr(maxNumberOfMatchesExpected)
starDataArrayForNonMatches = dblarr(17, maxNumberOfMatchesExpected)

nonMatchIndex = 0L

numberOfMatchesFound = 0L

for DB1starIndex=0L, DB1fieldNumberOfStars-1L, 1L do begin

        AfoundInDB2 = 0
    AindexInDB2 = 0
    for DB2starIndex=0L, DB2fieldNumberOfStars-1L, 1L do begin

        currentMaxDistanceBetweenMatchX = ABS(DB1AstarXCoordinates[DB1starIndex] -
            DB2starXCoordinates[DB2starIndex])
        currentMaxDistanceBetweenMatchY = ABS(DB1AstarYCoordinates[DB1starIndex] -
            DB2starYCoordinates[DB2starIndex])
        if (currentMaxDistanceBetweenMatchX lt maxDistanceForMatchedStarInArcDegrees) then begin

                if (currentMaxDistanceBetweenMatchY lt maxDistanceForMatchedStarInArcDegrees) then
                    begin
                 AfoundInDB2 = 1
                 AindexInDB2 = DB2starIndex

                 ;PRINT, "AfoundInDB2"

                endif
        endif

    endfor
    BfoundInDB2 = 0
    BindexInDB2 = 0
    for DB2starIndex=0L, DB2fieldNumberOfStars-1L, 1L do begin

        currentMaxDistanceBetweenMatchX = ABS(DB1BstarXCoordinates[DB1starIndex] -
            DB2starXCoordinates[DB2starIndex])
        currentMaxDistanceBetweenMatchY = ABS(DB1BstarYCoordinates[DB1starIndex] -
            DB2starYCoordinates[DB2starIndex])

        ;PRINT, " DB2starXCoordinates[DB2starIndex] = " +  STRING(DB2starXCoordinates[DB2starIndex])
        ;PRINT, " DB2starYCoordinates[DB2starIndex] = " +  STRING(DB2starYCoordinates[DB2starIndex])
        ;PRINT, " DB1BstarXCoordinates[DB1starIndex] = " +  STRING(DB1BstarXCoordinates[DB1starIndex
            ])
        ;PRINT, " DB1BstarYCoordinates[DB1starIndex] = " +  STRING(DB1BstarYCoordinates[DB1starIndex
            ])

        if (currentMaxDistanceBetweenMatchX lt maxDistanceForMatchedStarInArcDegrees) then begin

                if (currentMaxDistanceBetweenMatchY lt maxDistanceForMatchedStarInArcDegrees) then
                    begin
                 BfoundInDB2 = 1
                 BindexInDB2 = DB2starIndex

                 ;PRINT, "BfoundInDB2"

                endif
        endif

    endfor

    if (AfoundInDB2 eq 1) then begin
        numberOfMatchesFound = numberOfMatchesFound +1
    endif
    if (BfoundInDB2 eq 1) then begin
        numberOfMatchesFound = numberOfMatchesFound +1
    endif
```

```
    if (AfoundInDB2 eq 1) then begin
        if (BfoundInDB2 eq 1) then begin

            ;dual object found in pair

        endif else begin

            ;single object found in pair
        endelse

    endif else begin

        if (BfoundInDB2 eq 1) then begin

            ;single object found in pair

        endif else begin

            ;no objects found in pair

            indexOfSameStarID = where(starIDArrayForNonMatches eq DB1starID[0, DB1starIndex])
            indexOfSameStarIDb = where(starIDArrayForNonMatches eq DB1starID[1, DB1starIndex])

            if (indexOfSameStarID[0] eq -1) then begin  ;ORIGINAL: removes all duplicates + ~50% of
                 all triple systems
                if (indexOfSameStarIDb[0] eq -1) then begin ;NEW: removes the other ~50% of the
                     triples
                ;do not add double ups [comment out the above 2 conditions if creating a list of ra
                     dec values in order to perform a visual check for bad objects [ie, in order to
                     manually create a bad object list BOCPLVisualCheckBad.txt]

                    distanceBetweenPair = DB1Distances[DB1starIndex]

                    if (distanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees) then begin
                        starIDArrayForNonMatches[*, nonMatchIndex] = DB1starID[*, DB1starIndex]
                        starDistanceArrayForNonMatches[nonMatchIndex] = DB1Distances[DB1starIndex]
                        starDataArrayForNonMatches[*, nonMatchIndex] = DB1data[*, DB1starIndex]
                        nonMatchIndex = nonMatchIndex + 1
                    endif
                endif
            endif

        endelse
    endelse

    PRINT, "DB1starIndex = ", DB1starIndex

endfor


totalNumberOfNonMatchesFound = nonMatchIndex


stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
PRINT, stringmessage
PRINT, maxDistanceForConsideredPairInArcSeconds

stringmessage = "totalNumberOfNonMatchesFound = "
PRINT, stringmessage
PRINT, totalNumberOfNonMatchesFound

numberOfMatchesFound = numberOfMatchesFound/2L
PRINT, "numberOfMatchesFound = ", numberOfMatchesFound


;NB sorting is not necessary as the input DB1 is assumed already to be sorted wrt distance between
     pairs
starDistanceArrayForNonMatches = starDistanceArrayForNonMatches[0: (totalNumberOfNonMatchesFound-1)]
        ;clip starDistanceArrayForNonMatches
starDistanceArrayForNonMatchesSortIndicies = sort(starDistanceArrayForNonMatches)        ;NB to sort
     by Ra;  sort(starDataArrayForNonMatches[0, 0: (totalNumberOfNonMatchesFound-1)])
starIDArrayForNonMatchesSorted = starIDArrayForNonMatches[*,
     starDistanceArrayForNonMatchesSortIndicies]
;starDistanceArrayForNonMatchesSorted = starDistanceArrayForNonMatches[
     starDistanceArrayForNonMatchesSortIndicies]
starDataArrayForNonMatchesSorted = starDataArrayForNonMatches[*,
     starDistanceArrayForNonMatchesSortIndicies]
;starDistanceArrayForNonMatchesSorted = transpose(starDistanceArrayForNonMatchesSorted)



finalData = [starDataArrayForNonMatchesSorted]
finalDataID = [starIDArrayForNonMatchesSorted]



    ;PRINT, "finalData = "
    ;PRINT, finalData
    ;PRINT, "finalDataID = "
    ;PRINT, finalDataID
```

```
outputFileName1='BOCPLSDSSBOCPconfirmedNonMatchesData.txt'
OPENW,1,outputFileName1
PRINTF,1,finalData,FORMAT='(F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F)'
CLOSE,1
outputFileName2='BOCPLSDSSBOCPconfirmedNonMatchesIDs.txt'
OPENW,1,outputFileName2
PRINTF,1,finalDataID,FORMAT='(I18,1X,I18)'
CLOSE,1


end
```

A.2.10 create BOCPL Subset Using Bergeron Theoretical Data

This program generates a list of blue object pairs from a BOCPL which have a high mass ratio based upon bergeron theoretical DA WD data. This code was not used, as I did not wish to bias my target selection towards particular estimated masses (such that I could form a mass distribution).

A subset was create using Bergeron Theoretical Data to select those candidates estimated to have a significant difference in mass between their binary components. Their mass difference was estimated by either requiring a) significant magnitude difference but same temperature, or b) significant theoretical magnitude difference at a given arbitrary temperature derived from a theoretical cooling tract of arbitrary mass based on their current photometric derived temperatures (assuming this arbitrary mass in both cases). Ideally, this meant ideally selecting BOCPs with similar colours ($u-g$ and $g-r$) / temperatures, therefore any apparent luminosity difference being a product of a significant difference in mass. An alternative, theory dependent, method was applied as follows;

1. For both objects, in Bergeron theoretical database (DA WD $M_{\odot} = 0.6$ table, Table_Mass_0.6, http://www.astro.umontreal.ca/ bergeron/CoolingModels/), located the hypothetical WDs with the closest matching $u-r$ value (NB $u-g$ and $g-r$ can also be used).

2. For both objects, obtained the corresponding theoretical absolute r magnitude

3. calculated the theoretical r absolute magnitude difference between the binary WDs

4. calculated the measured r apparent magnitude difference between the binary WDs

5. the difference between 3. and 4. corresponds to differences in WD mass/radii, so if this difference is greater than some threshold (eg 0.5) I assumed this represented a binary system with a significant mass

The following code was created to execute this method.

BOCPL4createSubsetUsingBergeronTheoreticalData.pro

```
; Title: Blue Object Close Pair List - create Subset Using Bergeron Theoretical Data
; File: BOCPL4createSubsetUsingBergeronTheoreticalData.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 6 October 09 2pm
; Summary:
;
;   Postconditions;
;
;   This program plots u-g vs g-r for each blue object pair, and provides the apparent magnitudes (r
    ) for both objects
;
;       For every binary pair, calculate u-g and g-r
;           0. require binary WD pair to have similar colours (u-g and g-r)
;       1. In Bergeron theoretical database (DA WD solar mass 0.6 table), locate the hyp WD with
    closest matching u-g [or g-r] value ]
;           2. obtain the corresponding theoretical absolute r magnitude (and theoretical
    temperature, although not currently used)
;           3. calculate the theoretical r absolute magnitude difference between the binary WDs
;           4. calculate the measured r apparent magnitude difference between the binary WDs
;           5. the difference between 3. and 4. must correspond to differences in WD mass/radii, so
    if this difference is > some threshold, we can assume a binary system (assuming it actually is
    a real binary system - requiring proper motions for confirmation) with a low and a high mass WD
;
;       The output files are of the following format;
;       BOCPLBergeronTheoreticalComparisonPart1.txt [RIGHT_ASCENSION A, DECLINATION A,
    GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR,
    RIGHT_ASCENSION B, DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B
    ] for X blue objects Pairs, tab delimited
;       BOCPLBergeronTheoreticalComparisonPart2.txt [ID A, ID B] for Y blue objects Pairs, tab
    delimited
;
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   BOCPLSDSSDR5QuasarCatalogConfirmedStarsVisualCheckPart1.txt [RIGHT_ASCENSION A, DECLINATION A,
    GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR,
    RIGHT_ASCENSION B, DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B
    ] for X blue objects Pairs, tab delimited
;   BOCPLSDSSDR5QuasarCatalogConfirmedStarsVisualCheckPart2.txt [ID A, ID B]   for X blue objects
    Pairs, tab delimited
;       Table_Mass_0.6 [from http://www.astro.umontreal.ca/~bergeron/CoolingModels/, Teff, log g,
    Mbol, BC, U, B, V, R, I, J, H, K, u, g, r {not Teff, log g, Mbol, BC, U, B, V, R, I, J, H, K, u
    , g, r, i, z, y, b-y,u-b,v-y,V-I,G-R,U-V,U-G,B-V, Age}]


!p.multi = [0,6,6]

colourMatchWithUMinusGorGMinusR = 2 ;   2 = colourMatchWithUMinusR, 1 = colourMatchWithUMinusG, 0 =
    colourMatchWithGMinusR

rMagMinDifferenceForHighLowMassWDBinaryPair = 0.4  ;must be large [Eg 1.5; based upon the difference
     in magnitude between a 30k 0.3solar mass WD versus a 30k 1.0solar mass WD]
uMinusgMaxDifferenceForHighLowMassWDBinaryPair = 1.0  ;must be small
gMinusrMaxDifferenceForHighLowMassWDBinaryPair = 1.0  ;must be small

maxDistanceForConsideredPairInArcSeconds = double(30)   ;30
maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D

database1NameOfFilePart1 = 'BOCPLSDSSDR5QuasarCatalogConfirmedStarsVisualCheckPart1.txt'
database1NameOfFilePart2 = 'BOCPLSDSSDR5QuasarCatalogConfirmedStarsVisualCheckPart2.txt'

    ;long precision is required for ID
readcol64bitvars, database1NameOfFilePart1, RIGHT_ASCENSION_DB1_A, DECLINATION_DB1_A,
    GALATIC_LATITUDE_DB1_A, GALATIC_LONGITUDE_DB1_A, U_DB1_A, G_DB1_A, R_DB1_A, POK_DB1_A,
    DISTANCE_BETWEEN_PAIR_DB1, RIGHT_ASCENSION_DB1_B, DECLINATION_DB1_B, GALATIC_LATITUDE_DB1_B,
    GALATIC_LONGITUDE_DB1_B, U_DB1_B, G_DB1_B, R_DB1_B, POK_DB1_B, FORMAT='D,D,D,D,D,D,D,D,D,D,D,D,
    D,D,D,D,D,D,D,D,D'
readcol64bitvars, database1NameOfFilePart2, ID_DB1_A, ID_DB1_B, FORMAT='G,G'

;readcol cannot handle more than 16 columns of data
database2NameOfFile = 'Table_Mass_0.6'
bergeronMassXModelFileNumberOfRows = 57
readcol64bitvars, database2NameOfFile, Teff_DB2, logg_DB2, Mbol_DB2, BC_DB2, U_DB2, B_DB2, V_DB2,
    R_DB2, I_DB2, J_DB2, H_DB2, K_DB2, u_DB2, g_DB2, r_DB2, FORMAT='D,D,D,D,D,D,D,D,D,D,D,D,D,D',
     SKIPLINE=2


;Added by RBB for BOCPL4createSubsetUsingBergeronTheoreticalDataGEMINIOBJECTSPDSELECTIONDEBUG;
database1NameOfFilePart3 = 'BOCPLPDSelectionPart2.txt'
```

```
readcol64bitvars, database1NameOfFilePart3, PD_SELECTION_ID_DB1_A, PD_SELECTION_ID_DB1_B, FORMAT='G,
    G'
database3fieldNumberOfStars = (size(PD_SELECTION_ID_DB1_A))[1]


database1starID = [transpose(ID_DB1_A), transpose(ID_DB1_B)]
    ;database1AstarID = transpose(ID_DB1_A)
    ;database1BstarID = transpose(ID_DB1_B)

database1fieldNumberOfStars = (size(RIGHT_ASCENSION_DB1_A))[1]
;database2fieldNumberOfStars = (size(Teff_DB2))[1]

maxNumberOfMatchesExpected = 20000

database1data = [transpose(RIGHT_ASCENSION_DB1_A), transpose(DECLINATION_DB1_A), transpose(
    GALATIC_LATITUDE_DB1_A), transpose(GALATIC_LONGITUDE_DB1_A), transpose(U_DB1_A), transpose(
    G_DB1_A), transpose(R_DB1_A), transpose(POK_DB1_A), transpose(DISTANCE_BETWEEN_PAIR_DB1),
    transpose(RIGHT_ASCENSION_DB1_B), transpose(DECLINATION_DB1_B), transpose(
    GALATIC_LATITUDE_DB1_B), transpose(GALATIC_LONGITUDE_DB1_B), transpose(U_DB1_B), transpose(
    G_DB1_B), transpose(R_DB1_B), transpose(POK_DB1_B)]
database1Distances = transpose(DISTANCE_BETWEEN_PAIR_DB1)

;PRINT, "database1data = "
;PRINT, database1data


starIDArrayForMatches = lon64arr(2, maxNumberOfMatchesExpected)
starDistanceArrayForMatches = dblarr(maxNumberOfMatchesExpected)
starDataArrayForMatches = dblarr(17, maxNumberOfMatchesExpected)

matchIndex = 0L

for database1starIndex=0L, database1fieldNumberOfStars-1L, 1L do begin

    if (database1Distances[database1starIndex] lt maxDistanceForConsideredPairInArcDegrees) then
        begin


        uminusg = dblarr(2)
        gminusr = dblarr(2)
        uminusr = dblarr(2)
        r = dblarr(2)
    uminusg[0] = U_DB1_A[database1starIndex] - G_DB1_A[database1starIndex]
        uminusg[1] = U_DB1_B[database1starIndex] - G_DB1_B[database1starIndex]
    gminusr[0] = G_DB1_A[database1starIndex] - R_DB1_A[database1starIndex]
    gminusr[1] = G_DB1_B[database1starIndex] - R_DB1_B[database1starIndex]
    uminusr[0] = U_DB1_A[database1starIndex] - R_DB1_A[database1starIndex]
    uminusr[1] = U_DB1_B[database1starIndex] - R_DB1_B[database1starIndex]
    r[0] = R_DB1_A[database1starIndex]
    r[1] = R_DB1_B[database1starIndex]


    ;METHOD 1 - ADVANCED;

        if (ABS(uminusg[0] - uminusg[1]) lt uMinusgMaxDifferenceForHighLowMassWDBinaryPair) then
            begin

        if (ABS(gminusr[0] - gminusr[1]) lt gMinusrMaxDifferenceForHighLowMassWDBinaryPair) then
            begin

            database2correspondingTeff = dblarr(2)
            database2correspondingrMag = dblarr(2)

        for database1BinaryPairIndex=0L, 1L, 1L do begin

                database2closestMatchinguminusg = 9999999.99
            database2closestMatchinggminusr = 9999999.99
                    database2closestMatchinguminusr = 9999999.99
                database2closestMatchinguminusgIndex = 0
            database2closestMatchinggminusrIndex = 0
                    database2closestMatchinguminusrIndex = 0

                for database2starIndex=0L, bergeronMassXModelFileNumberOfRows-1L, 1L do begin

                database2gMinusr = g_DB2[database2starIndex] - r_DB2[database2starIndex]
                database2uMinusg = u_DB2[database2starIndex] - g_DB2[database2starIndex]
                        database2uMinusr = u_DB2[database2starIndex] - r_DB2[database2starIndex]

            if (ABS(database2gMinusr - gminusr[database1BinaryPairIndex]) lt
                database2closestMatchinggminusr) then begin
                    database2closestMatchinggminusr = ABS(database2gMinusr - gminusr[
                        database1BinaryPairIndex])
                database2closestMatchinggminusrIndex = database2starIndex
            endif

            if (ABS(database2uMinusg - uminusg[database1BinaryPairIndex]) lt
                database2closestMatchinguminusg) then begin
                    database2closestMatchinguminusg = ABS(database2uMinusg - uminusg[
                        database1BinaryPairIndex])
                database2closestMatchinguminusgIndex = database2starIndex
            endif
```

```
        if (ABS(database2uMinusr - uminusr[database1BinaryPairIndex]) lt
             database2closestMatchinguminusr) then begin
                database2closestMatchinguminusr = ABS(database2uMinusr - uminusr[
                     database1BinaryPairIndex])
            database2closestMatchinguminusrIndex = database2starIndex
        endif

        endfor

        ;now just use the closest matching u-g index [OR the closest matching g-r index]
            ;use this index to find the corresponding theoretical temperature and theoretical
                 absolute r magnitude
                    ;any remaining differences in r magnitude must correspond to differences in
                         WD mass/radii

                if (colourMatchWithUMinusGorGMinusR eq 1) then begin
                    database2correspondingTeff[database1BinaryPairIndex] = Teff_DB2[
                         database2closestMatchinguminusgIndex]
                    database2correspondingrMag[database1BinaryPairIndex] = r_DB2[
                         database2closestMatchinguminusgIndex]
        endif else begin

                if (colourMatchWithUMinusGorGMinusR eq 2) then begin
            ;print, "database2closestMatchinggminusr = ", database2closestMatchinggminusr
            ;print, "database2closestMatchinguminusrIndex = ",
                 database2closestMatchinguminusrIndex
                        database2correspondingTeff[database1BinaryPairIndex] = Teff_DB2[
                             database2closestMatchinguminusrIndex]
                        database2correspondingrMag[database1BinaryPairIndex] = r_DB2[
                             database2closestMatchinguminusrIndex]
        endif else begin
                        database2correspondingTeff[database1BinaryPairIndex] = Teff_DB2[
                             database2closestMatchinggminusrIndex]
                        database2correspondingrMag[database1BinaryPairIndex] = r_DB2[
                             database2closestMatchinggminusrIndex]
        endelse

        endelse

        ;PRINT, "database2closestMatchinguminusgIndex[", database1BinaryPairIndex, "] = ",
             database2closestMatchinguminusgIndex

        endfor

    differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair =
         database2correspondingrMag[0] - database2correspondingrMag[1]
    differenceBetweenMeasuredApparentMagnitudesForWDPair = r[0] - r[1]

        magnitudeDifferenceCondition = 0
        if(database2correspondingrMag[0] > database2correspondingrMag[1]) then begin
            ;then object0/A is theoretically less bright than object1/B at a fixed mass [due to
                 a lower temperature]
            ; differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair is positive
            if(differenceBetweenMeasuredApparentMagnitudesForWDPair lt
                 differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair) then begin
                ;then object0/A must be more massive (significantly less bright than their
                     theoretical absolute luminosities based upon their temperature) than
                     object1/B
                if (differenceBetweenMeasuredApparentMagnitudesForWDPair lt (
                     differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair-
                     rMagMinDifferenceForHighLowMassWDBinaryPair)) then begin
                    magnitudeDifferenceCondition = 1
                endif
            endif else begin
                if (differenceBetweenMeasuredApparentMagnitudesForWDPair gt (
                     differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair+
                     rMagMinDifferenceForHighLowMassWDBinaryPair)) then begin
                    magnitudeDifferenceCondition = 1
                endif
            endelse
        endif else begin
            ;then object0/A is theoretically more bright than object1/B at a fixed mass [due to
                 a higher temperature]
            ; differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair is negative
            if(differenceBetweenMeasuredApparentMagnitudesForWDPair gt
                 differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair) then begin
                if (differenceBetweenMeasuredApparentMagnitudesForWDPair gt (
                     differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair+
                     rMagMinDifferenceForHighLowMassWDBinaryPair)) then begin
                    magnitudeDifferenceCondition = 1
                endif
            endif else begin
                if (differenceBetweenMeasuredApparentMagnitudesForWDPair lt (
                     differenceBetweenTheoreticalAbsoluteMagnitudesForFixedMassWDPair-
                     rMagMinDifferenceForHighLowMassWDBinaryPair)) then begin
                    magnitudeDifferenceCondition = 1
                endif
            endelse
        endelse

        if (magnitudeDifferenceCondition eq 1) then begin
```

```
                ;'WD Binary Candidates (Blue Objects from SDSS DR7 Survey) -' +
                plotTitle = STRING(FORMAT='(I3)', database1starIndex) + ' r(A)=' + STRING(FORMAT='(
                    F0.2)', R_DB1_A[database1starIndex]) + ' r(B)=' + STRING(FORMAT='(F0.2)',
                    R_DB1_B[database1starIndex])
                    ;plotTitle = 'u-g vs g-r:' + STRING(FORMAT='(I3)', database1starIndex) + ', r(A)
                        =' + STRING(FORMAT='(F0.2)', R_DB1_A[database1starIndex]) + ', r(B)=' +
                        STRING(FORMAT='(F0.2)', R_DB1_B[database1starIndex])

                plot, gminusr, uminusg, title=plotTitle,xtitle='g-r',ytitle='u-g', xrange
                    =[-1.0,1.0],yrange=[-0.8,0.8],linestyle=5,psym=3, SYMSIZE=1

                starIDArrayForMatches[*, matchIndex] = database1starID[*, database1starIndex]
                starDistanceArrayForMatches[matchIndex] = database1Distances[database1starIndex]
                starDataArrayForMatches[*, matchIndex] = database1data[*, database1starIndex]

                matchIndex = matchIndex + 1L

            endif else begin

            endelse

        ;endif


        endif

    endif

    endif

endfor


PRINT, "rMagMinDifferenceForHighLowMassWDBinaryPair = ", rMagMinDifferenceForHighLowMassWDBinaryPair
PRINT, "uMinusgMaxDifferenceForHighLowMassWDBinaryPair = ",
    uMinusgMaxDifferenceForHighLowMassWDBinaryPair
PRINT, "uMinusgMaxDifferenceForHighLowMassWDBinaryPair = ",
    uMinusgMaxDifferenceForHighLowMassWDBinaryPair

PRINT, "Number of pairs found for given criteria = ", matchIndex    ;/2 is no longer required as
    BOCPLSDSSDR5QuasarCatalogConfirmedNotQuasarsPart1/2.txt is no longer assumed to be doubled up

totalNumberOfMatchesFound = matchIndex

    stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
    PRINT, stringmessage
    PRINT, maxDistanceForConsideredPairInArcSeconds

    stringmessage = "totalNumberOfMatchesFound = "
    PRINT, stringmessage
    PRINT, totalNumberOfMatchesFound


;NB sorting is not necessary as the input database1 is assumed already to be sorted wrt distance
    between pairs
starDistanceArrayForMatches = starDistanceArrayForMatches[0:(totalNumberOfMatchesFound-1)]     ;
    clip starDistanceArrayForMatches
starDistanceArrayForMatchesSortIndicies = sort(starDistanceArrayForMatches)
starIDArrayForMatchesSorted = starIDArrayForMatches[*, starDistanceArrayForMatchesSortIndicies]
;starDistanceArrayForMatchesSorted = starDistanceArrayForMatches[
    starDistanceArrayForMatchesSortIndicies]
starDataArrayForMatchesSorted = starDataArrayForMatches[*, starDistanceArrayForMatchesSortIndicies]
;starDistanceArrayForMatchesSorted = transpose(starDistanceArrayForMatchesSorted)

;print out coordinates in free format for Supercosmos manual proper motions check

numFields = (size(starDataArrayForMatchesSorted[0 ,*]))[1]

raDegA = starDataArrayForMatchesSorted[0 ,*]
decDegA = starDataArrayForMatchesSorted[1 ,*]
raDegB = starDataArrayForMatchesSorted[9 ,*]
decDegB = starDataArrayForMatchesSorted[10 ,*]

;for i=0L, numFields-1L, 1L do begin
    radec, raDegA,decDegA,ihrA,iminA,xsecA,idegA,imnA,xscA
    radec, raDegB,decDegB,ihrB,iminB,xsecB,idegB,imnB,xscB
    PRINT, "A=" + STRING(ihrA) + " " + STRING(iminA) + " " + STRING(xsecA) + " " + STRING(idegA) + "
        " + STRING(imnA) + " " + STRING(xscA)
    PRINT, "B=" + STRING(ihrB) + " " + STRING(iminB) + " " + STRING(xsecB) + " " + STRING(idegB) + "
        " + STRING(imnB) + " " + STRING(xscB)
;endfor

finalData = [starDataArrayForMatchesSorted]
finalDataID = [starIDArrayForMatchesSorted]

    outputFileName1='BOCPLBergeronTheoreticalComparisonPart1.txt'
    OPENW,1,outputFileName1
    PRINTF,1,finalData,FORMAT='(F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1
        X,F,1X,F,1X,F)'
    CLOSE,1
    outputFileName2='BOCPLBergeronTheoreticalComparisonPart2.txt'
```

```
    OPENW,1,outputFileName2
    PRINTF,1,finalDataID,FORMAT='(I18,1X,I18)'
    CLOSE,1

end
```

A.2.11 Calculate probability of chance alignment using distance limited volume method

This calculates the probability of chance alignment of each DD system using a distance limited volume method

probabilityOfChanceAlignmentUsingDistanceLimitedVolume-15Dec10a.ods

probabilityOfChanceAlignmentUsingDistanceLimitedVolume-15Dec10a.ods

	A	B	C	D	E	F	G	H	I
1									
2									
3	Mbol		logN	Density					
4		8	-4.6	=10^B4					
5		9	-4.2	=10^B5					
6		10	-3.95	=10^B6					
7		11	-3.8	=10^B7					
8		12	-3.5	=10^B8					
9		13	-3.2	=10^B9					
10				=SUM(C4:C9)					
11									
12						Total Sky coverage	42000		
13						SDSS sky coverage	7500		
14						SDSS percentage s	=G13/G12		
15									
16									
17		**Inputs**				**Inputs**			
18									
19			DD separation (arc s	component distance modulus				probability of c	probability of c
20						SDSS magnitu	20		
21			4.29	5.3		Absolute magr	7.49	=D42	=G42
22			4.29	4.9		maximum dista	=F20-F21		
23									
24									
25									
26									
27		**automatic**				**automatic**			
28						number of blue	56588		
29						number of blue	=F28/$G$14		
30						number of WD	=F29/2		
31		component 1 distance		=10^(0.2*D21+1)					
32		component 2 distance		=10^(0.2*D22+1)			=10^(0.2*F22+1)		
33									
34			*Method: distance limited volume*			*Method: volume (worst case)*			
35									
36			**equation number**			**equation number**			
37				1 =(PI()*($C$21/60/60)^2)/$G$12			1 =(PI()*($C$21/60/60)^2)/$G$12		
38				2 = 4/3 * PI() * (ABS($D$31 - $D$32))^3			2 = 4/3 * PI() * ($F$32)^3		
39				3 =D37*D38			3 =G37*G38		
40				4 =$C$10			4 =$C$10		
41			4b (?)	=D40*D39		4b	=G40*G39		
42				5 = ((D41^2) / (FACT(2))) * EXP(-D41)			5 = ((G41^2) / (FACT(2))) * EXP(-G41)		Monte carlo st
43									
44									

A.3 Proper Motions Code and Data (Target Selection Part II)

A.3.1 Proper Motion Calculation Procedure

The full proper motion calculation procedure is outlined here.

propermotions calculations.txt

```
//NB changes;
error columnns have been added
the formal SDSS gain (from SDSS plates data on internet) is used during object detection(set to 1.0
     in POSSI object detection)
the same mexhat mexhat_3.0_9x9 kernel is used for both POSSI and SDSS object detection.
a slightly greater detection radius is used for small database generation [4 arc min]
    NB DD3/12 use a 600 pixel radius for ~4arcmin extraction calculations
2.5sigma pmx-t cull
pmx-t.f bug correction

//do: Proper motion measurement instructions for DD systems (using SDSS r+b and POSSI/POSSII r+b
     data);

repeat the following for each DD system and for each colour band (ie, for POSSII blue and SDSS g,
     POSSII r and SDSS r, POSSI blue and SDSS g, and POSSI r and SDSS r);

0.
extract .tar/.gz image files to .fits


1.

? go to AAO (with a working version of gaia)
ssh -X rbaxter@aaolxa
source /star/etc/login
source /star/etc/cshrc
gaia

OR

    add the following to /home/rich/star/gaia/gaia.sh;
        if test "$EXTRACTOR_DIR" = ""; then
           EXTRACTOR_DIR="$HOME/star/bin/extractor"
           export EXTRACTOR_DIR
        fi
    /usr/bin/gaia

2.
open up SDSS image file of scene containing DD system
Run Image Analysis - object Detection;
    set catalouge name to something local, eg /home/aaossz/rbaxter/working/trialgaiaonserver/
         propermotionconfirmation/DD01/SDSS/catalog/rbb_sdss_g_werrors_cat
    Detector - set the detector gain to the formal SDSS values (from SDSS plates data on internet)
    Catalouge - select X_WORLD, Y_WORLD, X_IMAGE, Y_IMAGE, ERRX2_IMAGE, ERRY2_IMAGE, ERRXY_IMAGE,
         ERRX2_WORLD, ERRY2_WORLD, ERRXY_WORLD
    Set Detection Filter to mexhat_3.0_9x9 {OLD: gauss_3.0_7x7} for SDSS images, set contrast
         parameter to 0.005 [NO; or 0.00], set detection threshold and analysis thresholds to 1.5 [
         NO; or 2.0]
    set object size 5
    Detect Objects
verify correct objects have been detected
in catalouge box;
    set central position to the coordinates to either component of the DD system (see SDSS object
         table)
    set max radius to >4.0, eg 8.0 {OLD: ~3.0 [2.0 -> 3.0 depending upon number of objects around,
         such that ~20 objects are selected]}
    Save as ascii file list of 10 objects, eg /home/aaossz/rbaxter/working/trialgaiaonserver/
         propermotionconfirmation/DD01/SDSS/catalog/rbb_sdss_g_werrors_cat_small
    including their ra+dec coord (col 2+3), and their x+y coord (col 4+5)
verify files saved include ra+dec coord (col 2+3), and their x+y coord (col 4+5)

open up a POSSI/POSSII image file of scene containing DD system
Run Image Analysis - object Detection;
    set catalouge name to something local, eg /home/aaossz/rbaxter/working/trialgaiaonserver/
         propermotionconfirmation/DD01/SUPERCosmos-POSS-I/catalog/rbb_poss1_b_werrors_cat
    Detector - set the detector gain to 1.0 for POSSI object detection
    Catalouge - select X_WORLD, Y_WORLD, X_IMAGE, Y_IMAGE, ERRX2_IMAGE, ERRY2_IMAGE, ERRXY_IMAGE,
         ERRX2_WORLD, ERRY2_WORLD, ERRXY_WORLD
    Set Detection Filter to mexhat_3.0_9x9  {OLD: gauss_3.0_7x7} for POSS1 images
    set object size 5
    Run Image Analysis - object Detection
    verify parameters as above
    Detect Objects
verify correct objects have been detected
```

```
in catalouge box;
    verify central position is set to the coordinates to either component of the DD system (see SDSS
        object table)
    set max radius to 4.0 [or 5.0 if less than 20 eg DD10?] {OLD: ~4.0, ~3.0 [2.0 -> 3.0 depending
        upon number of objects around, such that ~20 objects are selected]}
    Save as ascii file list of 10 objects, eg /home/aaossz/rbaxter/working/trialgaiaonserver/
        propermotionconfirmation/DD01/SUPERCosmos-POSS-I/catalog/rbb_poss1_b_werrors_cat_small
verify files saved include ra+dec coord (col 2+3), and their x+y coord (col 4+5)

B. topcat
cd '/home/rich/utils/extrautilscentos5.3x64/21 topcat'/
java -jar topcat-full.jar
i. For DD-03 and DD-12 only - whose world coordinates could not be established by GAIA/extractor;
    0.
        download revised SDSS FITS catalog from SDSS website for a given photometric frame; eg drObj
            -002328-5-40-0113[-revised].fit
        open in GAIA, and save as ascii, DD-03catalogDownloadFits-revised.ASC
        open in nedit
        preferences - tab stops - 20
        trim off extra columns (shift ctrl - click and hold down) [NB there should only be 117
            columns]
        Save as DD-03catalogDownloadFits-revised2.ASC
    1. perform a pair match [2d-cartesian, not sky, error=6.0/7.0 respectively] of
        rbb_sdss_g_werrors_cat_small.ASC and DD-03/12catalogDownloadFits-revised2.ASC using their
        fits image pixel xy coordinates columns [col2+col3, col23+col21 respectively]
        this will determine the RA Dec coordinates of rbb_sdss_g_werrors_cat_small.ASC
        save this matched table as rbb_sdss_g_werrors_cat_small-withCoordinates-6pixMatch.ASC
    2. perform a pair match [2d-cartesian, not sky, error=1.0 respectively] of
        rbb_poss1_b_werrors_cat_small.ASC and rbb_poss1_b_werrors_cat.ASC using their fits image
        pixel xy coordinates columns [col4+5, col4+5 respectively]
        this will determine the RA Dec coordinates of rbb_poss1_b_werrors_cat_small.ASC in formats
            which match rbb_sdss_g_werrors_cat_small-withCoordinates-6pixMatch.ASC [ie, RA in hours
            , Dec in degrees decimal]
        save this matched table as rbb_poss1_b_werrors_cat_small-
            withCoordinatesInSDSSdownloadedCatalogFormat.ASC
    3. perform a pair match [sky, error=3.0arcsec] comparison of rbb_sdss_g_werrors_cat_small-
        withCoordinates-6pixMatch.ASC and rbb_poss1_b_werrors_cat_small-
        withCoordinatesInSDSSdownloadedCatalogFormat.ASC using their hours RA + decimal degrees Dec
         columns [col104+col105, col2_1+col3_2 respectively] {but note must set all matches to "
        degrees"; this is required for some reason, regardless of the fact the RA columns are in
        hours not degrees...}
        this will map the fits image xy pixels of the SDSS and POSSI data together
        save this matched table as POSSIversusSDSS.dat
ii. Or for all other DD sytems;
    Load Table - select ascii - select rbb_sdss_g_werrors_cat_small.ASC
    Load Table - select ascii - select rbb_poss1_b_werrors_cat_small.ASC
        OR [for DD-06 and DD-11];
        Load Table - select ascii - select rbb_sdss_i_werrors_cat_small.ASC {NB I actually used
            small for these calculations, in anticipation of reverting from 2011C medium/8as to
            2011B small/4as for all DD system proper motion calcualations}
        Load Table - select ascii - select rbb_int_i_werrors_cat_small.ASC {NB I actually used small
             for these calculations, in anticipation of reverting from 2011C medium/8as to 2011B
            small/4as for all DD system proper motion calcualations}
    Joins - Pair Match [match tables]
        max error 3.0 arc seconds
        type/algorithm = sky
        1.
        Col2    degrees
        Col3    degrees
        rbb_sdss_cat_small.ASC          [OR int]
        2.
        Col2    degrees
        Col3    degrees
        rbb_poss1_b_cat_small.ASC       [OR sdss]

        select 'Best Match only'
        Select join type '1 and 2'
        Go
            [should get >= ~30 matches]
    Select 'Match(1,2)', File - save table -
        create output ascii file called POSSIbversusSDSSg.dat   (POSSIb/r/versusSDSSg/r.dat)    [OR
            SDSSiversusINTi.dat]
    This matches/aligns the ra/dec coordinates of the matching objects, now use x,y (not ra,dec)
        coordinates from the matched data.

C.
Trim out all data except x/y columns from POSSIbversusSDSSg.dat, and create POSSIbversusSDSSgA.dat
remove DD data (2 rows) and create POSSIbversusSDSSgB.dat
edit pmx-t.f, and change the names of the two input files (POSSIbversusSDSSgB.dat for global
    transformation function fit [average RMS], and POSSIbversusSDSSgA.dat for individual residual
    fitting)

gfortran -c pmx-t.f
gfortran pmx-t.o -L$HOME/star/lib -lsla -o nige
setenv LD_LIBRARY_PATH $HOME/star/lib
./nige
    OR;
    bash
    export LD_LIBRARY_PATH=$HOME/star/lib
```

```
This prints out a list of x+y deviations between expected positions (without any real proper motions
    , just noise) and observed positions - highlighting therefore the large and similar direction
    of the DD component proper motions.
```

A.3.2 Proper Motion Calculations in Pixels

This code has been used during target selection to confirm common proper motions between components. It requires ASCII files to be previously calculated using GAIA and Topcat, indicating CCD frame relative positions of double degenerate candidate WDs along with a selection of nearby stars.

pmx-t.f is written by PDD (and updated by RBB) - it is used to calculate proper motions in pixels of one or more stars (eg a DD system) based upon astrometric reference field star list, where for each star, pixel positions are given for two photometric frames (eg SDSS and POSS-I).

pmx-t.f

```
      program pms_core
c
c     Quick check for Fraser on 1st/2nd epoch CCD data for Hyades BD
c
      implicit none
      integer nmax,itype,oldpt,newpt,i,ios,j,objno
      parameter (nmax=1000,itype=6)
      double precision xye(2,nmax),xym(2,nmax),coeffs(6),xyp(2,nmax)
      double precision xyee(2,nmax),xyme(2,nmax),fk5(2,nmax)
      double precision x1av,y1av,x2av,y2av,xrms,yrms,rrms,xt,yt
      double precision xz,yz,xs,ys,perp,orient,pi,tol,x1,y1,x2,y2
      double precision ep2xe,ep2ye

c     RBB added output without rejected stars
      double precision rejxy1(2,nmax)
      integer rejindex,rejfound,rejmax
      double precision x1Temp,y1Temp

      parameter (tol=2.5)
      real values(nmax),median,fk51,fk52,x1e,x2e,y2e,y1e

c     RBB added output without rejected stars
      rejindex = 0
      rejfound = 0
      rejmax = 0

      pi=4.0d0*atan(1.0d0)

      open (unit=1,file='POSSIbversusSDSSgB.dat',status='old')
      x1av=0.0d0
      x2av=0.0d0
      y1av=0.0d0
      y2av=0.0d0
      ios=0
      i=0
      do while (ios.eq.0)
       i=i+1
       if (i.gt.nmax) then
        write (*,*) 'WARNING: too many input stars for array size'
        ios=-1
        else
        read (1,*,iostat=ios) xye(1,i),xye(2,i),xym(1,i),xym(2,i)
          x1av=x1av+xye(1,i)
          x2av=x2av+xym(1,i)
          y1av=y1av+xye(2,i)
          y2av=y2av+xym(2,i)
       endif
      enddo
      newpt=i-1
      write (*,*) 'No. of xy pairs read in: ',newpt


c  Subtract of the means (makes least squares more robust):
      x1av=x1av/dble(newpt)
      x2av=x2av/dble(newpt)
      y1av=y1av/dble(newpt)
      y2av=y2av/dble(newpt)
      do i=1,newpt
       xye(1,i)=xye(1,i)-x1av
       xye(2,i)=xye(2,i)-y1av
       xym(1,i)=xym(1,i)-x2av
```

```
        xym(2,i)=xym(2,i)-y2av

        write (*,*) xye(1,i),xye(2,i),xym(1,i),xym(2,i)
       enddo
c   Initial fit:
       call sla_fitxy(itype,newpt,xye,xym,coeffs,j)
       call sla_pxy(newpt,xye,xym,coeffs,xyp,xrms,yrms,rrms)
c   Robust RMS:
       do i=1,newpt
        values(i)=abs(real(xye(1,i)-xyp(1,i)))
       enddo
       call sort(newpt,values)
       xrms=dble(1.48*median(values,newpt))
       do i=1,newpt
        values(i)=abs(real(xye(2,i)-xyp(2,i)))
       enddo
       call sort(newpt,values)
       yrms=dble(1.48*median(values,newpt))

       write (*,*) 'Initial RMS (xy in pix):',xrms,yrms

c   Iterate to chuck out the crap:
       oldpt=newpt+1
       do while (newpt.lt.oldpt)
        oldpt=newpt
        newpt=0
        do i=1,oldpt
         if (abs(xye(1,i)-xyp(1,i)).le.tol*xrms.and.
     :abs(xye(2,i)-xyp(2,i)).le.tol*yrms) then
           newpt=newpt+1
           xye(1,newpt)=xye(1,i)
           xye(2,newpt)=xye(2,i)
           xym(1,newpt)=xym(1,i)
           xym(2,newpt)=xym(2,i)
           else
              write (9,*) xye(1,i)+x1av,xye(2,i)+y1av
c     RBB added output without rejected stars
              rejxy1(1, rejindex) = xye(1,i)+x1av
              rejxy1(2, rejindex) = xye(2,i)+y1av
          rejindex = rejindex + 1
          endif
        enddo
        call sla_fitxy(itype,newpt,xye,xym,coeffs,j)
        call sla_pxy(newpt,xye,xym,coeffs,xyp,xrms,yrms,rrms)
        do i=1,newpt
         values(i)=abs(real(xye(1,i)-xyp(1,i)))
        enddo
        call sort(newpt,values)
        xrms=dble(1.48*median(values,newpt))
        do i=1,newpt
         values(i)=abs(real(xye(2,i)-xyp(2,i)))
        enddo
        call sort(newpt,values)
        yrms=dble(1.48*median(values,newpt))
       enddo
       rejmax = rejindex
c   Write out the details:
       write (*,*) 'Final results of fit:'
       write (*,*) 'Number of objects used: ',newpt
       call sla_dcmpf(coeffs,xz,yz,xs,ys,perp,orient)
       write (*,*) 'Zeropoints:',xz,yz
       write (*,*) 'Scales    :',xs,ys
       write (*,*) 'Orientation (deg):',orient*180.0d0/pi
       write (*,*) 'Shear of axes (deg):',perp*180.0d0/pi
       write (*,*) 'Final RMS (xy, pix):',xrms,yrms

c   Apply the fit and output displacements:
       rewind(1)
c
       close(unit=1)
       open (unit=4,file='xy.out',status='unknown')
       open (unit=3,file='xy2xy.log',status='unknown')
       open (unit=8,file='xyWithoutRejected.out',status='unknown')
       open (unit=7,file='xy2xyWithoutRejected.log',status='unknown')
       open (unit=10,file='POSSIbversusSDSSgA.dat',status='old')
       ios=0
       do while (ios.eq.0)
c        read (1,*,iostat=ios) x1,x1e,
c     *  y1,y1e,x2,x2e,y2,y2e
         read (10,*,iostat=ios) x1,y1,x2,y2
         x2=x2-x2av
         y2=y2-y2av
         x1=x1-x1av
         y1=y1-y1av
         call sla_xy2xy(x2,y2,coeffs,xt,yt)
c
c        next bit gives real direction of motion in x and y
c
         if (ios.eq.0) then

c     RBB added output without rejected stars
         x1Temp=x1+x1av
```

```
          y1Temp=y1+y1av
      rejindex = 0
      rejfound = 0
      do while (rejindex.lt.rejmax)
           if (rejxy1(1, rejindex).eq.x1Temp .AND.
       :rejxy1(2, rejindex).eq.y1Temp) then
        rejfound = 1
c       write (*,*) 'rejfound = 1'
       endif
       rejindex = rejindex + 1
      enddo



          write (3,*) xt-x1,yt-y1
c       RBB added output without rejected stars
      if(rejfound.eq.0) then
       write (7,*) xt-x1,yt-y1,xrms,yrms
      endif

          x1=x1+x1av
          y1=y1+y1av
          xt=xt+x1av
          yt=yt+y1av

          ep2xe=sqrt(xrms**2.0 + x2e**2.0)
          ep2ye=sqrt(yrms**2.0 + y2e**2.0)

          write (4,200) x1,x1e,y1,y1e,xt,ep2xe,yt,ep2ye
c       RBB added output without rejected stars
      if(rejfound.eq.0) then
       write (8,200) x1,x1e,y1,y1e,xt,ep2xe,yt,ep2ye
      endif
          endif
        enddo

        close (1)
c        close (2)
        close (3)
        close (4)
  200   format (f9.3,f9.3,f9.3,f9.3,f9.3,f9.3,f9.3,f9.3)
        end

        real function median(array,n)
        implicit none
        integer n
        real array(n)

        if (n.eq.0) then
         median=0.0
        else if (n.eq.1) then
         median=array(1)
        else if (mod(n,2).eq.0) then
         median=(array(n/2)+array(n/2 + 1))/2.0
        else
         median=array(n/2+1)
        endif

        return
        end

        SUBROUTINE SORT(N,RA)
        DIMENSION RA(N)
        L=N/2+1
        IR=N
10      CONTINUE
          IF(L.GT.1)THEN
            L=L-1
            RRA=RA(L)
          ELSE
            RRA=RA(IR)
            RA(IR)=RA(1)
            IR=IR-1
            IF(IR.EQ.1)THEN
              RA(1)=RRA
              RETURN
            ENDIF
          ENDIF
          I=L
          J=L+L
20        IF(J.LE.IR)THEN
            IF(J.LT.IR)THEN
              IF(RA(J).LT.RA(J+1))J=J+1
            ENDIF
            IF(RRA.LT.RA(J))THEN
              RA(I)=RA(J)
               I=J
               J=J+J
            ELSE
               J=IR+1
            ENDIF
          GO TO 20
```

```
   ENDIF
   RA(I)=RRA
GO TO 10
END
```

Results

SDSSDDproperMotionMeasurements-pmx-t-xy2xylog-28May11a.zip

Graphs indicating the relative displacement of WD double degenerate candidates and nearby stars over an epoch.

POSS-I[blue] : SDSS epoch;

- xy2xylogDD01.ods
- xy2xylogDD02.ods
- xy2xylogDD03.ods
- xy2xylogDD04.ods
- xy2xylogDD05.ods
- xy2xylogDD06.ods
- xy2xylogDD07.ods
- xy2xylogDD08.ods
- xy2xylogDD09.ods
- xy2xylogDD10.ods
- xy2xylogDD11.ods
- xy2xylogDD12.ods

SDSS : INT La Palma 12 February 2011 epoch;

- xy2xylogDD06.ods
- xy2xylogDD11.ods

A.3.3 Proper Motion Calculations in RA/Dec (Calibration of pixel values)

DDproperMotionWCScalibration-01June11a-2011B.ods

"M" = "Measure" = yellow
Measurement Difference in Pixels
10

INT only: [DD01 = INT, 2 = SDSS
normal: 1 = SDSS, 2 = POSS-I

Proper motion SDSS RA/Dec:X/Y Calibration

	x1	x2 M X	x diff M X	x2 M Y	x diff M Y	y1	y2 M X	y diff M X	y2 M Y	y diff M Y	ra1	ra2 M X	ra diff M X	ra2 M Y	ra diff M Y	de1	dec2 M X	dec diff M X	dec2 M Y	dec diff M Y	dec diff M Y C
DD-01A	725.4	=C13+E13	=$E$5	=C13+G13	0.000	635.5	=H13+J13	0.000	=H13+L13	=$E$5	12.742	12.754	=N13-M13	13.02	=P13-M13	0.940	4.890	=S13-R13	0.78	=U13-R13	=V13
DD-01B	=C13	=C14+E14	=$E$5	=C14+G14	0.000	=H13	=H14+J14	0.000	=H14+L14	=$E$5	=M13	12.754	=N14-M14	13.02	=P14-M14	=R13	4.890	=S14-R14	0.78	=U14-R14	=V14
DD-02A	1861.1	=C15+E15	=$E$5	=C15+G15	0.000	1089.5	=H15+J15	0.000	=H15+L15	=$E$5	46.492	46.741	=N15-M15	46.84	=P15-M15	1.780	5.010	=S15-R15	59.49	=U15-R15	=V15-60
DD-02B	=C15	=C16+E16	=$E$5	=C16+G16	0.000	=H15	=H16+J16	0.000	=H16+L16	=$E$5	=M15	46.741	=N16-M16	46.84	=P16-M16	=R15	5.010	=S16-R16	59.49	=U16-R16	=V16-60
DD-03A	1360.7	=C17+E17	=$E$5	=C17+G17	0.000	854.8	=H17+J17	0.000	=H17+L17	=$E$5	55.909	56.171	=N17-M17	56.079	=P17-M17	38.230	40.400	=S17-R17	34.900	=U17-R17	=V17
DD-03B	=C17	=C18+E18	=$E$5	=C18+G18	0.000	=H17	=H18+J18	0.000	=H18+L18	=$E$5	=M17	56.171	=N18-M18	56.079	=P18-M18	=R17	40.400	=S18-R18	34.900	=U18-R18	=V18
DD-04A	1590.5	=C19+E19	=$E$5	=C19+G19	0.000	1415.6	=H19+J19	0.000	=H19+L19	=$E$5	32.733	32.708	=N19-M19	33.004	=P19-M19	44.240	48.190	=S19-R19	44.580	=U19-R19	=V19
DD-04B	=C19	=C20+E20	=$E$5	=C20+G20	0.000	=H19	=H20+J20	0.000	=H20+L20	=$E$5	=M19	32.708	=N20-M20	33.004	=P20-M20	=R19	48.190	=S20-R20	44.580	=U20-R20	=V20
DD-05A	961.4	=C21+E21	=$E$5	=C21+G21	0.000	787.5	=H21+J21	0.000	=H21+L21	=$E$5	1.647	1.586	=N21-M21	1.924	=P21-M21	31.210	35.080	=S21-R21	32.030	=U21-R21	=V21
DD-05B	=C21	=C22+E22	=$E$5	=C22+G22	0.000	=H21	=H22+J22	0.000	=H22+L22	=$E$5	=M21	1.586	=N22-M22	1.924	=P22-M22	=R21	35.080	=S22-R22	32.030	=U22-R22	=V22
DD-06A	1015	=C23+E23	=$E$5	=C23+G23	0.000	2008	=H23+J23	0.000	=H23+L23	=$E$5	31.529	31.526	=N23-M23	31.297	=P23-M23	28.740	25.430	=S23-R23	28.790	=U23-R23	=V23
DD-06B	=C23	=C24+E24	=$E$5	=C24+G24	0.000	=H23	=H24+J24	0.000	=H24+L24	=$E$5	=M23	31.526	=N24-M24	31.297	=P24-M24	=R23	25.430	=S24-R24	28.790	=U24-R24	=V24
DD-07A	222.4	=C25+E25	=$E$5	=C25+G25	0.000	451	=H25+J25	0.000	=H25+L25	=$E$5	36.872	36.875	=N25-M25	37.138	=P25-M25	37.030	33.090	=S25-R25	37.050	=U25-R25	=V25
DD-07B	=C25	=C26+E26	=$E$5	=C26+G26	0.000	=H25	=H26+J26	0.000	=H26+L26	=$E$5	=M25	36.875	=N26-M26	37.138	=P26-M26	=R25	33.090	=S26-R26	37.050	=U26-R26	=V26
DD-08A	140.4	=C27+E27	=$E$5	=C27+G27	0.000	813.7	=H27+J27	0.000	=H27+L27	=$E$5	46.885	46.829	=N27-M27	47.150	=P27-M27	34.320	38.200	=S27-R27	35.160	=U27-R27	=V27
DD-08B	=C27	=C28+E28	=$E$5	=C28+G28	0.000	=H27	=H28+J28	0.000	=H28+L28	=$E$5	=M27	46.829	=N28-M28	47.150	=P28-M28	=R27	38.200	=S28-R28	35.160	=U28-R28	=V28
DD-09A	564.9	=C29+E29	=$E$5	=C29+G29	0.000	219.8	=H29+J29	0.000	=H29+L29	=$E$5	37.834	37.828	=N29-M29	38.106	=P29-M29	13.790	17.760	=S29-R29	13.890	=U29-R29	=V29
DD-09B	=C29	=C30+E30	=$E$5	=C30+G30	0.000	=H29	=H30+J30	0.000	=H30+L30	=$E$5	=M29	37.828	=N30-M30	38.106	=P30-M30	=R29	17.760	=S30-R30	13.890	=U30-R30	=V30
DD-10A	1625.8	=C31+E31	=$E$5	=C31+G31	0.000	1173.5	=H31+J31	0.000	=H31+L31	=$E$5	32.145	32.167	=N31-M31	32.427	=P31-M31	39.440	43.410	=S31-R31	39.150	=U31-R31	=V31
DD-10B	=C31	=C32+E32	=$E$5	=C32+G32	0.000	=H31	=H32+J32	0.000	=H32+L32	=$E$5	=M31	32.167	=N32-M32	32.427	=P32-M32	=R31	43.410	=S32-R32	39.150	=U32-R32	=V32
DD-11A	1004	=C33+E33	=$E$5	=C33+G33	0.000	2065	=H33+J33	0.000	=H33+L33	=$E$5	14.287	14.284	=N33-M33	14.056	=P33-M33	50.830	47.540	=S33-R33	50.880	=U33-R33	=V33
DD-11B	=C33	=C34+E34	=$E$5	=C34+G34	0.000	=H33	=H34+J34	0.000	=H34+L34	=$E$5	=M33	14.284	=N34-M34	14.056	=P34-M34	=R33	47.540	=S34-R34	50.880	=U34-R34	=V34
DD-12A	616	=C35+E35	=$E$5	=C35+G35	0.000	1127.3	=H35+J35	0.000	=H35+L35	=$E$5	36.305	36.328	=N35-M35	36.571	=P35-M35	7.950	4.020	=S35-R35	8.300	=U35-R35	=V35
DD-12B	=C35	=C36+E36	=$E$5	=C36+G36	0.000	=H35	=H36+J36	0.000	=H36+L36	=$E$5	=M35	36.328	=N36-M36	36.571	=P36-M36	=R35	4.020	=S36-R36	8.300	=U36-R36	=V36

	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN
1																	
2																	
3																	
4																	
5																	
6																	
7																	
8																	
9																	
10										Error Calculations							
11	Proper motion		Calculations using GAIA object detection for SDSS and POSS-I coordinates, with			reference star correction				Reference field star RMS Error Calculations							
12	x diff	y diff	RA x diff	RA y diff	Dec x diff	Dec y diff	RA diff	Dec diff		x reference fiely reference fie		RA x reference field star	RA y reference field star	Dec x reference field sta	Dec y reference field s	RA reference f	Dec reference fiel
13	=CN13	=CO13	=X13*$O13/$E$5	=Y13*$Q13/$E$5	=X13*$T13/$E$5	=Y13*$W13/$E$5	=Z13+AA13	=AB13+AC13		0.6497	0.8008	=AG13*$O13/$E$5	=AH13*$Q13/$E$5	=AG13*$T13/$E$5	=AH13*$W13/$E$5	=AI13+AJ13	=AK13+AL13
14	=CN14	=CO14	=X14*$O14/$E$5	=Y14*$Q14/$E$5	=X14*$T14/$E$5	=Y14*$W14/$E$5	=Z14+AA14	=AB14+AC14		0.6497	0.8008	=AG14*$O14/$E$5	=AH14*$Q14/$E$5	=AG14*$T14/$E$5	=AH14*$W14/$E$5	=AI14+AJ14	=AK14+AL14
15	=CN15	=CO15	=X15*$O15/$E$5	=Y15*$Q15/$E$5	=X15*$T15/$E$5	=Y15*$W15/$E$5	=Z15+AA15	=AB15+AC15		1.3820	1.7133	=AG15*$O15/$E$5	=AH15*$Q15/$E$5	=AG15*$T15/$E$5	=AH15*$W15/$E$5	=AI15+AJ15	=AK15+AL15
16	=CN16	=CO16	=X16*$O16/$E$5	=Y16*$Q16/$E$5	=X16*$T16/$E$5	=Y16*$W16/$E$5	=Z16+AA16	=AB16+AC16		1.3820	1.7133	=AG16*$O16/$E$5	=AH16*$Q16/$E$5	=AG16*$T16/$E$5	=AH16*$W16/$E$5	=AI16+AJ16	=AK16+AL16
17	=CN17	=CO17	=X17*$O17/$E$5	=Y17*$Q17/$E$5	=X17*$T17/$E$5	=Y17*$W17/$E$5	=Z17+AA17	=AB17+AC17		0.7837	0.6989	=AG17*$O17/$E$5	=AH17*$Q17/$E$5	=AG17*$T17/$E$5	=AH17*$W17/$E$5	=AI17+AJ17	=AK17+AL17
18	=CN18	=CO18	=X18*$O18/$E$5	=Y18*$Q18/$E$5	=X18*$T18/$E$5	=Y18*$W18/$E$5	=Z18+AA18	=AB18+AC18		0.7837	0.6989	=AG18*$O18/$E$5	=AH18*$Q18/$E$5	=AG18*$T18/$E$5	=AH18*$W18/$E$5	=AI18+AJ18	=AK18+AL18
19	=CN19	=CO19	=X19*$O19/$E$5	=Y19*$Q19/$E$5	=X19*$T19/$E$5	=Y19*$W19/$E$5	=Z19+AA19	=AB19+AC19		0.4703	0.4833	=AG19*$O19/$E$5	=AH19*$Q19/$E$5	=AG19*$T19/$E$5	=AH19*$W19/$E$5	=AI19+AJ19	=AK19+AL19
20	=CN20	=CO20	=X20*$O20/$E$5	=Y20*$Q20/$E$5	=X20*$T20/$E$5	=Y20*$W20/$E$5	=Z20+AA20	=AB20+AC20		0.4703	0.4833	=AG20*$O20/$E$5	=AH20*$Q20/$E$5	=AG20*$T20/$E$5	=AH20*$W20/$E$5	=AI20+AJ20	=AK20+AL20
21	=CN21	=CO21	=X21*$O21/$E$5	=Y21*$Q21/$E$5	=X21*$T21/$E$5	=Y21*$W21/$E$5	=Z21+AA21	=AB21+AC21		0.5701	0.5608	=AG21*$O21/$E$5	=AH21*$Q21/$E$5	=AG21*$T21/$E$5	=AH21*$W21/$E$5	=AI21+AJ21	=AK21+AL21
22	=CN22	=CO22	=X22*$O22/$E$5	=Y22*$Q22/$E$5	=X22*$T22/$E$5	=Y22*$W22/$E$5	=Z22+AA22	=AB22+AC22		0.5701	0.5608	=AG22*$O22/$E$5	=AH22*$Q22/$E$5	=AG22*$T22/$E$5	=AH22*$W22/$E$5	=AI22+AJ22	=AK22+AL22
23	=CN23	=CO23	=X23*$O23/$E$5	=Y23*$Q23/$E$5	=X23*$T23/$E$5	=Y23*$W23/$E$5	=Z23+AA23	=AB23+AC23		0.2059	0.1171	=AG23*$O23/$E$5	=AH23*$Q23/$E$5	=AG23*$T23/$E$5	=AH23*$W23/$E$5	=AI23+AJ23	=AK23+AL23
24	=CN24	=CO24	=X24*$O24/$E$5	=Y24*$Q24/$E$5	=X24*$T24/$E$5	=Y24*$W24/$E$5	=Z24+AA24	=AB24+AC24		0.2059	0.1171	=AG24*$O24/$E$5	=AH24*$Q24/$E$5	=AG24*$T24/$E$5	=AH24*$W24/$E$5	=AI24+AJ24	=AK24+AL24
25	=CN25	=CO25	=X25*$O25/$E$5	=Y25*$Q25/$E$5	=X25*$T25/$E$5	=Y25*$W25/$E$5	=Z25+AA25	=AB25+AC25		0.8335	1.0651	=AG25*$O25/$E$5	=AH25*$Q25/$E$5	=AG25*$T25/$E$5	=AH25*$W25/$E$5	=AI25+AJ25	=AK25+AL25
26	=CN26	=CO26	=X26*$O26/$E$5	=Y26*$Q26/$E$5	=X26*$T26/$E$5	=Y26*$W26/$E$5	=Z26+AA26	=AB26+AC26		0.8335	1.0651	=AG26*$O26/$E$5	=AH26*$Q26/$E$5	=AG26*$T26/$E$5	=AH26*$W26/$E$5	=AI26+AJ26	=AK26+AL26
27	=CN27	=CO27	=X27*$O27/$E$5	=Y27*$Q27/$E$5	=X27*$T27/$E$5	=Y27*$W27/$E$5	=Z27+AA27	=AB27+AC27		0.6123	0.1557	=AG27*$O27/$E$5	=AH27*$Q27/$E$5	=AG27*$T27/$E$5	=AH27*$W27/$E$5	=AI27+AJ27	=AK27+AL27
28	=CN28	=CO28	=X28*$O28/$E$5	=Y28*$Q28/$E$5	=X28*$T28/$E$5	=Y28*$W28/$E$5	=Z28+AA28	=AB28+AC28		0.6123	0.1557	=AG28*$O28/$E$5	=AH28*$Q28/$E$5	=AG28*$T28/$E$5	=AH28*$W28/$E$5	=AI28+AJ28	=AK28+AL28
29	=CN29	=CO29	=X29*$O29/$E$5	=Y29*$Q29/$E$5	=X29*$T29/$E$5	=Y29*$W29/$E$5	=Z29+AA29	=AB29+AC29		0.3208	0.6385	=AG29*$O29/$E$5	=AH29*$Q29/$E$5	=AG29*$T29/$E$5	=AH29*$W29/$E$5	=AI29+AJ29	=AK29+AL29
30	=CN30	=CO30	=X30*$O30/$E$5	=Y30*$Q30/$E$5	=X30*$T30/$E$5	=Y30*$W30/$E$5	=Z30+AA30	=AB30+AC30		0.3208	0.6385	=AG30*$O30/$E$5	=AH30*$Q30/$E$5	=AG30*$T30/$E$5	=AH30*$W30/$E$5	=AI30+AJ30	=AK30+AL30
31	=CN31	=CO31	=X31*$O31/$E$5	=Y31*$Q31/$E$5	=X31*$T31/$E$5	=Y31*$W31/$E$5	=Z31+AA31	=AB31+AC31		0.7398	0.5983	=AG31*$O31/$E$5	=AH31*$Q31/$E$5	=AG31*$T31/$E$5	=AH31*$W31/$E$5	=AI31+AJ31	=AK31+AL31
32	=CN32	=CO32	=X32*$O32/$E$5	=Y32*$Q32/$E$5	=X32*$T32/$E$5	=Y32*$W32/$E$5	=Z32+AA32	=AB32+AC32		0.7398	0.5983	=AG32*$O32/$E$5	=AH32*$Q32/$E$5	=AG32*$T32/$E$5	=AH32*$W32/$E$5	=AI32+AJ32	=AK32+AL32
33	=CN33	=CO33	=X33*$O33/$E$5	=Y33*$Q33/$E$5	=X33*$T33/$E$5	=Y33*$W33/$E$5	=Z33+AA33	=AB33+AC33		0.2330	0.2149	=AG33*$O33/$E$5	=AH33*$Q33/$E$5	=AG33*$T33/$E$5	=AH33*$W33/$E$5	=AI33+AJ33	=AK33+AL33
34	=CN34	=CO34	=X34*$O34/$E$5	=Y34*$Q34/$E$5	=X34*$T34/$E$5	=Y34*$W34/$E$5	=Z34+AA34	=AB34+AC34		0.2330	0.2149	=AG34*$O34/$E$5	=AH34*$Q34/$E$5	=AG34*$T34/$E$5	=AH34*$W34/$E$5	=AI34+AJ34	=AK34+AL34
35	=CN35	=CO35	=X35*$O35/$E$5	=Y35*$Q35/$E$5	=X35*$T35/$E$5	=Y35*$W35/$E$5	=Z35+AA35	=AB35+AC35		0.5792	0.3895	=AG35*$O35/$E$5	=AH35*$Q35/$E$5	=AG35*$T35/$E$5	=AH35*$W35/$E$5	=AI35+AJ35	=AK35+AL35
36	=CN36	=CO36	=X36*$O36/$E$5	=Y36*$Q36/$E$5	=X36*$T36/$E$5	=Y36*$W36/$E$5	=Z36+AA36	=AB36+AC36		0.5792	0.3895	=AG36*$O36/$E$5	=AH36*$Q36/$E$5	=AG36*$T36/$E$5	=AH36*$W36/$E$5	=AI36+AJ36	=AK36+AL36
37																	
38																	
39																	
40																	
41																	
42																	
43																	

	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ	BA	BB
1														
2														
3														
4														
5														
6														
7							//do: update POSSIb GAIA object detection errors							
8														
9														
10														
11							only for DD03 and DD12 (without RA/Dec errors)							
12	RA reference field star scatter RMS e	Dec reference	RA reference field star scatter RM	Dec reference field star scatter F	RA reference fie	Dec reference fiel	x SDSS GAIA	y SDSS GAIA	x SDSS GAIA me	y SDSS GAIA mex	RA x reference field s	RA y reference field sta	Dec x reference field :	Dec y reference field s
13	=AM13*COS(RADIANS($CQ13))	=AN13	=(AO13/60/60/24) * (360*60*60)	=(AP13/60/60) * (60*60)	=AQ13/60/60	=AR13/60/60	0.0016	0.0017	=SQRT(AU13)	=SQRT(AV13)	=AW13*$O13/$E$5	=AX13*$Q13/$E$5	=AW13*$T13/$E$5	=AX13*$W13/$E$5
14	=AM14*COS(RADIANS($CQ14))	=AN14	=(AO14/60/60/24) * (360*60*60)	=(AP14/60/60) * (60*60)	=AQ14/60/60	=AR14/60/60	0.0004	0.0003	=SQRT(AU14)	=SQRT(AV14)	=AW14*$O14/$E$5	=AX14*$Q14/$E$5	=AW14*$T14/$E$5	=AX14*$W14/$E$5
15	=AM15*COS(RADIANS($CQ15))	=AN15	=(AO15/60/60/24) * (360*60*60)	=(AP15/60/60) * (60*60)	=AQ15/60/60	=AR15/60/60	0.0000	0.0000	=SQRT(AU15)	=SQRT(AV15)	=AW15*$O15/$E$5	=AX15*$Q15/$E$5	=AW15*$T15/$E$5	=AX15*$W15/$E$5
16	=AM16*COS(RADIANS($CQ16))	=AN16	=(AO16/60/60/24) * (360*60*60)	=(AP16/60/60) * (60*60)	=AQ16/60/60	=AR16/60/60	0.0001	0.0001	=SQRT(AU16)	=SQRT(AV16)	=AW16*$O16/$E$5	=AX16*$Q16/$E$5	=AW16*$T16/$E$5	=AX16*$W16/$E$5
17	=AM17*COS(RADIANS($CQ17))	=AN17	=(AO17/60/60/24) * (360*60*60)	=(AP17/60/60) * (60*60)	=AQ17/60/60	=AR17/60/60	0.0001	0.0002	=SQRT(AU17)	=SQRT(AV17)	=AW17*$O17/$E$5	=AX17*$Q17/$E$5	=AW17*$T17/$E$5	=AX17*$W17/$E$5
18	=AM18*COS(RADIANS($CQ18))	=AN18	=(AO18/60/60/24) * (360*60*60)	=(AP18/60/60) * (60*60)	=AQ18/60/60	=AR18/60/60	0.0003	0.0004	=SQRT(AU18)	=SQRT(AV18)	=AW18*$O18/$E$5	=AX18*$Q18/$E$5	=AW18*$T18/$E$5	=AX18*$W18/$E$5
19	=AM19*COS(RADIANS($CQ19))	=AN19	=(AO19/60/60/24) * (360*60*60)	=(AP19/60/60) * (60*60)	=AQ19/60/60	=AR19/60/60	0.0000	0.0000	=SQRT(AU19)	=SQRT(AV19)	=AW19*$O19/$E$5	=AX19*$Q19/$E$5	=AW19*$T19/$E$5	=AX19*$W19/$E$5
20	=AM20*COS(RADIANS($CQ20))	=AN20	=(AO20/60/60/24) * (360*60*60)	=(AP20/60/60) * (60*60)	=AQ20/60/60	=AR20/60/60	0.0002	0.0003	=SQRT(AU20)	=SQRT(AV20)	=AW20*$O20/$E$5	=AX20*$Q20/$E$5	=AW20*$T20/$E$5	=AX20*$W20/$E$5
21	=AM21*COS(RADIANS($CQ21))	=AN21	=(AO21/60/60/24) * (360*60*60)	=(AP21/60/60) * (60*60)	=AQ21/60/60	=AR21/60/60	0.0000	0.0000	=SQRT(AU21)	=SQRT(AV21)	=AW21*$O21/$E$5	=AX21*$Q21/$E$5	=AW21*$T21/$E$5	=AX21*$W21/$E$5
22	=AM22*COS(RADIANS($CQ22))	=AN22	=(AO22/60/60/24) * (360*60*60)	=(AP22/60/60) * (60*60)	=AQ22/60/60	=AR22/60/60	0.0000	0.0000	=SQRT(AU22)	=SQRT(AV22)	=AW22*$O22/$E$5	=AX22*$Q22/$E$5	=AW22*$T22/$E$5	=AX22*$W22/$E$5
23	=AM23*COS(RADIANS($CQ23))	=AN23	=(AO23/60/60/24) * (360*60*60)	=(AP23/60/60) * (60*60)	=AQ23/60/60	=AR23/60/60	0.0015	0.0033	=SQRT(AU23)	=SQRT(AV23)	=AW23*$O23/$E$5	=AX23*$Q23/$E$5	=AW23*$T23/$E$5	=AX23*$W23/$E$5
24	=AM24*COS(RADIANS($CQ24))	=AN24	=(AO24/60/60/24) * (360*60*60)	=(AP24/60/60) * (60*60)	=AQ24/60/60	=AR24/60/60	0.0040	0.0056	=SQRT(AU24)	=SQRT(AV24)	=AW24*$O24/$E$5	=AX24*$Q24/$E$5	=AW24*$T24/$E$5	=AX24*$W24/$E$5
25	=AM25*COS(RADIANS($CQ25))	=AN25	=(AO25/60/60/24) * (360*60*60)	=(AP25/60/60) * (60*60)	=AQ25/60/60	=AR25/60/60	0.0002	0.0002	=SQRT(AU25)	=SQRT(AV25)	=AW25*$O25/$E$5	=AX25*$Q25/$E$5	=AW25*$T25/$E$5	=AX25*$W25/$E$5
26	=AM26*COS(RADIANS($CQ26))	=AN26	=(AO26/60/60/24) * (360*60*60)	=(AP26/60/60) * (60*60)	=AQ26/60/60	=AR26/60/60	0.0000	0.0000	=SQRT(AU26)	=SQRT(AV26)	=AW26*$O26/$E$5	=AX26*$Q26/$E$5	=AW26*$T26/$E$5	=AX26*$W26/$E$5
27	=AM27*COS(RADIANS($CQ27))	=AN27	=(AO27/60/60/24) * (360*60*60)	=(AP27/60/60) * (60*60)	=AQ27/60/60	=AR27/60/60	0.0002	0.0003	=SQRT(AU27)	=SQRT(AV27)	=AW27*$O27/$E$5	=AX27*$Q27/$E$5	=AW27*$T27/$E$5	=AX27*$W27/$E$5
28	=AM28*COS(RADIANS($CQ28))	=AN28	=(AO28/60/60/24) * (360*60*60)	=(AP28/60/60) * (60*60)	=AQ28/60/60	=AR28/60/60	0.0002	0.0002	=SQRT(AU28)	=SQRT(AV28)	=AW28*$O28/$E$5	=AX28*$Q28/$E$5	=AW28*$T28/$E$5	=AX28*$W28/$E$5
29	=AM29*COS(RADIANS($CQ29))	=AN29	=(AO29/60/60/24) * (360*60*60)	=(AP29/60/60) * (60*60)	=AQ29/60/60	=AR29/60/60	0.0002	0.0002	=SQRT(AU29)	=SQRT(AV29)	=AW29*$O29/$E$5	=AX29*$Q29/$E$5	=AW29*$T29/$E$5	=AX29*$W29/$E$5
30	=AM30*COS(RADIANS($CQ30))	=AN30	=(AO30/60/60/24) * (360*60*60)	=(AP30/60/60) * (60*60)	=AQ30/60/60	=AR30/60/60	0.0003	0.0003	=SQRT(AU30)	=SQRT(AV30)	=AW30*$O30/$E$5	=AX30*$Q30/$E$5	=AW30*$T30/$E$5	=AX30*$W30/$E$5
31	=AM31*COS(RADIANS($CQ31))	=AN31	=(AO31/60/60/24) * (360*60*60)	=(AP31/60/60) * (60*60)	=AQ31/60/60	=AR31/60/60	0.0000	0.0001	=SQRT(AU31)	=SQRT(AV31)	=AW31*$O31/$E$5	=AX31*$Q31/$E$5	=AW31*$T31/$E$5	=AX31*$W31/$E$5
32	=AM32*COS(RADIANS($CQ32))	=AN32	=(AO32/60/60/24) * (360*60*60)	=(AP32/60/60) * (60*60)	=AQ32/60/60	=AR32/60/60	0.0001	0.0001	=SQRT(AU32)	=SQRT(AV32)	=AW32*$O32/$E$5	=AX32*$Q32/$E$5	=AW32*$T32/$E$5	=AX32*$W32/$E$5
33	=AM33*COS(RADIANS($CQ33))	=AN33	=(AO33/60/60/24) * (360*60*60)	=(AP33/60/60) * (60*60)	=AQ33/60/60	=AR33/60/60	0.0000	0.0001	=SQRT(AU33)	=SQRT(AV33)	=AW33*$O33/$E$5	=AX33*$Q33/$E$5	=AW33*$T33/$E$5	=AX33*$W33/$E$5
34	=AM34*COS(RADIANS($CQ34))	=AN34	=(AO34/60/60/24) * (360*60*60)	=(AP34/60/60) * (60*60)	=AQ34/60/60	=AR34/60/60	0.0002	0.0002	=SQRT(AU34)	=SQRT(AV34)	=AW34*$O34/$E$5	=AX34*$Q34/$E$5	=AW34*$T34/$E$5	=AX34*$W34/$E$5
35	=AM35*COS(RADIANS($CQ35))	=AN35	=(AO35/60/60/24) * (360*60*60)	=(AP35/60/60) * (60*60)	=AQ35/60/60	=AR35/60/60	0.0001	0.0001	=SQRT(AU35)	=SQRT(AV35)	=AW35*$O35/$E$5	=AX35*$Q35/$E$5	=AW35*$T35/$E$5	=AX35*$W35/$E$5
36	=AM36*COS(RADIANS($CQ36))	=AN36	=(AO36/60/60/24) * (360*60*60)	=(AP36/60/60) * (60*60)	=AQ36/60/60	=AR36/60/60	0.0000	0.0001	=SQRT(AU36)	=SQRT(AV36)	=AW36*$O36/$E$5	=AX36*$Q36/$E$5	=AW36*$T36/$E$5	=AX36*$W36/$E$5
37														
38														
39														
40														
41														
42														
43														

	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN	BO	BP
1														
2														
3														
4														
5														
6										Fudge Factor	3.32			
7										(See checkSDSSraDecCalibration2.ods for GAIA SeExtractor RA/Dec versus pixel x/y differenc				
8														
9														
10														
11						SDSS GAIA mexhat Object Detection Error								
12	RA SDSS GAIA mexha	Dec SDSS GAIA mexhi	RA SDSS GAIA mexhat object detecti	Dec SDSS GARA	SDSS GAIA mexhat object detectior	Dec SDSS GAIA mexhat ot	RA SDSS GAI	Dec SDSS GARA	SDSS GAIA mexhat o	Dec SDSS GAIA mexhat	x POSSI mexhy	POSSI GAIA	x POSSI mexhat objy	POSSI GAIA mexh
13	=AY13+AZ13	=BA13+BB13	=BC13*COS(RADIANS($CQ13))	=BD13	=(BE13/60/24) * (360*60*60)	=(BF13/60/60) * (60*60)	2.00E-011	1.99E-011	=SQRT(BI13)	=SQRT(BJ13)	0.003	0.002	=SQRT(BM13)	=SQRT(BN13)
14	=AY14+AZ14	=BA14+BB14	=BC14*COS(RADIANS($CQ14))	=BD14	=(BE14/60/60/24) * (360*60*60)	=(BF14/60/60) * (60*60)	3.65E-012	4.29E-012	=SQRT(BI14)	=SQRT(BJ14)	0.003	0.003	=SQRT(BM14)	=SQRT(BN14)
15	=AY15+AZ15	=BA15+BB15	=BC15*COS(RADIANS($CQ15))	=BD15	=(BE15/60/60/24) * (360*60*60)	=(BF15/60/60) * (60*60)	3.09E-013	2.80E-013	=SQRT(BI15)	=SQRT(BJ15)	0.002	0.002	=SQRT(BM15)	=SQRT(BN15)
16	=AY16+AZ16	=BA16+BB16	=BC16*COS(RADIANS($CQ16))	=BD16	=(BE16/60/60/24) * (360*60*60)	=(BF16/60/60) * (60*60)	8.06E-013	7.37E-013	=SQRT(BI16)	=SQRT(BJ16)	0.003	0.003	=SQRT(BM16)	=SQRT(BN16)
17	=AY17+AZ17	=BA17+BB17	=BC17*COS(RADIANS($CQ17))	=BD17	=(BE17/60/60/24) * (360*60*60)	=(BF17/60/60) * (60*60)			=BG17*$BM$6/60/60	=BH17*$BM$6/60/60	0.002	0.002	=SQRT(BM17)	=SQRT(BN17)
18	=AY18+AZ18	=BA18+BB18	=BC18*COS(RADIANS($CQ18))	=BD18	=(BE18/60/60/24) * (360*60*60)	=(BF18/60/60) * (60*60)			=BG18*$BM$6/60/60	=BH18*$BM$6/60/60	0.001	0.001	=SQRT(BM18)	=SQRT(BN18)
19	=AY19+AZ19	=BA19+BB19	=BC19*COS(RADIANS($CQ19))	=BD19	=(BE19/60/60/24) * (360*60*60)	=(BF19/60/60) * (60*60)	2.35E-013	1.80E-013	=SQRT(BI19)	=SQRT(BJ19)	0.002	0.003	=SQRT(BM19)	=SQRT(BN19)
20	=AY20+AZ20	=BA20+BB20	=BC20*COS(RADIANS($CQ20))	=BD20	=(BE20/60/60/24) * (360*60*60)	=(BF20/60/60) * (60*60)	3.56E-012	2.56E-012	=SQRT(BI20)	=SQRT(BJ20)	0.002	0.002	=SQRT(BM20)	=SQRT(BN20)
21	=AY21+AZ21	=BA21+BB21	=BC21*COS(RADIANS($CQ21))	=BD21	=(BE21/60/60/24) * (360*60*60)	=(BF21/60/60) * (60*60)	8.10E-014	6.63E-014	=SQRT(BI21)	=SQRT(BJ21)	0.003	0.003	=SQRT(BM21)	=SQRT(BN21)
22	=AY22+AZ22	=BA22+BB22	=BC22*COS(RADIANS($CQ22))	=BD22	=(BE22/60/60/24) * (360*60*60)	=(BF22/60/60) * (60*60)	1.20E-013	1.04E-013	=SQRT(BI22)	=SQRT(BJ22)	0.003	0.003	=SQRT(BM22)	=SQRT(BN22)
23	=AY23+AZ23	=BA23+BB23	=BC23*COS(RADIANS($CQ23))	=BD23	=(BE23/60/60/24) * (360*60*60)	=(BF23/60/60) * (60*60)	2.81E-011	1.27E-011	=SQRT(BI23)	=SQRT(BJ23)	0.000	0.000	=SQRT(BM23)	=SQRT(BN23)
24	=AY24+AZ24	=BA24+BB24	=BC24*COS(RADIANS($CQ24))	=BD24	=(BE24/60/60/24) * (360*60*60)	=(BF24/60/60) * (60*60)	4.80E-011	3.37E-011	=SQRT(BI24)	=SQRT(BJ24)	0.000	0.001	=SQRT(BM24)	=SQRT(BN24)
25	=AY25+AZ25	=BA25+BB25	=BC25*COS(RADIANS($CQ25))	=BD25	=(BE25/60/60/24) * (360*60*60)	=(BF25/60/60) * (60*60)	2.23E-012	1.88E-012	=SQRT(BI25)	=SQRT(BJ25)	0.004	0.004	=SQRT(BM25)	=SQRT(BN25)
26	=AY26+AZ26	=BA26+BB26	=BC26*COS(RADIANS($CQ26))	=BD26	=(BE26/60/60/24) * (360*60*60)	=(BF26/60/60) * (60*60)	1.30E-013	1.20E-013	=SQRT(BI26)	=SQRT(BJ26)	0.003	0.003	=SQRT(BM26)	=SQRT(BN26)
27	=AY27+AZ27	=BA27+BB27	=BC27*COS(RADIANS($CQ27))	=BD27	=(BE27/60/60/24) * (360*60*60)	=(BF27/60/60) * (60*60)	3.09E-012	2.25E-012	=SQRT(BI27)	=SQRT(BJ27)	0.001	0.002	=SQRT(BM27)	=SQRT(BN27)
28	=AY28+AZ28	=BA28+BB28	=BC28*COS(RADIANS($CQ28))	=BD28	=(BE28/60/60/24) * (360*60*60)	=(BF28/60/60) * (60*60)	2.90E-012	2.05E-012	=SQRT(BI28)	=SQRT(BJ28)	0.002	0.002	=SQRT(BM28)	=SQRT(BN28)
29	=AY29+AZ29	=BA29+BB29	=BC29*COS(RADIANS($CQ29))	=BD29	=(BE29/60/60/24) * (360*60*60)	=(BF29/60/60) * (60*60)	2.52E-012	2.87E-012	=SQRT(BI29)	=SQRT(BJ29)	0.002	0.003	=SQRT(BM29)	=SQRT(BN29)
30	=AY30+AZ30	=BA30+BB30	=BC30*COS(RADIANS($CQ30))	=BD30	=(BE30/60/60/24) * (360*60*60)	=(BF30/60/60) * (60*60)	3.24E-012	3.66E-012	=SQRT(BI30)	=SQRT(BJ30)	0.003	0.003	=SQRT(BM30)	=SQRT(BN30)
31	=AY31+AZ31	=BA31+BB31	=BC31*COS(RADIANS($CQ31))	=BD31	=(BE31/60/60/24) * (360*60*60)	=(BF31/60/60) * (60*60)	9.17E-013	4.60E-013	=SQRT(BI31)	=SQRT(BJ31)	0.003	0.003	=SQRT(BM31)	=SQRT(BN31)
32	=AY32+AZ32	=BA32+BB32	=BC32*COS(RADIANS($CQ32))	=BD32	=(BE32/60/60/24) * (360*60*60)	=(BF32/60/60) * (60*60)	1.25E-012	8.19E-013	=SQRT(BI32)	=SQRT(BJ32)	0.005	0.002	=SQRT(BM32)	=SQRT(BN32)
33	=AY33+AZ33	=BA33+BB33	=BC33*COS(RADIANS($CQ33))	=BD33	=(BE33/60/60/24) * (360*60*60)	=(BF33/60/60) * (60*60)	5.79E-013	3.17E-013	=SQRT(BI33)	=SQRT(BJ33)	0.000	0.000	=SQRT(BM33)	=SQRT(BN33)
34	=AY34+AZ34	=BA34+BB34	=BC34*COS(RADIANS($CQ34))	=BD34	=(BE34/60/60/24) * (360*60*60)	=(BF34/60/60) * (60*60)	1.79E-012	1.63E-012	=SQRT(BI34)	=SQRT(BJ34)	0.000	0.000	=SQRT(BM34)	=SQRT(BN34)
35	=AY35+AZ35	=BA35+BB35	=BC35*COS(RADIANS($CQ35))	=BD35	=(BE35/60/60/24) * (360*60*60)	=(BF35/60/60) * (60*60)			=BG35*$BM$6/60/60	=BH35*$BM$6/60/60	0.001	0.001	=SQRT(BM35)	=SQRT(BN35)
36	=AY36+AZ36	=BA36+BB36	=BC36*COS(RADIANS($CQ36))	=BD36	=(BE36/60/60/24) * (360*60*60)	=(BF36/60/60) * (60*60)			=BG36*$BM$6/60/60	=BH36*$BM$6/60/60	0.001	0.001	=SQRT(BM36)	=SQRT(BN36)
37														
38														
39														
40														
41														
42														
43														

	BQ	BR	BS	BT	BU	BV	BW	BX	BY	BZ	CA	CB
1												
2												
3												
4												
5												
6												
7	es)											
8												
9												
10												
11	POSSI GAIA mexhat Object Detection Error				GAIA mexhat Object Detection Error				GAIA object detection & reference field star RMS Errors			
12	RA POSSI me	Dec POSSI G	RA POSSI mexhat ot	Dec POSSI GAIA	RA Detection error	Dec Detection error	x Detection error	y Detection error	Ra error (decimal degrees)	Dec error (decimal degrees)	Ra error (decimal as)	Dec error (decimal as
13	2.62E-010	1.86E-010	=SQRT(BQ13)	=SQRT(BR13)	=SQRT(BK13^2 + BS13^2)	=SQRT(BL13^2 + BT13^2)	=SQRT(AW13^2 + BO13^2)	=SQRT(AX13^2 + BP13^2)	=SQRT(AS13^2+ BU13^2)	=SQRT(AT13^2+ BV13^2)	=BY13*60*60	=BZ13*60*60
14	2.65E-010	2.63E-010	=SQRT(BQ14)	=SQRT(BR14)	=SQRT(BK14^2 + BS14^2)	=SQRT(BL14^2 + BT14^2)	=SQRT(AW14^2 + BO14^2)	=SQRT(AX14^2 + BP14^2)	=SQRT(AS14^2+ BU14^2)	=SQRT(AT14^2+ BV14^2)	=BY14*60*60	=BZ14*60*60
15	1.46E-010	1.36E-010	=SQRT(BQ15)	=SQRT(BR15)	=SQRT(BK15^2 + BS15^2)	=SQRT(BL15^2 + BT15^2)	=SQRT(AW15^2 + BO15^2)	=SQRT(AX15^2 + BP15^2)	=SQRT(AS15^2+ BU15^2)	=SQRT(AT15^2+ BV15^2)	=BY15*60*60	=BZ15*60*60
16	1.69E-010	1.69E-010	=SQRT(BQ16)	=SQRT(BR16)	=SQRT(BK16^2 + BS16^2)	=SQRT(BL16^2 + BT16^2)	=SQRT(AW16^2 + BO16^2)	=SQRT(AX16^2 + BP16^2)	=SQRT(AS16^2+ BU16^2)	=SQRT(AT16^2+ BV16^2)	=BY16*60*60	=BZ16*60*60
17	1.63E-010	1.77E-010	=SQRT(BQ17)	=SQRT(BR17)	=SQRT(BK17^2 + BS17^2)	=SQRT(BL17^2 + BT17^2)	=SQRT(AW17^2 + BO17^2)	=SQRT(AX17^2 + BP17^2)	=SQRT(AS17^2+ BU17^2)	=SQRT(AT17^2+ BV17^2)	=BY17*60*60	=BZ17*60*60
18	1.15E-010	1.13E-010	=SQRT(BQ18)	=SQRT(BR18)	=SQRT(BK18^2 + BS18^2)	=SQRT(BL18^2 + BT18^2)	=SQRT(AW18^2 + BO18^2)	=SQRT(AX18^2 + BP18^2)	=SQRT(AS18^2+ BU18^2)	=SQRT(AT18^2+ BV18^2)	=BY18*60*60	=BZ18*60*60
19	1.83E-010	2.49E-010	=SQRT(BQ19)	=SQRT(BR19)	=SQRT(BK19^2 + BS19^2)	=SQRT(BL19^2 + BT19^2)	=SQRT(AW19^2 + BO19^2)	=SQRT(AX19^2 + BP19^2)	=SQRT(AS19^2+ BU19^2)	=SQRT(AT19^2+ BV19^2)	=BY19*60*60	=BZ19*60*60
20	1.87E-010	1.79E-010	=SQRT(BQ20)	=SQRT(BR20)	=SQRT(BK20^2 + BS20^2)	=SQRT(BL20^2 + BT20^2)	=SQRT(AW20^2 + BO20^2)	=SQRT(AX20^2 + BP20^2)	=SQRT(AS20^2+ BU20^2)	=SQRT(AT20^2+ BV20^2)	=BY20*60*60	=BZ20*60*60
21	2.16E-010	2.49E-010	=SQRT(BQ21)	=SQRT(BR21)	=SQRT(BK21^2 + BS21^2)	=SQRT(BL21^2 + BT21^2)	=SQRT(AW21^2 + BO21^2)	=SQRT(AX21^2 + BP21^2)	=SQRT(AS21^2+ BU21^2)	=SQRT(AT21^2+ BV21^2)	=BY21*60*60	=BZ21*60*60
22	2.04E-010	2.48E-010	=SQRT(BQ22)	=SQRT(BR22)	=SQRT(BK22^2 + BS22^2)	=SQRT(BL22^2 + BT22^2)	=SQRT(AW22^2 + BO22^2)	=SQRT(AX22^2 + BP22^2)	=SQRT(AS22^2+ BU22^2)	=SQRT(AT22^2+ BV22^2)	=BY22*60*60	=BZ22*60*60
23	2.73E-012	2.84E-012	=SQRT(BQ23)	=SQRT(BR23)	=SQRT(BK23^2 + BS23^2)	=SQRT(BL23^2 + BT23^2)	=SQRT(AW23^2 + BO23^2)	=SQRT(AX23^2 + BP23^2)	=SQRT(AS23^2+ BU23^2)	=SQRT(AT23^2+ BV23^2)	=BY23*60*60	=BZ23*60*60
24	5.51E-012	5.89E-012	=SQRT(BQ24)	=SQRT(BR24)	=SQRT(BK24^2 + BS24^2)	=SQRT(BL24^2 + BT24^2)	=SQRT(AW24^2 + BO24^2)	=SQRT(AX24^2 + BP24^2)	=SQRT(AS24^2+ BU24^2)	=SQRT(AT24^2+ BV24^2)	=BY24*60*60	=BZ24*60*60
25	3.16E-010	2.82E-010	=SQRT(BQ25)	=SQRT(BR25)	=SQRT(BK25^2 + BS25^2)	=SQRT(BL25^2 + BT25^2)	=SQRT(AW25^2 + BO25^2)	=SQRT(AX25^2 + BP25^2)	=SQRT(AS25^2+ BU25^2)	=SQRT(AT25^2+ BV25^2)	=BY25*60*60	=BZ25*60*60
26	2.56E-010	2.41E-010	=SQRT(BQ26)	=SQRT(BR26)	=SQRT(BK26^2 + BS26^2)	=SQRT(BL26^2 + BT26^2)	=SQRT(AW26^2 + BO26^2)	=SQRT(AX26^2 + BP26^2)	=SQRT(AS26^2+ BU26^2)	=SQRT(AT26^2+ BV26^2)	=BY26*60*60	=BZ26*60*60
27	7.65E-011	1.39E-010	=SQRT(BQ27)	=SQRT(BR27)	=SQRT(BK27^2 + BS27^2)	=SQRT(BL27^2 + BT27^2)	=SQRT(AW27^2 + BO27^2)	=SQRT(AX27^2 + BP27^2)	=SQRT(AS27^2+ BU27^2)	=SQRT(AT27^2+ BV27^2)	=BY27*60*60	=BZ27*60*60
28	1.28E-010	1.22E-010	=SQRT(BQ28)	=SQRT(BR28)	=SQRT(BK28^2 + BS28^2)	=SQRT(BL28^2 + BT28^2)	=SQRT(AW28^2 + BO28^2)	=SQRT(AX28^2 + BP28^2)	=SQRT(AS28^2+ BU28^2)	=SQRT(AT28^2+ BV28^2)	=BY28*60*60	=BZ28*60*60
29	1.69E-010	1.97E-010	=SQRT(BQ29)	=SQRT(BR29)	=SQRT(BK29^2 + BS29^2)	=SQRT(BL29^2 + BT29^2)	=SQRT(AW29^2 + BO29^2)	=SQRT(AX29^2 + BP29^2)	=SQRT(AS29^2+ BU29^2)	=SQRT(AT29^2+ BV29^2)	=BY29*60*60	=BZ29*60*60
30	2.27E-010	2.20E-010	=SQRT(BQ30)	=SQRT(BR30)	=SQRT(BK30^2 + BS30^2)	=SQRT(BL30^2 + BT30^2)	=SQRT(AW30^2 + BO30^2)	=SQRT(AX30^2 + BP30^2)	=SQRT(AS30^2+ BU30^2)	=SQRT(AT30^2+ BV30^2)	=BY30*60*60	=BZ30*60*60
31	2.25E-010	2.25E-010	=SQRT(BQ31)	=SQRT(BR31)	=SQRT(BK31^2 + BS31^2)	=SQRT(BL31^2 + BT31^2)	=SQRT(AW31^2 + BO31^2)	=SQRT(AX31^2 + BP31^2)	=SQRT(AS31^2+ BU31^2)	=SQRT(AT31^2+ BV31^2)	=BY31*60*60	=BZ31*60*60
32	3.61E-010	1.91E-010	=SQRT(BQ32)	=SQRT(BR32)	=SQRT(BK32^2 + BS32^2)	=SQRT(BL32^2 + BT32^2)	=SQRT(AW32^2 + BO32^2)	=SQRT(AX32^2 + BP32^2)	=SQRT(AS32^2+ BU32^2)	=SQRT(AT32^2+ BV32^2)	=BY32*60*60	=BZ32*60*60
33	6.79E-013	4.85E-013	=SQRT(BQ33)	=SQRT(BR33)	=SQRT(BK33^2 + BS33^2)	=SQRT(BL33^2 + BT33^2)	=SQRT(AW33^2 + BO33^2)	=SQRT(AX33^2 + BP33^2)	=SQRT(AS33^2+ BU33^2)	=SQRT(AT33^2+ BV33^2)	=BY33*60*60	=BZ33*60*60
34	2.52E-012	2.52E-012	=SQRT(BQ34)	=SQRT(BR34)	=SQRT(BK34^2 + BS34^2)	=SQRT(BL34^2 + BT34^2)	=SQRT(AW34^2 + BO34^2)	=SQRT(AX34^2 + BP34^2)	=SQRT(AS34^2+ BU34^2)	=SQRT(AT34^2+ BV34^2)	=BY34*60*60	=BZ34*60*60
35	8.54E-011	7.08E-011	=SQRT(BQ35)	=SQRT(BR35)	=SQRT(BK35^2 + BS35^2)	=SQRT(BL35^2 + BT35^2)	=SQRT(AW35^2 + BO35^2)	=SQRT(AX35^2 + BP35^2)	=SQRT(AS35^2+ BU35^2)	=SQRT(AT35^2+ BV35^2)	=BY35*60*60	=BZ35*60*60
36	8.17E-011	7.86E-011	=SQRT(BQ36)	=SQRT(BR36)	=SQRT(BK36^2 + BS36^2)	=SQRT(BL36^2 + BT36^2)	=SQRT(AW36^2 + BO36^2)	=SQRT(AX36^2 + BP36^2)	=SQRT(AS36^2+ BU36^2)	=SQRT(AT36^2+ BV36^2)	=BY36*60*60	=BZ36*60*60
37												
38												
39												
40												
41												
42												
43												

DD

NB most of the error is contributed from the RMS error, not the GAIA object detection error

	CC	CD	CE	CF	CG	CH	CI	CJ	CK	CL	CM	CN	CO	CP	CQ	CR	CS	CT
9												GAIA/Topcat proper motion calculations						
10												Proper motion	Proper motion deviation	Pixel y				
11						Secondary Calculations: SDS		Secondary Calculations: GAIA object detection for POSS-I coordinates - without reference star correction)								GAIA/topcat calculated proper motions (as/xyears)		
12	x error	y error				ra1b	ra1b	ra2b	de2b						Dec	ra uncorrecteddec	ra	
13	=SQRT(AG13^2+ BW13^2)	=SQRT(AH13^2+ BX13^2)				12.73	1.1	12.734	3.050			3.8547	-0.3523		13.88365572	=AD13	=AE13	=CR13*COS(RADIANS($CQ13))
14	=SQRT(AG14^2+ BW14^2)	=SQRT(AH14^2+ BX14^2)				12.26	2	12.257	4.020			3.6437	0.2144		13.88390554	=AD14	=AE14	=CR14*COS(RADIANS($CQ14))
15	=SQRT(AG15^2+ BW15^2)	=SQRT(AH15^2+ BX15^2)				46.48	2.1	46.680	1.920			1.5526	2.7554		52.16725612	=AD15	=AE15	=CR15*COS(RADIANS($CQ15))
16	=SQRT(AG16^2+ BW16^2)	=SQRT(AH16^2+ BX16^2)				46.8	58	47.016	57.640			1.1961	3.1457		52.16612142	=AD16	=AE16	=CR16*COS(RADIANS($CQ16))
17	=SQRT(AG17^2+ BW17^2)	=SQRT(AH17^2+ BX17^2)				55.91	38.4	55.896	41.200			2.5872	-4.8598		33.07733943	=AD17	=AE17	=CR17*COS(RADIANS($CQ17))
18	=SQRT(AG18^2+ BW18^2)	=SQRT(AH18^2+ BX18^2)				56.77	35.7	56.760	38.600			2.5567	-5.0004		33.07660731	=AD18	=AE18	=CR18*COS(RADIANS($CQ18))
19	=SQRT(AG19^2+ BW19^2)	=SQRT(AH19^2+ BX19^2)				32.74	44.2	32.664	44.900			1.3458	-2.6787		14.07895147	=AD19	=AE19	=CR19*COS(RADIANS($CQ19))
20	=SQRT(AG20^2+ BW20^2)	=SQRT(AH20^2+ BX20^2)				32.21	39.2	32.136	39.940			1.3266	-2.9730		14.07756995	=AD20	=AE20	=CR20*COS(RADIANS($CQ20))
21	=SQRT(AG21^2+ BW21^2)	=SQRT(AH21^2+ BX21^2)				1.62	31.3	1.392	36.700			11.4193	-4.8810		22.02537688	=AD21	=AE21	=CR21*COS(RADIANS($CQ21))
22	=SQRT(AG22^2+ BW22^2)	=SQRT(AH22^2+ BX22^2)				1.72	24.9	1.512	29.100			8.3707	-4.7616		22.02359622	=AD22	=AE22	=CR22*COS(RADIANS($CQ22))
23	=SQRT(AG23^2+ BW23^2)	=SQRT(AH23^2+ BX23^2)									check order	-0.5320	0.3016		17.24121308	=AD23	=AE23	=CR23*COS(RADIANS($CQ23))
24	=SQRT(AG24^2+ BW24^2)	=SQRT(AH24^2+ BX24^2)									check order	-0.5757	0.2284		17.24177863	=AD24	=AE24	=CR24*COS(RADIANS($CQ24))
25	=SQRT(AG25^2+ BW25^2)	=SQRT(AH25^2+ BX25^2)				36.86	36.9	37.001	35.880			3.5187	5.7835		-0.82692479	=AD25	=AE25	=CR25*COS(RADIANS($CQ25))
26	=SQRT(AG26^2+ BW26^2)	=SQRT(AH26^2+ BX26^2)				36.6	18.4	36.742	17.540			3.1108	5.8023		-0.82178397	=AD26	=AE26	=CR26*COS(RADIANS($CQ26))
27	=SQRT(AG27^2+ BW27^2)	=SQRT(AH27^2+ BX27^2)				46.88	34.5	46.944	37.760			8.3146	4.1688		13.35958843	=AD27	=AE27	=CR27*COS(RADIANS($CQ27))
28	=SQRT(AG28^2+ BW28^2)	=SQRT(AH28^2+ BX28^2)				47	38.4	47.016	41.110			7.0938	2.1542		13.36069038	=AD28	=AE28	=CR28*COS(RADIANS($CQ28))
29	=SQRT(AG29^2+ BW29^2)	=SQRT(AH29^2+ BX29^2)				37.81	13.9	37.920	15.970			5.3160	2.1437		13.73720478	=AD29	=AE29	=CR29*COS(RADIANS($CQ29))
30	=SQRT(AG30^2+ BW30^2)	=SQRT(AH30^2+ BX30^2)				37.82	8.7	37.896	10.500			4.5708	1.5048		13.73576715	=AD30	=AE30	=CR30*COS(RADIANS($CQ30))
31	=SQRT(AG31^2+ BW31^2)	=SQRT(AH31^2+ BX31^2)				32.14	39.6	32.232	38.660			-3.5421	1.4438		20.51100050	=AD31	=AE31	=CR31*COS(RADIANS($CQ31))
32	=SQRT(AG32^2+ BW32^2)	=SQRT(AH32^2+ BX32^2)				32.56	39.3	32.688	38.590			-3.2085	2.2588		20.51094081	=AD32	=AE32	=CR32*COS(RADIANS($CQ32))
33	=SQRT(AG33^2+ BW33^2)	=SQRT(AH33^2+ BX33^2)									check order	0.1770	-1.4138		16.53098677	=AD33	=AE33	=CR33*COS(RADIANS($CQ33))
34	=SQRT(AG34^2+ BW34^2)	=SQRT(AH34^2+ BX34^2)									check order	0.2598	-1.1625		16.53080830	=AD34	=AE34	=CR34*COS(RADIANS($CQ34))
35	=SQRT(AG35^2+ BW35^2)	=SQRT(AH35^2+ BX35^2)				36.3	8	36.288	7.150			3.4697	-0.8088		-8.46888943	=AD35	=AE35	=CR35*COS(RADIANS($CQ35))
36	=SQRT(AG36^2+ BW36^2)	=SQRT(AH36^2+ BX36^2)				36.56	6	36.552	4.870			4.3171	-0.6499		-8.46834597	=AD36	=AE36	=CR36*COS(RADIANS($CQ36))

	CU	CV	CW	CX	CY	CZ	DA	DB	DC	DD	DE	DF	DG	DH
1					POSS-I Survey Year									
2					01/01/58									
3														
4					INT Survey Year									
5					02/12/11									
6														
7														
8														
9														
10														
11					GAIA/Topcat Proper motions				SDSS proper motions		SDSS date of	Years between POSS-I	and SDSS survey	
12	dec	ra (decimal as)	dec (decimal as)	WD Name	ra (mas/year)	dec (mas/year)	ra error (mas/year)	dec error (mas/year)	ra (mas/year)	dec (mas/year)				
13	=CS13	=(CT13/60/60/24) * (360*60*60)	=(CU13/60/60) * (60*60)	=B13	=-CV13*1000/$DF13	=-CW13*1000/$DF13	=(CA13)*1000/$DF13	=(CB13)*1000/$DF13	18.25	-39.65	10/14/99	=(DE13-$CY$2)/365		
14	=CS14	=(CT14/60/60/24) * (360*60*60)	=(CU14/60/60) * (60*60)	=B14	=-CV14*1000/$DF14	=-CW14*1000/$DF14	=(CA14)*1000/$DF14	=(CB14)*1000/$DF14	-2.98	-38.47	10/14/99	=(DE14-$CY$2)/365		
15	=CS15	=(CT15/60/60/24) * (360*60*60)	=(CU15/60/60) * (60*60)	=B15	=-CV15*1000/$DF15	=-CW15*1000/$DF15	=(CA15)*1000/$DF15	=(CB15)*1000/$DF15	-73.94	66.73	06/09/02	=(DE15-$CY$2)/365		
16	=CS16	=(CT16/60/60/24) * (360*60*60)	=(CU16/60/60) * (60*60)	=B16	=-CV16*1000/$DF16	=-CW16*1000/$DF16	=(CA16)*1000/$DF16	=(CB16)*1000/$DF16	0	0	06/09/02	=(DE16-$CY$2)/365		
17	=CS17	=(CT17/60/60/24) * (360*60*60)	=(CU17/60/60) * (60*60)	=B17	=-CV17*1000/$DF17	=-CW17*1000/$DF17	=(CA17)*1000/$DF17	=(CB17)*1000/$DF17	-1.79	-51.17	05/23/01	=(DE17-$CY$2)/365		
18	=CS18	=(CT18/60/60/24) * (360*60*60)	=(CU18/60/60) * (60*60)	=B18	=-CV18*1000/$DF18	=-CW18*1000/$DF18	=(CA18)*1000/$DF18	=(CB18)*1000/$DF18	0.32	-50.52	05/23/01	=(DE18-$CY$2)/365		
19	=CS19	=(CT19/60/60/24) * (360*60*60)	=(CU19/60/60) * (60*60)	=B19	=-CV19*1000/$DF19	=-CW19*1000/$DF19	=(CA19)*1000/$DF19	=(CB19)*1000/$DF19	28.09	-14.32	09/27/00	=(DE19-$CY$2)/365		
20	=CS20	=(CT20/60/60/24) * (360*60*60)	=(CU20/60/60) * (60*60)	=B20	=-CV20*1000/$DF20	=-CW20*1000/$DF20	=(CA20)*1000/$DF20	=(CB20)*1000/$DF20	28.57	-16.32	09/27/00	=(DE20-$CY$2)/365		
21	=CS21	=(CT21/60/60/24) * (360*60*60)	=(CU21/60/60) * (60*60)	=B21	=-CV21*1000/$DF21	=-CW21*1000/$DF21	=(CA21)*1000/$DF21	=(CB21)*1000/$DF21	0	0	10/13/04	=(DE21-$CY$2)/365		
22	=CS22	=(CT22/60/60/24) * (360*60*60)	=(CU22/60/60) * (60*60)	=B22	=-CV22*1000/$DF22	=-CW22*1000/$DF22	=(CA22)*1000/$DF22	=(CB22)*1000/$DF22	23.9	-5.01	10/13/04	=(DE22-$CY$2)/365		
23	=CS23	=(CT23/60/60/24) * (360*60*60)	=(CU23/60/60) * (60*60)	=B23	=-CV23*1000/$DF23	=-CW23*1000/$DF23	=(CA23)*1000/$DF23	=(CB23)*1000/$DF23	0	0	11/04/05	=($CY$5-DE23)/365		
24	=CS24	=(CT24/60/60/24) * (360*60*60)	=(CU24/60/60) * (60*60)	=B24	=-CV24*1000/$DF24	=-CW24*1000/$DF24	=(CA24)*1000/$DF24	=(CB24)*1000/$DF24	0	0	11/04/05	=($CY$5-DE24)/365		
25	=CS25	=(CT25/60/60/24) * (360*60*60)	=(CU25/60/60) * (60*60)	=B25	=-CV25*1000/$DF25	=-CW25*1000/$DF25	=(CA25)*1000/$DF25	=(CB25)*1000/$DF25	-30.91	-23.31	09/05/02	=(DE25-$CY$2)/365		
26	=CS26	=(CT26/60/60/24) * (360*60*60)	=(CU26/60/60) * (60*60)	=B26	=-CV26*1000/$DF26	=-CW26*1000/$DF26	=(CA26)*1000/$DF26	=(CB26)*1000/$DF26	-24.79	-23.63	09/05/02	=(DE26-$CY$2)/365		
27	=CS27	=(CT27/60/60/24) * (360*60*60)	=(CU27/60/60) * (60*60)	=B27	=-CV27*1000/$DF27	=-CW27*1000/$DF27	=(CA27)*1000/$DF27	=(CB27)*1000/$DF27	-50.89	-142.99	01/06/06	=(DE27-$CY$2)/365		
28	=CS28	=(CT28/60/60/24) * (360*60*60)	=(CU28/60/60) * (60*60)	=B28	=-CV28*1000/$DF28	=-CW28*1000/$DF28	=(CA28)*1000/$DF28	=(CB28)*1000/$DF28	11.46	-5.05	01/06/06	=(DE28-$CY$2)/365		
29	=CS29	=(CT29/60/60/24) * (360*60*60)	=(CU29/60/60) * (60*60)	=B29	=-CV29*1000/$DF29	=-CW29*1000/$DF29	=(CA29)*1000/$DF29	=(CB29)*1000/$DF29	-22.42	-26.25	01/28/03	=(DE29-$CY$2)/365		
30	=CS30	=(CT30/60/60/24) * (360*60*60)	=(CU30/60/60) * (60*60)	=B30	=-CV30*1000/$DF30	=-CW30*1000/$DF30	=(CA30)*1000/$DF30	=(CB30)*1000/$DF30	-19.23	-77.07	01/28/03	=(DE30-$CY$2)/365		
31	=CS31	=(CT31/60/60/24) * (360*60*60)	=(CU31/60/60) * (60*60)	=B31	=-CV31*1000/$DF31	=-CW31*1000/$DF31	=(CA31)*1000/$DF31	=(CB31)*1000/$DF31	-37.72	34.74	05/10/05	=(DE31-$CY$2)/365		
32	=CS32	=(CT32/60/60/24) * (360*60*60)	=(CU32/60/60) * (60*60)	=B32	=-CV32*1000/$DF32	=-CW32*1000/$DF32	=(CA32)*1000/$DF32	=(CB32)*1000/$DF32	-32.05	24.88	05/10/05	=(DE32-$CY$2)/365		
33	=CS33	=(CT33/60/60/24) * (360*60*60)	=(CU33/60/60) * (60*60)	=B33	=-CV33*1000/$DF33	=-CW33*1000/$DF33	=(CA33)*1000/$DF33	=(CB33)*1000/$DF33	-84.01	23.29	06/01/05	=($CY$5-DE33)/365		
34	=CS34	=(CT34/60/60/24) * (360*60*60)	=(CU34/60/60) * (60*60)	=B34	=-CV34*1000/$DF34	=-CW34*1000/$DF34	=(CA34)*1000/$DF34	=(CB34)*1000/$DF34	0	0	06/01/05	=($CY$5-DE34)/365		
35	=CS35	=(CT35/60/60/24) * (360*60*60)	=(CU35/60/60) * (60*60)	=B35	=-CV35*1000/$DF35	=-CW35*1000/$DF35	=(CA35)*1000/$DF35	=(CB35)*1000/$DF35	0	0	09/03/00	=(DE35-$CY$2)/365		
36	=CS36	=(CT36/60/60/24) * (360*60*60)	=(CU36/60/60) * (60*60)	=B36	=-CV36*1000/$DF36	=-CW36*1000/$DF36	=(CA36)*1000/$DF36	=(CB36)*1000/$DF36	0	0	09/03/00	=(DE36-$CY$2)/365		
37														
38														
39														
40														
41														
42														
43														

A.3.4 Calculation Probability of Chance Proper Motion Alignment - PDD Method

calcAveragePMofDDNearest150MDwarfObjects-19May11a.zip calcAveragePMofDDNearest150W
19May11a.zip

generateSQLQueryResultsRD7detailed-nearest150MDObjects.ods

start 5 arc minutes beyond target, and stop at a maximum distance such that ~ 150 nearest M [red/brown] dwarf objects are found

[just use square selection, not circular, as it is much easier this way]

						ra/dec min dis	ra/dec min dis	ra/dec max dis	ra/dec max dis	ra/dec min dis	ra/dec min dis	ra/dec max dis	ra/dec max distance			
ra/dec max distance (de		3		=$B$7	=$B$7	=$B$6	=$B$6	=$B$7	=$B$7	=$B$6	=$B$6					
ra/dec min distance (de		0.08	[5 arc minutes]													
empircal Mr max diff		0.1														
				nearest object conditions												
name	ra	dec		ra min neg	dec min neg	ra max neg	dec max neg	ra min pos	dec min pos	ra max pos	dec max pos	distance modu	app mag (r)	min abs mag r	max abs mag r	
DD-01A	13.053044	13.883656		=B19-D$6	=C19-E$6	=D19-F$6	=E19-G$6	=F19+H$6	=G19+I$6	=H19+J$6	=I19+K$6	7.059	18.915	=M19-L19-$B$8	=M19-L19+$B$8	
	13.053044	13.883656		=B20-D$6	=C20-E$6	=D20-F$6	=E20-G$6	=F20+H$6	=G20+I$6	=H20+J$6	=I20+K$6	7.062	17.981	=M20-L20-$B$8	=M20-L20+$B$8	
DD-02A	226.943691	52.167256		=B21-D$6	=C21-E$6	=D21-F$6	=E21-G$6	=F21+H$6	=G21+I$6	=H21+J$6	=I21+K$6	5.675	17.295	=M21-L21-$B$8	=M21-L21+$B$8	
	226.943691	52.167256		=B22-D$6	=C22-E$6	=D22-F$6	=E22-G$6	=F22+H$6	=G22+I$6	=H22+J$6	=I22+K$6	5.675	18.064	=M22-L22-$B$8	=M22-L22+$B$8	
DD-03A	255.982970	33.077339		=B23-D$6	=C23-E$6	=D23-F$6	=E23-G$6	=F23+H$6	=G23+I$6	=H23+J$6	=I23+K$6	6.316	18.858	=M23-L23-$B$8	=M23-L23+$B$8	
	255.982970	33.077339		=B24-D$6	=C24-E$6	=D24-F$6	=E24-G$6	=F24+H$6	=G24+I$6	=H24+J$6	=I24+K$6	6.356	18.271	=M24-L24-$B$8	=M24-L24+$B$8	
DD-04A	344.886423	14.078951		=B25-D$6	=C25-E$6	=D25-F$6	=E25-G$6	=F25+H$6	=G25+I$6	=H25+J$6	=I25+K$6	6.673	18.684	=M25-L25-$B$8	=M25-L25+$B$8	
	344.886423	14.078951		=B26-D$6	=C26-E$6	=D26-F$6	=E26-G$6	=F26+H$6	=G26+I$6	=H26+J$6	=I26+K$6	5.741	16.780	=M26-L26-$B$8	=M26-L26+$B$8	
DD-05A	335.756755	22.025377		=B27-D$6	=C27-E$6	=D27-F$6	=E27-G$6	=F27+H$6	=G27+I$6	=H27+J$6	=I27+K$6	4.298	15.911	=M27-L27-$B$8	=M27-L27+$B$8	
	335.756755	22.025377		=B28-D$6	=C28-E$6	=D28-F$6	=E28-G$6	=F28+H$6	=G28+I$6	=H28+J$6	=I28+K$6	4.552	16.205	=M28-L28-$B$8	=M28-L28+$B$8	
DD-06A	32.881340	17.241213		=B29-D$6	=C29-E$6	=D29-F$6	=E29-G$6	=F29+H$6	=G29+I$6	=H29+J$6	=I29+K$6	6.488	17.192	=M29-L29-$B$8	=M29-L29+$B$8	
	32.881340	17.241213		=B30-D$6	=C30-E$6	=D30-F$6	=E30-G$6	=F30+H$6	=G30+I$6	=H30+J$6	=I30+K$6	6.410	17.649	=M30-L30-$B$8	=M30-L30+$B$8	
DD-07A	53.153595	-0.826925		=B31-D$6	=C31-E$6	=D31-F$6	=E31-G$6	=F31+H$6	=G31+I$6	=H31+J$6	=I31+K$6	6.170	16.087	=M31-L31-$B$8	=M31-L31+$B$8	
	53.153595	-0.826925		=B32-D$6	=C32-E$6	=D32-F$6	=E32-G$6	=F32+H$6	=G32+I$6	=H32+J$6	=I32+K$6	6.169	18.300	=M32-L32-$B$8	=M32-L32+$B$8	
DD-08A	141.695334	13.359588		=B33-D$6	=C33-E$6	=D33-F$6	=E33-G$6	=F33+H$6	=G33+I$6	=H33+J$6	=I33+K$6	5.885	18.388	=M33-L33-$B$8	=M33-L33+$B$8	
	141.695334	13.359588		=B34-D$6	=C34-E$6	=D34-F$6	=E34-G$6	=F34+H$6	=G34+I$6	=H34+J$6	=I34+K$6	5.885	18.464	=M34-L34-$B$8	=M34-L34+$B$8	
DD-09A	179.907580	13.737205		=B35-D$6	=C35-E$6	=D35-F$6	=E35-G$6	=F35+H$6	=G35+I$6	=H35+J$6	=I35+K$6	6.469	18.124	=M35-L35-$B$8	=M35-L35+$B$8	
	179.907580	13.737205		=B36-D$6	=C36-E$6	=D36-F$6	=E36-G$6	=F36+H$6	=G36+I$6	=H36+J$6	=I36+K$6	5.563	18.522	=M36-L36-$B$8	=M36-L36+$B$8	
DD-10A	198.383940	20.511001		=B37-D$6	=C37-E$6	=D37-F$6	=E37-G$6	=F37+H$6	=G37+I$6	=H37+J$6	=I37+K$6	5.657	17.975	=M37-L37-$B$8	=M37-L37+$B$8	
	198.383940	20.511001		=B38-D$6	=C38-E$6	=D38-F$6	=E38-G$6	=F38+H$6	=G38+I$6	=H38+J$6	=I38+K$6	5.590	17.688	=M38-L38-$B$8	=M38-L38+$B$8	
DD-11A	202.059539	16.530987		=B39-D$6	=C39-E$6	=D39-F$6	=E39-G$6	=F39+H$6	=G39+I$6	=H39+J$6	=I39+K$6	5.367	16.629	=M39-L39-$B$8	=M39-L39+$B$8	
	202.059539	16.530987		=B40-D$6	=C40-E$6	=D40-F$6	=E40-G$6	=F40+H$6	=G40+I$6	=H40+J$6	=I40+K$6	5.606	17.741	=M40-L40-$B$8	=M40-L40+$B$8	
DD-12A	335.651282	-8.468889		=B41-D$6	=C41-E$6	=D41-F$6	=E41-G$6	=F41+H$6	=G41+I$6	=H41+J$6	=I41+K$6	5.297	16.669	=M41-L41-$B$8	=M41-L41+$B$8	
	335.651282	-8.468889		=B42-D$6	=C42-E$6	=D42-F$6	=E42-G$6	=F42+H$6	=G42+I$6	=H42+J$6	=I42+K$6	5.296	17.303	=M42-L42-$B$8	=M42-L42+$B$8	

```
SELECT P.ObjID as SDSSID, P.ra as RIGHT_ASCENSION, P.dec as DECLINATION, P.b as GALATIC_LATITUDE, P.l as GALATIC_LONGITUDE, P.psfmag_u as u, P.psfmag_g as g, P.psfmag_r as r,
FROM Star P JOIN properMotions U ON P.objID = U.objID
WHERE
  P.clean > 0
```

```
="   and (((p.ra < " & J19 & ") and (p.ra > " & F19 & ")) and ((p.dec < " & K19 & ") and (p.dec > " & G19 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J20 & ") and (p.ra > " & F20 & ")) and ((p.dec < " & K20 & ") and (p.dec > " & G20 & ")))"
="   and (((p.ra < " & J21 & ") and (p.ra > " & F21 & ")) and ((p.dec < " & K21 & ") and (p.dec > " & G21 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J22 & ") and (p.ra > " & F22 & ")) and ((p.dec < " & K22 & ") and (p.dec > " & G22 & ")))"
="   and (((p.ra < " & J23 & ") and (p.ra > " & F23 & ")) and ((p.dec < " & K23 & ") and (p.dec > " & G23 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J24 & ") and (p.ra > " & F24 & ")) and ((p.dec < " & K24 & ") and (p.dec > " & G24 & ")))"
="   and (((p.ra < " & J25 & ") and (p.ra > " & F25 & ")) and ((p.dec < " & K25 & ") and (p.dec > " & G25 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J26 & ") and (p.ra > " & F26 & ")) and ((p.dec < " & K26 & ") and (p.dec > " & G26 & ")))"
="   and (((p.ra < " & J27 & ") and (p.ra > " & F27 & ")) and ((p.dec < " & K27 & ") and (p.dec > " & G27 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J28 & ") and (p.ra > " & F28 & ")) and ((p.dec < " & K28 & ") and (p.dec > " & G28 & ")))"
="   and (((p.ra < " & J29 & ") and (p.ra > " & F29 & ")) and ((p.dec < " & K29 & ") and (p.dec > " & G29 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J30 & ") and (p.ra > " & F30 & ")) and ((p.dec < " & K30 & ") and (p.dec > " & G30 & ")))"
="   and (((p.ra < " & J31 & ") and (p.ra > " & F31 & ")) and ((p.dec < " & K31 & ") and (p.dec > " & G31 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J32 & ") and (p.ra > " & F32 & ")) and ((p.dec < " & K32 & ") and (p.dec > " & G32 & ")))"
="   and (((p.ra < " & J33 & ") and (p.ra > " & F33 & ")) and ((p.dec < " & K33 & ") and (p.dec > " & G33 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J34 & ") and (p.ra > " & F34 & ")) and ((p.dec < " & K34 & ") and (p.dec > " & G34 & ")))"
="   and (((p.ra < " & J35 & ") and (p.ra > " & F35 & ")) and ((p.dec < " & K35 & ") and (p.dec > " & G35 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J36 & ") and (p.ra > " & F36 & ")) and ((p.dec < " & K36 & ") and (p.dec > " & G36 & ")))"
="   and (((p.ra < " & J37 & ") and (p.ra > " & F37 & ")) and ((p.dec < " & K37 & ") and (p.dec > " & G37 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J38 & ") and (p.ra > " & F38 & ")) and ((p.dec < " & K38 & ") and (p.dec > " & G38 & ")))"
="   and (((p.ra < " & J39 & ") and (p.ra > " & F39 & ")) and ((p.dec < " & K39 & ") and (p.dec > " & G39 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J40 & ") and (p.ra > " & F40 & ")) and ((p.dec < " & K40 & ") and (p.dec > " & G40 & ")))"
="   and (((p.ra < " & J41 & ") and (p.ra > " & F41 & ")) and ((p.dec < " & K41 & ") and (p.dec > " & G41 & "))) and ((4.0 + 11.86*(P.psfmag_r - P.psfmag_i) - 10.74*(P.psfmag_r - P.psfmag_i)*(P.psfmag_
="   and (((p.ra < " & J42 & ") and (p.ra > " & F42 & ")) and ((p.dec < " & K42 & ") and (p.dec > " & G42 & ")))"
```

from baraffe 1998

V=551nm
I=806nm
SDSS g=475nm
SDSS i=763m

V-g = +76nm deviation
I-i = +43nm deviation

total deviation = V-g - I-i = +76 - +43 = 33nm
colour = V-I = 551nm - 806nm = 225nm

since deviation is small relative to colour photometric band measurement range (33 << 225), it is sufficient to use SDSS g-i as V-I.

min abs mag	max abs mag	diff
=#ref!62	=#ref!62	=V60-U60
V-I (g-i)	V-I (g-i)	
=#ref!62	=#ref!62	=V62-U62

gradient (V / V-I)	=W60/W62
max error in gradient	1

	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX
1	P.psfmag_i as i, P.psfmag_z as z, P.clean as pok, U.pmRa a				=$U1	=$U1	=$U1	=$U1	=$U1	=$U1	=$U1	=$U1	=$U1	=$U1	=$U1	=$U1
2					=$U2	=$U2	=$U2	=$U2	=$U2	=$U2	=$U2	=$U2	=$U2	=$U2	=$U2	=$U2
3					=$U3	=$U3	=$U3	=$U3	=$U3	=$U3	=$U3	=$U3	=$U3	=$U3	=$U3	=$U3
4					=$U4	=$U4	=$U4	=$U4	=$U4	=$U4	=$U4	=$U4	=$U4	=$U4	=$U4	=$U4
5					=$U5	=$U5	=$U5	=$U5	=$U5	=$U5	=$U5	=$U5	=$U5	=$U5	=$U5	=$U5
6					=$U6	=$U6	=$U6	=$U6	=$U6	=$U6	=$U6	=$U6	=$U6	=$U6	=$U6	=$U6
7					=$U7	=$U7	=$U7	=$U7	=$U7	=$U7	=$U7	=$U7	=$U7	=$U7	=$U7	=$U7
8					=$U8	=$U8	=$U8	=$U8	=$U8	=$U8	=$U8	=$U8	=$U8	=$U8	=$U8	=$U8
9					=$U9	=$U9	=$U9	=$U9	=$U9	=$U9	=$U9	=$U9	=$U9	=$U9	=$U9	=$U9
10					=$U10	=$U10	=$U10	=$U10	=$U10	=$U10	=$U10	=$U10	=$U10	=$U10	=$U10	=$U10
11					=$U11	=$U11	=$U11	=$U11	=$U11	=$U11	=$U11	=$U11	=$U11	=$U11	=$U11	=$U11
12					=$U12	=$U12	=$U12	=$U12	=$U12	=$U12	=$U12	=$U12	=$U12	=$U12	=$U12	=$U12
13					=$U13	=$U13	=$U13	=$U13	=$U13	=$U13	=$U13	=$U13	=$U13	=$U13	=$U13	=$U13
14					=$U14	=$U14	=$U14	=$U14	=$U14	=$U14	=$U14	=$U14	=$U14	=$U14	=$U14	=$U14
15					=$U15	=$U15	=$U15	=$U15	=$U15	=$U15	=$U15	=$U15	=$U15	=$U15	=$U15	=$U15
16					=$U16	=$U16	=$U16	=$U16	=$U16	=$U16	=$U16	=$U16	=$U16	=$U16	=$U16	=$U16
17					=$U17	=$U17	=$U17	=$U17	=$U17	=$U17	=$U17	=$U17	=$U17	=$U17	=$U17	=$U17
18					=U18	=$U18	=$U18	=$U18	=$U18	=$U18	=$U18	=$U18	=$U18	=$U18	=$U18	=$U18
19	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_				=U19											
20					=U20											
21	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i				=U21											
22						=U22										
23	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r						=U23									
24							=U24									
25	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)							=U25								
26								=U26								
27	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfm								=U27							
28									=U28							
29	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfm									=U29						
30										=U30						
31	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfmag_i)*(P.psfma										=U31					
32											=U32					
33	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag											=U33				
34												=U34				
35	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_												=U35			
36													=U36			
37	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i													=U37		
38														=U38		
39	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r														=U39	
40															=U40	
41	- P.psfmag_i) + 5.99*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i) - 1.20*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i)*(P.psfmag_r - P.psfmag_i))=U41															
42																=U42
43				DD System	1	2	3	4	5	6	7	8	9	10	11	12
44																
45																
46																
47																
48																
49																
50																
51																
52																
53																
54																
55																
56																
57																
58																
59																
60																
61																
62																
63																
64																
65																
66																

calcAveragePMofNearest150MDwarfObjects.pro

```
; Title: calculate Average Proper Motion of Nearest 150 M Dwarf Objects
; File: calcAveragePMofNearest150MDwarfObjects.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 7 Jan 11 2pm
; Updated: 1 Feb 11b 8pm [Added errors; to be considered a chance pm alignment, the MD has to be
    within the bounds of not only the rectange formed by the proper motions (plus the pm errors) of
     the two WD components
; Summary:
;
;   Postconditions;
;
;   This program calculates the Average Proper Motion of Nearest 150 M Dwarf Objects
;
;   For each component/system; NearestMDs-DDX.csv [SDSSID,RIGHT_ASCENSION,DECLINATION,
    GALATIC_LATITUDE,GALATIC_LONGITUDE,u,g,r,i,z,pok,RIGHT_ASCENSION, DECLINATION, PMRA, PMDEC,
    PMRAErr, PMDECErr]     [X = DD component/system index]
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   DDpropermotionsA/B.txt [RIGHT_ASCENSION,DECLINATION,PMRA,PMDEC,PMRAERR,PMDECErr] for X blue
    objects pair A/B components, tab delimited   [this list was generated using GAIA/Topcat]
;   NearestMDs-DDX.csv [SDSSID,RIGHT_ASCENSION,DECLINATION,GALATIC_LATITUDE,GALATIC_LONGITUDE,u,g,r,
    i,z,pok,PMRA,PMDEC,PMRAERR,PMDECErr,SIGRA,SIGDEC] for X DD components, comma delimited


minDistanceForConsideredNearestNeighborInArcMinutes = 5.0
maxDistanceForConsideredNearestNeighborInArcDegrees = 3.0
minDistanceForConsideredNearestNeighborInArcDegrees =
    minDistanceForConsideredNearestNeighborInArcMinutes/60D

;maxNumberOfNearestNeighbors = 150
maxNumberOfMatchesExpected = 2000

database1NameOfFile = 'DDpropermotionsA.txt'
readcol64bitvars, database1NameOfFile, RIGHT_ASCENSION_DB1A, DECLINATION_DB1A, PMRA_DB1A, PMDEC_DB1A
    , PMRAErr_DB1A, PMDECErr_DB1A, FORMAT='D,D,D,D,D,D'

database1BNameOfFile = 'DDpropermotionsB.txt'
readcol64bitvars, database1BNameOfFile, RIGHT_ASCENSION_DB1B, DECLINATION_DB1B, PMRA_DB1B,
    PMDEC_DB1B, PMRAErr_DB1B, PMDECErr_DB1B, FORMAT='D,D,D,D,D,D'

database1starXCoordinates = transpose(RIGHT_ASCENSION_DB1A)
database1starYCoordinates = transpose(DECLINATION_DB1A)

database1fieldNumberOfStars = (size(RIGHT_ASCENSION_DB1A))[1]

;PRINT, "database1fieldNumberOfStars = ", database1fieldNumberOfStars

PRINT, "printing in the following format; "
PRINT, "properMotionDDRA, properMotionDDDec, properMotion150NearestMDwarfObjectsRAaverage,
    properMotion150NearestMDwarfObjectsDECaverage, probabilityOfChanceCommonProperMotion"

for database1starIndex=0L, database1fieldNumberOfStars-1L, 1L do begin

    ; calculate max deviation between DD-xA and DD-xB
    ;to calculate number of nearest Neighbor M Dwarfs That Would Have Been Classified As Common
        Proper Motion Systems
    DDsystemMaxRAErr = MAX([PMRAErr_DB1B[database1starIndex], PMRAErr_DB1A[database1starIndex]])
    DDsystemMaxDecErr = MAX([PMDECErr_DB1B[database1starIndex], PMDECErr_DB1A[database1starIndex]])
    DDsystemMaxRA = MAX([PMRA_DB1B[database1starIndex], PMRA_DB1A[database1starIndex]]) +
        DDsystemMaxRAErr
    DDsystemMinRA = MIN([PMRA_DB1B[database1starIndex], PMRA_DB1A[database1starIndex]]) -
        DDsystemMaxRAErr
    DDsystemMaxDec = MAX([PMDEC_DB1B[database1starIndex], PMDEC_DB1A[database1starIndex]]) +
        DDsystemMaxDecErr
    DDsystemMinDec = MIN([PMDEC_DB1B[database1starIndex], PMDEC_DB1A[database1starIndex]]) -
        DDsystemMaxDecErr
    numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems = 0
    ;PRINT, "DDsystemMaxRA = ", DDsystemMaxRA
    ;PRINT, "DDsystemMinRA = ", DDsystemMinRA
    ;PRINT, "DDsystemMaxDec = ", DDsystemMaxDec
    ;PRINT, "DDsystemMinDec = ", DDsystemMinDec


    ;maximumRAdeviationBetweenDDcomponentPropermotions = PMRA_DB1B[database1starIndex] - PMRA_DB1A[
        database1starIndex]
    ;maximumDECdeviationBetweenDDcomponentPropermotions =  PMDEC_DB1B[database1starIndex] -
        PMDEC_DB1A[database1starIndex]
```

```
DDnumber = database1starIndex+1

database2NameOfFilePartA = 'NearestMDs-DD'
database2NameOfFilePartC = '.csv'
database2NameOfFile = database2NameOfFilePartA + STRTRIM(DDnumber,2) + database2NameOfFilePartC

;PRINT, "database2NameOfFile = ", database2NameOfFile

readcol64bitvars, database2NameOfFile, ID, RIGHT_ASCENSION_DB2, DECLINATION_DB2,
     GALATIC_LATITUDE, GALATIC_LONGITUDE, U, G, R, I, Z, POK, PMRA_DB2, PMDEC_DB2, PMRAERR_DB2,
     PMDECErr_DB2, SIGRA_DB2, SIGDEC_DB2, FORMAT='G,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D', SKIPLINE
     =1, /SILENT

database2starXCoordinates = transpose(RIGHT_ASCENSION_DB2)
database2starYCoordinates = transpose(DECLINATION_DB2)

database2fieldNumberOfStars = (size(RIGHT_ASCENSION_DB2))[1]
;PRINT, "database2fieldNumberOfStars = ", database2fieldNumberOfStars

properMotionRAaverage = 0.0
properMotionDecaverage = 0.0

nearestNeighborIndex = 0

for database2starIndex=0L, database2fieldNumberOfStars-1L, 1L do begin

    currentMaxDistanceBetweenMatchX = ABS(database1starXCoordinates[database1starIndex] -
        database2starXCoordinates[database2starIndex])
    currentMaxDistanceBetweenMatchY = ABS(database1starYCoordinates[database1starIndex] -
        database2starYCoordinates[database2starIndex])

    averageYDeclination = (database1starYCoordinates[database1starIndex] +
        database2starYCoordinates[database2starIndex])/2
    currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
        currentMaxDistanceBetweenMatchX
    currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenMatchY

    currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
        currentMaxDistanceBetweenPairYCorrected^2)

    ;PRINT, "database1starXCoordinates[database1starIndex] = ", database1starXCoordinates[
        database1starIndex]
    ;PRINT, "database2starXCoordinates[database2starIndex] = ", database2starXCoordinates[
        database2starIndex]

    ;PRINT, "currentMaxDistanceBetweenPair = ", currentMaxDistanceBetweenPair
    ;PRINT, "maxDistanceForConsideredNearestNeighborInArcDegrees = ",
        maxDistanceForConsideredNearestNeighborInArcDegrees

    properMotionRAaverage = properMotionRAaverage + PMRA_DB2[database2starIndex]
    properMotionDecaverage = properMotionDecaverage + PMDEC_DB2[database2starIndex]

    ;PRINT, "PMRA_DB2[database2starIndex] = ", PMRA_DB2[database2starIndex]
    ;PRINT, "PMDEC_DB2[database2starIndex]   = ", PMDEC_DB2[database2starIndex]

    if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredNearestNeighborInArcDegrees)
        then begin

        if (currentMaxDistanceBetweenPair gt minDistanceForConsideredNearestNeighborInArcDegrees
            ) then begin

            nearestNeighborIndex = nearestNeighborIndex + 1

            ;to calculate number of nearest Neighbor M Dwarfs That Would Have Been Classified As
                Common Proper Motion Systems
            if (((PMRA_DB2[database2starIndex] lt DDsystemMaxRA) AND (PMRA_DB2[
                database2starIndex] gt DDsystemMinRA)) AND ((PMDEC_DB2[database2starIndex] lt
                DDsystemMaxDec) AND (PMDEC_DB2[database2starIndex] gt DDsystemMinDec)))then
                begin
                numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems
                    =
                    numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems
                    + 1
            endif

        endif

    endif

endfor

properMotionRAaverage = properMotionRAaverage/database2fieldNumberOfStars
properMotionDecaverage = properMotionDecaverage/database2fieldNumberOfStars

properMotionDDRA = PMRA_DB1A[database1starIndex]
properMotionDDDec = PMDEC_DB1A[database1starIndex]

    ;PRINT, "properMotionRAaverage = ", properMotionRAaverage
    ;PRINT, "properMotionDecaverage = ", properMotionDecaverage
```

```
    ;to calculate number of nearest Neighbor M Dwarfs That Would Have Been Classified As Common
        Proper Motion Systems
    probabilityOfChanceCommonProperMotion =  double(
        numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems)/
        double(database2fieldNumberOfStars)
    ;PRINT, "numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems =
        ", numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems
    ;PRINT, "database2fieldNumberOfStars = ", database2fieldNumberOfStars
    ;PRINT, "probability of chance common proper motion (using this method) = ",
        probabilityOfChanceCommonProperMotion


    ;PRINT, "properMotion150NearestMDwarfObjectsRAaverage,
        properMotion150NearestMDwarfObjectsDECaverage:"
    ;PRINT, [properMotionRAaverage, properMotionDecaverage]
    ;PRINT, "properMotionDDRA, properMotionDDDec:"
    ;PRINT, [properMotionDDRA, properMotionDDDec]

    ;DDname = "DD-" + STRTRIM(DDnumber,2)
    ;PRINT, DDname

    PRINT, [properMotionDDRA, properMotionDDDec, properMotionRAaverage, properMotionDecaverage,
        probabilityOfChanceCommonProperMotion]

endfor


end
```

findNearest150WDObjects-usingEisensteinSDSSDR4WDData.pro

```
; Title: Blue Object Close Pair List - create Subset using Eisenstein SDSS DR4 WD Data
; File: BOCPLcreateSubsetusingEisensteinSDSSDR4WDData.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 8 April 09 3pm
; Summary:
;
;   Postconditions;
;
;   This program compares RA and Dec data from rows 49 and 50 [48 and 49] of
    EisensteinSDSSDR4WDDataSmall.txt to find matching stars with RA and Dec data from rows 10 and
    11 [9 and 10] of BOCPLPart1.txt data
;
;   This program generates a list of blue object pairs from database 1 which match with objects from
     database 2 within a given angular separation. The output files are of the following format;
;
;   For each component/system; BOCPLEisensteinABNearestWDsX.txt [RIGHT_ASCENSION A, DECLINATION A,
    proper motion magnitude, proper motion angle]     [X = component/system index]

;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   DDtargetList.txt [RIGHT_ASCENSION A, DECLINATION A] for X blue objects Pairs, tab delimited
;   BOCPLEisensteinSDSSDR4WDDataSmallWithProperMotions.txt [RIGHT_ASCENSION A, DECLINATION A, proper
     motion magnitude, proper motion angle]
;   SDSSblueObjectsDR7detailedWithProperMotions.txt [ID, RIGHT_ASCENSION, DECLINATION,
    GALATIC_LATITUDE, GALATIC_LONGITUDE, ID, U, G, R, POK, PMRA, PMDEC, PMRAERR, PMDECErr, SIGRA,
    SIGDEC] for ~56587 blue objects, tab delimited


minDistanceForConsideredNearestNeighborInArcMinutes = 5.0
maxDistanceForConsideredNearestNeighborInArcDegrees = 10.0
minDistanceForConsideredNearestNeighborInArcDegrees =
    minDistanceForConsideredNearestNeighborInArcMinutes/60D

;maxNumberOfNearestNeighbors = 150
maxNumberOfMatchesExpected = 2000

maxDistanceForMatchedStarInArcSeconds = double(0.1)
maxDistanceForMatchedStarInArcDegrees = maxDistanceForMatchedStarInArcSeconds/60D/60D

database1NameOfFile = 'DDtargetList.txt'
database2NameOfFile = 'BOCPLEisensteinSDSSDR4WDDataSmallWithProperMotions.txt'
database3NameOfFile = 'SDSSblueObjectsDR7detailedWithProperMotions.txt'

    ;long precision is required for ID
readcol64bitvars, database1NameOfFile, RIGHT_ASCENSION_DB1, DECLINATION_DB1, FORMAT='D,D'
readcol64bitvars, database2NameOfFile, RIGHT_ASCENSION_DB2, DECLINATION_DB2, PROPERMOTION_MAG_DB2,
    PROPERMOTION_ANGLE_DB2, FORMAT='D,D'
```

```
readcol64bitvars, database3NameOfFile, ID, RIGHT_ASCENSION_DB3, DECLINATION_DB3, GALATIC_LATITUDE,
     GALATIC_LONGITUDE, U, G, R, POK, PMRA, PMDEC, PMRAERR, PMDECErr, SIGRA, SIGDEC, FORMAT='G,D,D,D
     ,D,D,D,D,D,D,D,D,D,D,D,D,D', SKIPLINE=1

database1starXCoordinates = transpose(RIGHT_ASCENSION_DB1)
database1starYCoordinates = transpose(DECLINATION_DB1)
database2starXCoordinates = transpose(RIGHT_ASCENSION_DB2)
database2starYCoordinates = transpose(DECLINATION_DB2)
database3starXCoordinates = transpose(RIGHT_ASCENSION_DB3)
database3starYCoordinates = transpose(DECLINATION_DB3)
;database2starData = [transpose(RIGHT_ASCENSION_DB2), transpose(DECLINATION_DB2), transpose(
     PROPERMOTION_MAG_DB2), transpose(PROPERMOTION_ANGLE_DB2)]
database3starData = [transpose(RIGHT_ASCENSION_DB3), transpose(DECLINATION_DB3), transpose(
     GALATIC_LATITUDE), transpose(GALATIC_LONGITUDE), transpose(U), transpose(G), transpose(R),
     transpose(POK), transpose(PMRA), transpose(PMDEC), transpose(PMRAERR), transpose(PMDECErr),
     transpose(SIGRA), transpose(SIGDEC)]
database3starID = ID

database1fieldNumberOfStars = (size(RIGHT_ASCENSION_DB1))[1]
database2fieldNumberOfStars = (size(RIGHT_ASCENSION_DB2))[1]
database3fieldNumberOfStars = (size(RIGHT_ASCENSION_DB3))[1]
    PRINT, "database1fieldNumberOfStars = ", database1fieldNumberOfStars
    PRINT, "database2fieldNumberOfStars = ", database2fieldNumberOfStars
    PRINT, "database3fieldNumberOfStars = ", database3fieldNumberOfStars


;averageAnglePart1
;totalAnglePart1

PRINT, "generating nearest neighbor files and printing proper motions [Ra Dec]..."

for database1starIndex=0L, database1fieldNumberOfStars-1L, 1L do begin

    nearestNeighborData = dblarr(14, maxNumberOfMatchesExpected)
    nearestNeighborID = lon64arr(maxNumberOfMatchesExpected)

    nearestNeighborIndex = 0

    properMotionRAaverage = 0.0
    properMotionDecaverage = 0.0

    for database2starIndex=0L, database2fieldNumberOfStars-1L, 1L do begin

        currentMaxDistanceBetweenMatchX = ABS(database1starXCoordinates[database1starIndex] -
             database2starXCoordinates[database2starIndex])
        currentMaxDistanceBetweenMatchY = ABS(database1starYCoordinates[database1starIndex] -
             database2starYCoordinates[database2starIndex])

        averageYDeclination = (database1starYCoordinates[database1starIndex] +
             database2starYCoordinates[database2starIndex])/2
        currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
             currentMaxDistanceBetweenMatchX
        currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenMatchY

        currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
             currentMaxDistanceBetweenPairYCorrected^2)

        if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredNearestNeighborInArcDegrees)
             then begin

            if (currentMaxDistanceBetweenPair gt minDistanceForConsideredNearestNeighborInArcDegrees
                 ) then begin

                        ;nearestNeighborData[*, nearestNeighborIndex] = database2starData[*,
                             database2starIndex]

                numberMatchesFound = 0
                ;now extract DR7 proper motions from
                     BOCPLSDSSblueObjectsDR7detailedWithProperMotions.txt;

                for database3starIndex=0L, database3fieldNumberOfStars-1L, 1L do begin

                    ;PRINT, "database3starIndex = ", database3starIndex
                    currentMaxDistanceBetweenMatchX = ABS(database2starXCoordinates[
                         database2starIndex] - database3starXCoordinates[database3starIndex])
                    currentMaxDistanceBetweenMatchY = ABS(database2starYCoordinates[
                         database2starIndex] - database3starYCoordinates[database3starIndex])

                    averageYDeclination = (database2starYCoordinates[database2starIndex] +
                         database3starYCoordinates[database3starIndex])/2

                    currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
                         currentMaxDistanceBetweenMatchX
                    currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenMatchY

                    currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
                          currentMaxDistanceBetweenPairYCorrected^2)

                    if (currentMaxDistanceBetweenPair lt maxDistanceForMatchedStarInArcDegrees) then
                          begin
```

```
                        nearestNeighborData[*, nearestNeighborIndex] = database3starData[*,
                             database3starIndex]
                        nearestNeighborID[nearestNeighborIndex] = database3starID[database3starIndex
                             ]

                            ;PRINT, "database3starIndex = ", database3starIndex
                            ;PRINT, "currentMaxDistanceBetweenPair = ",
                                 currentMaxDistanceBetweenPair

                                ;error checking ... [NB one error occurs for DD11, but otherwise
                                     everything is OK]
                            numberMatchesFound = numberMatchesFound + 1
                            if(numberMatchesFound gt 1) then begin
                                ;PRINT, "ERROR; numberMatchesFound gt 1"
                                ;exit
                            endif else begin
                                ;PRINT, "nearestNeighborData[*, nearestNeighborIndex] = ",
                                     nearestNeighborData[*, nearestNeighborIndex]

                                nearestNeighborIndex = nearestNeighborIndex +1

                                properMotionRAaverage = properMotionRAaverage + PMRA[
                                     database3starIndex]
                                properMotionDecaverage = properMotionDecaverage + PMDEC[
                                     database3starIndex]
                            endelse

                        endif
                    endfor

                    ;[NB no match occurs for ~50% of objects; is this because Eisenstein WD database
                         covers whole sky not just SDSS region?]
                    if(numberMatchesFound lt 1) then begin
                        ;PRINT, "no SDSS match found for this Eisenstein WD object..."
                        ;PRINT, [database2starXCoordinates(database2starIndex),
                             database2starYCoordinates(database2starIndex)]
                    endif


                endif

            endif

        endfor

        properMotionRAaverage = properMotionRAaverage/nearestNeighborIndex
        properMotionDecaverage = properMotionDecaverage/nearestNeighborIndex
            ;PRINT, "properMotionRAaverage = ", properMotionRAaverage
            ;PRINT, "properMotionDecaverage = ", properMotionDecaverage
        PRINT, [properMotionRAaverage, properMotionDecaverage]

        ;NB sorting is not necessary as the input database1 is assumed already to be sorted wrt distance
             between pairs
        nearestNeighborData = nearestNeighborData[*, 0:(nearestNeighborIndex-1)]
        nearestNeighborID = nearestNeighborID[0:(nearestNeighborIndex-1)]

                ;ABstarDistanceArrayForMatchesSortIndicies = sort(ABstarDistanceArrayForMatches)
                ;ABstarIDArrayForMatchesSorted = ABstarIDArrayForMatches[*,
                     ABstarDistanceArrayForMatchesSortIndicies]

        ;PRINT, "nearestNeighborData = "
        ;PRINT, nearestNeighborData

        outputFileName='BOCPLEisensteinABNearestWDsPart1-DD' + STRTRIM((database1starIndex+1),2) + '.txt
             '
        ;PRINT, "outputFileName = ", outputFileName
        OPENW,1,outputFileName
        PRINTF,1,nearestNeighborData,FORMAT='(F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1
             X,F,1X,F)'
        CLOSE,1

        outputFileName='BOCPLEisensteinABNearestWDsPart2-DD' + STRTRIM((database1starIndex+1),2) + '.txt
             '
        ;PRINT, "outputFileName = ", outputFileName
        OPENW,1,outputFileName
        PRINTF,1,nearestNeighborID,FORMAT='(I18)'
        CLOSE,1

endfor


end
```

calcAveragePMofNearest150WDwarfObjects.pro

```
; Title: calculate Average Proper Motion of Nearest 150 W Dwarf Objects
; File: calcAveragePMofNearest150WDwarfObjects.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 19 May 11 9am [based on calcAveragePMofNearest150MDwarfObjects.pro]
```

```
; Summary:
;
;    Postconditions;
;
;    This program calculates the Average Proper Motion of Nearest 150 M Dwarf Objects
;
;    For each component/system; NearestMDs-DDX.csv [SDSSID,RIGHT_ASCENSION,DECLINATION,
     GALATIC_LATITUDE,GALATIC_LONGITUDE,u,g,r,i,z,pok,RIGHT_ASCENSION, DECLINATION, PMRA, PMDEC,
     PMRAErr, PMDECErr]     [X = DD component/system index]
;
;    Preconditions
;
;    The input files are of the following format;
;
;    repchr.pro [NASA]
;    gettok.pro [NASA]
;    readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;    remchar.pro [NASA]
;    skip_lun.pro [required for GDL]
;    strsplit.pro [GDL library]
;    strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;    DDpropermotionsA/B.txt [RIGHT_ASCENSION,DECLINATION,PMRA,PMDEC,PMRAERR,PMDECErr] for X blue
     objects pair A/B components, tab delimited  [this list was generated using GAIA/Topcat]
;    BOCPLEisensteinABNearestWDsPart1-DDX.csv [RIGHT_ASCENSION,DECLINATION,GALATIC_LATITUDE,
     GALATIC_LONGITUDE,u,g,r,pok,PMRA,PMDEC,PMRAERR,PMDECErr,SIGRA,SIGDEC] for X DD components,
     space delimited


minDistanceForConsideredNearestNeighborInArcMinutes = 5.0
maxDistanceForConsideredNearestNeighborInArcDegrees = 3.0
minDistanceForConsideredNearestNeighborInArcDegrees =
     minDistanceForConsideredNearestNeighborInArcMinutes/60D

;maxNumberOfNearestNeighbors = 150
maxNumberOfMatchesExpected = 2000

database1NameOfFile = 'DDpropermotionsA.txt'
readcol64bitvars, database1NameOfFile, RIGHT_ASCENSION_DB1A, DECLINATION_DB1A, PMRA_DB1A, PMDEC_DB1A
     , PMRAErr_DB1A, PMDECErr_DB1A, FORMAT='D,D,D,D,D,D'

database1BNameOfFile = 'DDpropermotionsB.txt'
readcol64bitvars, database1BNameOfFile, RIGHT_ASCENSION_DB1B, DECLINATION_DB1B, PMRA_DB1B,
     PMDEC_DB1B, PMRAErr_DB1B, PMDECErr_DB1B, FORMAT='D,D,D,D,D,D'

database1starXCoordinates = transpose(RIGHT_ASCENSION_DB1A)
database1starYCoordinates = transpose(DECLINATION_DB1A)

database1fieldNumberOfStars = (size(RIGHT_ASCENSION_DB1A))[1]

;PRINT, "database1fieldNumberOfStars = ", database1fieldNumberOfStars

PRINT, "printing in the following format; "
PRINT, "properMotionDDRA, properMotionDDDec, properMotion150NearestMDwarfObjectsRAaverage,
     properMotion150NearestMDwarfObjectsDECaverage, probabilityOfChanceCommonProperMotion"

for database1starIndex=0L, database1fieldNumberOfStars-1L, 1L do begin

     ; calculate max deviation between DD-xA and DD-xB
     ;to calculate number of nearest Neighbor M Dwarfs That Would Have Been Classified As Common
          Proper Motion Systems
     DDsystemMaxRAErr = MAX([PMRAErr_DB1B[database1starIndex], PMRAErr_DB1A[database1starIndex]])
     DDsystemMaxDecErr = MAX([PMDECErr_DB1B[database1starIndex], PMDECErr_DB1A[database1starIndex]])
     DDsystemMaxRA = MAX([PMRA_DB1B[database1starIndex], PMRA_DB1A[database1starIndex]]) +
          DDsystemMaxRAErr
     DDsystemMinRA = MIN([PMRA_DB1B[database1starIndex], PMRA_DB1A[database1starIndex]]) -
          DDsystemMaxRAErr
     DDsystemMaxDec = MAX([PMDEC_DB1B[database1starIndex], PMDEC_DB1A[database1starIndex]]) +
          DDsystemMaxDecErr
     DDsystemMinDec = MIN([PMDEC_DB1B[database1starIndex], PMDEC_DB1A[database1starIndex]]) -
          DDsystemMaxDecErr
     numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems = 0
     ;PRINT, "DDsystemMaxRA = ", DDsystemMaxRA
     ;PRINT, "DDsystemMinRA = ", DDsystemMinRA
     ;PRINT, "DDsystemMaxDec = ", DDsystemMaxDec
     ;PRINT, "DDsystemMinDec = ", DDsystemMinDec


     ;maximumRAdeviationBetweenDDcomponentPropermotions = PMRA_DB1B[database1starIndex] - PMRA_DB1A[
          database1starIndex]
     ;maximumDECdeviationBetweenDDcomponentPropermotions =  PMDEC_DB1B[database1starIndex] -
          PMDEC_DB1A[database1starIndex]

     DDnumber = database1starIndex+1

     database2NameOfFilePartA = 'BOCPLEisensteinABNearestWDsPart1-DD'
     database2NameOfFilePartC = '.txt'
     database2NameOfFile = database2NameOfFilePartA + STRTRIM(DDnumber,2) + database2NameOfFilePartC

     ;PRINT, "database2NameOfFile = ", database2NameOfFile
```

```
readcol64bitvars, database2NameOfFile, RIGHT_ASCENSION_DB2, DECLINATION_DB2, GALATIC_LATITUDE,
     GALATIC_LONGITUDE, U, G, R, POK, PMRA_DB2, PMDEC_DB2, PMRAERR_DB2, PMDECErr_DB2, SIGRA_DB2,
      SIGDEC_DB2, FORMAT='D,D,D,D,D,D,D,D,D,D,D,D,D', /SILENT

database2starXCoordinates = transpose(RIGHT_ASCENSION_DB2)
database2starYCoordinates = transpose(DECLINATION_DB2)

database2fieldNumberOfStars = (size(RIGHT_ASCENSION_DB2))[1]
;PRINT, "database2fieldNumberOfStars = ", database2fieldNumberOfStars

properMotionRAaverage = 0.0
properMotionDecaverage = 0.0

nearestNeighborIndex = 0

for database2starIndex=0L, database2fieldNumberOfStars-1L, 1L do begin

    currentMaxDistanceBetweenMatchX = ABS(database1starXCoordinates[database1starIndex] -
         database2starXCoordinates[database2starIndex])
    currentMaxDistanceBetweenMatchY = ABS(database1starYCoordinates[database1starIndex] -
         database2starYCoordinates[database2starIndex])

    averageYDeclination = (database1starYCoordinates[database1starIndex] +
         database2starYCoordinates[database2starIndex])/2
    currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
         currentMaxDistanceBetweenMatchX
    currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenMatchY

    currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
         currentMaxDistanceBetweenPairYCorrected^2)

    ;PRINT, "database1starXCoordinates[database1starIndex] = ", database1starXCoordinates[
         database1starIndex]
    ;PRINT, "database2starXCoordinates[database2starIndex] = ", database2starXCoordinates[
         database2starIndex]

    ;PRINT, "currentMaxDistanceBetweenPair = ", currentMaxDistanceBetweenPair
    ;PRINT, "maxDistanceForConsideredNearestNeighborInArcDegrees = ",
         maxDistanceForConsideredNearestNeighborInArcDegrees

    ;PRINT, "PMRA_DB2[database2starIndex] = ", PMRA_DB2[database2starIndex]
    ;PRINT, "PMDEC_DB2[database2starIndex]    = ", PMDEC_DB2[database2starIndex]

    properMotionRAaverage = properMotionRAaverage + PMRA_DB2[database2starIndex]
    properMotionDecaverage = properMotionDecaverage + PMDEC_DB2[database2starIndex]

    ;PRINT, "properMotionRAaverage = ", properMotionRAaverage
    ;PRINT, "properMotionDecaverage = ", properMotionDecaverage

    if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredNearestNeighborInArcDegrees)
         then begin

        if (currentMaxDistanceBetweenPair gt minDistanceForConsideredNearestNeighborInArcDegrees
             ) then begin

            nearestNeighborIndex = nearestNeighborIndex + 1

            ;to calculate number of nearest Neighbor M Dwarfs That Would Have Been Classified As
                 Common Proper Motion Systems
            if (((PMRA_DB2[database2starIndex] lt DDsystemMaxRA) AND (PMRA_DB2[
                 database2starIndex] gt DDsystemMinRA)) AND ((PMDEC_DB2[database2starIndex] lt
                 DDsystemMaxDec) AND (PMDEC_DB2[database2starIndex] gt DDsystemMinDec)))then
                 begin
                numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems
                     =
                    numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems
                     + 1
            endif

        endif

    endif

endfor

properMotionRAaverage = properMotionRAaverage/database2fieldNumberOfStars
properMotionDecaverage = properMotionDecaverage/database2fieldNumberOfStars

properMotionDDRA = PMRA_DB1A[database1starIndex]
properMotionDDDec = PMDEC_DB1A[database1starIndex]

    ;PRINT, "properMotionRAaverage = ", properMotionRAaverage
    ;PRINT, "properMotionDecaverage = ", properMotionDecaverage


;to calculate number of nearest Neighbor M Dwarfs That Would Have Been Classified As Common
     Proper Motion Systems
probabilityOfChanceCommonProperMotion =  double(
     numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems)/
     double(database2fieldNumberOfStars)
;PRINT, "numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems =
     ", numberOfnearestNeighborMDwarfsThatWouldHaveBeenClassifiedAsCommonProperMotionSystems
```

```
    ;PRINT, "database2fieldNumberOfStars = ", database2fieldNumberOfStars
    ;PRINT, "probability of chance common proper motion (using this method) = ",
        probabilityOfChanceCommonProperMotion


    ;PRINT, "properMotion150NearestMDwarfObjectsRAaverage,
        properMotion150NearestMDwarfObjectsDECaverage:"
    ;PRINT, [properMotionRAaverage, properMotionDecaverage]
    ;PRINT, "properMotionDDRA, properMotionDDDec:"
    ;PRINT, [properMotionDDRA, properMotionDDDec]

    ;DDname = "DD-" + STRTRIM(DDnumber,2)
    ;PRINT, DDname

    PRINT, [properMotionDDRA, properMotionDDDec, properMotionRAaverage, properMotionDecaverage,
        probabilityOfChanceCommonProperMotion]

endfor


end
```

A.3.5 Calculation Probability of Chance Proper Motion Alignment - RBB Method

calculateRBBprobabilityOfChanceAlignment-13April11a.zip

calculateRBBprobabilityOfChanceAlignment.pro

```
; Title: calculate RBB probability Of Chance Alignment
; File: calculateRBBprobabilityOfChanceAlignment.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 14 April 11a 5am
; Summary:
;
;   Postconditions;
;
;   This program plots an ASCII Table Using GDL
;
;       The output files are of the following format;
;
;   graph.eps
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   plotASCIITableUsingGDL.pro [NASA]
;   asciiTable.txt [X Axis, Y Axis 1, Y Axis 2] for X variables, tab delimited


pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black      : 0, $
                white      : 1, $
                red        : 2, $
                green      : 3, $
                blue       : 4, $
                yellow     : 5, $
                magenta    : 6, $
                cyan       : 7, $
                orange     : 8, $
                l_green    : 9, $
                violet     : 10,$
                turquoise  : 11,$
                l_blue     : 12,$
                l_red      : 13 $
               }
;         0  1    2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]    ;blue
end
```

```
my_colors


;plotTitle = 'Number of BOCPs'
;plotXAxisTitle = 'Max Separation Between Pairs (Arc Seconds)'
;plotYAxisTitle = 'Number of Blue Object Pairs'
;plotXmin = 0
;plotXmax = 200
;plotYmin = 0.0
;plotYmax = 2000
;asciiTablefileName = 'CCPtestsweepAndCCPsweepResults2-1.txt'
;psGraphfileName = 'CCPtestsweepAndCCPsweepResults2-1.ps'
;pdfGraphfileName = 'CCPtestsweepAndCCPsweepResults2-1.pdf'
;headerNumLines = 2
;multipleFiles = 0
;numYcols = 2
;numPlots=numYcols
;lineStyleY = intarr(numPlots)
;lineStyleY[0] = 3
;lineStyleY[1] = 0
;psymY = intarr(numPlots)
;psymY[0]=-0
;psymY[1]=-0
;normalise = 0
;specifyYRange = 1
;plotErrorBars = 0

;plotTitle = 'Number of BOCPs'
;plotXAxisTitle = 'Max Separation Between Pairs (Arc Seconds)'
;plotYAxisTitle = 'Number of Blue Object Pairs'
;plotXmin = 0
;plotXmax = 200
;plotYmin = 0.0
;plotYmax = 2000
;asciiTablefileName = 'CCPtestsweepAndCCPsweepResults2-2.txt'
;psGraphfileName = 'CCPtestsweepAndCCPsweepResults2-2.ps'
;pdfGraphfileName = 'CCPtestsweepAndCCPsweepResults2-2.pdf'
;headerNumLines = 2
;multipleFiles = 0
;numYcols = 2
;numPlots=numYcols
;lineStyleY = intarr(numPlots)
;lineStyleY[0] = 3
;lineStyleY[1] = 0
;psymY = intarr(numPlots)
;psymY[0]=-0
;psymY[1]=-0
;normalise = 0
;specifyYRange = 1
;plotErrorBars = 0

;plotTitle = 'Number of BOCPs'
;plotXAxisTitle = 'Max Separation Between Pairs (Arc Seconds)'
;plotYAxisTitle = 'Estimated number binaries'
;plotXmin = 0
;plotXmax = 200
;plotYmin = 0.0
;plotYmax = 200
;asciiTablefileName = 'CCPtestsweepAndCCPsweepResults2-3.txt'
;psGraphfileName = 'CCPtestsweepAndCCPsweepResults2-3.ps'
;pdfGraphfileName = 'CCPtestsweepAndCCPsweepResults2-3.pdf'
;headerNumLines = 2
;multipleFiles = 0
;numYcols = 1
;numPlots=numYcols
;lineStyleY = intarr(numPlots)
;lineStyleY[0] = 0
;psymY = intarr(numPlots)
;psymY[0]=-0
;normalise = 0
;specifyYRange = 1
;plotErrorBars = 0

;plotTitle = 'Number of BOCPs'
;plotXAxisTitle = 'Max Separation Between Pairs (Arc Seconds)'
;plotYAxisTitle = 'probability real binary detection (percentage)'
;plotXmin = 0
;plotXmax = 200
;plotYmin = 0.0
;plotYmax = 100
;asciiTablefileName = 'CCPtestsweepAndCCPsweepResults2-4.txt'
;psGraphfileName = 'CCPtestsweepAndCCPsweepResults2-4.ps'
;pdfGraphfileName = 'CCPtestsweepAndCCPsweepResults2-4.pdf'
;headerNumLines = 2
;multipleFiles = 0
;numYcols = 1
;numPlots=numYcols
;lineStyleY = intarr(numPlots)
;lineStyleY[0] = 0
;psymY = intarr(numPlots)
;psymY[0]=-0
```

```
;normalise = 0
;specifyYRange = 1
;plotErrorBars = 0

plotTitle = 'Proper Motions within Field'
plotXAxisTitle = 'X displacement from mean proper motion'
plotYAxisTitle = 'Y displacement from mean proper motion'
plotXmin = -10
plotXmax = 10
plotYmin = -10
plotYmax = 10
fileNameBase1 = 'DD'
fileNameBase2 = 'xy2xy'
asciiTablefileNameExtension = '.log'
psGraphfileNameExtension = '.ps'
pdfGraphfileNameExtension = '.pdf'
headerNumLines = 0
multipleFiles = 1
numberFiles = 12
multipleSubFiles = 1
numberSubFiles = 2
numYcols = 1
numPlots=numberSubFiles
lineStyleY = intarr(numPlots)
lineStyleY[0] = 0
lineStyleY[1] = 0
psymY = intarr(numPlots)
psymY[0]=1
psymY[1]=2
normalise = 0
specifyYRange = 1
plotErrorBars = 1
properMotionCalculations = 1

;plotTitle = 'Spectra Flux'
;plotXAxisTitle = 'Wavelength (Angstroms)'
;plotYAxisTitle = 'Amplitude'
;plotXmin = 3600
;plotXmax = 5200
;plotYmin = 0
;plotYmax = 1.5
;fileNameBase1 = 'SSWD'
;fileNameBase2 = 'B'     ;change to B for other half
;asciiTablefileNameExtension = '.dat'
;psGraphfileNameExtension = '.ps'
;pdfGraphfileNameExtension = '.pdf'
;headerNumLines = 0
;multipleFiles = 1
;numberFiles = 12
;multipleSubFiles = 0
;numberSubFiles = 1
;numYcols = 1
;numPlots=numberSubFiles
;lineStyleY = intarr(numPlots)
;lineStyleY[0] = 0
;psymY = intarr(numPlots)
;psymY[0]=-0
;normalise = 1
;specifyYRange = 0
;plotErrorBars = 0


;plotTitle = 'Mass Distribution (SDSS DR4 WD mass distribution versus DD mass distribution)'
;plotXAxisTitle = 'WD mass'
;plotYAxisTitle = 'frequency'
;plotXmin = 0.2
;plotXmax = 1.5
;plotYmin = 0
;plotYmax = 1.0
;fileNameBase1 = 'WDversusDDmassDistribution'
;fileNameBase2 = ''
;asciiTablefileNameExtension = '.txt'
;psGraphfileNameExtension = '.ps'
;pdfGraphfileNameExtension = '.pdf'
;headerNumLines = 2
;multipleFiles = 1
;numberFiles = 1
;multipleSubFiles = 1
;numberSubFiles = 2
;numYcols = 1
;numPlots=numberSubFiles
;lineStyleY = intarr(numPlots)
;lineStyleY[0] = 3
;lineStyleY[1] = 0
;psymY = intarr(numPlots)
;psymY[0]=-0
;psymY[1]=-0
;normalise = 1
;specifyYRange = 1
;plotErrorBars = 0

useGDL = 1
```

```
xMarginDefault = 0
yMarginDefault = 0
xPlotDefault = 0.0
yPlotDefault = 0.0
if(useGDL eq 1) then begin
    xMarginDefault = 32
    yMarginDefault = 18
    xPlotDefault = 0
    yPlotDefault = 0
    positionDefault=ASPECT(1.0)
endif else begin
    xMarginDefault = 3
    yMarginDefault = 1
    xPlotDefault = [1e10,1e10]
    yPlotDefault = [1e10,1e10]
    positionDefault=ASPECT(1.0)
endelse

formatString = 'D'
for yIndex=1, numYcols, 1 do begin
    formatString = formatString + ',D'
endfor

PRINT, "formatString = ", formatString

if(multipleFiles eq 1) then begin

    if(numYcols eq 1) then begin

    endif else begin
        PRINT, "error numYcols > 1 not supported with multipleFiles"
        exit
    endelse

    DDprobabilityOfChanceAlignmentArray = dblarr(numberFiles)

    for systemIndex=1, numberFiles, 1 do begin

        if(numberFiles gt 1) then begin
            systemIndexString = String(systemIndex, Format='(I2.2)')
            PRINT,systemIndexString
        endif else begin
            systemIndexString = ''
        endelse

        psGraphfileName = fileNameBase1 + systemIndexString + fileNameBase2 +
            psGraphfileNameExtension
        pdfGraphfileName = fileNameBase1 + systemIndexString + fileNameBase2 +
            pdfGraphfileNameExtension

        set_plot,'ps'
        device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE      ;SET_RESOLUTION
            =[320,200]    ;SET_CHARACTER_SIZE=[200,40]


        ;if(specifyYRange eq 1) then begin
            if(numberSubFiles gt 1) then begin
                plot, [-100, 100, 0, 0], [0, 0, -100, 100], xtitle=plotXAxisTitle, ytitle=
                    plotYAxisTitle, xrange=[plotXmin, plotXmax], yrange=[plotYmin, plotYmax],
                    BACKGROUND=1, CHARSIZE=2.0,XMARGIN=xMarginDefault,YMARGIN=yMarginDefault,
                    POSITION=positionDefault       ;title=plotTitle,
            endif
        ;endif else begin
            ;plot, xPlotDefault, yPlotDefault, xtitle=plotXAxisTitle, ytitle=plotYAxisTitle, xrange
                =[plotXmin, plotXmax], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=xMarginDefault,YMARGIN=
                yMarginDefault      ;title=plotTitle,
        ;endelse

        averageX = 0.0
        varianceX = 0.0
        stddevX = 0.0
        averageY = 0.0
        varianceY = 0.0
        stddevY = 0.0

        dataFileContainsData = 0

        for subIndex=0, numberSubFiles-1, 1 do begin

            if(numberSubFiles gt 1) then begin
                asciiTablefileName = fileNameBase1 + systemIndexString + fileNameBase2 + STRTRIM(
                    subIndex, 2) + asciiTablefileNameExtension
            endif else begin
                asciiTablefileName = fileNameBase1 + systemIndexString + fileNameBase2 +
                    asciiTablefileNameExtension
            endelse
            PRINT, "asciiTablefileName = ", asciiTablefileName

            readcol64bitvars, asciiTablefileName, Xarray, Yarray, FORMAT=formatString,SKIPLINE=
                headerNumLines

            if(normalise eq 1) then begin
```

```
    maxValue = max(Yarray)
    ;PRINT, "maxValue = ", maxValue
    numberValues = (size(Yarray))[1]
    ;PRINT, "numberValues = ", numberValues
    for arrayIndex=0, numberValues-1, 1 do begin
        Yarray[arrayIndex] = (Yarray[arrayIndex]/maxValue)*plotYmax
    endfor
endif

if(properMotionCalculations eq 1) then begin

    ;error in readcol64bitvars, always creates an array with at least 2 elements
    sz2 = SIZE(Xarray)
    n2 = N_ELEMENTS(Xarray)
    ;PRINT, "systemIndex = ", systemIndex
    ;PRINT, "Yarray(0) = ", Yarray(0)
    ;PRINT, "Yarray(1) = ", Yarray(1)
    ;PRINT, "n2 = ", n2
    ;exit

    fieldminX = 0.0
    fieldmaxX = 0.0
    fieldminY = 0.0
    fieldmaxY = 0.0
    AminX = 0.0
    AmaxX = 0.0
    AminY = 0.0
    AmaxY = 0.0
    BminX = 0.0
    BmaxX = 0.0
    BminY = 0.0
    BmaxY = 0.0

    if(subIndex eq 0) then begin

        if(n2 gt 2) then begin

            AminX = MIN(Xarray)
            AmaxX = MAX(Xarray)
            AminY = MIN(Yarray)
            AmaxY = MAX(Yarray)
            ;calculate average

            ;averageX = MEAN(Xarray)
            ;varianceX = VARIANCE(Xarray)
            ;stddevX = STDDEV(Xarray)
            ;averageY = MEAN(Yarray)
            ;varianceY = VARIANCE(Yarray)
            ;stddevY = STDDEV(Yarray)

            ;deviationFromMeanXarray = ABS(ABS(Xarray)-ABS(averageX))
            ;deviationFromMeanYarray = ABS(ABS(Yarray)-ABS(averageY))
            ;averageDeviationFromMeanX = MEAN(deviationFromMeanXarray)
            ;stddevDeviationFromMeanX = VARIANCE(deviationFromMeanXarray)
            ;varianceDeviationFromMeanX = STDDEV(deviationFromMeanXarray)
            ;averageDeviationFromMeanY = MEAN(deviationFromMeanYarray)
            ;stddevDeviationFromMeanY = VARIANCE(deviationFromMeanYarray)
            ;varianceDeviationFromMeanY = STDDEV(deviationFromMeanYarray)

                ;PRINT, "Yarray = ", Yarray
                ;PRINT, "deviationFromMeanYarray = ", deviationFromMeanYarray
                dataFileContainsData = 1
        endif

    endif else begin

        if(dataFileContainsData eq 1) then begin

            BminX = MIN(Xarray)
            BmaxX = MAX(Xarray)
            BminY = MIN(Yarray)
            BmaxY = MAX(Yarray)

            fieldminX = MIN([AminX, BminX])
            fieldmaxX = MAX([AmaxX, BmaxX])
            fieldminY = MIN([AminY, BminY])
            fieldmaxY = MAX([AmaxY, BmaxY])

            pi = 3.1416

            fieldDiameter = MAX([fieldmaxX-fieldminX,fieldmaxY-fieldminY])

            fieldArea = pi*((fieldDiameter/2.0)^2)
            PRINT, "fieldDiameter = ", fieldDiameter
            PRINT, "fieldArea = ", fieldArea



            BOCPdiameter  = MAX([BmaxX-BminX,BmaxY-BminY])
            BOCPArea = pi*((BOCPdiameter/2.0)^2)
            PRINT, "BOCPDiameter = ", BOCPDiameter
            PRINT, "BOCPArea = ", BOCPArea
```

```
DDprobabilityOfChanceAlignment = BOCPArea/fieldArea

PRINT, "DDprobabilityOfChanceAlignment = ", DDprobabilityOfChanceAlignment
DDprobabilityOfChanceAlignmentArray(systemIndex-1) =
     DDprobabilityOfChanceAlignment
;PRINT, "DDprobabilityOfChanceAlignmentArray = ",
     DDprobabilityOfChanceAlignmentArray


     ;PRINT, "(Xarray(0) - averageX) = ", (Xarray(0) - averageX)
     ;PRINT, "(Yarray(0) - averageY) = ", (Yarray(0) - averageY)
     ;PRINT, "(Xarray(1) - averageX) = ", (Xarray(1) - averageX)
     ;PRINT, "(Yarray(1) - averageY) = ", (Yarray(1) - averageY)
     ;PRINT, "(Xarray(1) - Xarray(0)) = ", (Xarray(1) - Xarray(0))
     ;PRINT, "(Yarray(1) - Yarray(0)) = ", (Yarray(1) - Yarray(0))
     ;PRINT, GAUSS_PDF(2.0/SQRT(varianceY))
     ;PRINT, varianceY


;deviationBetweenComponentsX = ABS(Xarray(1) - Xarray(0))
;deviationBetweenComponentsY = ABS(Yarray(1) - Yarray(0))
;DDcomponentAIndependentProbabilityOfLyingWithinDistributionX = GAUSS_PDF(
     ABS(Xarray(0) - averageX)/SQRT(varianceX))
;DDcomponentAIndependentProbabilityOfLyingWithinDistributionY = GAUSS_PDF(
     ABS(Yarray(0) - averageY)/SQRT(varianceY))
;DDcomponentBIndependentProbabilityOfLyingWithinDistributionX = GAUSS_PDF(
     ABS(Xarray(1) - averageX)/SQRT(varianceX))
;DDcomponentBIndependentProbabilityOfLyingWithinDistributionY = GAUSS_PDF(
     ABS(Yarray(1) - averageY)/SQRT(varianceY))
;DDcomponentBIndependentProbabilityOfLyingNextToComponentAX = GAUSS_PDF(ABS(
     deviationBetweenComponentsX - averageDeviationFromMeanX)/SQRT(
     varianceDeviationFromMeanX))
;DDcomponentBIndependentProbabilityOfLyingNextToComponentAY = GAUSS_PDF(ABS(
     deviationBetweenComponentsY - averageDeviationFromMeanY)/SQRT(
     varianceDeviationFromMeanY))
     ;DDcomponentBIndependentProbabilityOfLyingNextToComponentAX = GAUSS_PDF(
          ABS((Xarray(1) - Xarray(0)))/SQRT(varianceX))
     ;DDcomponentBIndependentProbabilityOfLyingNextToComponentAY = GAUSS_PDF(
          ABS((Yarray(1) - Yarray(0)))/SQRT(varianceY))
          ;DDcomponentBIndependentProbabilityOfLyingNextToComponentAX =
               GAUSS_PDF(ABS((Xarray(1) - Xarray(0)) - varianceX)/SQRT(1.0))
          ;DDcomponentBIndependentProbabilityOfLyingNextToComponentAY =
               GAUSS_PDF(ABS((Yarray(1) - Yarray(0)) - varianceY)/SQRT(1.0))

;PRINT, "GAUSS_PDF(0.0) = ", GAUSS_PDF(3.0)
;PRINT, "systemIndex = ", systemIndex


;DDcomponentAIndependentProbabilityOfLyingWithinDistributionX = 1.0-
     DDcomponentAIndependentProbabilityOfLyingWithinDistributionX
;DDcomponentAIndependentProbabilityOfLyingWithinDistributionY = 1.0-
     DDcomponentAIndependentProbabilityOfLyingWithinDistributionY
;DDcomponentBIndependentProbabilityOfLyingWithinDistributionX = 1.0-
     DDcomponentBIndependentProbabilityOfLyingWithinDistributionX
;DDcomponentBIndependentProbabilityOfLyingWithinDistributionY = 1.0-
     DDcomponentBIndependentProbabilityOfLyingWithinDistributionY
;DDcomponentBIndependentProbabilityOfLyingNextToComponentAX = 1.0-
     DDcomponentBIndependentProbabilityOfLyingNextToComponentAX
;DDcomponentBIndependentProbabilityOfLyingNextToComponentAY = 1.0-
     DDcomponentBIndependentProbabilityOfLyingNextToComponentAY
               ;if(systemIndex eq 5) then begin

     ;PRINT, "DDcomponentAIndependentProbabilityOfLyingWithinDistributionX =
          ", (DDcomponentAIndependentProbabilityOfLyingWithinDistributionX)
     ;PRINT, "DDcomponentAIndependentProbabilityOfLyingWithinDistributionY =
          ", (DDcomponentAIndependentProbabilityOfLyingWithinDistributionY)
     ;PRINT, "DDcomponentBIndependentProbabilityOfLyingWithinDistributionX =
          ", (DDcomponentBIndependentProbabilityOfLyingWithinDistributionX)
     ;PRINT, "DDcomponentBIndependentProbabilityOfLyingWithinDistributionY =
          ", (DDcomponentBIndependentProbabilityOfLyingWithinDistributionY)
     ;PRINT, "DDcomponentBIndependentProbabilityOfLyingNextToComponentAX = ",
          (DDcomponentBIndependentProbabilityOfLyingNextToComponentAX)
     ;PRINT, "DDcomponentBIndependentProbabilityOfLyingNextToComponentAY = ",
          (DDcomponentBIndependentProbabilityOfLyingNextToComponentAY)
     ;exit
               ;endif
                    ;
                         DDcomponentAIndependentProbabilityOfLyingWithinDistribution
                          = SQRT(
                         DDcomponentAIndependentProbabilityOfLyingWithinDistributionX
                         ^2 +
                         DDcomponentAIndependentProbabilityOfLyingWithinDistributionY
                         ^2)
                    ;
                         DDcomponentBIndependentProbabilityOfLyingWithinDistribution
                          = SQRT(
                         DDcomponentBIndependentProbabilityOfLyingWithinDistributionX
                         ^2 +
                         DDcomponentBIndependentProbabilityOfLyingWithinDistributionY
                         ^2)
```

```
                                        ;
                                            DDcomponentBIndependentProbabilityOfLyingNextToComponentA
                                             = SQRT(
                                            DDcomponentBIndependentProbabilityOfLyingNextToComponentAX
                                            ^2 +
                                            DDcomponentBIndependentProbabilityOfLyingNextToComponentAY
                                            ^2)
                                        ;PRINT, "
                                            DDcomponentAIndependentProbabilityOfLyingWithinDistribution
                                             = ",
                                            DDcomponentAIndependentProbabilityOfLyingWithinDistribution

                                        ;PRINT, "
                                            DDcomponentBIndependentProbabilityOfLyingWithinDistribution
                                             = ",
                                            DDcomponentBIndependentProbabilityOfLyingWithinDistribution

                                        ;PRINT, "
                                            DDcomponentBIndependentProbabilityOfLyingNextToComponentA
                                             = ",
                                            DDcomponentBIndependentProbabilityOfLyingNextToComponentA

                                        ;DDprobabilityOfChanceAlignment =
                                            DDcomponentAIndependentProbabilityOfLyingWithinDistribution
                                            *
                                            DDcomponentBIndependentProbabilityOfLyingWithinDistribution
                                            *
                                            DDcomponentBIndependentProbabilityOfLyingNextToComponentA


                                ;DDprobabilityOfChanceAlignment = (
                                    DDcomponentAIndependentProbabilityOfLyingWithinDistributionX
                                    *
                                    DDcomponentAIndependentProbabilityOfLyingWithinDistributionY
                                    *
                                    DDcomponentBIndependentProbabilityOfLyingWithinDistributionX
                                    *
                                    DDcomponentBIndependentProbabilityOfLyingWithinDistributionY
                                    )
                        ;DDprobabilityOfChanceAlignment = (
                            DDcomponentAIndependentProbabilityOfLyingWithinDistributionX*
                            DDcomponentAIndependentProbabilityOfLyingWithinDistributionY*
                            DDcomponentBIndependentProbabilityOfLyingWithinDistributionX*
                            DDcomponentBIndependentProbabilityOfLyingWithinDistributionY*
                            DDcomponentBIndependentProbabilityOfLyingNextToComponentAX*
                            DDcomponentBIndependentProbabilityOfLyingNextToComponentAY)

                        ;PRINT, "DDprobabilityOfChanceAlignment = ", DDprobabilityOfChanceAlignment
                        ;DDprobabilityOfChanceAlignmentArray(systemIndex-1) =
                            DDprobabilityOfChanceAlignment
                        ;PRINT, "DDprobabilityOfChanceAlignmentArray = ",
                            DDprobabilityOfChanceAlignmentArray
                        ;exit
                    endif
                endelse
            endif


            if(specifyYRange eq 1) then begin
            endif else begin
                plotYmax = mean(Yarray)*2.0
            endelse

            if(numberSubFiles gt 1) then begin

                if(plotErrorBars eq 1) then begin

                    sz = SIZE(Xarray)
                    n = N_ELEMENTS(Xarray)
                    XarrayError = dblarr(n)
                    YarrayError = dblarr(n)
                    maxMexicanHatObjectDetectionErrorInPixels = 1.0
                    properMotionErrorInPixels = maxMexicanHatObjectDetectionErrorInPixels/sqrt(2.0)
                    for i=0, n-1, 1 do begin
                        XarrayError[i] = properMotionErrorInPixels
                        YarrayError[i] = properMotionErrorInPixels
                    endfor
                    oploterror, Xarray, Yarray, XarrayError, YarrayError, psym=psymY[subIndex],
                        LINESTYLE=lineStyleY[subIndex], COLOR=0
                endif else begin
                    oplot, Xarray, Yarray, psym=psymY[subIndex], LINESTYLE=lineStyleY[subIndex],
                        COLOR=0
                endelse

            endif else begin
                ;if(plotErrorBars eq 1) then begin
                ;    plot, Xarray, Yarray, psym=psymY[subIndex], LINESTYLE=lineStyleY[subIndex],
                    COLOR=0, xtitle=plotXAxisTitle, ytitle=plotYAxisTitle, xrange=[plotXmin,
                    plotXmax], yrange=[plotYmin, plotYmax], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=
                    xMarginDefault,YMARGIN=yMarginDefault       ;title=plotTitle,
                ;endif else begin
```

```
                    plot,  Xarray, Yarray, psym=psymY[subIndex], LINESTYLE=lineStyleY[subIndex],
                        COLOR=0, xtitle=plotXAxisTitle, ytitle=plotYAxisTitle, xrange=[plotXmin,
                        plotXmax], yrange=[plotYmin, plotYmax], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=
                        xMarginDefault,YMARGIN=yMarginDefault       ;title=plotTitle,
                ;endelse
            endelse
        endfor

        device,/close

        ;convert ps to pdf;
        convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
        PRINT, "convertPStoPDFcommand = ", convertPStoPDFcommand
        spawn,convertPStoPDFcommand,result

        if(properMotionCalculations eq 1) then begin

            PRINT, "DDprobabilityOfChanceAlignmentArray = "
            sz = SIZE(DDprobabilityOfChanceAlignmentArray)
            n = N_ELEMENTS(DDprobabilityOfChanceAlignmentArray)
            for i=0, n-1, 1 do begin
                PRINT, DDprobabilityOfChanceAlignmentArray(i)
                PRINT, " "
            endfor
            ;PRINT, "DDprobabilityOfChanceAlignmentArray = ", DDprobabilityOfChanceAlignmentArray
        endif

    endfor

endif else begin

    set_plot,'ps'
    device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE       ;SET_RESOLUTION
        =[320,200]    ;SET_CHARACTER_SIZE=[200,40]
    plot, xPlotDefault, yPlotDefault, xtitle=plotXAxisTitle, ytitle=plotYAxisTitle, xrange=[plotXmin
        , plotXmax], yrange=[plotYmin, plotYmax], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=xMarginDefault
        ,YMARGIN=yMarginDefault       ;title=plotTitle,

    for subIndex=0, numYcols-1, 1 do begin

        if(subIndex eq 1) then begin
            readcol64bitvars, asciiTablefileName, Xarray, Yarray, FORMAT=formatString, SKIPLINE=
                headerNumLines
        endif else begin
            readcol64bitvars, asciiTablefileName, Xarray, YarrayTemp, Yarray, FORMAT=formatString,
                SKIPLINE=headerNumLines
        endelse

        oplot, Xarray, Yarray, LINESTYLE=lineStyleY[subIndex], psym=psymY[subIndex], COLOR=0

    endfor

    device,/close
    ;convert ps to pdf;
    convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
    spawn,convertPStoPDFcommand,result
endelse


end
```

A.4 Observations Code and Data

A.4.1 calculate the angle between WD components in double degenerate system (for aligning slit)

bearing-23Aug10a.ods

	A	B	C	D	E	F	G	H	I	J	K
1	bearing.ods (NB bearing.pro behaves badly due to gdl/idl PRINT function limitations; http://www.dfanning.com/misc_tips/dbl_to_str.html)										
2											
3	VLT name	x1	y1	x2	y2	dx	dy	dxcorrected	theta	thetaDegrees	thetaDegrees (+1
4	WDB02	53.15359480	-0.82692479	53.15251716	-0.82178397	=B4-D4	=C4-E4	=F4*COS(C4/180*PI())	=ATAN(H4/G4)	=(I4/PI())*180	=J4+180
5	WDB04	141.69533369	13.35958843	141.69584285	13.36069038	=B5-D5	=C5-E5	=F5*COS(C5/180*PI())	=ATAN(H5/G5)	=(I5/PI())*180	=J5
6	WDB10	179.90758042	13.73720478	179.90760447	13.73576715	=B6-D6	=C6-E6	=F6*COS(C6/180*PI())	=ATAN(H6/G6)	=(I6/PI())*180	=J6+180
7	WDB11	198.38394049	20.51100050	198.38569698	20.51094081	=B7-D7	=C7-E7	=F7*COS(C7/180*PI())	=ATAN(H7/G7)	=(I7/PI())*180	=J7+180
8	WDB12	202.05953933	16.53098677	202.05987454	16.53080830	=B8-D8	=C8-E8	=F8*COS(C8/180*PI())	=ATAN(H8/G8)	=(I8/PI())*180	=J8+180
9											
10											
11											

(bearing.pro, although used for many calculations in this project - is no longer used, as it's printed values cannot be relied upon to a high degree of accuracy due to an IDL/GDL print function limitation).

A.4.2 Flux Calibrated WD Data

WHT Flux calibrated Data

PDWHTJuly08ReducedData15Nov09c-usingNewfcBB.zip:

- SSWD01Aw.dat - szbc1dwcfcSDSS0052a.dat
- SSWD01Bw.dat - szbc1dwcfcSDSS0052b.dat
- SSWD02Aw.dat - szbc1dwcfcSDSS1507a.dat
- SSWD02Bw.dat - szbc1dwcfcSDSS1507b.dat
- SSWD03Aw.dat - szbc1dwcfcSDSS1703a.dat
- SSWD03Bw.dat - szbc1dwcfcSDSS1703b.dat
- SSWD04Aw.dat - szbc1dwcfcSDSS2259a.dat
- SSWD04Bw.dat - szbc1dwcfcSDSS2259b.dat

WHT Flux Re-calibrated Magnetic WD Candidate Data

SDSSDDmagneticWDdata-30Aug10a.zip:

- fcSDSS1507a-SSWD02Aw.dat.FluxRecalibratedUsingModel
- fcSDSS1507b-SSWD02Bw.dat.FluxRecalibratedUsingModel
- fcSDSS2259a-SSWD04Aw.dat.FluxRecalibratedUsingModel

Gemini North Flux calibrated Data

geminiN2009BdatafluxcalibratedSept09c-usingNewfcBB.zip:

- SSWD05Ag.dat - fcde1stgsN20090720S0083.dat
- SSWD05Bg.dat - fcde2stgsN20090720S0083.dat
- SSWD06Ag.dat - fcde1stgsN20090720S0094.dat
- SSWD06Bg.dat - fcde2stgsN20090720S0094.dat

Gemini South Flux calibrated Data

geminiS2010AdatafluxcalibratedJune10a.zip:

- SSWD12Ag.dat - f2010A-DD-07A.dat
- SSWD12Bg.dat - f2010A-DD-07B.dat

VLT Flux calibrated Data

IRAFreducedVLTDataByPDD-richard.tar.gz:

- SSWD07Av.dat - WD2-T-WAV-FLX-A.dat
- SSWD07Bv.dat - WD2-T-WAV-FLX-B.dat
- SSWD08Av.dat - WD4-T-WAV-FLX-A.dat
- SSWD08Bv.dat - WD4-T-WAV-FLX-B.dat
- SSWD09Av.dat - WD10-T-A-WAV-FLX.dat
- SSWD09Bv.dat - WD10-T-B-WAV-FLX.dat
- SSWD10Av.dat - WD11-T-WAV-FLX-A.dat
- SSWD10Bv.dat - WD11-T-WAV-FLX-B.dat
- SSWD11Av.dat - WD12-T-WAV-FLX-A.dat
- SSWD11Bv.dat - WD12-T-WAV-FLX-B.dat

WiFeS Flux Re-calibrated Magnetic WD Candidate Data - low S/N - NOT USED

N/A

A.4.3 WHT Data Reduction

WHT Data was reduced using IRAF scripts.

IRAF Software Installation

IRAF Software (version 2.14.1);

- as.pcix.gen.gz
- ib.rhux.x86.gz
- nb.rhux.x86.gz
- tcsh-6.15-7.fc10.x86_64.rpm
- iraf_install.sh.centos5

Instructions for IRAF software installation on Centos 5 are found here;

```
//how to install iraf using the script iraf_install.sh;

IRAF README

1. The script used to install IRAF creates a user account named iraf. It has no password set so it
     is advised to set one of your own.
After installation IRAF can be run from any user account, instructions are given below.

2. During installation the script will perform a trial installation, after that it will return a
     prompt. At this point you must type "exit" and it will proceed to the full installation.
When the install configuration asks for default locations for folders keep hitting enter to accept
     the default locations.

2.a.    Post install configuration
The default options are enclosed in () while the option i chose is given after

    Configure IRAF Networking on this machine? (yes): no
Unless you are installing this on a networked machine choose no

    Create a default tapecap file? (yes): no    / Do you wish to create a default dev$tapecap link?
        (yes): no

    Do you wish to delete these unused HSI binaries? (yes): yes
```

```
This deletes the binaries used in Mac OS, Cygwin (Windows compatability), and other Linux
     architectures

    Do you wish to strip the system of sources? (no): no
Unless you are planning on developing the code then select yes to free up disk space

2b. install X11 libraries using [optional - can use skycat/gaia instead]
    x11iraf-v2.0BETA-bin.redhat/install
2e. System - Administration - Users and Groups - change iraf password to something secure

3. To set up IRAF on your user account it is first recommended that you make a directory in your
     home folder named iraf (mkdir ~/iraf).
The following steps must be done in a command line terminal
Go into the folder (cd iraf) and type in the command "mkiraf". When it asks for preferred terminal
     type select xgterm, this will have been installed along with IRAF.
Two objects should be created in your iraf folder, a directory named uparm and file named login.cl

4. To start IRAF open an xgterm window (this can done by typing "xgterm &" into a command line
     terminal) and type "cd iraf" followed by "cl"
This will start IRAF
```

Lacos Spec Software Installation

Laplacian Cosmic Ray Removal Software created by Pieter van Dokkum, April 2001 (van Dokkum 2001);
lacos_spec.cl

Reduction Archive Files

Reduced Files: PDWHTJuly08ReducedData15Nov09c-usingNewfcBB.zip
Raw and Reduced Files: backupIRAFWHT08workingDir15Nov09c.zip
WHT Reduced Files:

- 24july08WHTlog.txt
- 25july08WHTlog.txt
- biaslistblue.txt
- caliblamplistblue0.6.txt
- caliblamplistblue0.6zerocorrectedMappedToScience1d.0001.txt
- caliblamplistblue0.6zerocorrectedMappedToScience1d.txt
- caliblamplistblue0.6zerocorrectedMappedToScience.txt
- caliblamplistblue0.6zerocorrected.txt
- caliblamplistblue1.0.txt
- caliblamplistblue1.0zerocorrectedMappedToScience1d.0001.txt
- caliblamplistblue1.0zerocorrectedMappedToScience1d.0002.txt
- caliblamplistblue1.0zerocorrectedMappedToScience1d.txt
- caliblamplistblue1.0zerocorrectedMappedToScience.txt
- caliblamplistblue1.0zerocorrected.txt
- flatlistblue0.6.txt
- flatlistblue0.6zerocorrected.txt
- flatlistblue1.0.txt
- flatlistblue1.0zerocorrected.txt
- fluxCalDemo.ods
- fluxCalibrationUsingDCWD.perl
- listAllObjectsExceptForFluxCalDCCWDdat.txt
- listAllObjectsExceptForFluxCalDCCWDfits.txt

- listszbc1dwcComaWD.txt
- listszbc1dwcSDSS0052a.txt
- listszbc1dwcSDSS0052b.txt
- listszbc1dwcSDSS1507a.txt
- listszbc1dwcSDSS1507b.txt
- listszbc1dwcSDSS1703a-24July.txt
- listszbc1dwcSDSS1703a-25July.txt
- listszbc1dwcSDSS1703a.txt
- listszbc1dwcSDSS1703b-24July.txt
- listszbc1dwcSDSS1703b-25July.txt
- listszbc1dwcSDSS1703b.txt
- listszbc1dwcSDSS2259a-24July.txt
- listszbc1dwcSDSS2259a-25July.txt
- listszbc1dwcSDSS2259a.txt
- listszbc1dwcSDSS2259b-24July.txt
- listszbc1dwcSDSS2259b-25July.txt
- listszbc1dwcSDSS2259b.txt
- listszbc1dwcSP2157-24July.txt
- listszbc1dwcSP2157-25July.txt
- listszbc1dwcWD1918-24July.txt
- listszbc1dwcWD1918-25July.txt
- login.cl
- sciencelistblue0.6.txt
- sciencelistblue0.6zeroandbadpixcorrected1d.0001.txt
- sciencelistblue0.6zeroandbadpixcorrected1dS.0001.txt
- sciencelistblue0.6zeroandbadpixcorrected1dS.txt
- sciencelistblue0.6zeroandbadpixcorrected1dwavelengthCalibrated.0001.txt
- sciencelistblue0.6zeroandbadpixcorrected1dwavelengthCalibrated.txt
- sciencelistblue0.6zeroandbadpixcorrectedS.txt
- sciencelistblue0.6zeroandbadpixcorrected.txt
- sciencelistblue0.6zerocorrected.txt
- sciencelistblue1.0.txt
- sciencelistblue1.0zeroandbadpixcorrected1d.0001.txt
- sciencelistblue1.0zeroandbadpixcorrected1d.0002.txt
- sciencelistblue1.0zeroandbadpixcorrected1d.000x.txt
- sciencelistblue1.0zeroandbadpixcorrected1dDD.0001.txt
- sciencelistblue1.0zeroandbadpixcorrected1dDD.0002.txt
- sciencelistblue1.0zeroandbadpixcorrected1dDD.txt
- sciencelistblue1.0zeroandbadpixcorrected1dS.0001.txt
- sciencelistblue1.0zeroandbadpixcorrected1dS.txt
- sciencelistblue1.0zeroandbadpixcorrected1d.txt
- sciencelistblue1.0zeroandbadpixcorrected1dwavelengthCalibrated.0001.txt
- sciencelistblue1.0zeroandbadpixcorrected1dwavelengthCalibrated.0002.txt
- sciencelistblue1.0zeroandbadpixcorrectedDD.txt
- sciencelistblue1.0zeroandbadpixcorrectedS.txt

- sciencelistblue1.0zeroandbadpixcorrected.txt
- sciencelistblue1.0zerocorrected.txt
- spectraReductionMethod13Nov09b.txt
- szbc1dwcComaWD.dat
- szbc1dwcComaWD.fits
- szbc1dwcfcComaWD.dat
- szbc1dwcfcSDSS0052a.dat
- szbc1dwcfcSDSS0052b.dat
- szbc1dwcfcSDSS1507a.dat
- szbc1dwcfcSDSS1507b.dat
- szbc1dwcfcSDSS1703a-24July.dat
- szbc1dwcfcSDSS1703a-25July.dat
- szbc1dwcfcSDSS1703a.dat
- szbc1dwcfcSDSS1703b-24July.dat
- szbc1dwcfcSDSS1703b-25July.dat
- szbc1dwcfcSDSS1703b.dat
- szbc1dwcfcSDSS2259a-24July.dat
- szbc1dwcfcSDSS2259a-25July.dat
- szbc1dwcfcSDSS2259a.dat
- szbc1dwcfcSDSS2259b-24July.dat
- szbc1dwcfcSDSS2259b-25July.dat
- szbc1dwcfcSDSS2259b.dat
- szbc1dwcfcSP2157-24July.dat
- szbc1dwcfcSP2157-25July.dat
- szbc1dwcSDSS0052a.dat
- szbc1dwcSDSS0052a.fits
- szbc1dwcSDSS0052b.dat
- szbc1dwcSDSS0052b.fits
- szbc1dwcSDSS1507a.dat
- szbc1dwcSDSS1507a.fits
- szbc1dwcSDSS1507b.dat
- szbc1dwcSDSS1507b.fits
- szbc1dwcSDSS1703a-24July.dat
- szbc1dwcSDSS1703a-24July.fits
- szbc1dwcSDSS1703a-25July.dat
- szbc1dwcSDSS1703a-25July.fits
- szbc1dwcSDSS1703a.dat
- szbc1dwcSDSS1703a.fits
- szbc1dwcSDSS1703b-24July.dat
- szbc1dwcSDSS1703b-24July.fits
- szbc1dwcSDSS1703b-25July.dat
- szbc1dwcSDSS1703b-25July.fits
- szbc1dwcSDSS1703b.dat
- szbc1dwcSDSS1703b.fits
- szbc1dwcSDSS2259a-24July.dat

- szbc1dwcSDSS2259a-24July.fits
- szbc1dwcSDSS2259a-25July.dat
- szbc1dwcSDSS2259a-25July..fits
- szbc1dwcSDSS2259a.dat
- szbc1dwcSDSS2259a.fits
- szbc1dwcSDSS2259b-24July.dat
- szbc1dwcSDSS2259b-24July.fits
- szbc1dwcSDSS2259b-25July.dat
- szbc1dwcSDSS2259b-25July..fits
- szbc1dwcSDSS2259b.dat
- szbc1dwcSDSS2259b.fits
- szbc1dwcSP2157-24July.dat
- szbc1dwcSP2157-24July.fits
- szbc1dwcSP2157-25July.dat
- szbc1dwcSP2157-25July.fits
- szbc1dwcWD1918-24July.fits
- szbc1dwcWD1918-25July.fits
- szbc1dwcWD1918-25JulyInterpolated.dat
- WHTfileListCreationRecord.txt
- whtreduce15Nov09b.cl
- whtreduce15Nov09c.cl
- whtreduce.cl
- zerolistblue.txt

WHT Raw Files:

- r1175417.fit
- r1175419.fit
- r1175420.fit
- r1175421.fit
- r1175422.fit
- r1175423.fit
- r1175429.fit
- r1175430.fit
- r1175431.fit
- r1175433.fit
- r1175439.fit
- r1175440.fit
- r1175441.fit
- r1175442.fit
- r1175443.fit
- r1175444.fit
- r1175445.fit
- r1175446.fit
- r1175447.fit

- r1175448.fit
- r1175449.fit
- r1175450.fit
- r1175451.fit
- r1175452.fit
- r1175453.fit
- r1175454.fit
- r1175455.fit
- r1175456.fit
- r1175457.fit
- r1175458.fit
- r1175459.fit
- r1175460.fit
- r1175461.fit
- r1175462.fit
- r1175463.fit
- r1175464.fit
- r1175465.fit
- r1175466.fit
- r1175467.fit
- r1175468.fit
- r1175469.fit
- r1175470.fit
- r1175471.fit
- r1175472.fit
- r1175473.fit
- r1175474.fit
- r1175475.fit
- r1175476.fit
- r1175477.fit
- r1175478.fit
- r1175479.fit
- r1175480.fit
- r1175481.fit
- r1175482.fit
- r1175483.fit
- r1175484.fit
- r1175485.fit
- r1175486.fit
- r1175487.fit
- r1175488.fit
- r1175489.fit
- r1175490.fit
- r1175491.fit
- r1175492.fit

- r1175493.fit
- r1175494.fit
- r1175495.fit
- r1175496.fit
- r1175497.fit
- r1175498.fit
- r1175499.fit
- r1175500.fit
- r1175501.fit
- r1175502.fit
- r1175503.fit
- r1175504.fit
- r1175505.fit
- r1175506.fit
- r1175507.fit
- r1175508.fit
- r1175509.fit
- r1175510.fit
- r1175511.fit
- r1175512.fit
- r1175513.fit
- r1175514.fit
- r1175515.fit
- r1175516.fit
- r1175517.fit
- r1175518.fit
- r1175519.fit
- r1175520.fit
- r1175521.fit
- r1175522.fit
- r1175523.fit
- r1175524.fit
- r1175525.fit
- r1175526.fit
- r1175527.fit
- r1175528.fit
- r1175529.fit
- r1175530.fit
- r1175531.fit
- r1175532.fit
- r1175533.fit
- r1175534.fit
- r1175535.fit
- r1175536.fit
- r1175537.fit

- r1175538.fit
- r1175539.fit
- r1175540.fit
- r1175541.fit
- r1175542.fit
- r1175543.fit
- r1175544.fit
- r1175545.fit
- r1175546.fit
- r1175547.fit
- r1175548.fit
- r1175549.fit
- r1175551.fit
- r1175552.fit
- r1175553.fit
- r1175554.fit
- r1175555.fit
- r1175556.fit
- r1175557.fit
- r1175558.fit
- r1175559.fit
- r1175560.fit
- r1175561.fit
- r1175562.fit
- r1175563.fit
- r1175564.fit
- r1175565.fit
- r1175566.fit
- r1175567.fit
- r1175568.fit
- r1175569.fit
- r1175570.fit
- r1175571.fit
- r1175572.fit
- r1175573.fit
- r1175574.fit
- r1175575.fit
- r1175576.fit
- r1175577.fit
- r1175578.fit
- r1175579.fit
- r1175580.fit
- r1175581.fit
- r1175582.fit
- r1175583.fit

- r1175584.fit
- r1175585.fit
- r1175586.fit
- r1175587.fit
- r1175588.fit
- r1175589.fit
- r1175590.fit
- r1175591.fit
- r1175592.fit
- r1175593.fit
- r1175594.fit
- r1175595.fit
- r1175596.fit
- r1175597.fit
- r1175598.fit
- r1175599.fit
- r1175600.fit
- r1175601.fit
- r1175602.fit
- r1175603.fit
- r1175604.fit
- r1175605.fit
- r1175606.fit
- r1175607.fit
- r1175608.fit
- r1175609.fit
- r1175610.fit
- r1175611.fit
- r1175612.fit
- r1175613.fit
- r1175614.fit
- r1175615.fit
- r1175616.fit
- r1175617.fit
- r1175618.fit
- r1175619.fit
- r1175620.fit
- r1175621.fit
- r1175622.fit
- r1175623.fit
- r1175624.fit
- r1175625.fit
- r1175626.fit
- r1175627.fit
- r1175628.fit

- r1175629.fit
- r1175630.fit
- r1175631.fit
- r1175632.fit
- r1175633.fit
- r1175634.fit
- r1175635.fit
- r1175636.fit
- r1175637.fit
- r1175638.fit
- r1175639.fit
- r1175640.fit
- r1175641.fit
- r1175642.fit
- r1175643.fit
- r1175644.fit
- r1175645.fit
- r1175646.fit
- r1175647.fit
- r1175648.fit
- r1175649.fit
- r1175650.fit
- r1175651.fit
- r1175652.fit
- r1175653.fit
- r1175654.fit
- r1175655.fit
- r1175656.fit
- r1175657.fit
- r1175658.fit
- r1175659.fit
- r1175660.fit
- r1175661.fit
- r1175662.fit
- r1175663.fit
- r1175664.fit
- r1175665.fit
- r1175666.fit
- r1175667.fit
- r1175668.fit
- r1175669.fit
- r1175670.fit
- r1175671.fit
- r1175672.fit
- r1175673.fit

- r1175674.fit
- r1175675.fit
- r1175676.fit
- r1175677.fit
- r1175678.fit
- r1175679.fit
- r1175680.fit
- r1175681.fit
- r1175682.fit
- r1175683.fit
- r1175684.fit
- r1175685.fit
- r1175686.fit
- r1175687.fit
- r1175688.fit
- r1175689.fit
- r1175690.fit
- r1175691.fit
- r1175692.fit
- r1175693.fit
- r1175694.fit
- r1175695.fit
- r1175696.fit
- r1175697.fit
- r1175698.fit
- r1175699.fit
- r1175700.fit
- r1175701.fit
- r1175702.fit
- r1175703.fit
- r1175704.fit
- r1175705.fit
- r1175706.fit
- r1175707.fit
- r1175708.fit
- r1175709.fit
- r1175710.fit
- r1175711.fit
- r1175712.fit
- r1175713.fit
- r1175714.fit
- r1175715.fit
- r1175716.fit
- r1175717.fit
- r1175718.fit

- r1175719.fit
- r1175720.fit
- r1175721.fit
- r1175722.fit
- r1175723.fit
- r1175724.fit
- r1175725.fit
- r1175726.fit
- r1175727.fit
- r1175728.fit
- r1175729.fit
- r1175730.fit
- r1175731.fit
- r1175732.fit
- r1175733.fit
- r1175734.fit
- r1175735.fit
- r1175736.fit
- r1175737.fit
- r1175738.fit
- r1175739.fit
- r1175740.fit
- r1175741.fit
- r1175742.fit
- r1175743.fit
- r1175744.fit
- r1175745.fit
- r1175746.fit
- r1175747.fit
- r1175748.fit
- r1175749.fit
- r1175750.fit
- r1175751.fit
- r1175752.fit
- r1175753.fit
- r1175754.fit
- r1175755.fit
- r1175756.fit
- r1175757.fit
- r1175758.fit
- r1175759.fit
- r1175760.fit
- r1175761.fit
- r1175762.fit
- r1175763.fit

- r1175764.fit
- r1175765.fit
- r1175766.fit
- r1175767.fit
- r1175768.fit
- r1175769.fit
- r1175770.fit
- r1175771.fit
- r1175772.fit
- r1175773.fit
- r1175774.fit
- r1175775.fit
- r1175776.fit
- r1175777.fit
- r1175778.fit
- r1175779.fit
- r1175780.fit
- r1175781.fit
- r1175782.fit
- r1175783.fit
- r1175784.fit
- r1175785.fit
- r1175786.fit
- r1175787.fit
- r1175788.fit
- r1175789.fit
- r1175790.fit
- r1175791.fit
- r1175792.fit
- r1175793.fit
- r1175794.fit
- r1175795.fit
- r1175796.fit
- r1175797.fit
- r1175798.fit
- r1175799.fit
- r1175800.fit
- r1175801.fit
- r1175802.fit
- r1175803.fit
- r1175804.fit
- r1175805.fit
- r1175806.fit
- r1175807.fit
- r1175808.fit

- r1175809.fit
- r1175810.fit
- r1175811.fit
- r1175812.fit
- r1175813.fit
- r1175814.fit
- r1175815.fit
- r1175816.fit
- r1175817.fit
- r1175818.fit
- r1175819.fit
- r1175820.fit
- r1175821.fit

whtreduce.cl

```
procedure whtreduce()

string file_name
struct *flist
string file_name2
struct *flist2
string file_name3
struct *flist3

begin

    # RBB wht reduction script 15 Nov 09c [cosmic rays removal using lacos_spec.cl];
    # instructions;
    #   cd /home/rich/iraf
    #   delete all files in /home/rich/iraf/uparm
    #   delete all temporary files in /home/rich/iraf/database/
    #   delete all temporary files in /home/rich/iraf/
    #   add to /home/rich/iraf/login.cl;
    #       task whtreduce = /home/rich/iraf/whtreduce.cl
    #       task lacos_spec=/home/rich/iraf/lacos_spec.cl
    #   cl
    #   whtreduce
    #
    # IRAF Reduction Process for Paul Dobbie WHT July 08 Data;
    #    "IRAF V2.14.1 September 2008"
    # preconditions;
    #   Arc + Flat + Bias; ensure window sizes are the same for each file.
    #   ensure to use the correct flat field slit date
    #       (0.6 = high res, 0.1 = low res)
    #   biassec/trimsec values - see fits header for exact parameters and modify accordingly
    #
    # notes:
    #   1. regarding apall background subtraction - ensure background subtraction areas are correct
        (currently use +interac here),
    #   2. apall fit - ensure rejection is applied and the correct fit order is applied (currently
        use +interac here),
    #   3. have not separated flat (and bias) data from 24 July and 25 July (currently they are all
        combined together)
    #   4. flux calibration is currently done manually by executing a separate perl script


    string tempfile
    string tempstring2
    string rawdirString = "/home/rich/iraf/"
    string outpufileNameTempTypeCharacter
    int junk
    int aperatureNumToBeExtracted
    int skipAlreadyDone = 0
    struct line
    struct line2
    struct line3
    string outpuStringPart1
    string outpuStringPart2
    string outpuStringPart3
    string outpufileName
    string outpufileNameTemp
    string outpufileNameTempDatabase
```

```
if(skipAlreadyDone != 1)
{
    #1
    #noao
    #imred
    #ccdred
    ccdproc(images="@biaslistblue.txt", output="@zerolistblue.txt", biassec="[101:501,
        4105:4190]", trimsec="[125:390,250:3750]", ccdtype="",  fixpix-, oversca+, trim+,
        zerocor-, darkcor-, flatcor-, readaxis="column")

    #2
    #noao
    #imred
    #ccdred
    zerocombine(input="@zerolistblue.txt", output="Zero.fits", ccdtype="")

    #3
    #noao
    #imred
    #ccdred
    ccdproc(images="@flatlistblue0.6.txt", output="@flatlistblue0.6zerocorrected.txt", biassec
        ="[101:501, 4105:4190]", trimsec="[125:390,250:3750]", ccdtype="", fixpix-, oversca+,
        trim+, zerocor+, darkcor-, flatcor-, zero="Zero.fits", readaxis="column")
    ccdproc(images="@flatlistblue1.0.txt", output="@flatlistblue1.0zerocorrected.txt", biassec
        ="[101:501, 4105:4190]", trimsec="[125:390,250:3750]", ccdtype="", fixpix-, oversca+,
        trim+, zerocor+, darkcor-, flatcor-, zero="Zero.fits", readaxis="column")

    #4
    #noao
    #imred
    #ccdred
    flatcombine(process-, reject="none", input="@flatlistblue0.6zerocorrected.txt", output="
        Flat0.6.fits", ccdtype="", subsets-)
    flatcombine(process-, reject="none", input="@flatlistblue1.0zerocorrected.txt", output="
        Flat1.0.fits", ccdtype="", subsets-)

    #5
    #noao
    #twodspec
    #longslit
    response(normaliz="Flat0.6.fits", response="nFlat0.6.fits", calibrat="Flat0.6.fits",
        function="spline3", order=61, niterat=0, interac-)
    response(normaliz="Flat1.0.fits", response="nFlat1.0.fits", calibrat="Flat1.0.fits",
        function="spline3", order=61, niterat=0, interac-)

    #6
    #noao
    #imred
    #ccdred
    ccdproc(images="@sciencelistblue0.6.txt", output="@sciencelistblue0.6zerocorrected.txt",
        biassec="[101:501, 4105:4190]", trimsec="[125:390,250:3750]", ccdtype="", fixpix-,
        oversca+, trim+, zerocor+, darkcor-, flatcor+, zero="Zero.fits", flat="nFlat0.6.fits",
        readaxis="column")
    ccdproc(images="@sciencelistblue1.0.txt", output="@sciencelistblue1.0zerocorrected.txt",
        biassec="[101:501, 4105:4190]", trimsec="[125:390,250:3750]", ccdtype="", fixpix-,
        oversca+, trim+, zerocor+, darkcor-, flatcor+, zero="Zero.fits", flat="nFlat1.0.fits",
        readaxis="column")



    #6b
    file_name = "sciencelistblue0.6zerocorrected.txt"
    flist = file_name
    while (fscan (flist, line) != EOF) {

        printf ("line = %s\n", line)

        imdelete ("bad.fits", verify-, >& "dev$null")
        outpuStringPart1 = substr(line, 1, 2)  # Eg s1 = "sz"
        outpuStringPart3 = substr(line, 3, 14)        # Eg s1 = "c1175758.fit"
        outpuStringPart2 = "b"
        outpufileName = outpuStringPart1//outpuStringPart2//outpuStringPart3

        #printf ("%s\n", outpuStringPart1)
        #printf ("%s\n", outpuStringPart2)
        #printf ("%s\n", outpuStringPart3)
        printf ("outpufileName = %s\n", outpufileName)

        lacos_spec(input=line, output=outpufileName, outmask="bad.fits", verbose+, sigclip=7.5,
            sigfra=0.5, xorder=15, yorder=15, niter=5, verbose-)
    }
    file_name = "sciencelistblue1.0zerocorrected.txt"
    flist = file_name
    while (fscan(flist, line) != EOF) {

        printf ("line = %s\n", line)

        imdelete ("bad.fits", verify-, >& "dev$null")
        outpuStringPart1 = substr(line, 1, 2)         # Eg s1 = "sz"
        outpuStringPart3 = substr(line, 3, 14)        # Eg s1 = "c1175758.fit"
        outpuStringPart2 = "b"
        outpufileName = outpuStringPart1//outpuStringPart2//outpuStringPart3
```

```
    #printf ("%s\n", outpuStringPart1)
    #printf ("%s\n", outpuStringPart2)
    #printf ("%s\n", outpuStringPart3)
    printf ("outpufileName = %s\n", outpufileName)

    lacos_spec(input=line, output=outpufileName, outmask="bad.fits", verbose+, sigclip=7.5,
        sigfra=0.5, xorder=15, yorder=15, niter=5, verbose-)
}



#7
#noao
#twodspec
#apextract
apall(input="@sciencelistblue0.6zeroandbadpixcorrectedS.txt", output="@sciencelistblue0.6
    zeroandbadpixcorrected1dS.txt", format="onedspec", referen="", background="fit",
    weights="variance",  t_order=5, b_sample="-20:-10, 10:20", b_order=4, interac+, nfind
    =1)

aperatureNumToBeExtracted = 1
file_name = "sciencelistblue0.6zeroandbadpixcorrected1dS.txt"
flist = file_name
while (fscan(flist, line) != EOF) {

    printf ("line = %s\n", line)          # Eg s1 = "szbc1d1175758.fit"

    outpufileNameTempDatabase = "ap"//substr(line, 1, 4)//substr(line, 7, 13)   # Eg  = "
        szbc1175758"

    #temporarily move files in database - required for apall to continue to extract multiple
        aperatures
    tempfile = rawdirString//"database/"//outpufileNameTempDatabase
    copy (tempfile, tempfile//"TEMP")
    delete(tempfile)

    printf ("outpufileNameTempDatabase = %s\n", outpufileNameTempDatabase)
}



apall(input="@sciencelistblue1.0zeroandbadpixcorrectedS.txt", output="@sciencelistblue1.0
    zeroandbadpixcorrected1dS.txt", format="onedspec", referen="", background="fit",
    weights="variance",  t_order=5, b_sample="-20:-10, 10:20", b_order=4, interac+, nfind
    =1)

aperatureNumToBeExtracted = 1
file_name = "sciencelistblue1.0zeroandbadpixcorrected1dS.txt"
flist = file_name
while (fscan(flist, line) != EOF) {

    printf ("line = %s\n", line)          # Eg s1 = "szbc1d1175758.fit"

    outpufileNameTempDatabase = "ap"//substr(line, 1, 4)//substr(line, 7, 13)   # Eg  = "
        szbc1175758"

    #temporarily move files in database - required for apall to continue to extract multiple
        aperatures
    tempfile = rawdirString//"database/"//outpufileNameTempDatabase
    copy (tempfile, tempfile//"TEMP")
    delete(tempfile)

    printf ("outpufileNameTempDatabase = %s\n", outpufileNameTempDatabase)
}



# may not work in all scenarios - does in this case however; multiple aperatures do not work
    well with lists, unless every image has the same number of aperatures
apall(input="@sciencelistblue1.0zeroandbadpixcorrectedDD.txt", output="@sciencelistblue1.0
    zeroandbadpixcorrected1dDD.txt", format="onedspec", referen="", background="fit",
    weights="variance", t_order=5, b_sample="-20:-10, 10:20", b_order=4, interac+, nfind=2)

aperatureNumToBeExtracted = 2
file_name = "sciencelistblue1.0zeroandbadpixcorrected1dDD.txt"
flist = file_name
while (fscan(flist, line) != EOF)
{
    printf ("line = %s\n", line)          # Eg s1 = "szbc1d1175758.fit"

    outpufileNameTempDatabase = "ap"//substr(line, 1, 4)//substr(line, 7, 13)   # Eg  = "
        szbc1175758"

    #temporarily move files in database - required for apall to continue to extract multiple
        aperatures
    tempfile = rawdirString//"database/"//outpufileNameTempDatabase
    copy (tempfile, tempfile//"TEMP")
    delete(tempfile)

    printf ("outpufileNameTempDatabase = %s\n", outpufileNameTempDatabase)
```

```
}


        #either use b_sampl="-12:-6, 6:12"  # (either side of WD spectra)
        #or use b_sampl= [or :sample -25:-15, 15:25] [or :sample -20:-10, 10:20] [or :sample
              -18:-10, 10:18] [or :sample -12:-6, 6:12]  [or :sample -12:-6, 11:17]  [or :
             sample -10:-6, 6:10] [or :sample -25:-10, 10:25] [or :sample -20:-10, 10:20]
             custom; different for each object
        #or use b_sampl="-100:-40, 40:100" (either side of DD spectra) [note this is the
             current method used for Gemini reduction]


            #OLD interactive options;
            #defaults/yes to all
            #select original selection, press d
            #select spectra line, press m
            #press b to look at background
            #:order 2
            #:sample -25:-15, 15:25      [or :sample -20:-10, 10:20] [or :sample -18:-10,
                 10:18] [or :sample -12:-6, 6:12]  [or :sample -12:-6, 11:17]  [or :sample
                 -10:-6, 6:10] [or :sample -25:-10, 10:25] [or :sample -20:-10, 10:20]
            #f
            #q
            #q
            #defaults/yes to all
            #delete all wild points, select and press d [that may result in bad fit]
            #:order 4
            #f
            #q
            #defaults/yes to all
            #q
            #[q]
            #rename new file to .fit afterwards



#8
#noao
#imred
#ccdred
ccdproc(images="@caliblamplistblue0.6.txt", output="@caliblamplistblue0.6zerocorrected.txt",
      biassec="[101:501, 4105:4190]", trimsec="[125:390,250:3750]", ccdtype="", fixpix-,
     oversca+, trim+, zerocor+, darkcor-, flatcor+, zero="Zero.fits", flat="nFlat0.6.fits",
     readaxis="column")
ccdproc(images="@caliblamplistblue1.0.txt", output="@caliblamplistblue1.0zerocorrected.txt",
      biassec="[101:501, 4105:4190]", trimsec="[125:390,250:3750]", ccdtype="", fixpix-,
     oversca+, trim+, zerocor+, darkcor-, flatcor+, zero="Zero.fits", flat="nFlat1.0.fits",
     readaxis="column")




#now restore database file that were temporarily moved
file_name = "sciencelistblue0.6zeroandbadpixcorrected1dS.txt"
flist = file_name
while (fscan(flist, line) != EOF) {

    printf ("line = %s\n", line)                              # Eg s1 = "szbc1d1175758.0001.
         fits"

    outpufileNameTempDatabase = "ap"//substr(line, 1, 4)//substr(line, 7, 13)   # Eg  = "
         szbc1175758"

    printf ("outpufileNameTempDatabase = %s\n", outpufileNameTempDatabase)

    tempfile = rawdirString//"database/"//outpufileNameTempDatabase

    copy (tempfile//"TEMP", tempfile)

}


file_name = "sciencelistblue1.0zeroandbadpixcorrected1dS.txt"
flist = file_name
while (fscan(flist, line) != EOF) {

    printf ("line = %s\n", line)                              # Eg s1 = "szbc1d1175758.0001.
         fits"

    outpufileNameTempDatabase = "ap"//substr(line, 1, 4)//substr(line, 7, 13)   # Eg  = "
         szbc1175758"

    printf ("outpufileNameTempDatabase = %s\n", outpufileNameTempDatabase)

    tempfile = rawdirString//"database/"//outpufileNameTempDatabase
```

```
    copy (tempfile//"TEMP", tempfile)

}



file_name = "sciencelistblue1.0zeroandbadpixcorrected1dDD.txt"
flist = file_name
while (fscan(flist, line) != EOF) {


    printf ("line = %s\n", line)                                  # Eg s1 = "szbc1d1175758.fit"

    outpufileNameTempDatabase = "ap"//substr(line, 1, 4)//substr(line, 7, 13)   # Eg  = "
        szbc1175758"

    printf ("outpufileNameTempDatabase = %s\n", outpufileNameTempDatabase)

    tempfile = rawdirString//"database/"//outpufileNameTempDatabase

    copy (tempfile//"TEMP", tempfile)


}



#9
#noao
#twodspec
#apextract
apall(input="@caliblamplistblue0.6zerocorrectedMappedToScience.txt", output="
     @caliblamplistblue0.6zerocorrectedMappedToScience1d.txt", format="onedspec", t_order=5,
      referen="@sciencelistblue0.6zeroandbadpixcorrected.txt", background="none", weights="
     variance", interac-)
apall(input="@caliblamplistblue1.0zerocorrectedMappedToScience.txt", output="
     @caliblamplistblue1.0zerocorrectedMappedToScience1d.txt", format="onedspec", t_order=5,
      referen="@sciencelistblue1.0zeroandbadpixcorrected.txt", background="none", weights="
     variance", interac-)

        #OLD interactive options;
        #default/yes to all
        #q
        #default/yes to all
        #q
        #default/yes to all
        #q
        #default/yes to all
        #rename new file to .fit afterwards


#10
#perform manual wavelength identification for two/three images
#noao
#twodspec
#longslit

file_name = "caliblamplistblue0.6zerocorrectedMappedToScience1d.0001.txt"
flist = file_name
junk = fscan(flist, line)   #first image from 0.6
copy(line, "identifyReferenceImage0.6.0001.fits")

file_name = "caliblamplistblue1.0zerocorrectedMappedToScience1d.0001.txt"
flist = file_name
junk = fscan(flist, line)   #first image from 1.0
copy(line, "identifyReferenceImage1.0.0001.fits")

file_name = "caliblamplistblue1.0zerocorrectedMappedToScience1d.0002.txt"
flist = file_name
junk = fscan(flist, line)   #first image from 1.0
copy(line, "identifyReferenceImage1.0.0002.fits")


identify(images="identifyReferenceImage0.6.0001.fits", function="legendre", order=4)
identify(images="identifyReferenceImage1.0.0001.fits", function="legendre", order=4)
identify(images="identifyReferenceImage1.0.0002.fits", function="legendre", order=4)



        #interactive options;
        #
        #(compare with tn125.pdf from;
        #---
        #http://www.ing.iac.es/Astronomy/instruments/isis/calibration_info.html
        #http://www.ing.iac.es/Astronomy/instruments/isis/index.html
        #   ccd EEV12   [obtain from website / fits header] [ccdtype EEV4280]
        #   R300B       [obtain from telescope usage log file / fits header]
        #   (CuNe + CUAR)   [obtain from fits header]
        #
        #---
```

```
        #CAGLAMPS   'CuNe+CuAr          '                                   Cass. A&G box
             comparison lamps
        #ISIGRAT    'R300B              '                                   Name of grating
        #       (or R1200 for red)
        #
        #   PD WHT Data 08 -Calibration Lamp Document Extract.png
        #   PD WHT Data 08 - Calibration Lamp Document Extract with IRAF Mappings.png
        #   PD WHT Data 08 -IRAF Calibration Lamp Mappings.png
        #   [from right to left];
        #
        #   5944.83
        #   5881.90
        #
        #   5400.56
        #   5330.78
        #   5218.20
        #   5105.54
        #   4965.08
        #   4806.02
        #   4589.90
        #
        #   4510.73
        #   4426.00
        #   4131.72
        #
        # default/yes to all
        # mark around 4 to 5 lines with m, and enter wavelength
        # (delete wrong lines with d)
        # f
        # q
        # mark more lines (up to around 10)
        # f
        # :order 4
        # f (delete any obviously bad points with d, there should not be any, or only 1 max
            - should be aiming for approximately 0.075 angstroms error max)
        # q
        # q
        # answer 'yes' to write to database


file_name = "sciencelistblue0.6zeroandbadpixcorrected1d.0001.txt"
file_name2 = "caliblamplistblue0.6zerocorrectedMappedToScience1d.0001.txt"
file_name3 = "sciencelistblue0.6zeroandbadpixcorrected1dwavelengthCalibrated.0001.txt"
flist = file_name
flist2 = file_name2
flist3 = file_name3
while (fscan(flist, line) != EOF) {

    junk = fscan(flist2, line2)
    junk = fscan(flist3, line3)

    printf ("line = %s\n", line)                          # Eg s1 = "szbc1d1175758.0001.
        fits"
    printf ("line2 = %s\n", line2)
    printf ("line3 = %s\n", line3)

    #10a
    #noao
    #twodspec
    #longslit
    reidentify(referen="identifyReferenceImage0.6.0001.fits", images=line2, interac-)

    #10b. (required according to  specred.dvi;)
    hedit(images=line, value=line2, fields="refspec1", add+, verify-)

    #10c. (required according to  specred.dvi;)
    dispcor(input=line, output=line3, w1=3600, w2=5250, dw=1, nw=INDEF)

}




file_name = "sciencelistblue1.0zeroandbadpixcorrected1d.0001.txt"
file_name2 = "caliblamplistblue1.0zerocorrectedMappedToScience1d.0001.txt"
file_name3 = "sciencelistblue1.0zeroandbadpixcorrected1dwavelengthCalibrated.0001.txt"
flist = file_name
flist2 = file_name2
flist3 = file_name3
while (fscan(flist, line) != EOF) {

    junk = fscan(flist2, line2)
    junk = fscan(flist3, line3)

    printf ("line = %s\n", line)                          # Eg s1 = "szbc1d1175758.0001.
        fits"
    printf ("line2 = %s\n", line2)
    printf ("line3 = %s\n", line3)
```

```
    #10a
    #noao
    #twodspec
    #longslit
    reidentify(referen="identifyReferenceImage1.0.0001.fits", images=line2, interac-)

    #10b. (required according to specred.dvi;)
    hedit(images=line, value=line2, fields="refspec1", add+, verify-)

    #10c. (required according to specred.dvi;)
    dispcor(input=line, output=line3, w1=3600, w2=5250, dw=1, nw=INDEF)

}

file_name = "sciencelistblue1.0zeroandbadpixcorrected1d.0002.txt"
file_name2 = "caliblamplistblue1.0zerocorrectedMappedToScience1d.0002.txt"
file_name3 = "sciencelistblue1.0zeroandbadpixcorrected1dwavelengthCalibrated.0002.txt"
flist = file_name
flist2 = file_name2
flist3 = file_name3
while (fscan(flist, line) != EOF) {

    junk = fscan(flist2, line2)
    junk = fscan(flist3, line3)

    printf ("line = %s\n", line)                              # Eg s1 = "szbc1d1175758.0001.
        fits"
    printf ("line2 = %s\n", line2)
    printf ("line3 = %s\n", line3)

    #10a
    #noao
    #twodspec
    #longslit
    reidentify(referen="identifyReferenceImage1.0.0002.fits", images=line2, interac-)

    #10b. (required according to specred.dvi;)
    hedit(images=line, value=line2, fields="refspec1", add+, verify-)

    #10c. (required according to specred.dvi;)
    dispcor(input=line, output=line3, w1=3600, w2=5250, dw=1, nw=INDEF)

}


#10d imcombine

imcombine(input="@listszbc1dwcComaWD.txt", output="szbc1dwcComaWD.fits")
imcombine(input="@listszbc1dwcWD1918-25July.txt", output="szbc1dwcWD1918-25July.fits")


imcombine(input="@listszbc1dwcSDSS1507a.txt", output="szbc1dwcSDSS1507a.fits")
imcombine(input="@listszbc1dwcSDSS1507b.txt", output="szbc1dwcSDSS1507b.fits")
imcombine(input="@listszbc1dwcSDSS1703a-24July.txt", output="szbc1dwcSDSS1703a-24July.fits")
imcombine(input="@listszbc1dwcSDSS1703b-24July.txt", output="szbc1dwcSDSS1703b-24July.fits")
imcombine(input="@listszbc1dwcWD1918-24July.txt", output="szbc1dwcWD1918-24July.fits")
imcombine(input="@listszbc1dwcSDSS2259a-24July.txt", output="szbc1dwcSDSS2259a-24July.fits")
imcombine(input="@listszbc1dwcSDSS2259b-24July.txt", output="szbc1dwcSDSS2259b-24July.fits")
imcombine(input="@listszbc1dwcSP2157-24July.txt", output="szbc1dwcSP2157-24July.fits")

imcombine(input="@listszbc1dwcSDSS1703a-25July.txt", output="szbc1dwcSDSS1703a-25July.fits")
imcombine(input="@listszbc1dwcSDSS1703b-25July.txt", output="szbc1dwcSDSS1703b-25July.fits")
imcombine(input="@listszbc1dwcSDSS2259a-25July.txt", output="szbc1dwcSDSS2259a-25July.fits")
imcombine(input="@listszbc1dwcSDSS2259b-25July.txt", output="szbc1dwcSDSS2259b-25July.fits")
imcombine(input="@listszbc1dwcSDSS0052a.txt", output="szbc1dwcSDSS0052a.fits")
imcombine(input="@listszbc1dwcSDSS0052b.txt", output="szbc1dwcSDSS0052b.fits")
imcombine(input="@listszbc1dwcSP2157-25July.txt", output="szbc1dwcSP2157-25July.fits")

imcombine(input="@listszbc1dwcSDSS1703a.txt", output="szbc1dwcSDSS1703a.fits")
imcombine(input="@listszbc1dwcSDSS1703b.txt", output="szbc1dwcSDSS1703b.fits")
imcombine(input="@listszbc1dwcSDSS2259a.txt", output="szbc1dwcSDSS2259a.fits")
imcombine(input="@listszbc1dwcSDSS2259b.txt", output="szbc1dwcSDSS2259b.fits")

#flux cal method part 1
wspectext("@listAllObjectsExceptForFluxCalDCCWDfits.txt", "
    @listAllObjectsExceptForFluxCalDCCWDdat.txt", head-, wformat="%.4f")

#10e CHECK ONLY
#noao
#onedspec
#splot file.fits

#11a. flux calibration {Use DC White Dwarf Spectra to calibrate science spectra flux}
#
#first interpolate over the DC White Dwarf Spectra to calibrate science spectra flux [
    interpolate over Ca H and Ca K lines]
#splot szbc1dwcWD1918-25July.fits
#t
#f
#:order 36  [order 50] [OLD: 100]
#f
```

```
        #:function spline3  [or spline1???]
        #f
        #:/color 2
        #f
        #:low_reject 3
        #:high_reject 3
        #f
        #[remove using d anomolies due to Ca, H and K???]
        #shift-? for help
        #:xyshow szbc1dwcWD1918-25JulyInterpolated.dat
        #    [may need to delete iraf/uparm/oncsplot.par afterwards]
        #
        #11b.
        # open temp.dat with openoffice calc [use fixed width, and set separators immediately after
            the end of the text of each column (not start)]
        # copy and paste all data {except first row if the header has not yet been removed} into a
            new calc file
        # save the calc file as szbc1dwcWD1918-25JulyInterpolated.dat (using file format .csv, and
            choose 'fixed width' when asked)
        #
        # perl fluxCalibrationUsingDCWD.perl > file.dat
        #
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcComaWD.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcComaWD.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1507a.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1507a.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1507b.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1507b.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and  szbc1dwcSDSS1703a-24July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1703a-24July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1703b-24July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1703b-24July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS2259a-24July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS2259a-24July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS2259b-24July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS2259b-24July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSP2157-24July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSP2157-24July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1703a-25July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1703a-25July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1703b-25July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1703b-25July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS2259a-25July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS2259a-25July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS2259b-25July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS2259b-25July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS0052a.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS0052a.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS0052b.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS0052b.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSP2157-25July.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSP2157-25July.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1703a.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1703a.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS1703b.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS1703b.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS2259a.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS2259a.dat
        #       edit fluxCalibrationUsingDCWD.perl and change var names (Eg szbc1dwcWD1918-25
            JulyInterpolated.dat, and szbc1dwcSDSS2259b.dat);
        #       perl fluxCalibrationUsingDCWD.perl > szbc1dwcfcSDSS2259b.dat

    }
end
```

A.4.4 Gemini South/North Data Reduction

Gemini Data was reduced using IRAF scripts (implementing Gemini GMOS IRAF software).

Gemini Reduction Software Installation

GMOS Software (version 1.10);
gemini_v110.tar.gz
GMOS Software additional IRAF packages required;

- color-bin.redhat.tar.gz
- color.tar.Z
- extern.pkg
- fitsutil-bin.redhat.tar.gz
- fitsutil.tar.gz
- gmisc-bin.redhat.tar.gz
- gmisc.tar.Z
- nmisc-bin.redhat.tgz
- nmisc.tar.Z
- stsdas3.10.bin.redhat.tar.gz
- stsdas3.10.tar.gz
- tables3.10.bin.redhat.tar.gz
- tables3.10.tar.gz

Instructions for Gemini software installation are found here;

```
for all packages x;
    extract extern source data for package x to /iraf/extern/x/
    extract extern bin data for package x to /iraf/extern/x/bin/
su iraf
cd $hlib    /   cd /iraf/iraf/unix/hlib
su root
chmod 777 extern.pkg
su rich
nedit extern.pkg; [and update the file's contents with the following]

# External (non core-system) packages.  To install a new package, add the
# two statements to define the package root directory and package task,
# then add the package helpdb to the 'helpdb' list.

reset   noao        = iraf$noao/
task    noao.pkg    = noao$noao.cl

reset   tables      = /iraf/extern/tables/
task    tables.pkg  = tables$tables.cl

reset   stsdas      = /iraf/extern/stsdas/
task    stsdas.pkg  = stsdas$stsdas.cl

reset   color       = /iraf/extern/color/
task    color.pkg   = color$color.cl

reset   fitsutil    = /iraf/extern/fitsutil/
task    fitsutil.pkg    = fitsutil$fitsutil.cl

reset   gemini      = /iraf/extern/gemini/
task    gemini.pkg  = gemini$gemini.cl

reset   gmisc       = /iraf/extern/gmisc/
task    gmisc.pkg   = gmisc$gmisc.cl

reset   nmisc       = /iraf/extern/nmisc/
task    nmisc.pkg   = nmisc$nmisc.cl

reset   helpdb      = "lib$helpdb.mip\
                      ,noao$lib/helpdb.mip\
                      ,tables$lib/helpdb.mip\
                      ,stsdas$lib/helpdb.mip\
                      ,color$lib/helpdb.mip\
                      ,fitsutil$lib/helpdb.mip\
                      ,gemini$lib/helpdb.mip\
                      ,gmisc$lib/helpdb.mip\
                      ,nmisc$lib/helpdb.mip\
                      "
```

```
keep
copy lacos_spec.cl to /home/rich/iraf/lacos_spec.cl
```

Reduction Archive Files

Reduced Files: geminiN2009Bdatafluxcalibrated23Sept09c-usingNewfcBB.zip Reduced Files: geminiS2010Adatafluxcalibrated-15June10a.zip Raw and Reduced Files: backupIRAFGeminiN2009BworkingDir23Sept09a (with CuAr_GMOS.dat.rbbmod5).zip Raw and Reduced Files: backupIRAFGeminiS2010AworkingDir15June10a.zip

Gemini North Reduced Files:

- cde1stgsN20090720S0067.dat
- cde1stgsN20090720S0067.fits
- cde1stgsN20090720S0067interpolated.dat
- cde1stgsN20090720S0083.dat
- cde1stgsN20090720S0083.fits
- cde1stgsN20090720S0094.dat
- cde1stgsN20090720S0094.fits
- cde2stgsN20090720S0067.dat
- cde2stgsN20090720S0067.fits
- cde2stgsN20090720S0083.dat
- cde2stgsN20090720S0083.fits
- cde2stgsN20090720S0094.dat
- cde2stgsN20090720S0094.fits
- CuAr_GMOS.dat.orig
- CuAr_GMOS.dat.rbbmod5
- fcde1stgsN20090720S0083.dat
- fcde1stgsN20090720S0094.dat
- fcde2stgsN20090720S0083.dat
- fcde2stgsN20090720S0094.dat
- flux cal example1.ods
- flux cal example2.ods
- fluxCalibrationUsingDCWD.perl
- geminireduce15Sept09a.cl
- gsextractmulti.cl

Gemini South Reduced Files:

- 2010A-DD-07A.dat
- 2010A-DD-07A.fits
- 2010A-DD-07B.dat
- 2010A-DD-07B.fits
- 2010A-WD0000-345A.dat
- 2010A-WD0000-345A.fits
- 2010A-WD0000-345Ainterpolated.dat

- 2010A-WD0000-345Atemp1interp.dat
- 2010A-WD0000-345Atemp2interp.dat
- 2010A-WD0000-345Atempinterp.dat
- 2010A-WD0000-345B.dat
- 2010A-WD0000-345B.fits
- CuAr_GMOS.dat.orig
- CuAr_GMOS.dat.rbbmod5
- f2010A-DD-07A.dat
- f2010A-DD-07A.ods
- f2010A-DD-07B.dat
- f2010A-DD-07B.ods
- fluxCalibrationUsingDCWD.perl
- geminireduce.cl
- geminireduce.cl.backup15June10a
- gsextractmulti.cl

Gemini North Raw File List:

- N20090716S0152_bias.fits
- N20090720S0062.fits
- N20090720S0063.fits
- N20090720S0064.fits
- N20090720S0065.fits
- N20090720S0066.fits
- N20090720S0067.fits
- N20090720S0068.fits
- N20090720S0069.fits
- N20090720S0070.fits
- N20090720S0071.fits
- N20090720S0077.fits
- N20090720S0078.fits
- N20090720S0079.fits
- N20090720S0080.fits
- N20090720S0081.fits
- N20090720S0082.fits
- N20090720S0083.fits
- N20090720S0084.fits
- N20090720S0085.fits
- N20090720S0086.fits
- N20090720S0087.fits
- N20090720S0088.fits
- N20090720S0089.fits
- N20090720S0090.fits
- N20090720S0091.fits
- N20090720S0092.fits

- N20090720S0093.fits
- N20090720S0094.fits
- N20090725S0267.fits
- N20090725S0268.fits
- N20090725S0269.fits
- N20090725S0270.fits
- N20090725S0271.fits
- N20090725S0272.fits
- N20090725S0273.fits
- N20090725S0274.fits

Gemini South Raw File List:

- gS20100512S0284_bias.fits
- S20100303S0071.fits
- S20100303S0072.fits
- S20100303S0073.fits
- S20100303S0074.fits
- S20100303S0075.fits
- S20100303S0076.fits
- S20100303S0077.fits
- S20100303S0092.fits
- S20100512S0216.fits
- S20100512S0217.fits
- S20100512S0218.fits
- S20100512S0219.fits
- S20100512S0220.fits
- S20100512S0221.fits
- S20100512S0222.fits
- S20100512S0223.fits
- S20100512S0224.fits
- S20100512S0225.fits
- S20100512S0226.fits
- S20100512S0227.fits
- S20100512S0228.fits
- S20100512S0230.fits
- S20100512S0231.fits
- S20100512S0232.fits
- S20100512S0233.fits
- S20100512S0284.fits
- S20100512S0285.fits
- S20100512S0286.fits
- S20100512S0287.fits
- S20100512S0288.fits
- S20100520S0203.fits

- S20100520S0204.fits
- S20100520S0205.fits
- S20100520S0206.fits
- S20100520S0207.fits
- S20100520S0208.fits
- S20100520S0209.fits
- S20100520S0210.fits
- S20100520S0211.fits
- S20100520S0212.fits
- S20100520S0213.fits
- S20100520S0214.fits
- S20100520S0215.fits
- S20100520S0216.fits
- S20100520S0217.fits
- S20100522S0045.fits
- S20100522S0046.fits
- S20100522S0047.fits
- S20100522S0048.fits
- S20100522S0049.fits

geminireduce.cl

```
procedure geminireduce()

string file_name
struct *flist
string file_name2
struct *flist2
string file_name3
struct *flist3


begin

    # RBB gemini reduction script 27 May 10a[supports gemini 1.10] [cosmic rays removal using
        lacos_spec.cl];
    # instructions;
    #   install iraf with gemini 1.10 and all required packages [http://www.gemini.edu/sciops/data-
        and-results/processing-software]
    #   cd /home/rich/iraf
    #   delete all files in /home/rich/iraf/uparm
    #   delete all temporary files in /home/rich/iraf/database/
    #   delete all temporary files in /home/rich/iraf/
    #   add to /home/rich/iraf/login.cl;
    #       gemini
    #       gmos
    #       noao
    #       onedspec
    #       task geminireduce=/home/rich/iraf/geminireduce.cl
    #       task lacos_spec=/home/rich/iraf/lacos_spec.cl
    #       task gsextractmulti=/home/rich/iraf/gsextractmulti.cl
    #   cp /iraf/extern/gemini/gmos/data/CuAr_GMOS.dat /iraf/extern/gemini/gmos/data/CuAr_GMOS.dat.
        backup
    #   cp CuAr_GMOS.dat.rbbmod5 /iraf/extern/gemini/gmos/data/CuAr_GMOS.dat
    #   cl
    #   geminireduce
    #       use cosmic ray removal method 5 (lacos spec after bias subtraction, flat field)
    #       not 1 (gemini gsrej),
    #       not 2 (lacos spec after bias subtraction, flat field and combined),
    #       not 3 (lacos spec before bias subtraction)
    #       not 4 (lacos spec after bias subtraction)
    #
    #   cp


    #critical parameters to set;
```

```
int observationNumber = 2    #1 = GMOSN 2009B, 2 = GMOSS 2010A, 3 = GMOSN 2010A
string rawdirString="/media/newharddisk/iraf/"



int numberOfObjects = 0
int reverseextractspectraordering

int firstObject = 1
int firstScience = 1

int o
int s

string flatNumber[9]     #max 3objects * max 3sciencedata
string arcNumber[9]
string scienceNumber[9]
int numberOfScienceImages[3]     #max 3objects
string objectName[9]

string flatNumberString
string arcNumberString
string scienceNumberString

string basescienceNumberString
string scombineInputString1
string scombineInputString2

string rawflatfilename
string rawarcfilename
string rawsciencefilename
string flatfilename
string combinedbiasfilename[3]
string arcfilenamecalibrated
string sciencefilenamecalibrated
string arcfilenamecalibratedNoFits
string sciencefilenamewavelengthcalibrated
string sciencefilenamewavelengthcalibratedskysubtracted
string sciencefilenamewavelengthcalibratedskysubtracted1dTemp
string sciencefilenamewavelengthcalibratedskysubtracted1d
string sciencefilenamewavelengthcalibratedskysubtracted1d1
string sciencefilenamewavelengthcalibratedskysubtracted1d2
string sciencefilenamewavelengthcalibratedskysubtracted1d1dispcored
string sciencefilenamewavelengthcalibratedskysubtracted1d2dispcored
#string sciencefilenamewavelengthcalibratedskysubtracted1d1dat
#string sciencefilenamewavelengthcalibratedskysubtracted1d2dat
string sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombined
string sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombined
string sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombineddat
string sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombineddat

string rawsciencefilenameafterLacosSpecCosmicRayRemovalMethod3
string rawsciencefilenamebeforeLacosSpecCosmicRayRemovalMethod3
string sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4
string sciencefilenameBiasSubtractedBeforeLacosSpecMethodCosmicRayRemoval4
string sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5
string sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip1
string sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip2
string sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip3
string sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod2

string tempfile

objectName[0*3 + 1] = "2009B-WD1918+38"
objectName[0*3 + 2] = "2009B-DD-08"
objectName[0*3 + 3] = "2009B-DD-20"

objectName[1*3 + 1] = "2010A-DD-07"
objectName[1*3 + 2] = "2010A-WD0000-345"
objectName[1*3 + 3] = "notused"

objectName[2*3 + 1] = "WD1918+38"
objectName[2*3 + 2] = "unknown"
objectName[2*3 + 3] = "notused"


if(observationNumber == 1)
{#GMOSN 2009B
    numberOfObjects = 3
    numberOfScienceImages[1] = 3
    numberOfScienceImages[2] = 3
    numberOfScienceImages[3] = 3
}
else if(observationNumber == 2)
{#GMOSS 2010A
    numberOfObjects = 2
    numberOfScienceImages[1] = 3
    numberOfScienceImages[2] = 2
    numberOfScienceImages[3] = 0
}
else if(observationNumber == 3)
{#GMOSN 2010A
```

```
    numberOfObjects = 2
    numberOfScienceImages[1] = 3
    numberOfScienceImages[2] = 3
    numberOfScienceImages[3] = 0
}

if(observationNumber == 1)
{#GMOSN 2009B


    flatNumber[0*3 + 1] = "N20090720S0065"
    arcNumber[0*3 + 1] = "N20090720S0066"
    scienceNumber[0*3 + 1] = "N20090720S0067"
    flatNumber[0*3 + 2] = "N20090720S0065"
    arcNumber[0*3 + 2] = "N20090720S0066"
    scienceNumber[0*3 + 2] = "N20090720S0068"
    flatNumber[0*3 + 3] = "N20090720S0065"
    arcNumber[0*3 + 3] = "N20090720S0066"
    scienceNumber[0*3 + 3] = "N20090720S0069"

    combinedbiasfilename[1] = "N20090716S0152_bias.fits"

    flatNumber[1*3 + 1] = "N20090720S0081"
    arcNumber[1*3 + 1] = "N20090720S0082"
    scienceNumber[1*3 + 1] = "N20090720S0083"
    flatNumber[1*3 + 2] = "N20090720S0081"
    arcNumber[1*3 + 2] = "N20090720S0082"
    scienceNumber[1*3 + 2] = "N20090720S0084"
    flatNumber[1*3 + 3] = "N20090720S0081"
    arcNumber[1*3 + 3] = "N20090720S0082"
    scienceNumber[1*3 + 3] = "N20090720S0085"

    combinedbiasfilename[2] = "N20090716S0152_bias.fits"

    flatNumber[2*3 + 1] = "N20090720S0092"
    arcNumber[2*3 + 1] = "N20090720S0093"
    scienceNumber[2*3 + 1] = "N20090720S0094"
    flatNumber[2*3 + 2] = "N20090725S0274"
    arcNumber[2*3 + 2] = "N20090725S0273"
    scienceNumber[2*3 + 2] = "N20090725S0271"
    flatNumber[2*3 + 3] = "N20090725S0274"
    arcNumber[2*3 + 3] = "N20090725S0273"
    scienceNumber[2*3 + 3] = "N20090725S0272"

    combinedbiasfilename[3] = "N20090716S0152_bias.fits"
}
else if(observationNumber == 2)
{#GMOSS 2010A


    flatNumber[0*3 + 1] = "S20100512S0224"  #S20100512S0224
    arcNumber[0*3 + 1] = "S20100512S0233"
    scienceNumber[0*3 + 1] = "S20100512S0221"
    flatNumber[0*3 + 2] = "S20100512S0224"  #S20100512S0224
    arcNumber[0*3 + 2] = "S20100512S0233"
    scienceNumber[0*3 + 2] = "S20100512S0222"
    flatNumber[0*3 + 3] = "S20100512S0224"  #S20100512S0224
    arcNumber[0*3 + 3] = "S20100512S0233"
    scienceNumber[0*3 + 3] = "S20100512S0223"

        #this works also;
        #combinedbiasfilename[1] = "combinedBias2010A.fits"
        #gbias(inimages="@rawBList2010A.txt", outbias=combinedbiasfilename[1])

    #this works;
    combinedbiasfilename[1] = "gS20100512S0284_bias"    #this is wrong

        #this doesnt work;
        #combinedbiasfilename[1] = "gS20100508S0338_bias"   #this is wrong

        #combinedbiasfilename[1] = "combinedBias1.fits"     #gS20100508S0338_bias
        #gbias(inimages="gS20100508S0338_bias.fits", outbias=combinedbiasfilename[1])

        #combinedbiasfilename[1] = "combinedBias1.fits"
        #gbias(inimages=scienceNumber[0*3 + 1], outbias=combinedbiasfilename[1], fl_over=yes)



    flatNumber[1*3 + 1] = "S20100520S0208"
    arcNumber[1*3 + 1] = "S20100520S0216"
    scienceNumber[1*3 + 1] = "S20100520S0206"
    flatNumber[1*3 + 2] = "S20100520S0208"
    arcNumber[1*3 + 2] = "S20100520S0216"
    scienceNumber[1*3 + 2] = "S20100520S0207"

        #this works also;
        #combinedbiasfilename[2] = "combinedBias2010A.fits"

    #this works;
    combinedbiasfilename[2] = "gS20100512S0284_bias"    #this is wrong
```

```
        #this doesnt work;
        #combinedbiasfilename[2] = "gS20100520S0250_bias"   # this is wrong

        #combinedbiasfilename[2] = "combinedBias2.fits"     #gS20100520S0250_bias
        #gbias(inimages="gS20100520S0250_bias.fits", outbias=combinedbiasfilename[2])

        #combinedbiasfilename[2] = "combinedBias2.fits"
        #gbias(inimages=scienceNumber[1*3 + 1], outbias=combinedbiasfilename[2], fl_over=yes)

        #exit

}
else if(observationNumber == 3)
{#GMOSN 2010A


    #use DC white dwarf for flux calibration from previous gemini north semester observation
    flatNumber[0*3 + 1] = "N20090720S0065"
    arcNumber[0*3 + 1] = "N20090720S0066"
    scienceNumber[0*3 + 1] = "N20090720S0067"
    flatNumber[0*3 + 2] = "N20090720S0065"
    arcNumber[0*3 + 2] = "N20090720S0066"
    scienceNumber[0*3 + 2] = "N20090720S0068"
    flatNumber[0*3 + 3] = "N20090720S0065"
    arcNumber[0*3 + 3] = "N20090720S0066"
    scienceNumber[0*3 + 3] = "N20090720S0069"

    combinedbiasfilename[1] = "N20090716S0152_bias"

    flatNumber[1*3 + 1] = "unknown"
    arcNumber[1*3 + 1] = "unknown"
    scienceNumber[1*3 + 1] = "unknown"
    flatNumber[1*3 + 2] = "unknown"
    arcNumber[1*3 + 2] = "unknown"
    scienceNumber[1*3 + 2] = "unknown"
    flatNumber[1*3 + 3] = "unknown"
    arcNumber[1*3 + 3] = "unknown"
    scienceNumber[1*3 + 3] = "unknown"

    combinedbiasfilename[2] = "unknown"
}



for (o=firstObject; o <= numberOfObjects; o+=1)
{

    basescienceNumberString = scienceNumber[(o-1)*3 + 1]

    sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombined = ""//objectName[(
        observationNumber-1)*3 + o]//"A"//".fits"
    sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombined = ""//objectName[(
        observationNumber-1)*3 + o]//"B"//".fits"
    sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombineddat = ""//objectName[(
        observationNumber-1)*3 + o]//"A"//".dat"
    sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombineddat = ""//objectName[(
        observationNumber-1)*3 + o]//"B"//".dat"

    for (s=firstScience; s <= numberOfScienceImages[o]; s+=1)
    {


        flatNumberString = flatNumber[(o-1)*3 + s]
        arcNumberString = arcNumber[(o-1)*3 + s]
        scienceNumberString = scienceNumber[(o-1)*3 + s]

        rawflatfilename = ""//flatNumberString//".fits"
        rawarcfilename = ""//arcNumberString//".fits"
        rawsciencefilename = ""//scienceNumberString//".fits"
        flatfilename = ""//flatNumberString//"_flat.fits"
        arcfilenamecalibrated = "gs"//arcNumberString//".fits"
        sciencefilenamecalibrated = "gs"//scienceNumberString//".fits"

        arcfilenamecalibratedNoFits = "gs"//arcNumberString
        sciencefilenamewavelengthcalibrated = "tgs"//scienceNumberString//".fits"
        sciencefilenamewavelengthcalibratedskysubtracted = "stgs"//scienceNumberString//".fits"
        sciencefilenamewavelengthcalibratedskysubtracted1dTemp = "apstgs"//scienceNumberString
            //"_SCI_1_"
        sciencefilenamewavelengthcalibratedskysubtracted1d = "estgs"//scienceNumberString//".
            fits"
        sciencefilenamewavelengthcalibratedskysubtracted1d1 = "e1stgs"//scienceNumberString//".
            fits"
        sciencefilenamewavelengthcalibratedskysubtracted1d2 = "e2stgs"//scienceNumberString//".
            fits"
        sciencefilenamewavelengthcalibratedskysubtracted1d1dispcored = "de1stgs"//
            scienceNumberString//".fits"
        sciencefilenamewavelengthcalibratedskysubtracted1d2dispcored = "de2stgs"//
            scienceNumberString//".fits"
```

```
#sciencefilenamewavelengthcalibratedskysubtracted1d1dat = "e1stgs"//scienceNumberString
    //".dat"
#sciencefilenamewavelengthcalibratedskysubtracted1d2dat = "e2stgs"//scienceNumberString
    //".dat"

rawsciencefilenameafterLacosSpecCosmicRayRemovalMethod3 = ""//scienceNumberString//"
    afterLacosSpecCosmicRayRemoval.fits"
rawsciencefilenamebeforeLacosSpecCosmicRayRemovalMethod3 = ""//scienceNumberString//"
    beforeLacosSpecCosmicRayRemoval.fits"
sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4 = "gs"//
    scienceNumberString//"biassubtractedwithoutrays.fits"
sciencefilenameBiasSubtractedBeforeLacosSpecMethodCosmicRayRemoval4 = "gs"//
    scienceNumberString//"biassubtractedwithrays.fits"
sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5 = "gs"//
    scienceNumberString//"withrays.fits"
sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip1 = "gs"//
    scienceNumberString//"withrayschip1.fits"
sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip2 = "gs"//
    scienceNumberString//"withrayschip2.fits"
sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip3 = "gs"//
    scienceNumberString//"withrayschip3.fits"
sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod2 = "gs"//
    scienceNumberString//"withrays.fits"



# set up the logfile for this reduction
gmos.logfile="geminireductionlogfile.txt"


# Make the flat field -- bias has been overscan corrected

gsflat (rawflatfilename, flatfilename, order=23, fl_bias+, bias=combinedbiasfilename[o],
     niter=5)   #order=23



    #COSMIC RAY REMOVAL METHOD ADDITION 3b/4b;
    #copy original raw science file (without cosmic ray removal);
    #cp(rawsciencefilename, rawsciencefilenamebeforeLacosSpecCosmicRayRemovalMethod3)


    #COSMIC RAY REMOVAL METHOD 3;
    #delete("temp1.fits")
    #delete("temp2.fits")
    #delete("temp3.fits")
    #delete("bad1.fits")
    #delete("bad2.fits")
    #delete("bad3.fits")
    #lacos_spec (input=rawsciencefilename//"[1]", output="temp1.fits", outmask="bad1.
        fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, yorder=6, gain
        =2.337, readn=3.3, niter=5)
    #lacos_spec (input=rawsciencefilename//"[2]", output="temp2.fits", outmask="bad2.
        fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, yorder=6, gain
        =2.337, readn=3.3, niter=5)
    #lacos_spec (input=rawsciencefilename//"[3]", output="temp3.fits", outmask="bad3.
        fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, yorder=6, gain
        =2.337, readn=3.3, niter=5)
    #imsum(input="temp1.fits", output=rawsciencefilename//"[1,overwrite]")
    #imsum(input="temp2.fits", output=rawsciencefilename//"[2,overwrite]")
    #imsum(input="temp3.fits", output=rawsciencefilename//"[3,overwrite]")
    #hedit(images=sciencefilenamecalibrated//"[1]", add+, fields="EXTVER", value="-1",
        verify-)
    #hedit(images=sciencefilenamecalibrated//"[2]", add+, fields="EXTVER", value="-1",
        verify-)
    #hedit(images=sciencefilenamecalibrated//"[3]", add+, fields="EXTVER", value="-1",
        verify-)
    #cp(rawsciencefilename, rawsciencefilenameafterLacosSpecCosmicRayRemovalMethod3)

gscut(flatfilename, fl_update+, gradimage=flatfilename)

    #COSMIC RAY REMOVAL METHOD 1; if using gemini (not lacos_spec) cosmic ray
        elimiination;
    gscrrej.datares=5.0 #if this is raised, more false positives
    #gscrrej.fnsigma=8  #if this is raised, less false positives
    #gscrrej.tnsigma=10 #if this is raised, les s false positives
    #gscrrej.niter=5
    #if using gemini (not lacos_spec) cosmic ray elimiination; add this to gsreduce
        fl_gscrrej+


    #COSMIC RAY REMOVAL METHOD 4;
    #delete("temp1.fits")
    #delete("temp2.fits")
    #delete("temp3.fits")
    #delete("bad1.fits")
    #delete("bad2.fits")
    #delete("bad3.fits")
        #do bias subtract
    #gsreduce(rawsciencefilename, rawpath=rawdirString, fl_flat-, fl_bias+, bias=
        combinedbiasfilename[o], fl_gmosaic-, fl_fixpix-, fl_gscrrej-, fl_cut-)
        #remove cosmic rays lacos spec;
```

```
    #cp(sciencefilenamecalibrated,
        sciencefilenameBiasSubtractedBeforeLacosSpecMethodCosmicRayRemoval4)
                #old before 4b
                #cp(sciencefilenamecalibrated,
                    sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4)
    #new after 4b
    #cp(rawsciencefilename,
        sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4)
    #lacos_spec (input=sciencefilenamecalibrated//"[1]", output="temp1.fits", outmask="
        bad1.fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, yorder=6,
        gain=2.337, readn=3.3, niter=5)
    #lacos_spec (input=sciencefilenamecalibrated//"[2]", output="temp2.fits", outmask="
        bad2.fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, yorder=6,
        gain=2.337, readn=3.3, niter=5)
    #lacos_spec (input=sciencefilenamecalibrated//"[3]", output="temp3.fits", outmask="
        bad3.fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, yorder=6,
        gain=2.337, readn=3.3, niter=5)
    #imsum(input="temp1.fits", output=
        sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4//"[1,
        overwrite]")
    #imsum(input="temp2.fits", output=
        sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4//"[2,
        overwrite]")
    #imsum(input="temp3.fits", output=
        sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4//"[3,
        overwrite]")
    #hedit(images=sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4
        //"[1]", add+, fields="EXTVER", value="-1", verify-)
    #hedit(images=sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4
        //"[2]", add+, fields="EXTVER", value="-1", verify-)
    #hedit(images=sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4
        //"[3]", add+, fields="EXTVER", value="-1", verify-)
    #delete(sciencefilenamecalibrated)
        #flat field subtract and combine images
                #old before 4b
                #gsreduce(
                     sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4,
                      outimag=sciencefilenamecalibrated, rawpath=rawdirString, fl_flat+,
                      fl_bias-, flatim=flatfilename, fl_gmosaic+, fl_fixpix-, fl_gscrrej
                     -, fl_cut+)
    #new after 4b
    #cp(sciencefilenameBiasSubtractedAfterLacosSpecMethodCosmicRayRemoval4,
        rawsciencefilename)
    #gsreduce(rawsciencefilename, rawpath=rawdirString, fl_flat+, fl_bias-, flatim=
        flatfilename, fl_gmosaic+, fl_fixpix-, fl_gscrrej-, fl_cut+)


#NOT COSMIC RAY REMOVAL METHOD 4;

#print, "flatfilename = ", flatfilename

gsreduce(rawsciencefilename, fl_flat+, fl_bias+, bias=combinedbiasfilename[o], flatim=
    flatfilename, fl_fixpix-, fl_gscrrej-)

#exit

    #COSMIC RAY REMOVAL METHOD 2;
    #delete("temp.fits")
    #delete("bad.fits")
    #lacos_spec(input=sciencefilenamecalibrated//"[2]", output="temp.fits", outmask="bad
        .fits", verbose+, sigclip=7.5, sigfra=0.5, objlim=1, xorder=9, order=9, gain
        =2.337, readn=3.3)
    #cp(sciencefilenamecalibrated,
        sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod2)
    #imsum(input="temp.fits", output=sciencefilenamecalibrated//"[2,overwrite]")
    #hedit(images=sciencefilenamecalibrated//"[2]", add+, fields="EXTNAME", value="SCI
            " verify-)
    #hedit(images=sciencefilenamecalibrated//"[2]", add+, fields="EXTVER", value="1"
        verify-)

    #COSMIC RAY REMOVAL METHOD 5;
    delete("temp1.fits")
    delete("temp2.fits")
    delete("temp3.fits")
    delete("bad1.fits")
    delete("bad2.fits")
    delete("bad3.fits")
    cp(sciencefilenamecalibrated,
        sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5)
    imcopy (sciencefilenamecalibrated//"[sci][1:511,1:1152]",
        sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip1)
    imcopy (sciencefilenamecalibrated//"[sci][522:1033,1:1152]",
        sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip2)
    imcopy (sciencefilenamecalibrated//"[sci][1045:1554,1:1152]",
        sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip3)
        # x/width=1554
        # y/height=1152
        # 511-10-512-11-510
        # x/width=544
        # y/height=1152
```

```
    if(observationNumber == 1)
    {
        lacos_spec (input=
            sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip1,
            output="temp1.fits", outmask="bad1.fits", verbose+, sigclip=7.5, sigfra
            =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
        lacos_spec (input=
            sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip2,
            output="temp2.fits", outmask="bad2.fits", verbose+, sigclip=7.5, sigfra
            =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
        lacos_spec (input=
            sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip3,
            output="temp3.fits", outmask="bad3.fits", verbose+, sigclip=7.5, sigfra
            =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
    }
    else
    {
        if(observationNumber == 2)
        {
            lacos_spec (input=
                sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip1,
                output="temp1.fits", outmask="bad1.fits", verbose+, sigclip=7.5, sigfra
                =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
            lacos_spec (input=
                sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip2,
                output="temp2.fits", outmask="bad2.fits", verbose+, sigclip=7.5, sigfra
                =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
            lacos_spec (input=
                sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip3,
                output="temp3.fits", outmask="bad3.fits", verbose+, sigclip=7.5, sigfra
                =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
        }
        else
        {
            lacos_spec (input=
                sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip1,
                output="temp1.fits", outmask="bad1.fits", verbose+, sigclip=7.5, sigfra
                =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
            lacos_spec (input=
                sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip2,
                output="temp2.fits", outmask="bad2.fits", verbose+, sigclip=7.5, sigfra
                =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)
            lacos_spec (input=
                sciencefilenamecalibratedBeforeLacosSpecCosmicRayRemovalMethod5chip3,
                output="temp3.fits", outmask="bad3.fits", verbose+, sigclip=7.5, sigfra
                =0.5, objlim=1, xorder=100, yorder=6, gain=2.337, readn=3.3, niter=5)

        }
    }
    imcopy ("temp1.fits", sciencefilenamecalibrated//"[sci][1:511,1:1152]")
    imcopy ("temp2.fits", sciencefilenamecalibrated//"[sci][522:1033,1:1152]")
    imcopy ("temp3.fits", sciencefilenamecalibrated//"[sci][1045:1554,1:1152]")

# Reduce the CuAr B600/500.  The CuAr is not flat fielded,
# gaps are not fixpixed.


gsreduce(rawarcfilename, fl_flat-, fl_bias+, bias=combinedbiasfilename[o], fl_fixpix-,
    fl_gscrrej-)


# For CuAr taken through the 2.0 arcsec slit must smooth the data so that
# accurate line centers can be found by GSWAVELENGTH.  Also need to adjust
# the line width, centering radius and minimum separation parameters.

# Establish the wavelength calibration
#gswavelength(arcfilenamecalibrated, nsum=5, fwidth=11, gsigma=1.5, cradius=5, threshold
    =1000, trace-, fitcxord=4, fitcyord=3, fl_inter-)
gswavelength(arcfilenamecalibrated, trace-, fl_inter-)

# Transform the CuAr spectrum, for checking that the transformation is OK.
# Output image name is defined by the default outpref="t"
gstransform(arcfilenamecalibrated, wavtran=arcfilenamecalibratedNoFits)

# Transform the science spectrum
gstransform(sciencefilenamecalibrated, wavtran=arcfilenamecalibratedNoFits)

# Sky subtraction, the sample is manually selected after inspection of
# the image
gsskysub(sciencefilenamewavelengthcalibrated, long_sample="490:535,650:695", order=4,
    fl_inter-)
    #gsskysub sciencefilenamewavelengthcalibrated long_sample="580:590,605:615" order=4
        fl_inter-

# Extract both spectra
gsextractmulti(sciencefilenamewavelengthcalibratedskysubtracted, apwidth=1, fl_inter-,
    find+, recenter+, trace-, aperatureNumToBeExtracted=1)

mv(sciencefilenamewavelengthcalibratedskysubtracted1d,
    sciencefilenamewavelengthcalibratedskysubtracted1d1)
    #delete temporary files in database
    tempfile = rawdirString//"database/aplast"
```

```
        delete(tempfile)
        tempfile = rawdirString//"database/"//
            sciencefilenamewavelengthcalibratedskysubtracted1dTemp
        delete(tempfile)

    if(observationNumber == 1)
    {
        if(o == 3)
        {
            reverseextractspectraordering = 1
        }
        else
        {
            reverseextractspectraordering = 0
        }
    }
    else
    {
        reverseextractspectraordering = 0

        #if(observationNumber == 2)
        #{
        #    if(o == 2)
        #    {
        #        reverseextractspectraordering = 1
        #    }
        #    else
        #    {
        #        reverseextractspectraordering = 0
        #    }
        #}
    }

    gsextractmulti(sciencefilenamewavelengthcalibratedskysubtracted, apwidth=1, fl_inter-,
        find+, recenter+, trace-, aperatureNumToBeExtracted=2,
        reversegsextractspectraordering=reverseextractspectraordering)
    mv(sciencefilenamewavelengthcalibratedskysubtracted1d,
        sciencefilenamewavelengthcalibratedskysubtracted1d2)
        #delete temporary files in database
        tempfile = rawdirString//"database/aplast"
        delete(tempfile)
        tempfile = rawdirString//"database/"//
            sciencefilenamewavelengthcalibratedskysubtracted1dTemp
        delete(tempfile)

    #COSMIC RAY REMOVAL METHOD ADDITION 3b/4b;
    #restore original raw science file (without cosmic ray removal);
    #cp(rawsciencefilenamebeforeLacosSpecCosmicRayRemovalMethod3, rawsciencefilename)

    #now prepare science spectras for combination (dispcor)
    dispcor(input=sciencefilenamewavelengthcalibratedskysubtracted1d1//"[sci]", output=
        sciencefilenamewavelengthcalibratedskysubtracted1d1dispcored//"[sci]", w1=3600, w2
        =5250, dw=1.0, nw=INDEF)
    dispcor(input=sciencefilenamewavelengthcalibratedskysubtracted1d2//"[sci]", output=
        sciencefilenamewavelengthcalibratedskysubtracted1d2dispcored//"[sci]", w1=3600, w2
        =5250, dw=1.0, nw=INDEF)

    if(s == firstScience)
    {
        scombineInputString1 = sciencefilenamewavelengthcalibratedskysubtracted1d1dispcored
            //"[sci]"
        scombineInputString2 = sciencefilenamewavelengthcalibratedskysubtracted1d2dispcored
            //"[sci]"
    }
    else
    {
        scombineInputString1 = scombineInputString1//","//
            sciencefilenamewavelengthcalibratedskysubtracted1d1dispcored//"[sci]"
        scombineInputString2 = scombineInputString2//","//
            sciencefilenamewavelengthcalibratedskysubtracted1d2dispcored//"[sci]"
    }


}

#now combine the three 1D science spectras... (scombine)
scombine(input=scombineInputString1, output=
    sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombined//"[sci]")
scombine(input=scombineInputString2, output=
    sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombined//"[sci]")

#flux cal method part 1
wspectext(sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombined//"[sci]",
    sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombineddat, head-, wformat
    ="%.4f")
wspectext(sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombined//"[sci]",
    sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombineddat, head-, wformat
    ="%.4f")

# Inspect the extracted spectrum [temporarily removed due to iraf error "INTERNAL ERROR:
    write error within fprop()"]
#splot(sciencefilenamewavelengthcalibratedskysubtracted1d1dispcoredCombined//"[sci]")
```

```
        #splot(sciencefilenamewavelengthcalibratedskysubtracted1d2dispcoredCombined//"[sci]")

    }

end

#flux cal method part 2 [this example is for observation number 1/2 only].
#first interpolate over the DC White Dwarf Spectra to calibrate science spectra flux [interpolate
     over Ca H and Ca K lines]
#splot 2009B-WD1918+38A.fits[sci]
#splot 2010A-WD0000-345A.fits[sci]
#t
#f
#:order 36  [order 50] [OLD: 100]
#f
#:function spline3  [or spline1???]
#f
#:/color 2
#f
#:low_reject 3
#:high_reject 3
#f
#[remove using d anomolies due to Ca (H and K lines)]
#(NB shift-? for help)
#:xyshow 2009B-WD1918+38Ainterpolated.dat
#:xyshow 2010A-WD0000-345Ainterpolated.dat
# remove first line (header) from DC WD data (Eg 2009B-WD1918+38Ainterpolated.dat)

#rename 2009B-WD1918+38Ainterpolated.dat to temp.dat
#open temp.dat with openoffice calc [use fixed width, and set separators immediately after the end
     of the text of each column (not start)]
#copy and paste all data except first row(header) into a new calc file
#save the calc file as 2009B-WD1918+38Ainterpolated.dat/2010A-WD0000-345Ainterpolated.dat (using
     file format .csv, and choose 'fixed width' when asked)

#flux cal method part 3.
#edit perl and change input var names (2009B-WD1918+38Ainterpolated.dat, and 2009B-DD-08A.dat);
#perl fluxCalibrationUsingDCWD.perl > f2009B-DD-08A.dat
#edit perl and change input var names (2009B-WD1918+38Ainterpolated.dat, and 2009B-DD-08B.dat);
#perl fluxCalibrationUsingDCWD.perl > f2009B-DD-08B.dat
#edit perl and change input var names (2009B-WD1918+38Ainterpolated.dat, and 2009B-DD-20A.dat);
#perl fluxCalibrationUsingDCWD.perl > f2009B-DD-20A.dat
#edit perl and change input var names (2009B-WD1918+38Ainterpolated.dat, and 2009B-DD-20B.dat);
#perl fluxCalibrationUsingDCWD.perl > f2009B-DD-20B.dat

#OR
#edit perl and change input var names (2010A-WD0000-345Ainterpolated.dat, and 2010A-DD-07A.dat);
#perl fluxCalibrationUsingDCWD.perl > f2010A-DD-07A.dat
#edit perl and change input var names (2010A-WD0000-345Ainterpolated, and 2010A-DD-07B.dat);
#perl fluxCalibrationUsingDCWD.perl > f2010A-DD-07B.dat
```

gsextractmulti.cl

gsextractmulti.cl is a modified version of gsextract.cl enabling the extraction of multiple spectral lines from a single long slit image.

A.4.5 VLT Data Reduction

VLT Data reduced using IRAF by PDD. RBB VLT data reductions not used due to Gasgano pipeline lacos_spec cosmic ray removal integration difficulties.

Reduction Archive Files

richard.tar.gz

A.4.6 Flux Calibration

Flux Calibration using DC WD (PDD Code).

fluxCalibrationUsingDCWD.perl

```
#!/usr/bin/perl
#
use Carp;
use POSIX;
#
open (input,"2010A-WD0000-345Ainterpolated.dat");
#
$bbt=6240.0;
$pc=6.626e-27;
$c=2.9979e10;
$bol=1.381e-16;
#
$pcc=$pc*$c;
$c2=$c*$c;
my @response;
#
#
$cnt=-1;
<input>;
while (<input>) {
    ($wav,$crap1,$cts,$crap2)=split " ",$_;  #use y fit data, not y data from DC white dwarf
    $owav=$wav;
    $wav=$wav*1e-8;
    $bflux=2*($pc/($wav**5.0))*$c2 * (exp($pcc/($bol*$bbt)/$wav) -
1.0)**-1.0;
    $out=sprintf("%8.2f %12.6e\n",$owav,$bflux/$cts);
    $cnt++;
    $response[$cnt]=$bflux/$cts/5e14;
#     printf $out;
}
close(input);
#
open (input,"2010A-DD-07B.dat");
$cnt=-1;
while (<input>) {
    $cnt++;
    ($wav,$cts)=split " ",$_;     #use y data, not y fit data from object
    $flux=$response[$cnt]*$cts;
    $out=sprintf("%8.2f %12.6e\n",$wav,$flux);
    printf $out;
}
close (input);
```

A.4.7 Signal to Noise Calculations

This program calculates the signal to noise of a set of reduced 1D WD spectra.

calculateSignalToNoise.perl

```
; Title: Calculate Signal to Noise
; File: calculateSignalToNoise.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 12 November 10 6pm (updated 15 Mar 11)
; Summary:
;
;   Postconditions;
;
;   This program calculates the signal to noise of a set of ASCII Tables Using GDL
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   calculateSignalToNoise.pro [NASA]
;   SSWD0XA/B.txt [Wavelength, Flux] for 12 WDs (A/B), tab delimited
;   instrumentResolution.txt for X WDs

pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
```

```
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
                l_blue    : 12,$
                l_red     : 13 $
               }
;         0   1    2    3    4    5    6    7    8    9   10   11   12   13
tvlct,[   0,255, 255,   0,   0,255,255,   0,255,125,125,   0,   0,255],$ ;red
      [   0,255,   0,255,   0,255,   0,255,125,255,   0,255,125,   0],$ ;green
      [   0,255,   0,   0,255,   0,255,255,   0,   0,255,125,255,125]    ;blue
end


my_colors


plotTitle = 'Spectra Flux'
plotXAxisTitle = 'Wavelength (Angstroms)'
plotYAxisTitle = 'Amplitude'
plotXmin = 3600
plotXmax = 5200
plotYmin = 0
plotYmax = 1.5
fileNameBase1 = 'SSWD'
fileNameBase2 = 'B'     ;change to B for other half
asciiTablefileNameExtension = '.dat'
psGraphfileNameExtension = '.ps'
pdfGraphfileNameExtension = '.pdf'
headerNumLines = 0
multipleFiles = 1
numberFiles = 12
multipleSubFiles = 0
numberSubFiles = 1
numYcols = 1
numPlots=numberSubFiles
lineStyleY = intarr(numPlots)
lineStyleY[0] = 0
psymY = intarr(numPlots)
psymY[0]=-0
normalise = 1
specifyYRange = 0
firstWavelengthForNoiseLevelCalculation = 4600
lastWavelengthForNoiseLevelCalculation = 4800

formatString = 'D'
for yIndex=1, numYcols, 1 do begin
    formatString = formatString + ',D'
endfor

PRINT, "formatString = ", formatString

if(multipleFiles eq 1) then begin

    instrumentResolutionFileName = 'instrumentResolution.txt'
    PRINT, "instrumentResolutionFileName = ", instrumentResolutionFileName
    readcol64bitvars, instrumentResolutionFileName, instrumentResolutionArray, FORMAT='D',SKIPLINE=1
    PRINT, "instrumentResolutionArray = ", instrumentResolutionArray

    if(numYcols eq 1) then begin

    endif else begin
        PRINT, "error numYcols > 1 not supported with multipleFiles"
        exit
    endelse

    signalToNoiseForAllFiles = dblarr(numberFiles)

    for systemIndex=1, numberFiles, 1 do begin

        if(numberFiles gt 1) then begin
            systemIndexString = String(systemIndex, Format='(I2.2)')
            PRINT,systemIndexString
        endif else begin
            systemIndexString = ''
        endelse

        plot, [0], [0], xtitle=plotXAxisTitle, ytitle=plotYAxisTitle, xrange=[plotXmin, plotXmax]

        asciiTablefileName = fileNameBase1 + systemIndexString + fileNameBase2 +
            asciiTablefileNameExtension

        PRINT, "asciiTablefileName = ", asciiTablefileName

        readcol64bitvars, asciiTablefileName, Xarray, Yarray, FORMAT=formatString,SKIPLINE=
            headerNumLines

        wavelengthInterval = Xarray(2) - Xarray(1)
        firstPointForNoiseLevelCalculation=(firstWavelengthForNoiseLevelCalculation-plotXmin)/
            wavelengthInterval
```

```
        lastPointForNoiseLevelCalculation=firstPointForNoiseLevelCalculation + (
            lastWavelengthForNoiseLevelCalculation-firstWavelengthForNoiseLevelCalculation)/
            wavelengthInterval
        numberOfPointsForNoiseLevelCalculation=(lastPointForNoiseLevelCalculation-
            firstPointForNoiseLevelCalculation)

        ;INTERPOLATION METHOD1
        ;smoothedYArray = SMOOTH(Yarray, 10.0)
        ;INTERPOLATION METHOD2
        ;smoothedYArray = INTERPOL(Yarray, Xarray, Xarray, /SPLINE) ;requires IDL to execute (GDL
            does not have spline.pro)
        ;INTERPOLATION METHOD3
        ;YarraySplineInterpolationParameters = SPL_INIT(Xarray, Yarray)
        ;smoothedYArray = SPL_INTERP(Xarray, Yarray, YarraySplineInterpolationParameters, Xarray)


        ;PRINT, "smoothedYArray = ", smoothedYArray

        smoothedYArraySmall = dblarr(numberOfPointsForNoiseLevelCalculation)
        YarraySmall = dblarr(numberOfPointsForNoiseLevelCalculation)
        XarraySmall = dblarr(numberOfPointsForNoiseLevelCalculation)
        for i=0, numberOfPointsForNoiseLevelCalculation-1, 1 do begin
            YarraySmall(i) = Yarray(firstPointForNoiseLevelCalculation + i)
            XarraySmall(i) = Xarray(firstPointForNoiseLevelCalculation + i)
            ;smoothedYArraySmall(i) = smoothedYArray(firstPointForNoiseLevelCalculation + i)
        endfor

        ;INTERPOLATION METHOD4
        polyFitOrder = 3
        YarraySmallSplinePolyParameters = poly_fit(XarraySmall, YarraySmall, polyFitOrder)
        smoothedYArraySmall = poly(XarraySmall, YarraySmallSplinePolyParameters)

        noiseArraySmall = smoothedYArraySmall-YArraySmall

        ;oplot, Xarray, Yarray
        oplot, XarraySmall, smoothedYArraySmall
        oplot, XarraySmall, noiseArraySmall
        ;oplot, Xarray, smoothedYarray
        oplot, Xarray, Yarray

        averageSignal = mean(smoothedYArraySmall)
        standardDeviation = stddev(noiseArraySmall)
        RMS = standardDeviation
        PRINT, "RMS = ", RMS
        SignalToNoisePerPixel = averageSignal/RMS
        resolutionElementsPerPixel = instrumentResolutionArray(systemIndex-1)    ;eg 3 instrument
            resolution/
        SignalToNoisePerResolutionElement = SignalToNoisePerPixel/SQRT(resolutionElementsPerPixel)
        SignalToNoise = SignalToNoisePerResolutionElement
        PRINT, "SignalToNoisePerResolutionElement = ", SignalToNoisePerResolutionElement

        wait, 3

        ;noiseArray = YArray/smoothedYArray
        ;;PRINT, noiseArray
        ;now calculate the standard deviation
        ;averageNoiseRatio = 0.0
        ;for index=firstPointForNoiseLevelCalculation, lastPointForNoiseLevelCalculation, 1 do begin
        ;    ;PRINT, "index = ", index
        ;    ;PRINT, "firstPointForNoiseLevelCalculation = ", firstPointForNoiseLevelCalculation
        ;    ;PRINT, "lastPointForNoiseLevelCalculation = ", lastPointForNoiseLevelCalculation
        ;
        ;    averageNoiseRatio = averageNoiseRatio + (noiseArray(index))
        ;endfor
        ;averageNoiseRatio = averageNoiseRatio/numberOfPointsForNoiseLevelCalculation
        ;
        ;deviationsArray = noiseArray    ;easy declaration
        ;deviationsSquaredArray = noiseArray     ;easy declaration
        ;sumOfDeviationsSquared = 0.0
        ;for index=firstPointForNoiseLevelCalculation, lastPointForNoiseLevelCalculation, 1 do begin
        ;    deviationsArray(index) = noiseArray(index) - averageNoiseRatio
        ;    deviationsSquaredArray(index) = (deviationsArray(index))^2
        ;    sumOfDeviationsSquared = sumOfDeviationsSquared + deviationsSquaredArray(index)
        ;endfor
        ;
        ;standardDeviationSquared = sumOfDeviationsSquared/(numberOfPointsForNoiseLevelCalculation
            -1)
        ;standardDeviation = standardDeviationSquared^0.5
        ;RMS = standardDeviation
        ;signalToNoise = 1.0/RMS
        ;;PRINT,  "standardDeviation = ", standardDeviation
        ;;PRINT,  "signalToNoise = ", signalToNoise
        signalToNoiseForAllFiles(systemIndex-1) = signalToNoise

    endfor

    PRINT, "signalToNoiseForAllFiles = "
    for systemIndex=1, numberFiles, 1 do begin
        PRINT, signalToNoiseForAllFiles(systemIndex-1)
    endfor
endif
```

```
end
```

A.5 Stellar Modelling Scripts

A.5.1 Balmer Line Extraction

The Balmer Line Extraction software is used to verify the wavelength calibration of reduced (spectroscopic) WD data. It also serves for some basic normalisation purposes (NOT USED).

'RBBBalmerLineExtraction11Sept09a.pro'

RBBBalmerLineExtraction 13 Sept 09a.zip

```
; Title: RBB Balmer Line Extraction
; File: RBBBalmerLineExtraction.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 31 July 09 3:00pm
; Summary:
;
;   Postconditions;
;
;   This program calculates the X positions of the balmer lines for a WD flux calibrated .dat file -
     such that normalisation can be applied and models can be fitted [2 step process]
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   szbc1dwcfcSDSS0052a.dat [wavelengthArray, amplitudeArray], space delimited
;   Usage: gdl
;   .run RBBBalmerLineExtraction.pro


pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
                l_blue    : 12,$
                l_red     : 13 $
               }
;         0   1    2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]   ;blue
end

my_colors

numberBalmerLines = 6
numberBalmerLinesUsed = 5
balmerLinesLabXPos = dblarr(numberBalmerLines)
;balmerLinesLabXPos[6] = 6562.852    ;H-Alpha - not used
balmerLinesLabXPos[0] = 4861.33      ;H-Beta
balmerLinesLabXPos[1] = 4340.47      ;H-Gamma
balmerLinesLabXPos[2] = 4101.74      ;H-Delta
balmerLinesLabXPos[3] = 3970.072     ;H-Epsilon
balmerLinesLabXPos[4] = 3889.049     ;H-8
balmerLinesLabXPos[5] = 3835.384     ;H-..
```

```
balmerLineOffsetMax = (20)              ;+/-20 angstroms - subject to max radial velocity
balmerLineOffsetIncrement = (1.0)           ;0.1 angstroms - subject to max precision available
balmerLineHalfWidthUsedForCentreCalculation = dblarr(numberBalmerLines)
;balmerLineHalfWidthUsedForCentreCalculation[6] = 20     ;H-Alpha     ;50 angstrom
balmerLineHalfWidthUsedForCentreCalculation[0] = 60 ;80 ;100     ;H-Beta
balmerLineHalfWidthUsedForCentreCalculation[1] = 60 ;80 ;100     ;H-Gamma
balmerLineHalfWidthUsedForCentreCalculation[2] = 35 ;50 ;60 ;H-Delta
balmerLineHalfWidthUsedForCentreCalculation[3] = 15 ;30 ;40 ;H-Epsilon
balmerLineHalfWidthUsedForCentreCalculation[4] = 7  ;15 ;25 ;H-8
balmerLineHalfWidthUsedForCentreCalculation[5] = 8  ;10 ;25 ;H-9

;Normalisation Method 1 Parameters Only
balmerLineHalfWidthUsedForCentreCalculationRatioToPointUsedForNormalisationMethod1 = 0.75    ;0.6

;Normalisation Method 2 Parameters Only
balmerLinesLabXNormPointMethod2 = dblarr(numberBalmerLines, 2)
balmerLinesLabXNormPointMethod2Width = dblarr(numberBalmerLines, 2)
;NB left and right normalisation points are always the same distance from the centre
balmerLinesLabXNormPointMethod2[0, 0] = 120         ;H-Beta
balmerLinesLabXNormPointMethod2[1, 0] = 120         ;H-Gamma
balmerLinesLabXNormPointMethod2[2, 0] = 70          ;H-Delta
balmerLinesLabXNormPointMethod2[3, 0] = 30          ;H-Epsilon
balmerLinesLabXNormPointMethod2[4, 0] = 30          ;H-8
balmerLinesLabXNormPointMethod2[5, 0] = 20          ;H-..
balmerLinesLabXNormPointMethod2Width[0, 0] = 100       ;H-Beta
balmerLinesLabXNormPointMethod2Width[1, 0] = 100       ;H-Gamma
balmerLinesLabXNormPointMethod2Width[2, 0] = 30        ;H-Delta
balmerLinesLabXNormPointMethod2Width[3, 0] = 20        ;H-Epsilon
balmerLinesLabXNormPointMethod2Width[4, 0] = 20        ;H-8
balmerLinesLabXNormPointMethod2Width[5, 0] = 20        ;H-9
balmerLinesLabXNormPointMethod2[0, 1] = balmerLinesLabXNormPointMethod2[0, 0]            ;H-Beta
balmerLinesLabXNormPointMethod2[1, 1] = balmerLinesLabXNormPointMethod2[1, 0]                ;H-Gamma
balmerLinesLabXNormPointMethod2[2, 1] = balmerLinesLabXNormPointMethod2[2, 0]                ;H-Delta
balmerLinesLabXNormPointMethod2[3, 1] = balmerLinesLabXNormPointMethod2[3, 0]                ;H-
    Epsilon
balmerLinesLabXNormPointMethod2[4, 1] = balmerLinesLabXNormPointMethod2[4, 0]                ;H-8
balmerLinesLabXNormPointMethod2[5, 1] = balmerLinesLabXNormPointMethod2[5, 0]                ;H-..
balmerLinesLabXNormPointMethod2Width[0, 1] = balmerLinesLabXNormPointMethod2Width[0, 0] ;100 ;H-Beta
balmerLinesLabXNormPointMethod2Width[1, 1] = balmerLinesLabXNormPointMethod2Width[1, 0]     ;100 ;H-
    Gamma
balmerLinesLabXNormPointMethod2Width[2, 1] = balmerLinesLabXNormPointMethod2Width[2, 0]       ;H-
    Delta
balmerLinesLabXNormPointMethod2Width[3, 1] = balmerLinesLabXNormPointMethod2Width[3, 0]       ;50
        ;H-Epsilon
balmerLinesLabXNormPointMethod2Width[4, 1] = balmerLinesLabXNormPointMethod2Width[4, 0]       ;20
        ;H-8
balmerLinesLabXNormPointMethod2Width[5, 1] = balmerLinesLabXNormPointMethod2Width[5, 0]       ;H-9

;Normalisation Method 3 Parameters Only
balmerLinesLabXNormPointMethod3Width = dblarr(numberBalmerLines)
balmerLinesLabXNormPointMethod3Width[0] = 140       ;H-Beta
balmerLinesLabXNormPointMethod3Width[1] = 140       ;H-Gamma
balmerLinesLabXNormPointMethod3Width[2] = 90        ;H-Delta
balmerLinesLabXNormPointMethod3Width[3] = 60        ;H-Epsilon
balmerLinesLabXNormPointMethod3Width[4] = 40        ;H-8
balmerLinesLabXNormPointMethod3Width[5] = 30        ;H-9



numberOfbalmerLineOffsets = long(balmerLineOffsetMax/balmerLineOffsetIncrement)
;PRINT, "numberOfbalmerLineOffsets = "
;PRINT, numberOfbalmerLineOffsets

fitBothSidesSimultaneously = 0
polyFitOrder = 2
if (fitBothSidesSimultaneously eq 1) then begin
    polyFitOrder = 2
endif else begin
    polyFitOrder = 2
endelse


minWavelength = 3600;
maxWavelength = 5250;
interval = 1.0;
SOME_LARGE_VALUE = 999999.0;

fluxCalibratedWDDatFileName = "fcde1stgsN20090720S0083.dat" ;"szbc1dwcfcSDSS0052a.dat"
readcol64bitvars, fluxCalibratedWDDatFileName, wavelengthArray, amplitudeArray, FORMAT='D,D'

wavelengthErrorsAll = dblarr(numberOfbalmerLineOffsets, numberBalmerLines)
wavelengthErrorsTotalAcrossAllBalmerLines = dblarr(numberOfbalmerLineOffsets)


; Stage A - calculate best poly fit and calculate centre points

for balmerLineWavelengthOffsetIndex=0, numberOfbalmerLineOffsets-1, 1 do begin

    balmerLineWavelengthOffset = (balmerLineWavelengthOffsetIndex-(numberOfbalmerLineOffsets/2))*
        balmerLineOffsetIncrement
```

```
;PRINT, "balmerLineWavelengthOffset = "
;PRINT, balmerLineWavelengthOffset

plotTitle = 'balmerLineWavelengthOffsetIndex:' + STRING(FORMAT='(I3)',
    balmerLineWavelengthOffsetIndex) + ', balmerLineWavelengthOffset=' + STRING(FORMAT='(F0.2)
    ', balmerLineWavelengthOffset)

plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,maxWavelength],
    xtitle='wavelength (Ang)', ytitle='amplitude', COLOR=140  ;yrange=[-0.8,0.8],linestyle=5,
    psym=3, SYMSIZE=1

PRINT, "balmerLineWavelengthOffset", balmerLineWavelengthOffset

totalWaveLengthErrorAcrossAllBalmerLines = 0.0

for balmerLineIndex=0, numberBalmerLinesUsed-1, 1 do begin

    centreSubsetIndex = long((balmerLinesLabXPos[balmerLineIndex]-minWavelength+
        balmerLineWavelengthOffset)/interval)
    wavelengthAtCentre = balmerLinesLabXPos[balmerLineIndex]+balmerLineWavelengthOffset
    subsetNumValues = long(balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex]/interval
        )
    leftSubsetIndex = (centreSubsetIndex - subsetNumValues)
    leftSubsetNumValues = subsetNumValues
    rightSubsetNumValues = subsetNumValues
    rightSubsetIndex = centreSubsetIndex + subsetNumValues

    wavelengthAtMinimaTheory = 0.0

    wavelengthSubset = wavelengthArray(leftSubsetIndex:rightSubsetIndex)
    amplitudeSubset = amplitudeArray(leftSubsetIndex:rightSubsetIndex)

    if fitBothSidesSimultaneously eq 1 then begin
        ;alternatively, fit both sides simultaneously

        subsetPolyFitParameters = poly_fit(wavelengthSubset, amplitudeSubset, polyFitOrder)

        amplitudeSubsetTheory = poly(wavelengthSubset, subsetPolyFitParameters)
        fitError = amplitudeSubsetTheory - amplitudeSubset
            ;centre = subsetPolyFitParameters(2)     ; third parameter
        indexOfMinima = min(amplitudeSubset)     ; the index of the minima
        amplitudeAtMinima = amplitudeSubset(indexOfMinima)  ; the amplitude of the minima
        ;now find the fit x position at this amplitude..

        closestDiff = SOME_LARGE_VALUE
        closestIndex = 0
        for tempindex=0, (subsetNumValues*2-1), 1 do begin
            currentDiff = abs(amplitudeSubsetTheory(tempindex) - amplitudeAtMinima)
            if(currentDiff lt closestDiff) then begin
                closestDiff = currentDiff
                closestIndex =  tempindex
            endif
        endfor

        oplot, wavelengthSubset, amplitudeSubsetTheory, COLOR=230

        wavelengthAtMinimaTheory = amplitudeSubsetTheory(tempindex)

    endif else begin

        ;line left side polyfit

        leftwavelengthSubset = wavelengthArray(leftSubsetIndex:centreSubsetIndex)
        leftamplitudeSubset = amplitudeArray(leftSubsetIndex:centreSubsetIndex)

        leftSubsetPolyFitParameters = poly_fit(leftwavelengthSubset, leftamplitudeSubset,
            polyFitOrder)
        leftamplitudeSubsetTheory = poly(wavelengthSubset, leftSubsetPolyFitParameters)
            ;PRINT, "size(wavelengthArray)= "
            ;PRINT, size(wavelengthArray)
        ;leftfitError = leftamplitudeSubsetTheory - leftamplitudeSubset
            ;leftcentre = leftSubsetPolyFitParameters(2)     ; third parameter
        temp = min(amplitudeSubset, indexOfMinima)  ; the index of the minima
        temp = max(amplitudeSubset, indexOfMaxima)
        amplitudeAtMinima = amplitudeSubset(indexOfMinima)  ; the amplitude of the minima
        wavelengthAtMinima = wavelengthSubset(indexOfMinima)
        amplitudeAtMaxima = amplitudeSubset(indexOfMaxima)  ; the amplitude of the minima
        wavelengthAtMaxima = wavelengthSubset(indexOfMaxima)

        ;now find the fit x position at this amplitude..
        leftclosestDiff = SOME_LARGE_VALUE
        leftclosestIndex = 0
        for lefttempindex=0, (subsetNumValues*2-1), 1 do begin
            ;PRINT, "amplitudeSubsetTheory(lefttempindex)= "
            ;PRINT, amplitudeSubsetTheory(lefttempindex)
            leftcurrentDiff = abs(leftamplitudeSubsetTheory(lefttempindex) - amplitudeAtMinima)
            if(leftcurrentDiff lt leftclosestDiff) then begin
                leftclosestDiff = leftcurrentDiff
                leftclosestIndex =  lefttempindex
            endif
        endfor
        leftwavelengthAtMinimaTheory = wavelengthSubset(leftclosestIndex)
```

```
            ;line right side polyfit
        rightwavelengthSubset = wavelengthArray(centreSubsetIndex:rightSubsetIndex)
        rightamplitudeSubset = amplitudeArray(centreSubsetIndex:rightSubsetIndex)
        rightSubsetPolyFitParameters = poly_fit(rightwavelengthSubset, rightamplitudeSubset,
            polyFitOrder)
        rightamplitudeSubsetTheory = poly(wavelengthSubset, rightSubsetPolyFitParameters)

        ;now find the fit x position at this amplitude..
        rightclosestDiff = SOME_LARGE_VALUE
        rightclosestIndex = 0
        for righttempindex=0, (subsetNumValues*2-1), 1 do begin
            ;PRINT, "amplitudeSubsetTheory(righttempindex)= "
            ;PRINT, amplitudeSubsetTheory(righttempindex)
            rightcurrentDiff = abs(rightamplitudeSubsetTheory(righttempindex) -
                amplitudeAtMinima)
            if(rightcurrentDiff lt rightclosestDiff) then begin
                rightclosestDiff = rightcurrentDiff
                rightclosestIndex =  righttempindex
            endif
        endfor
        rightwavelengthAtMinimaTheory = wavelengthSubset(rightclosestIndex)

        oplot, wavelengthSubset, leftamplitudeSubsetTheory,  col = !col.red
        oplot, wavelengthSubset, rightamplitudeSubsetTheory,  col = !col.blue

        if (leftwavelengthAtMinimaTheory gt rightwavelengthAtMinimaTheory) then begin
            ;wavelengthAtMinimaTheory = SOME_LARGE_VALUE
            ;PRINT, "quite a bad fit; leftwavelengthAtMinimaTheory gt
                rightwavelengthAtMinimaTheory"
            wavelengthAtMinimaTheory = (leftwavelengthAtMinimaTheory +
                rightwavelengthAtMinimaTheory) / 2.0
        endif else begin
            wavelengthAtMinimaTheory = (leftwavelengthAtMinimaTheory +
                rightwavelengthAtMinimaTheory) / 2.0
        endelse

        leftOptDisplayAmplitude = dblarr(2)
        leftOptDisplayWavelength = dblarr(2)
        rightOptDisplayAmplitude = dblarr(2)
        rightOptDisplayWavelength = dblarr(2)
        minDisplayAmplitude = dblarr(2)
        minDisplayWavelength = dblarr(2)
        centreDisplayAmplitude = dblarr(2)
        centreDisplayWavelength = dblarr(2)
        leftOptDisplayAmplitude[0] = 1.5
        leftOptDisplayAmplitude[1] = 0.0
        leftOptDisplayWavelength[0] = leftwavelengthAtMinimaTheory
        leftOptDisplayWavelength[1] = leftwavelengthAtMinimaTheory
        rightOptDisplayAmplitude[0] = 1.5
        rightOptDisplayAmplitude[1] = 0.0
        rightOptDisplayWavelength[0] = rightwavelengthAtMinimaTheory
        rightOptDisplayWavelength[1] = rightwavelengthAtMinimaTheory
        minDisplayAmplitude[0] = 1.5
        minDisplayAmplitude[1] = 0.0
        minDisplayWavelength[0] = wavelengthAtMinima
        minDisplayWavelength[1] = wavelengthAtMinima
        centreDisplayAmplitude[0] = 1.5
        centreDisplayAmplitude[1] = 0.0
        centreDisplayWavelength[0] = wavelengthAtCentre
        centreDisplayWavelength[1] = wavelengthAtCentre
        oplot, leftOptDisplayWavelength, leftOptDisplayAmplitude, col = !col.red
        oplot, rightOptDisplayWavelength, rightOptDisplayAmplitude, col = !col.blue
        ;oplot, minDisplayWavelength, minDisplayAmplitude, col = !col.green
        oplot, centreDisplayWavelength, centreDisplayAmplitude, col = !col.orange

            ; CANNOT DO IN IDL - can do this in Matlab; constrain the polyfits with the centre
                line
            ; calculate the midpoint between the two fits, and measure the deviation of the mid
                point from the ideal midpoint

    endelse


    wavelengthAtMinimaTheoryError = abs(wavelengthAtMinimaTheory - balmerLinesLabXPos[
        balmerLineIndex])
    wavelengthErrorsAll(balmerLineWavelengthOffsetIndex, balmerLineIndex) =
        wavelengthAtMinimaTheoryError
    totalWaveLengthErrorAcrossAllBalmerLines = totalWaveLengthErrorAcrossAllBalmerLines +
        wavelengthAtMinimaTheoryError

endfor

PRINT, "Waiting for 1 second to display progress graph..."
WAIT, 5

wavelengthErrorsTotalAcrossAllBalmerLines(balmerLineWavelengthOffsetIndex) =
    totalWaveLengthErrorAcrossAllBalmerLines


PRINT, "totalWaveLengthErrorAcrossAllBalmerLines = ", totalWaveLengthErrorAcrossAllBalmerLines
```

```
endfor

    ; now find the fit (the given wavelength offset) which gave the minium total wavelength error
        across all balmer lines

minError = SOME_LARGE_VALUE
minErrorbalmerLineWavelengthIndex = 0
minErrorbalmerLineWavelengthOffset = 0
for balmerLineWavelengthOffsetIndex=0, numberOfbalmerLineOffsets-1, 1 do begin

    balmerLineWavelengthOffset = (balmerLineWavelengthOffsetIndex-(numberOfbalmerLineOffsets/2))/
        balmerLineOffsetIncrement

    if (wavelengthErrorsTotalAcrossAllBalmerLines(balmerLineWavelengthOffsetIndex) lt minError) then
         begin
        minError = wavelengthErrorsTotalAcrossAllBalmerLines(balmerLineWavelengthOffsetIndex)
        minErrorbalmerLineWavelengthIndex = balmerLineWavelengthOffsetIndex
        minErrorbalmerLineWavelengthOffset = balmerLineWavelengthOffset
    endif

endfor

PRINT, "min error = ", minError
PRINT, "minErrorbalmerLineWavelengthIndex = ", minErrorbalmerLineWavelengthIndex
PRINT, "minErrorbalmerLineWavelengthOffset = ", minErrorbalmerLineWavelengthOffset


; Stage B - calculate normalisation points and perform normalisation

    ;print out final best poly fits

    ;at a given amplitude value, calculate the wavelength difference from the centre between the
        left and right fits of each balmer line fit, and adjust the amplitude of the data on each/
        one side of the balmer line to normalise the balmer line (use a linear shear?)


balmerLineWavelengthOffsetIndex = minErrorbalmerLineWavelengthIndex
balmerLineWavelengthOffset = minErrorbalmerLineWavelengthOffset

plotTitle = 'balmerLineWavelengthOffsetIndex:' + STRING(FORMAT='(I3)',
    balmerLineWavelengthOffsetIndex) + ', balmerLineWavelengthOffset=' + STRING(FORMAT='(F0.2)',
    balmerLineWavelengthOffset)

plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,maxWavelength], xtitle
    ='wavelength (Ang)', ytitle='amplitude', COLOR=140  ;yrange=[-0.8,0.8],linestyle=5,psym=3,
    SYMSIZE=1

for balmerLineIndex=0, numberBalmerLinesUsed-1, 1 do begin

    centreSubsetIndex = long((balmerLinesLabXPos[balmerLineIndex]-minWavelength+
        balmerLineWavelengthOffset)/interval)
    subsetNumValues = long(balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex]/interval)
    leftSubsetIndex = (centreSubsetIndex - subsetNumValues)
    leftSubsetNumValues = subsetNumValues
    rightSubsetNumValues = subsetNumValues
    rightSubsetIndex = centreSubsetIndex + subsetNumValues

        ;line left side polyfit

    leftwavelengthSubset = wavelengthArray(leftSubsetIndex:centreSubsetIndex)
    leftamplitudeSubset = amplitudeArray(leftSubsetIndex:centreSubsetIndex)
    leftSubsetPolyFitParameters = poly_fit(leftwavelengthSubset, leftamplitudeSubset, polyFitOrder)
    leftamplitudeSubsetTheory = poly(leftwavelengthSubset, leftSubsetPolyFitParameters)

        ;line right side polyfit

    rightwavelengthSubset = wavelengthArray(centreSubsetIndex:rightSubsetIndex)
    rightamplitudeSubset = amplitudeArray(centreSubsetIndex:rightSubsetIndex)
    rightSubsetPolyFitParameters = poly_fit(rightwavelengthSubset, rightamplitudeSubset,
        polyFitOrder)
    rightamplitudeSubsetTheory = poly(rightwavelengthSubset, rightSubsetPolyFitParameters)

    oplot, leftwavelengthSubset, leftamplitudeSubsetTheory,  COLOR=230
    oplot, rightwavelengthSubset, rightamplitudeSubsetTheory,  COLOR=230

endfor

PRINT, "Waiting for 10 seconds to display balmer line centre calculation graph (B1a, B2a, B3a...)"
WAIT, 3


plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,maxWavelength], xtitle
    ='wavelength (Ang)', ytitle='amplitude', COLOR=140  ;yrange=[-0.8,0.8],linestyle=5,psym=3,
    SYMSIZE=1


for balmerLineIndex=0, numberBalmerLinesUsed-1, 1 do begin

    centreSubsetIndex = long((balmerLinesLabXPos[balmerLineIndex]-minWavelength+
        balmerLineWavelengthOffset)/interval)
    subsetNumValues = long(balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex]/interval)
    leftSubsetIndex = (centreSubsetIndex - subsetNumValues)
```

```
    leftSubsetNumValues = subsetNumValues
    rightSubsetNumValues = subsetNumValues
    rightSubsetIndex = centreSubsetIndex + subsetNumValues

        ;line left side polyfit
    leftwavelengthSubset = wavelengthArray(leftSubsetIndex:centreSubsetIndex)
    leftamplitudeSubset = amplitudeArray(leftSubsetIndex:centreSubsetIndex)
    leftSubsetPolyFitParameters = poly_fit(leftwavelengthSubset, leftamplitudeSubset, polyFitOrder)
    leftamplitudeSubsetTheory = poly(leftwavelengthSubset, leftSubsetPolyFitParameters)

        ;line right side polyfit
    rightwavelengthSubset = wavelengthArray(centreSubsetIndex:rightSubsetIndex)
    rightamplitudeSubset = amplitudeArray(centreSubsetIndex:rightSubsetIndex)
    rightSubsetPolyFitParameters = poly_fit(rightwavelengthSubset, rightamplitudeSubset,
        polyFitOrder)
    rightamplitudeSubsetTheory = poly(rightwavelengthSubset, rightSubsetPolyFitParameters)

    ;now find the fit x position at an arbitrary amplitude,
        balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex]/2.0
    ;at a given amplitude value, calculate the wavelength difference from the centre between the
        left and right fits of each balmer line fit,

    wavelengthCentre = balmerLinesLabXPos[balmerLineIndex]+balmerLineWavelengthOffset
    leftPosArray = dblarr(1)
    leftWavelength = wavelengthCentre - (balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex
        ]*balmerLineHalfWidthUsedForCentreCalculationRatioToPointUsedForNormalisationMethod1)
    leftPosArray[0] = leftWavelength
    rightPosArray = dblarr(1)
    rightWavelength = wavelengthCentre + (balmerLineHalfWidthUsedForCentreCalculation[
        balmerLineIndex]*
        balmerLineHalfWidthUsedForCentreCalculationRatioToPointUsedForNormalisationMethod1)
    rightPosArray[0] = rightWavelength
    leftValArray = poly(leftPosArray, leftSubsetPolyFitParameters)
    rightValArray = poly(rightPosArray, rightSubsetPolyFitParameters)
    leftVal = leftValArray[0]
    rightVal = rightValArray[0]

    ;adjust the amplitude of the data on each/one side of the balmer line to normalise the balmer
        line (use a linear shear?)
    differenceBetweenAmplitudeValues = leftVal - rightVal

    PRINT, "differenceBetweenAmplitudeValues = ", differenceBetweenAmplitudeValues

    ;adjust the data such that leftVal = leftVal - differenceBetweenAmplitudeValues/2,  and rightVal
         = rightVal + differenceBetweenAmplitudeValues/2
    ;create a linear shear function - for a given wavelength, multiply by some function of the
        wavelength diff

    amplitudeShearPerLambda = differenceBetweenAmplitudeValues / ((double(
        balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex])*
        balmerLineHalfWidthUsedForCentreCalculationRatioToPointUsedForNormalisationMethod1)*2.0)

    PRINT, "amplitudeShearPerLambda = ", amplitudeShearPerLambda

    subsetNumValues2 = long(balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex]/interval) *
         2
    leftSubsetIndex2 = (centreSubsetIndex - subsetNumValues2)
    leftSubsetNumValues2 = subsetNumValues2
    rightSubsetNumValues2 = subsetNumValues2
    rightSubsetIndex2 = centreSubsetIndex + subsetNumValues2
    leftwavelengthSubset2 = wavelengthArray(leftSubsetIndex2:centreSubsetIndex)
    leftamplitudeSubset2 = amplitudeArray(leftSubsetIndex2:centreSubsetIndex)
    rightwavelengthSubset2 = wavelengthArray(centreSubsetIndex:rightSubsetIndex2)
    rightamplitudeSubset2 = amplitudeArray(centreSubsetIndex:rightSubsetIndex2)
    leftamplitudeSubsetSheared2 = leftamplitudeSubset2   ;dummy data for array declaration
    rightamplitudeSubsetSheared2 = rightamplitudeSubset2     ;dummy data for array declaration

    for leftSubsetIndex2=0, leftSubsetNumValues2-1, 1 do begin
        leftSubsetIndexInverse2 = (leftSubsetNumValues2-1)-leftSubsetIndex2
        currentWavelength2 = wavelengthCentre-(leftSubsetIndexInverse2*interval)
        leftamplitudeSubsetSheared2[leftSubsetIndex2] = leftamplitudeSubset2[leftSubsetIndex2] - (
            double(leftSubsetIndexInverse2)*interval*amplitudeShearPerLambda)
    endfor

    for rightSubsetIndex2=0, rightSubsetNumValues2-1, 1 do begin
        currentWavelength2 = wavelengthCentre+(rightSubsetIndex2*interval)
        rightamplitudeSubsetSheared2[rightSubsetIndex2] = rightamplitudeSubset2[rightSubsetIndex2] +
             (double(rightSubsetIndex2)*interval*amplitudeShearPerLambda)
    endfor

    oplot, leftwavelengthSubset2, leftamplitudeSubsetSheared2,  COLOR=250
    oplot, rightwavelengthSubset2, rightamplitudeSubsetSheared2,  COLOR=250

endfor


PRINT, "Waiting for 1 seconds to display Normalisation Method B1b..."
WAIT, 1

plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,maxWavelength], xtitle
    ='wavelength (Ang)', ytitle='amplitude', COLOR=140   ;yrange=[-0.8,0.8], linestyle=5,psym=3,
    SYMSIZE=1
```

```
for balmerLineIndex=0, numberBalmerLinesUsed-1, 1 do begin

    centreSubsetIndex = long((balmerLinesLabXPos[balmerLineIndex]-minWavelength+
        balmerLineWavelengthOffset)/interval)
    subsetNumValuesLeft = long(balmerLinesLabXNormPointMethod2Width[balmerLineIndex, 0]/interval)
    subsetNumValuesRight = long(balmerLinesLabXNormPointMethod2Width[balmerLineIndex, 1]/interval)
    leftSubsetCentreIndex = centreSubsetIndex - long(balmerLinesLabXNormPointMethod2[balmerLineIndex
        , 0]/interval)
    leftSubsetIndex = (leftSubsetCentreIndex - subsetNumValuesLeft/2)
    leftSubsetIndexB = leftSubsetIndex + subsetNumValuesLeft
    rightSubsetCentreIndex = centreSubsetIndex + long(balmerLinesLabXNormPointMethod2[
        balmerLineIndex, 1]/interval)
    rightSubsetIndex = (rightSubsetCentreIndex - subsetNumValuesRight/2)
    rightSubsetIndexB = rightSubsetIndex + subsetNumValuesRight

        ;line left side polyfit

    leftwavelengthSubset = wavelengthArray(leftSubsetIndex:leftSubsetIndexB)
    leftamplitudeSubset = amplitudeArray(leftSubsetIndex:leftSubsetIndexB)
    leftSubsetPolyFitParameters = poly_fit(leftwavelengthSubset, leftamplitudeSubset, polyFitOrder)
    leftamplitudeSubsetTheory = poly(leftwavelengthSubset, leftSubsetPolyFitParameters)

        ;line right side polyfit

    rightwavelengthSubset = wavelengthArray(rightSubsetIndex:rightSubsetIndexB)
    rightamplitudeSubset = amplitudeArray(rightSubsetIndex:rightSubsetIndexB)
    rightSubsetPolyFitParameters = poly_fit(rightwavelengthSubset, rightamplitudeSubset,
        polyFitOrder)
    rightamplitudeSubsetTheory = poly(rightwavelengthSubset, rightSubsetPolyFitParameters)

    oplot, leftwavelengthSubset, leftamplitudeSubsetTheory, COLOR=230
    oplot, rightwavelengthSubset, rightamplitudeSubsetTheory, COLOR=230

endfor

PRINT, "Waiting for 1 seconds to display Normalisation Method B2b..."
WAIT, 1

plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,maxWavelength], xtitle
    ='wavelength (Ang)', ytitle='amplitude', COLOR=140  ;yrange=[-0.8,0.8],linestyle=5,psym=3,
    SYMSIZE=1

for balmerLineIndex=0, numberBalmerLinesUsed-1, 1 do begin

    centreSubsetIndex = long((balmerLinesLabXPos[balmerLineIndex]-minWavelength+
        balmerLineWavelengthOffset)/interval)
    subsetNumValuesLeft = long(balmerLinesLabXNormPointMethod2Width[balmerLineIndex, 0]/interval)
    subsetNumValuesRight = long(balmerLinesLabXNormPointMethod2Width[balmerLineIndex, 1]/interval)
    leftSubsetCentreIndex = centreSubsetIndex - long(balmerLinesLabXNormPointMethod2[balmerLineIndex
        , 0]/interval)
    leftSubsetIndex = (leftSubsetCentreIndex - subsetNumValuesLeft/2)
    leftSubsetIndexB = leftSubsetIndex + subsetNumValuesLeft
    rightSubsetCentreIndex = centreSubsetIndex + long(balmerLinesLabXNormPointMethod2[
        balmerLineIndex, 1]/interval)
    rightSubsetIndex = (rightSubsetCentreIndex - subsetNumValuesRight/2)
    rightSubsetIndexB = rightSubsetIndex + subsetNumValuesRight

        ;line left side polyfit
    leftwavelengthSubset = wavelengthArray(leftSubsetIndex:leftSubsetIndexB)
    leftamplitudeSubset = amplitudeArray(leftSubsetIndex:leftSubsetIndexB)
    leftSubsetPolyFitParameters = poly_fit(leftwavelengthSubset, leftamplitudeSubset, polyFitOrder)
    leftamplitudeSubsetTheory = poly(leftwavelengthSubset, leftSubsetPolyFitParameters)

        ;line right side polyfit
    rightwavelengthSubset = wavelengthArray(rightSubsetIndex:rightSubsetIndexB)
    rightamplitudeSubset = amplitudeArray(rightSubsetIndex:rightSubsetIndexB)
    rightSubsetPolyFitParameters = poly_fit(rightwavelengthSubset, rightamplitudeSubset,
        polyFitOrder)
    rightamplitudeSubsetTheory = poly(rightwavelengthSubset, rightSubsetPolyFitParameters)

    ;now find the fit x position at an arbitrary amplitude,
        balmerLineHalfWidthUsedForCentreCalculation[balmerLineIndex]/2.0
    ;at a given amplitude value, calculate the wavelength difference from the centre between the
        left and right fits of each balmer line fit,

    wavelengthCentre = balmerLinesLabXPos[balmerLineIndex]+balmerLineWavelengthOffset
    leftPosArray = dblarr(1)
    leftWavelength = wavelengthCentre - (balmerLinesLabXNormPointMethod2[balmerLineIndex, 0])
    leftPosArray[0] = leftWavelength
    rightPosArray = dblarr(1)
    rightWavelength = wavelengthCentre + (balmerLinesLabXNormPointMethod2[balmerLineIndex, 1])
    rightPosArray[0] = rightWavelength
    leftValArray = poly(leftPosArray, leftSubsetPolyFitParameters)
    rightValArray = poly(rightPosArray, rightSubsetPolyFitParameters)
    leftVal = leftValArray[0]
    rightVal = rightValArray[0]

    ;adjust the amplitude of the data on each/one side of the balmer line to normalise the balmer
        line (use a linear shear?)
    differenceBetweenAmplitudeValues = leftVal - rightVal
```

```
    PRINT, "differenceBetweenAmplitudeValues = ", differenceBetweenAmplitudeValues

    ;adjust the data such that leftVal = leftVal - differenceBetweenAmplitudeValues/2,  and rightVal
         = rightVal + differenceBetweenAmplitudeValues/2
    ;create a linear shear function - for a given wavelength, multiply by some function of the
        wavelength diff

    amplitudeShearPerLambda = differenceBetweenAmplitudeValues / ((double(
        balmerLinesLabXNormPointMethod2[balmerLineIndex, 0]+balmerLinesLabXNormPointMethod2[
        balmerLineIndex, 1])))

    PRINT, "amplitudeShearPerLambda = ", amplitudeShearPerLambda

    subsetNumValues2 = long(balmerLinesLabXNormPointMethod2[balmerLineIndex]/interval) * 1.5
    leftSubsetIndex2 = (centreSubsetIndex - subsetNumValues2)
    leftSubsetNumValues2 = subsetNumValues2
    rightSubsetNumValues2 = subsetNumValues2
    rightSubsetIndex2 = centreSubsetIndex + subsetNumValues2
    leftwavelengthSubset2 = wavelengthArray(leftSubsetIndex2:centreSubsetIndex)
    leftamplitudeSubset2 = amplitudeArray(leftSubsetIndex2:centreSubsetIndex)
    rightwavelengthSubset2 = wavelengthArray(centreSubsetIndex:rightSubsetIndex2)
    rightamplitudeSubset2 = amplitudeArray(centreSubsetIndex:rightSubsetIndex2)
    leftamplitudeSubsetSheared2 = leftamplitudeSubset2   ;dummy data for array declaration
    rightamplitudeSubsetSheared2 = rightamplitudeSubset2     ;dummy data for array declaration

    for leftSubsetIndex2=0, leftSubsetNumValues2-1, 1 do begin
        leftSubsetIndexInverse2 = (leftSubsetNumValues2-1)-leftSubsetIndex2
        currentWavelength2 = wavelengthCentre-(leftSubsetIndexInverse2*interval)
        leftamplitudeSubsetSheared2[leftSubsetIndex2] = leftamplitudeSubset2[leftSubsetIndex2] - (
            double(leftSubsetIndexInverse2)*interval*amplitudeShearPerLambda)
    endfor

    for rightSubsetIndex2=0, rightSubsetNumValues2-1, 1 do begin
        currentWavelength2 = wavelengthCentre+(rightSubsetIndex2*interval)
        rightamplitudeSubsetSheared2[rightSubsetIndex2] = rightamplitudeSubset2[rightSubsetIndex2] +
             (double(rightSubsetIndex2)*interval*amplitudeShearPerLambda)
    endfor

    oplot, leftwavelengthSubset2, leftamplitudeSubsetSheared2,  COLOR=250
    oplot, rightwavelengthSubset2, rightamplitudeSubsetSheared2,  COLOR=250


endfor

PRINT, "Waiting for 1 seconds to display Normalisation Method B2c..."
WAIT, 1


plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,maxWavelength], xtitle
    ='wavelength (Ang)', ytitle='amplitude', COLOR=140   ;yrange=[-0.8,0.8],linestyle=5,psym=3,
    SYMSIZE=1

for balmerLineIndex=0, numberBalmerLinesUsed-1, 1 do begin

    centreSubsetIndex = long((balmerLinesLabXPos[balmerLineIndex]-minWavelength+
        balmerLineWavelengthOffset)/interval)

    subsetNumValues = long(balmerLinesLabXNormPointMethod3Width[balmerLineIndex]/interval)
    leftSubsetIndex = (centreSubsetIndex - subsetNumValues)
    leftSubsetNumValues = subsetNumValues

    PRINT, "leftSubsetNumValues = "
    PRINT, leftSubsetNumValues

    rightSubsetNumValues = subsetNumValues
    rightSubsetIndex = centreSubsetIndex + subsetNumValues

        ;line left side polyfit
    leftwavelengthSubset = wavelengthArray(leftSubsetIndex:centreSubsetIndex)
    leftamplitudeSubset = amplitudeArray(leftSubsetIndex:centreSubsetIndex)

        ;line right side polyfit
    rightwavelengthSubset = wavelengthArray(centreSubsetIndex:rightSubsetIndex)
    rightamplitudeSubset = amplitudeArray(centreSubsetIndex:rightSubsetIndex)

    averagedLeftAmplitudeSubset = leftamplitudeSubset    ;dummy value for initialisation purposes
        only
    averagedRightAmplitudeSubset = leftamplitudeSubset   ;dummy value for initialisation purposes
        only

    ;PRINT, size(averagedLeftAmplitudeSubset)

    for i=0, subsetNumValues, 1 do begin
        averagedLeftAmplitudeSubset[i] = (leftamplitudeSubset[i] + rightamplitudeSubset[(
            subsetNumValues)-i]) /2.0
        averagedRightAmplitudeSubset[i] = (rightamplitudeSubset[i] + leftamplitudeSubset[(
            subsetNumValues)-i]) /2.0
    endfor

    oplot, leftwavelengthSubset, averagedLeftAmplitudeSubset,  COLOR=230
    oplot, rightwavelengthSubset, averagedRightAmplitudeSubset,  COLOR=230
```

```
endfor

PRINT, "Waiting for 1 seconds to display Normalisation Method B3b..."
;WAIT, 1


; Stage C - calculate fit... [not yet implemented]


end
```

A.5.2 White Dwarf Parameter Extraction

This software is otherwise known as the "Bergeron" code.

This code has been used to interpolate and derive a best fit WD model from Bergeron DA/DB WD colour-magnitude / evolutionary grids, and a given set of WD match criteria (parameters).

The PDD Bergeron grid interpolation code (pda_wd_mag_sigma.f) has been generalised and extended to support the following operational modes;

Mode Norm

input filename; loggtempmag.dat (PDD cooler component mass calculations) replaces pda_wd_mag_sigma.f/pda_wd_Vmag.f + fontaine g2m_fbbCO.f [minus gravitational redshift and radius parameters] inputs; logg + temperature

Mode Cool

input filename; teffdistmag.dat (RBB cooler component calculations) replaces pda_wd_dd_cool_n_ho / rbb_wd_dd_cool_n_hot.f) inputs; temperature + distance

Mode Cool 2

input filename; masstempmag.dat (PDD cooler component mass calculations) inputs; mass + temperature

Mode DC

input filename; distmag.dat (PDD cooler component mass calculations) replaces rbb_dd_DC.f inputs; distance + observed mag

'rbb_wd_dd_param.f'

rbb_wd_dd_param-08Nov10d.zip

```
      program rbb_wd_dd_param
c
c         bicubic spline interpolation of Bergeron WD model grid
c         version 9 Aug 10a
c
c         tested modes;
c
c         par_DC; [distMag.dat]
c         input: distance modulus, distance modulus error, observed u_mag, observed g_mag, observed
     r_mag, observed i_mag, observed z_mag, observed u_mag error, observed g_mag error, observed
     r_mag error, observed i_mag error, observed z_mag error
```

```
c       output: distance modulus, observed g_mag, mass, mass error, age, age error, logg, logg error
     , abs u_mag, abs g_mag, abs r_mag, abs i_mag, abs z_mag, abs u_mag error, abs g_mag error, abs
     r_mag error, abs i_mag error, abs z_mag error, Teff, Teff error
c
c       par_cool; [teffdistmag.dat]
c       input: Teff, distance modulus, distance modulus error, observed u_mag, observed g_mag,
     observed r_mag, observed i_mag, observed z_mag, observed u_mag error, observed g_mag error,
     observed r_mag error, observed i_mag error, observed z_mag error
c       output: Teff, distance modulus, observed g_mag, mass, mass error, age, age error, logg, logg
      error, abs u_mag, abs g_mag, abs r_mag, abs i_mag, abs z_mag, abs u_mag error, abs g_mag error
     , abs r_mag error, abs i_mag error, abs z_mag error
c
c       par_norm; [loggtempmag.dat]
c       input: logg, logg error, temp, temp error, observed u_mag, observed g_mag, observed r_mag,
     observed i_mag, observed z_mag, observed u_mag error, observed g_mag error, observed r_mag
     error, observed i_mag error, observed z_mag error
c       output: observed g_mag, mass, mass error, age, age error, logg, logg error, abs u_mag, abs
     g_mag, abs r_mag, abs i_mag, abs z_mag, abs u_mag error, abs g_mag error, abs r_mag error, abs
     i_mag error, abs z_mag error
c
c       par_cool_2; [masstempmag.dat]   - required for DD system with both a DC and a low
     temperature component (where the low temperature component's spectroscopic mass has been
     corrected for estimated offset in low temp spectroscopic determination [~0.15Msol])
c       input: mass, mass error, temp, temp error, observed u_mag, observed g_mag, observed r_mag,
     observed i_mag, observed z_mag, observed u_mag error, observed g_mag error, observed r_mag
     error, observed i_mag error, observed z_mag error
c       output: observed g_mag, mass, mass error, age, age error, logg, logg error, abs u_mag, abs
     g_mag, abs r_mag, abs i_mag, abs z_mag, abs u_mag error, abs g_mag error, abs r_mag error, abs
     i_mag error, abs z_mag error
c
c       par_DC_3/par_cool; [teffdistmag.dat]
c       input: Teff, distance modulus, distance modulus error, observed u_mag, observed g_mag,
     observed r_mag, observed i_mag, observed z_mag, observed u_mag error, observed g_mag error,
     observed r_mag error, observed i_mag error, observed z_mag error
c       output: Teff, distance modulus, observed g_mag, mass, mass error, age, age error, logg, logg
      error, abs u_mag, abs g_mag, abs r_mag, abs i_mag, abs z_mag, abs u_mag error, abs g_mag error
     , abs r_mag error, abs i_mag error, abs z_mag error
c
c       not yet coded modes;
c
c       par_DC_2; [mag.dat] - NOT USED
c       input: observed u_mag, observed g_mag, observed r_mag, observed i_mag, observed z_mag,
     observed u_mag error, observed g_mag error, observed r_mag error, observed i_mag error,
     observed z_mag error
c       output: observed g_mag, mass, mass error, age, age error, logg, logg error, abs u_mag, abs
     g_mag, abs r_mag, abs i_mag, abs z_mag, abs u_mag error, abs g_mag error, abs r_mag error, abs
     i_mag error, abs z_mag error, Teff, Teff error,distance modulus, distance modulus error,
c


      implicit none
c
c       declarations
c
      character(len=120) crap
      character(len=9) name(1)
c
c       variables for spline fitting
c

      integer par_norm
      integer par_cool
      integer par_DC
      integer par_DC_2
      integer par_coo2
      integer par_type
      integer tempLogg
      integer distTemp
      integer distCol
      integer colCol
      integer massTemp
      integer isDA
      integer DAmxmx
      integer DAmymx
      integer DBmxmx
      integer DBmymx

      integer mxmx,mymx
      Character(50) WDfiName
      Character(50) fnNorm
      Character(50) fnCool
      Character(50) fnDC
      Character(50) fnDC_2

c     fnNorm='tempLoggmag.dat'
c     fnCool='teffdistmag.dat'
c     fnDC='distmag.dat'
c     fnDC_2='mag.dat'

      parameter (distCol=1,distTemp=2,tempLogg=3,colCol=4)
      parameter (massTemp=5)
      parameter (par_DC=distCol,par_cool=distTemp)
```

```
      parameter (par_norm=tempLogg,par_DC_2=colCol)
      parameter (par_coo2=massTemp)
      parameter (DAmxmx=57,DAmymx=6)
      parameter (DBmxmx=31,DBmymx=5)

      integer in1,out1


c     tested modes;
      parameter (par_type=par_cool,isDA=1,in1=55,out1=21)
      parameter (mxmx=DAmxmx,mymx=DAmymx,WDfiName='teffdistmag.dat')
c     parameter (par_type=par_DC,isDA=0,in1=56,out1=22)
c     parameter (mxmx=DBmxmx,mymx=DBmymx,WDfiName='distmag.dat')
c     parameter (par_type=par_norm,isDA=1,in1=55,out1=24)
c     parameter (mxmx=DAmxmx,mymx=DAmymx,WDfiName='loggtempmag.dat')
c     parameter (par_type=par_coo2,isDA=1,in1=55,out1=23)
c     parameter (mxmx=DAmxmx,mymx=DAmymx,WDfiName='masstempmag.dat')

c     not yet coded modes;
c     parameter (par_type=par_DC_2,isDA=0,in1=56,out1=25)
c     parameter (mxmx=DBmxmx,mymx=DBmymx,WDfiName='mag.dat')
c     parameter (par_type=par_cool,isDA=0,in1=56,out1=26)
c     parameter (mxmx=DBmxmx,mymx=DBmymx,WDfiName='teffdistmag.dat')
c     NB 26 is for a magentic DB WD [DC_3]

      integer mx,my,r0,r1,r2,cnt



      integer nmax,maxwd
      parameter (nmax=mxmx)
      parameter (maxwd=50)

      integer nest1,nest2,nest3
      parameter (nest1=nmax+4,nest2=nmax+4,nest3=nmax+4)

      integer lwrk1,lwrk2,lwrk3
      parameter (lwrk1=4*nmax+nest1*16,lwrk2=4*nmax+nest2*16)
      parameter (lwrk3=4*nmax+nest3*16)
      integer loop,var

      real fp1,fp2,fp3
      real s1,s2,s3

      integer ifail1,ifail2,ifail3
      integer m1,m2,m3
      integer n1,n2,n3

      real c1(nest1),k1(nest1),w1(nmax),wrk1(lwrk1)
      integer iwrk1(nest1)

      real c2(nest2),k2(nest2),w2(nmax),wrk2(lwrk2)
      integer iwrk2(nest2)

      real c3(nest3),k3(nest3),w3(nmax),wrk3(lwrk3)
      integer iwrk3(nest3)

      real x(mymx,mxmx),y(mymx,mxmx),mass(mymx,mxmx),mbol(mymx,mxmx),
     +           bc(mymx,mxmx),mv(mymx,mxmx),mc(mymx,mxmx),mj(mymx,mxmx),
     +           mh(mymx,mxmx),mk(mymx,mxmx),mu(mymx,mxmx),mg(mymx,mxmx),
     +           mr(mymx,mxmx),mi(mymx,mxmx),mz(mymx,mxmx),mb(mymx,mxmx),
     +           um(mymx,mxmx),age(mymx,mxmx)

      real w(mymx,mxmx)
      real w1g(mymx)
      real w1RBc(mxmx)

      integer wd,ios,i

c     for rbb_wd_dd_cool_n_hot.f/rbb_wd_dd_DC.f/rbb_wd_dd_DC_2.f

      real mbigU(mymx,mxmx)
      real mbigB(mymx,mxmx)
      real mbigV(mymx,mxmx)
      real mbigR(mymx,mxmx)
      real mbigI(mymx,mxmx)
      real mbigJ(mymx,mxmx)
      real mbigH(mymx,mxmx)
      real mbigK(mymx,mxmx)
      real mSmally(mymx,mxmx)
      real mbMinusy(mymx,mxmx)
      real muMinusb(mymx,mxmx)
      real mvMinusy(mymx,mxmx)
      real mVMinusI(mymx,mxmx)
      real mGMinusR(mymx,mxmx)
      real mUMinusV(mymx,mxmx)
      real mUMinusG(mymx,mxmx)
      real mBMinusV(mymx,mxmx)

      real obsmagg(maxwd),obsmagge(maxwd)
      real obsmagr(maxwd),obsmagre(maxwd)
      real obsmagu(maxwd),obsmague(maxwd)
```

```
      real obsmagi(maxwd),obsmagie(maxwd)
      real obsmagz(maxwd),obsmagze(maxwd)

      integer ColMatch
      integer CMatch2
      integer CM_UMinR
      integer CM_UMinG
      integer CM_GMinR
      integer CM_GMinZ
      real obscol
      real obscole
      real obscol2
      real obscol2e
      real maxvar1,minvar1
      real var1temp

      real qmagsc(maxwd)
      real obsdist(maxwd),obsdiste(maxwd)
      real obsmag(maxwd),obsmage(maxwd)
      real obsmass(maxwd),obsmasse(maxwd)
      real obstemp(maxwd),obstempe(maxwd)
          real obslogg(maxwd), obslogge(maxwd)

      real wd_var1a(7)
      real wd_var2a(7)
      real wd_cola(7)
      real wd_col2a(7)
      real wd_teffa(7)
      real wd_massa(7)
      real wd_logga(7)
      real absmaga(7)
      real obsmaga(7)
      real dista(7)

c     for error calculations
      real wd_varae1(7)
      real wd_varae2(7)
      real wd_varae3(7)

      parameter (CM_GMinZ=3,CM_UMinR=2,CM_UMinG=1,CM_GMinR=0)
      parameter (ColMatch=CM_UMinR)
      parameter (CMatch2=CM_GMinR)

      real var1_RBc(mxmx),var2_RBc(mymx)
      real qmassRBc(mxmx),teffRBc(mxmx),qmagRBc(mxmx),loggc(mymx)
      real mgRBc(mxmx),mrRBc(mxmx),muRBc(mxmx)
      real miRBc(mxmx),mzRBc(mxmx)
      real ageRBc(mxmx),colRBc(mxmx),col2RBc(mxmx)
      real loggRBc(mxmx)

      real qmasss2(mymx),teffRBs2(mymx),loggRBs2(mymx)
      real mgRBs2(mymx),mrRBs2(mymx),muRBs2(mymx)
      real miRBs2(mymx),mzRBs2(mymx)
      real ageRBs2(mymx),colRBs2(mymx),col2RBs2(mymx)

      real var1_RBs(mymx),var2_RBs(mymx)
      real qmasss(mymx),teffs(mymx),qmags(mymx),qloggs(mymx)
      real loggRBs(mymx),mgRBs(mymx),mrRBs(mymx),muRBs(mymx)
      real miRBs(mymx),mzRBs(mymx)
      real ageRBs(mymx),colRBs(mymx),col2RBs(mymx)

      real calcMass(7)
      real calcAge(7)
      real calcLogg(7)
      real calcAbsu(7)
      real calcAbsg(7)
      real calcAbsr(7)
      real calcAbsi(7)
      real calcAbsz(7)
      real calcTemp(7)
      real calcDist(7)

      real dmdV1,dmdD
      real dmdV2,dmdV3,dm2,dm
      real dAgedV1,dAgedD
      real dAgedV2,dAgedV3,dAge2,dAge
      real dlgdV1,dlgdD
      real dlgdV2,dlgdV3,dlg2,dlg
      real dMudV1,dMudD
      real dMudV2,dMudV3,dMu2,dMu
      real dMgdV1,dMgdD
      real dMgdV2,dMgdV3,dMg2,dMg
      real dMrdV1,dMrdD
      real dMrdV2,dMrdV3,dMr2,dMr
      real dMidV1,dMidD
      real dMidV2,dMidV3,dMi2,dMi
      real dMzdV1,dMzdD
      real dMzdV2,dMzdV3,dMz2,dMz
      real dTdV1,dTdD
      real dTdV2,dTdV3,dT2,dT
      real dD
```

```
      integer index1
      integer index2

      integer useV3

      external pda_curfit,pda_splev
c
c         read in the bergeron model grid
c
      print *, 'Reading in the Bergeron Helium model grid: '
      print *, '3500K < Teff < 30000K 7.0 < log g < 9.0'
c
c     switch is required for some kind of Fortran reason...   (??)
      read (in1,*,end=120) mx,my

      print *,"MX: ",mx," MY: ",my
      read (in1,*) crap


c     print *, mx
c     print *, mxmx
c     print *, my
c     print *, mymx

      if (mx.gt.0 .and. mx.le.mxmx .and. my.gt.0 .and. my.le.mymx) then

          do r1 = 1,my
              do r2 = 1,mx

                        w(r1,r2)=1.0
                        read (in1,*) x(r1,r2),y(r1,r2),mass(r1,r2),mbol(r1,r2),
     +                   bc(r1,r2),mbigU(r1,r2),mbigB(r1,r2),mbigV(r1,r2),
     +                   mbigR(r1,r2),mbigI(r1,r2),mbigJ(r1,r2),mbigH(r1,r2),
     +                   mbigK(r1,r2),mu(r1,r2),mg(r1,r2),mr(r1,r2),
     +                   mi(r1,r2),
     +                   mz(r1,r2),mSmally(r1,r2),mbMinusy(r1,r2),muMinusb(r1,r2),
     +                   mvMinusy(r1,r2),mVMinusI(r1,r2),mGMinusR(r1,r2),
     +                   mUMinusV(r1,r2),mUMinusG(r1,r2),mBMinusV(r1,r2),age(r1,r2)


              end do
          end do
      end if
      s1=0

      close(unit=55)
c
c         read in effective temperatures and surface gravities from file
c
      open (unit=1,file=WDfiName,status='old')
      ios=0
      i=0
      if(par_type.eq.par_norm) then
          do while (ios.eq.0)
              i=i+1
              read (1,*,iostat=ios) obslogg(i),obslogge(i),
     +             obstemp(i),obstempe(i),
     +             obsmagu(i),obsmagg(i),obsmagr(i),obsmagi(i),
     +             obsmagz(i),
     +             obsmague(i),obsmagge(i),obsmagre(i),
     +             obsmagie(i),
     +             obsmagze(i)

                  obsmag(i) = obsmagg(i)
                  obsmage(i) = obsmagge(i)

          enddo
      else if(par_type.eq.par_cool) then
          do while (ios.eq.0)
              i=i+1
              read (1,*,iostat=ios) obstemp(i),obstempe(i),
     +             obsdist(i),obsdiste(i),
     +             obsmagu(i),obsmagg(i),obsmagr(i),obsmagi(i),
     +             obsmagz(i),
     +             obsmague(i),obsmagge(i),obsmagre(i),
     +             obsmagie(i),
     +             obsmagze(i)

                  obsmag(i) = obsmagg(i)
                  obsmage(i) = obsmagge(i)

          enddo
      else if(par_type.eq.par_DC) then
          do while (ios.eq.0)
              i=i+1
              read (1,*,iostat=ios) obsdist(i),obsdiste(i),
     +             obsmagu(i),obsmagg(i),obsmagr(i),obsmagi(i),
     +             obsmagz(i),
     +             obsmague(i),obsmagge(i),obsmagre(i),
     +             obsmagie(i),
     +             obsmagze(i)
```

```
                obsmag(i) = obsmagg(i)
                obsmage(i) = obsmagge(i)

            enddo
        else if(par_type.eq.par_DC_2) then
            do while (ios.eq.0)
                i=i+1
                read (1,*,iostat=ios)
     +              obsmagu(i),obsmagg(i),obsmagr(i),obsmagi(i),
     +              obsmagz(i),
     +              obsmague(i),obsmagge(i),obsmagre(i),
     +              obsmagie(i),
     +              obsmagze(i)

                obsmag(i) = obsmagg(i)
                obsmage(i) = obsmagge(i)

            enddo
        else if(par_type.eq.par_coo2) then
            do while (ios.eq.0)
                i=i+1
                read (1,*,iostat=ios) obsmass(i),obsmasse(i),
     +              obstemp(i),obstempe(i),
     +              obsmagu(i),obsmagg(i),obsmagr(i),obsmagi(i),
     +              obsmagz(i),
     +              obsmague(i),obsmagge(i),obsmagre(i),
     +              obsmagie(i),
     +              obsmagze(i)

                obsmag(i) = obsmagg(i)
                obsmage(i) = obsmagge(i)

            enddo
        end if
        close (unit=1)


c       print *, x
c       stop


        i=i-1
c
c       loop over each white dwarf
c
        do wd=1,i


            if(par_type.eq.par_norm) then

                wd_teffa(1)=obstemp(wd)
                wd_teffa(2)=obstemp(wd)+obstempe(wd)
                wd_teffa(3)=obstemp(wd)-obstempe(wd)
                wd_teffa(4)=obstemp(wd)
                wd_teffa(5)=obstemp(wd)
                wd_teffa(6)=obstemp(wd)
                wd_teffa(7)=obstemp(wd)

                wd_logga(1)=obslogg(wd)
                wd_logga(2)=obslogg(wd)
                wd_logga(3)=obslogg(wd)
                wd_logga(4)=obslogg(wd)+obslogge(wd)
                wd_logga(5)=obslogg(wd)-obslogge(wd)
                wd_logga(6)=obslogg(wd)+obslogge(wd)
                wd_logga(7)=obslogg(wd)-obslogge(wd)

                wd_var1a(1)=wd_teffa(1)
                wd_var1a(2)=wd_teffa(2)
                wd_var1a(3)=wd_teffa(3)
                wd_var1a(4)=wd_teffa(4)
                wd_var1a(5)=wd_teffa(5)
                wd_var1a(6)=wd_teffa(6)
                wd_var1a(7)=wd_teffa(7)

                wd_var2a(1)=wd_logga(1)
                wd_var2a(2)=wd_logga(2)
                wd_var2a(3)=wd_logga(3)
                wd_var2a(4)=wd_logga(4)
                wd_var2a(5)=wd_logga(5)
                wd_var2a(6)=wd_logga(6)
                wd_var2a(7)=wd_logga(7)

                wd_varae1(1)=wd_teffa(1)
                wd_varae1(2)=wd_teffa(2)
                wd_varae1(3)=wd_teffa(3)
                wd_varae1(4)=wd_teffa(4)
                wd_varae1(5)=wd_teffa(5)
                wd_varae1(6)=wd_teffa(6)
                wd_varae1(7)=wd_teffa(7)

                wd_varae2(1)=wd_logga(1)
```

```
      wd_varae2(2)=wd_logga(2)
      wd_varae2(3)=wd_logga(3)
      wd_varae2(4)=wd_logga(4)
      wd_varae2(5)=wd_logga(5)
      wd_varae2(6)=wd_logga(6)
      wd_varae2(7)=wd_logga(7)

      useV3 = 0

      wd_varae3(1)=0
      wd_varae3(2)=0
      wd_varae3(3)=0
      wd_varae3(4)=0
      wd_varae3(5)=0
      wd_varae3(6)=0
      wd_varae3(7)=0

    else if(par_type.eq.par_cool) then

      wd_teffa(1)=obstemp(wd)
      wd_teffa(2)=obstemp(wd)+obstempe(wd)
      wd_teffa(3)=obstemp(wd)-obstempe(wd)
      wd_teffa(4)=obstemp(wd)
      wd_teffa(5)=obstemp(wd)
      wd_teffa(6)=obstemp(wd)
      wd_teffa(7)=obstemp(wd)

      wd_var1a(1)=wd_teffa(1)
      wd_var1a(2)=wd_teffa(2)
      wd_var1a(3)=wd_teffa(3)
      wd_var1a(4)=wd_teffa(4)
      wd_var1a(5)=wd_teffa(5)
      wd_var1a(6)=wd_teffa(6)
      wd_var1a(7)=wd_teffa(7)

      dista(1)=obsdist(wd)
      dista(2)=obsdist(wd)
      dista(3)=obsdist(wd)
      dista(4)=obsdist(wd)+obsdiste(wd)
      dista(5)=obsdist(wd)-obsdiste(wd)
      dista(6)=obsdist(wd)
      dista(7)=obsdist(wd)

      obsmaga(1)=obsmag(wd)
      obsmaga(2)=obsmag(wd)
      obsmaga(3)=obsmag(wd)
      obsmaga(4)=obsmag(wd)
      obsmaga(5)=obsmag(wd)
      obsmaga(6)=obsmag(wd)+obsmage(wd)
      obsmaga(7)=obsmag(wd)-obsmage(wd)

      absmaga(1)=obsmaga(1)-dista(1)
      absmaga(2)=obsmaga(2)-dista(2)
      absmaga(3)=obsmaga(3)-dista(3)
      absmaga(4)=obsmaga(4)-dista(4)
      absmaga(5)=obsmaga(5)-dista(5)
      absmaga(6)=obsmaga(6)-dista(6)
      absmaga(7)=obsmaga(7)-dista(7)

      wd_var2a(1)=absmaga(1)
      wd_var2a(2)=absmaga(2)
      wd_var2a(3)=absmaga(3)
      wd_var2a(4)=absmaga(4)
      wd_var2a(5)=absmaga(5)
      wd_var2a(6)=absmaga(6)
      wd_var2a(7)=absmaga(7)

      wd_col2a(1)=0.0
      wd_col2a(2)=0.0
      wd_col2a(3)=0.0
      wd_col2a(4)=0.0
      wd_col2a(5)=0.0
      wd_col2a(6)=0.0
      wd_col2a(7)=0.0

      wd_varae1(1)=wd_teffa(1)
      wd_varae1(2)=wd_teffa(2)
      wd_varae1(3)=wd_teffa(3)
      wd_varae1(4)=wd_teffa(4)
      wd_varae1(5)=wd_teffa(5)
      wd_varae1(6)=wd_teffa(6)
      wd_varae1(7)=wd_teffa(7)

      wd_varae2(1)=dista(1)
      wd_varae2(2)=dista(2)
      wd_varae2(3)=dista(3)
      wd_varae2(4)=dista(4)
      wd_varae2(5)=dista(5)
      wd_varae2(6)=dista(6)
      wd_varae2(7)=dista(7)

      useV3 = 1
```

```
            wd_varae3(1)=obsmaga(1)
            wd_varae3(2)=obsmaga(2)
            wd_varae3(3)=obsmaga(3)
            wd_varae3(4)=obsmaga(4)
            wd_varae3(5)=obsmaga(5)
            wd_varae3(6)=obsmaga(6)
            wd_varae3(7)=obsmaga(7)


        else if(par_type.eq.par_DC) then

c           print *, "wd="
c           print *, i
c           print *, "he"
c           print *, "obsmagu(i) =", obsmagu(i)
c           print *, "obsmagr(i) =", obsmagr(i)
c           print *, "obsmagg(i) =", obsmagg(i)

            if (ColMatch.eq.CM_UMinR) then
                obscol = obsmagu(i)-obsmagr(i)
                obscole = (obsmague(i) + obsmagre(i))*(1**(-0.5))
            else if (ColMatch.eq.CM_UMinG) then
                obscol = obsmagu(i)-obsmagg(i)
                obscole = (obsmague(i) + obsmagge(i))*(1**(-0.5))
            else if (ColMatch.eq.CM_GMinR) then
                obscol = obsmagg(i)-obsmagr(i)
                obscole = (obsmagge(i) + obsmagre(i))*(1**(-0.5))
            else if (ColMatch.eq.CM_GMinZ) then
                obscol = obsmagg(i)-obsmagz(i)
                obscole = (obsmagge(i) + obsmagze(i))*(1**(-0.5))
            else

            endif

            wd_cola(1)=obscol
            wd_cola(2)=obscol+obscole
            wd_cola(3)=obscol-obscole
            wd_cola(4)=obscol
            wd_cola(5)=obscol
            wd_cola(6)=obscol
            wd_cola(7)=obscol

            dista(1)=obsdist(wd)
            dista(2)=obsdist(wd)
            dista(3)=obsdist(wd)
            dista(4)=obsdist(wd)+obsdiste(wd)
            dista(5)=obsdist(wd)-obsdiste(wd)
            dista(6)=obsdist(wd)
            dista(7)=obsdist(wd)

            obsmaga(1)=obsmag(wd)
            obsmaga(2)=obsmag(wd)
            obsmaga(3)=obsmag(wd)
            obsmaga(4)=obsmag(wd)
            obsmaga(5)=obsmag(wd)
            obsmaga(6)=obsmag(wd)+obsmage(wd)
            obsmaga(7)=obsmag(wd)-obsmage(wd)

            absmaga(1)=obsmaga(1)-dista(1)
            absmaga(2)=obsmaga(2)-dista(2)
            absmaga(3)=obsmaga(3)-dista(3)
            absmaga(4)=obsmaga(4)-dista(4)
            absmaga(5)=obsmaga(5)-dista(5)
            absmaga(6)=obsmaga(6)-dista(6)
            absmaga(7)=obsmaga(7)-dista(7)


            wd_var1a(1)=absmaga(1)
            wd_var1a(2)=absmaga(2)
            wd_var1a(3)=absmaga(3)
            wd_var1a(4)=absmaga(4)
            wd_var1a(5)=absmaga(5)
            wd_var1a(6)=absmaga(6)
            wd_var1a(7)=absmaga(7)

            wd_var2a(1)=wd_cola(1)
            wd_var2a(2)=wd_cola(2)
            wd_var2a(3)=wd_cola(3)
            wd_var2a(4)=wd_cola(4)
            wd_var2a(5)=wd_cola(5)
            wd_var2a(6)=wd_cola(6)
            wd_var2a(7)=wd_cola(7)

            wd_col2a(1)=0.0
            wd_col2a(2)=0.0
            wd_col2a(3)=0.0
            wd_col2a(4)=0.0
            wd_col2a(5)=0.0
            wd_col2a(6)=0.0
            wd_col2a(7)=0.0
```

```
            wd_varae1(1)=wd_cola(1)
            wd_varae1(2)=wd_cola(2)
            wd_varae1(3)=wd_cola(3)
            wd_varae1(4)=wd_cola(4)
            wd_varae1(5)=wd_cola(5)
            wd_varae1(6)=wd_cola(6)
            wd_varae1(7)=wd_cola(7)

            wd_varae2(1)=dista(1)
            wd_varae2(2)=dista(2)
            wd_varae2(3)=dista(3)
            wd_varae2(4)=dista(4)
            wd_varae2(5)=dista(5)
            wd_varae2(6)=dista(6)
            wd_varae2(7)=dista(7)

            useV3 = 1

            wd_varae3(1)=obsmaga(1)
            wd_varae3(2)=obsmaga(2)
            wd_varae3(3)=obsmaga(3)
            wd_varae3(4)=obsmaga(4)
            wd_varae3(5)=obsmaga(5)
            wd_varae3(6)=obsmaga(6)
            wd_varae3(7)=obsmaga(7)


c               print *, "absmaga(1) =", absmaga(1)
c               print *, "absmaga(2) =", absmaga(2)
c               print *, "absmaga(3) =", absmaga(3)
c               print *, "absmaga(4) =", absmaga(4)
c               print *, "absmaga(5) =", absmaga(5)
c               print *, "absmaga(6) =", absmaga(6)
c               print *, "absmaga(7) =", absmaga(7)

        else if(par_type.eq.par_DC_2) then

c           print *, "wd="
c           print *, i
c           print *, "he"
c           print *, "obsmagu(i) =", obsmagu(i)
c           print *, "obsmagr(i) =", obsmagr(i)
c           print *, "obsmagg(i) =", obsmagg(i)

            if (ColMatch.eq.CM_UMinR) then
                obscol = obsmagu(i)-obsmagr(i)
                obscole = (obsmague(i) + obsmagre(i))*(1**(-0.5))
            else if (ColMatch.eq.CM_UMinG) then
                obscol = obsmagu(i)-obsmagg(i)
                obscole = (obsmague(i) + obsmagge(i))*(1**(-0.5))
            else if (ColMatch.eq.CM_GMinR) then
                obscol = obsmagg(i)-obsmagr(i)
                obscole = (obsmagge(i) + obsmagre(i))*(1**(-0.5))
            else if (ColMatch.eq.CM_GMinZ) then
                obscol = obsmagg(i)-obsmagz(i)
                obscole = (obsmagge(i) + obsmagze(i))*(1**(-0.5))
            else

            endif

            wd_cola(1)=obscol
            wd_cola(2)=obscol+obscole
            wd_cola(3)=obscol-obscole
            wd_cola(4)=obscol
            wd_cola(5)=obscol
            wd_cola(6)=obscol
            wd_cola(7)=obscol

            wd_var1a(1)=wd_cola(1)
            wd_var1a(2)=wd_cola(2)
            wd_var1a(3)=wd_cola(3)
            wd_var1a(4)=wd_cola(4)
            wd_var1a(5)=wd_cola(5)
            wd_var1a(6)=wd_cola(6)
            wd_var1a(7)=wd_cola(7)

            if (CMatch2.eq.CM_UMinR) then
                obscol2 = obsmagu(i)-obsmagr(i)
                obscol2e = (obsmague(i) + obsmagre(i))*(1**(-0.5))
            else if (CMatch2.eq.CM_UMinG) then
                obscol2 = obsmagu(i)-obsmagg(i)
                obscol2e = (obsmague(i) + obsmagge(i))*(1**(-0.5))
            else if (CMatch2.eq.CM_GMinR) then
                obscol2 = obsmagg(i)-obsmagr(i)
                obscol2e = (obsmagge(i) + obsmagre(i))*(1**(-0.5))
            else if (CMatch2.eq.CM_GMinZ) then
                obscol2 = obsmagg(i)-obsmagz(i)
                obscol2e = (obsmagge(i) + obsmagze(i))*(1**(-0.5))
            else

            endif
```

```
      wd_col2a(1)=obscol2
      wd_col2a(2)=obscol2
      wd_col2a(3)=obscol2
      wd_col2a(4)=obscol2+obscol2e
      wd_col2a(5)=obscol2-obscol2e
      wd_col2a(6)=obscol2+obscol2e
      wd_col2a(7)=obscol2-obscol2e


      wd_var2a(1)=wd_col2a(1)
      wd_var2a(2)=wd_col2a(2)
      wd_var2a(3)=wd_col2a(3)
      wd_var2a(4)=wd_col2a(4)
      wd_var2a(5)=wd_col2a(5)
      wd_var2a(6)=wd_col2a(6)
      wd_var2a(7)=wd_col2a(7)


      wd_varae1(1)=wd_cola(1)
      wd_varae1(2)=wd_cola(2)
      wd_varae1(3)=wd_cola(3)
      wd_varae1(4)=wd_cola(4)
      wd_varae1(5)=wd_cola(5)
      wd_varae1(6)=wd_cola(6)
      wd_varae1(7)=wd_cola(7)

      wd_varae2(1)=wd_col2a(1)
      wd_varae2(2)=wd_col2a(2)
      wd_varae2(3)=wd_col2a(3)
      wd_varae2(4)=wd_col2a(4)
      wd_varae2(5)=wd_col2a(5)
      wd_varae2(6)=wd_col2a(6)
      wd_varae2(7)=wd_col2a(7)

      useV3 = 0

      wd_varae3(1)=0
      wd_varae3(2)=0
      wd_varae3(3)=0
      wd_varae3(4)=0
      wd_varae3(5)=0
      wd_varae3(6)=0
      wd_varae3(7)=0



else if(par_type.eq.par_coo2) then

      wd_teffa(1)=obstemp(wd)
      wd_teffa(2)=obstemp(wd)+obstempe(wd)
      wd_teffa(3)=obstemp(wd)-obstempe(wd)
      wd_teffa(4)=obstemp(wd)
      wd_teffa(5)=obstemp(wd)
      wd_teffa(6)=obstemp(wd)
      wd_teffa(7)=obstemp(wd)

      wd_massa(1)=obsmass(wd)
      wd_massa(2)=obsmass(wd)
      wd_massa(3)=obsmass(wd)
      wd_massa(4)=obsmass(wd)+obsmasse(wd)
      wd_massa(5)=obsmass(wd)-obsmasse(wd)
      wd_massa(6)=obsmass(wd)+obsmasse(wd)
      wd_massa(7)=obsmass(wd)-obsmasse(wd)

      wd_var1a(1)=wd_teffa(1)
      wd_var1a(2)=wd_teffa(2)
      wd_var1a(3)=wd_teffa(3)
      wd_var1a(4)=wd_teffa(4)
      wd_var1a(5)=wd_teffa(5)
      wd_var1a(6)=wd_teffa(6)
      wd_var1a(7)=wd_teffa(7)

      wd_var2a(1)=wd_massa(1)
      wd_var2a(2)=wd_massa(2)
      wd_var2a(3)=wd_massa(3)
      wd_var2a(4)=wd_massa(4)
      wd_var2a(5)=wd_massa(5)
      wd_var2a(6)=wd_massa(6)
      wd_var2a(7)=wd_massa(7)

      wd_varae1(1)=wd_teffa(1)
      wd_varae1(2)=wd_teffa(2)
      wd_varae1(3)=wd_teffa(3)
      wd_varae1(4)=wd_teffa(4)
      wd_varae1(5)=wd_teffa(5)
      wd_varae1(6)=wd_teffa(6)
      wd_varae1(7)=wd_teffa(7)

      wd_varae2(1)=wd_massa(1)
      wd_varae2(2)=wd_massa(2)
      wd_varae2(3)=wd_massa(3)
      wd_varae2(4)=wd_massa(4)
```

```
            wd_varae2(5)=wd_massa(5)
            wd_varae2(6)=wd_massa(6)
            wd_varae2(7)=wd_massa(7)

            useV3 = 0

            wd_varae3(1)=0
            wd_varae3(2)=0
            wd_varae3(3)=0
            wd_varae3(4)=0
            wd_varae3(5)=0
            wd_varae3(6)=0
            wd_varae3(7)=0


        end if



        maxvar1 = -1E15
        minvar1 = 1E15


ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

        do var=1,7
c
c           loop over surface gravities
c
            do r1 = 1,my

                loggc(r1)=y(r1,1)
                w1g(r1)=w(r1,1)

                do r2 = 1,mx

                    if(par_type.eq.par_norm) then

                        index2 = r2

                    else if(par_type.eq.par_cool) then

                        index2 = r2

                    else if(par_type.eq.par_DC) then

                        index2 = (mx-r2+1)

                    else if(par_type.eq.par_DC_2) then

                        index2 = (mx-r2+1)

                    else if(par_type.eq.par_coo2) then

                        index2 = r2

                    end if


                    if (ColMatch.eq.CM_UMinR) then
                        colRBc(index2) = mu(r1,r2)-mr(r1,r2)
                    else if (ColMatch.eq.CM_UMinG) then
                        colRBc(index2) = mu(r1,r2)-mg(r1,r2)
                    else if (ColMatch.eq.CM_GMinR) then
                        colRBc(index2) = mg(r1,r2)-mr(r1,r2)
                    else if (ColMatch.eq.CM_GMinZ) then
                        colRBc(index2) = mg(r1,r2)-mz(r1,r2)
                    else
                    endif

                    if (CMatch2.eq.CM_UMinR) then
                        col2RBc(index2) = mu(r1,r2)-mr(r1,r2)
                    else if (CMatch2.eq.CM_UMinG) then
                        col2RBc(index2) = mu(r1,r2)-mg(r1,r2)
                    else if (CMatch2.eq.CM_GMinR) then
                        col2RBc(index2) = mg(r1,r2)-mr(r1,r2)
                    else if (CMatch2.eq.CM_GMinZ) then
                        col2RBc(index2) = mg(r1,r2)-mz(r1,r2)
                    else
                    endif


                    teffRBc(index2)=x(r1,r2)
c                       change here depending on what photometric band you wish to work with
                    qmagRBc(index2)=mg(r1,r2)
                    qmassRBc(index2)=mass(r1,r2)
                    w1RBc(index2)=w(r1,r2)

                    muRBc(index2) = mu(r1,r2)
                    mgRBc(index2) = mg(r1,r2)
```

```
                    mrRBc(index2) = mr(r1,r2)
                    miRBc(index2) = mi(r1,r2)
                    mzRBc(index2) = mz(r1,r2)

                    ageRBc(index2) = age(r1,r2)

c                   shouldnt have to use this;
                    loggRBc(index2) = loggc(r1)


c                               print *,qmagRBc(r2),teffRBc(r2)

                end do

                if(par_type.eq.par_norm) then

                    var1_RBc = teffRBc
                    index1 = r1

                else if(par_type.eq.par_cool) then

                    var1_RBc = teffRBc
                    index1 = r1

                else if(par_type.eq.par_DC) then

                    var1_RBc = mgRBc
                    index1 = (my-r1+1)

                else if(par_type.eq.par_DC_2) then

                    var1_RBc = colRBc
                    index1 = (my-r1+1)

                else if(par_type.eq.par_coo2) then

                    var1_RBc = teffRBc
                    index1 = r1

                end if

c               print *, "var1_RBc = ", var1_RBc
c               print *, "qmassRBc = ", qmassRBc
c               print *, "wd_var1a(var) = ", wd_var1a(var)

c               For mass calculations;      [for each gravity determine mass at a given var1 (eg Mg)
     ]
                call pda_curfit(0,mx,var1_RBc,qmassRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),qmasss2(index1),
     +                  1,ifail2)

c               print *, "qmasss2(index1) = ", qmasss2(index1)
c               stop


c               For age calculations;
                call pda_curfit(0,mx,var1_RBc,ageRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),ageRBs2(index1),
     +                  1,ifail2)

c               For abs u calculations;
                call pda_curfit(0,mx,var1_RBc,muRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),muRBs2(index1),
     +                  1,ifail2)

c                  print *, "var1_RBc = ", var1_RBc
c                  print *, "mgRBc = ", mgRBc
c                  print *, "wd_var1a(var) = ", wd_var1a(var)

c               For abs g calculations;
                call pda_curfit(0,mx,var1_RBc,mgRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),mgRBs2(index1),
     +                  1,ifail2)

c               print *, "mgRBs2(index1) = ", mgRBs2(index1)
c               stop

c               For abs r calculations;
                call pda_curfit(0,mx,var1_RBc,mrRBc,w1RBc,var1_RBc(1),
```

```
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),mrRBs2(index1),
     +                  1,ifail2)

c               For abs i calculations;
                call pda_curfit(0,mx,var1_RBc,miRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),miRBs2(index1),
     +                  1,ifail2)

c               For abs z calculations;
                call pda_curfit(0,mx,var1_RBc,mzRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),mzRBs2(index1),
     +                  1,ifail2)

c               For logg calculations;
                loggRBs2(r1) = loggc(r1)

c               For temperature calculations;
                call pda_curfit(0,mx,var1_RBc,teffRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),teffRBs2(index1),
     +                  1,ifail2)

c               print *, "var1_RBc = ", var1_RBc
c               print *, "colRBc = ", colRBc
c               print *, "wd_var1a(var) = ", wd_var1a(var)

c               For colour calculations;
                call pda_curfit(0,mx,var1_RBc,colRBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),colRBs2(index1),
     +                  1,ifail2)

c               print *, "colRBs2(index1) = ", colRBs2(index1)
c               stop

c               print *, "var1_RBc = ", var1_RBc
c               print *, "col2RBc = ", col2RBc
c               print *, "wd_var1a(var) = ", wd_var1a(var)

c               For colour2 calculations;
                call pda_curfit(0,mx,var1_RBc,col2RBc,w1RBc,var1_RBc(1),
     +                  var1_RBc(mx),3,
     +                  s2,nest2,n2,k2,c2,fp2,wrk2,lwrk2,iwrk2,ifail2)
                ifail2=0
                call pda_splev(k2,n2,c2,3,wd_var1a(var),col2RBs2(index1),
     +                  1,ifail2)

c               print *, "col2RBs2(index1) = ", col2RBs2(index1)
c               stop

c               print *, "r1 = ", r1
c               print *, qmassRBc
c               print *, ageRBc
c               print *, mgRBc
c               print *, muRBc
c               print *, muRBc
c               print *, teffRBc

c               print *, "mgRBc = ", mgRBc

c               print *, "var1_RBc = ", var1_RBc
c               print *, "wd_var1a(var) = ", wd_var1a(var)
c                  print *, "r1 = ", r1
c                  print *, "var1_RBs(r1) = ", var1_RBs(r1)
c                  print *, "qmasss2(r1) = ", qmasss2(r1)
c                  print *, "ageRBs2(r1) = ", ageRBs2(r1)
c                  print *, "mgRBs2(r1) = ", mgRBs2(r1)
c                  print *, "muRBs2(r1) = ", muRBs2(r1)
c                  print *, "muRBs2(r1) = ", muRBs2(r1)
c                  print *, "teffRBs2(r1) = ", teffRBs2(r1)
c               stop


c               print *, "var1_RBc = ", var1_RBc
c               print *, "wd_var1a(var) = ", wd_var1a(var)


            end do
```

```
            if(par_type.eq.par_norm) then

                var2_RBc = loggc

            else if(par_type.eq.par_cool) then

                var2_RBc = mgRBs2

            else if(par_type.eq.par_DC) then

                var2_RBc = colRBs2

            else if(par_type.eq.par_DC_2) then

                var2_RBc = col2RBs2

            else if(par_type.eq.par_coo2) then

                var2_RBc = qmasss2

            end if

c           print *, "var2_RBc = ", var2_RBc
c           print *, "wd_var2a(var) = ", wd_var2a(var)
c           print *, "qmasss2 = ", qmasss2

c
c           interpolate mass over Mg now - only 6 points
c


c           For Mass calculations;
            call pda_curfit(0,my,var2_RBc,qmasss2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcMass(var),1,ifail1)

c           print *, "calcMass(var) = ", calcMass(var)
c           stop

c           For age calculations;
            call pda_curfit(0,my,var2_RBc,ageRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcAge(var),1,ifail1)

c           For abs u calculations;
            call pda_curfit(0,my,var2_RBc,muRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcAbsu(var),1,ifail1)

c           For abs g calculations;
            call pda_curfit(0,my,var2_RBc,mgRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcAbsg(var),1,ifail1)

c           For abs r calculations;
            call pda_curfit(0,my,var2_RBc,mrRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcAbsr(var),1,ifail1)

c           For abs i calculations;
            call pda_curfit(0,my,var2_RBc,miRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcAbsi(var),1,ifail1)

c           For abs z calculations;
            call pda_curfit(0,my,var2_RBc,mzRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcAbsz(var),1,ifail1)

c           For logg calculations;
            call pda_curfit(0,my,var2_RBc,loggRBs2,w1g,var2_RBc(1),
     +          var2_RBc(my),3,
     +          s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcLogg(var),1,ifail1)

c           For temperature calculations;
```

```
            call pda_curfit(0,my,var2_RBc,teffRBs2,wlg,var2_RBc(1),
     +           var2_RBc(my),3,
     +           s1,nest1,n1,k1,c1,fp1,wrk1,lwrk1,iwrk1,ifail1)
            ifail1=0
            call pda_splev(k1,n1,c1,3,wd_var2a(var),calcTemp(var),1,ifail1)

               print *, "calculations:"
              print *, calcMass(var)
              print *, calcAge(var)
              print *, calcAbsg(var)
              print *, calcAbsr(var)
              print *, calcAbsu(var)
              print *, calcLogg(var)
              print *, calcTemp(var)


          enddo


          do var=1,7
              print *,var,calcMass(var)
          enddo


c             For Mass calculations;
          dmdV1 = (calcMass(2)-calcMass(3)) / (wd_varae1(2)-wd_varae1(3))
          dmdV2 = (calcMass(4)-calcMass(5)) / (wd_varae2(4)-wd_varae2(5))
          if(useV3.eq.1) then
          dmdV3 = (calcMass(6)-calcMass(7)) / (wd_varae3(6)-wd_varae3(7))
          else
          dmdV3 = 0.0
          endif

          dm2=dmdV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +       dmdV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +       dmdV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

          dm=dm2**0.5


c             For age calculations;
          dAgedV1 = (calcAge(2)-calcAge(3)) / (wd_varae1(2)-wd_varae1(3))
          dAgedV2 = (calcAge(4)-calcAge(5)) / (wd_varae2(4)-wd_varae2(5))
          if(useV3.eq.1) then
          dAgedV3 = (calcAge(6)-calcAge(7)) / (wd_varae3(6)-wd_varae3(7))
          else
          dAgedV3 = 0.0
          endif

          dAge2=dAgedV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +       dAgedV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +       dAgedV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

          dAge=dAge2**0.5


c             For logg calculations;
          dlgdV1 = (calcLogg(2)-calcLogg(3)) / (wd_varae1(2)-wd_varae1(3))
          dlgdV2 = (calcLogg(4)-calcLogg(5)) / (wd_varae2(4)-wd_varae2(5))
          if(useV3.eq.1) then
          dlgdV3 = (calcLogg(6)-calcLogg(7)) / (wd_varae3(6)-wd_varae3(7))
          else
          dlgdV3 = 0.0
          endif

          dlg2=dlgdV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +       dlgdV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +       dlgdV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

          dlg=dlg2**0.5


c             For abs u calculations;
          dMudV1 = (calcAbsu(2)-calcAbsu(3)) / (wd_varae1(2)-wd_varae1(3))
          dMudV2 = (calcAbsu(4)-calcAbsu(5)) / (wd_varae2(4)-wd_varae2(5))
          if(useV3.eq.1) then
          dMudV3 = (calcAbsu(6)-calcAbsu(7)) / (wd_varae3(6)-wd_varae3(7))
          else
          dMudV3 = 0.0
          endif

          dMu2=dMudV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +       dMudV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +       dMudV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

          dMu=dMu2**0.5


c             For abs g calculations;
          dMgdV1 = (calcAbsg(2)-calcAbsg(3)) / (wd_varae1(2)-wd_varae1(3))
          dMgdV2 = (calcAbsg(4)-calcAbsg(5)) / (wd_varae2(4)-wd_varae2(5))
          if(useV3.eq.1) then
```

```
      dMgdV3 = (calcAbsg(6)-calcAbsg(7)) / (wd_varae3(6)-wd_varae3(7))
      else
      dMgdV3 = 0.0
      endif

      dMg2=dMgdV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +     dMgdV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +     dMgdV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

      dMg=dMg2**0.5


c          For abs r calculations;
      dMrdV1 = (calcAbsr(2)-calcAbsr(3)) / (wd_varae1(2)-wd_varae1(3))
      dMrdV2 = (calcAbsr(4)-calcAbsr(5)) / (wd_varae2(4)-wd_varae2(5))
      if(useV3.eq.1) then
      dMrdV3 = (calcAbsr(6)-calcAbsr(7)) / (wd_varae3(6)-wd_varae3(7))
      else
      dMrdV3 = 0.0
      endif

      dMr2=dMrdV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +     dMrdV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +     dMrdV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

      dMr=dMr2**0.5


c          For abs i calculations;
      dMidV1 = (calcAbsi(2)-calcAbsi(3)) / (wd_varae1(2)-wd_varae1(3))
      dMidV2 = (calcAbsi(4)-calcAbsi(5)) / (wd_varae2(4)-wd_varae2(5))
      if(useV3.eq.1) then
      dMidV3 = (calcAbsi(6)-calcAbsi(7)) / (wd_varae3(6)-wd_varae3(7))
      else
      dMidV3 = 0.0
      endif

      dMi2=dMidV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +     dMidV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +     dMidV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

      dMi=dMi2**0.5

c          For abs z calculations;
      dMzdV1 = (calcAbsz(2)-calcAbsz(3)) / (wd_varae1(2)-wd_varae1(3))
      dMzdV2 = (calcAbsz(4)-calcAbsz(5)) / (wd_varae2(4)-wd_varae2(5))
      if(useV3.eq.1) then
      dMzdV3 = (calcAbsz(6)-calcAbsz(7)) / (wd_varae3(6)-wd_varae3(7))
      else
      dMzdV3 = 0.0
      endif

      dMz2=dMzdV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +     dMzdV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +     dMzdV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

      dMz=dMz2**0.5



c          For temperature calculations;
      dTdV1 = (calcTemp(2)-calcTemp(3)) / (wd_varae1(2)-wd_varae1(3))
      dTdV2 = (calcTemp(4)-calcTemp(5)) / (wd_varae2(4)-wd_varae2(5))
      if(useV3.eq.1) then
      dTdV3 = (calcTemp(6)-calcTemp(7)) / (wd_varae3(6)-wd_varae3(7))
      else
      dTdV3 = 0.0
      endif

      dT2=dTdV1**2.0 * (wd_varae1(1)-wd_varae1(3))**2.0 +
     +     dTdV2**2.0 * (wd_varae2(1)-wd_varae2(5))**2.0 +
     +     dTdV3**2.0 * (wd_varae3(1)-wd_varae3(7))**2.0

      dT=dT2**0.5


c             print *, "dista(1) = ", dista(1)
c             print *, "obsmag(wd) = ", obsmaga(1)
c             print *, "calcMass(1) = ", calcMass(1)
c             print *, "dm = ", dm
c             print *, "calcAge(1) = ", calcAge(1)
c          print *, "dAge = ", dAge
c          print *, "calcLogg(1) = ", calcLogg(1)
c          print *, "dlg = ", dlg
c          print *, "calcAbsg(1) = ", calcAbsg(1)
c          print *, "dMg = ", dMg
c          print *, "dMr = ", dMr
c          print *, "dMu = ", dMu
c          print *, "calcAbsr(1) = ", calcAbsr(1)
c          print *, "dMr = ", dMr
c          print *, "calcAbsu(1) = ", calcAbsu(1)
c             print *, "dMu = ", dMu
```

```
        calcDist(1) = obsmag(wd) - calcAbsg(1)
        dD = (obsmage(wd)**2 + dMg**2)**0.5

        if(par_type.eq.par_norm) then

            write(out1,150) obsmag(wd),
     +           calcMass(1),dm,calcAge(1),dAge,
     +           obslogg(wd),obslogge(wd),calcAbsu(1),calcAbsg(1),
     +           calcAbsr(1),calcAbsi(1),calcAbsz(1),
     +           dMu,dMg,dMr,dMi,dMz,
     +           obstemp(wd),obstempe(wd),calcDist(1),dD

        else if(par_type.eq.par_cool) then

            write(out1,150) obsmag(wd),
     +           calcMass(1),dm,calcAge(1),dAge,
     +           calcLogg(1),dlg,calcAbsu(1),calcAbsg(1),
     +           calcAbsr(1),calcAbsi(1),calcAbsz(1),
     +           dMu,dMg,dMr,dMi,dMz,
     +           obstemp(wd),obstempe(wd),obsdist(wd),obsdiste(wd)

        else if(par_type.eq.par_DC) then

            write(out1,150) obsmag(wd),
     +           calcMass(1),dm,calcAge(1),dAge,
     +           calcLogg(1),dlg,calcAbsu(1),calcAbsg(1),
     +           calcAbsr(1),calcAbsi(1),calcAbsz(1),
     +           dMu,dMg,dMr,dMi,dMz,
     +           calcTemp(1),dT,obsdist(wd),obsdiste(wd)

        else if(par_type.eq.par_DC_2) then

            write(out1,150) obsmag(wd),
     +           calcMass(1),dm,calcAge(1),dAge,
     +           calcLogg(1),dlg,calcAbsu(1),calcAbsg(1),
     +           calcAbsr(1),calcAbsi(1),calcAbsz(1),
     +           dMu,dMg,dMr,dMi,dMz,
     +           calcTemp(1),dT,calcDist(1),dD

        else if(par_type.eq.par_coo2) then

            write(out1,150) obsmag(wd),
     +           obsmass(wd),obsmasse(wd),calcAge(1),dAge,
     +           calcLogg(1),dlg,calcAbsu(1),calcAbsg(1),
     +           calcAbsr(1),calcAbsi(1),calcAbsz(1),
     +           dMu,dMg,dMr,dMi,dMz,
     +           obstemp(wd),obstempe(wd),calcDist(1),dD

        end if


c           print *, calcAge(1)

    end do

 120    stop

 150    format(f6.3,1x,
     +  f6.3,1x,f6.3,1x,ES14.7,1x,ES14.7,
     +  1x,f6.3,1x,f6.3,1x,f6.3,1x,f6.3,1x,f6.3,
     +  1x,f6.3,1x,f6.3,1x,f6.3,1x,f6.3,1x,f6.3,
     +  1x,f6.3,1x,f6.3,1x,f6.3,1x,f6.3,1x,f6.3,
     +  1x,f7.1,1x,f7.1,1x,f6.3,1x,f6.3)

        stop


        end
```

A.5.3 Tlusty And Koester ASCII Model Grid Comparison Using GDL

This code has been used to verify that the PDD calculated TLusty and Koester WD ASCII WD model grids align.

compareTlustyAndKoesterModelGrid

tlustyAndKoesterASCIIModelGridComparisonUsingGDL-26May10a-includingModelGrids.zip

```
; Title: compare Tlusty And Koester Model Grid
; File: compareTlustyAndKoesterModelGrid.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 26 May 10 2:00pm
; Summary:
;
;   Postconditions;
;
;   This program flux normalises and compares Tlusty And Koester Model Grid files
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   ngc3532-lls300750.dat [wavelengthArray, amplitudeArray], space delimited
;   da15000_900.dk_tr_rg [wavelengthArray, amplitudeArray], space delimited
;   Usage:  gdl
;   .run compareTlustyAndKoesterModelGrid.pro


pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
                l_blue    : 12,$
                l_red     : 13 $
               }
;         0  1   2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]    ;blue
end


my_colors

minTemperatureForComparison = 12000
maxTemperatureForComparison = 20000
incrementTemperatureForComparison = 2000
minLoggForComparison = 7.25
maxLoggForComparison = 8.50
incrementLoggForComparison = 0.25

minWavelength = 3500.0
maxWavelength = 5300.0

for logg=minLoggForComparison, maxLoggForComparison, incrementLoggForComparison do begin
    for teff=minTemperatureForComparison, maxTemperatureForComparison,
         incrementTemperatureForComparison do begin

        ;read in koester and tlusty model
        teffString = STRING(teff, FORMAT='(I5.5)')
        loggString = STRING(logg, FORMAT='(F0.2)')
        PRINT, "teffString = ", teffString
        PRINT, "loggString = ", loggString

        koesterModelFileName = "da" + teffString + "_" + STRMID(loggString,0,1) + STRMID(loggString
             ,2,2) + ".dk_tr_rg"
        tlustyModelFileName = "ngc3532-lls" + STRMID(teffString,0,3) + STRMID(loggString,0,1) +
             STRMID(loggString,2,2) + ".dat"

        PRINT, "koesterModelFileName = ", koesterModelFileName
        PRINT, "tlustyModelFileName = ", tlustyModelFileName
        readcol64bitvars, koesterModelFileName, koesterwavelengthArray, koesteramplitudeArray,
             FORMAT='D,D'
        readcol64bitvars, tlustyModelFileName, tlustywavelengthArray, tlustyamplitudeArray, FORMAT='
             D,D'

        ;normalise koester and tlusty model (to 1.0 max)
        maxkoesteramplitudeArrayValue = max(koesteramplitudeArray)
        maxtlustyamplitudeArrayValue = max(tlustyamplitudeArray)
```

```
        PRINT, "maxkoesteramplitudeArrayValue = ", maxkoesteramplitudeArrayValue
        PRINT, "maxtlustyamplitudeArrayValue = ", maxtlustyamplitudeArrayValue
        koesteramplitudeArrayNormalised = koesteramplitudeArray/maxkoesteramplitudeArrayValue
        tlustyamplitudeArrayNormalised = tlustyamplitudeArray/maxtlustyamplitudeArrayValue

        ;graph in tlusty and koester model
        plotTitle = 'Logg:' + loggString + ', teff=' + teffString
        plot, koesterwavelengthArray, koesteramplitudeArrayNormalised, title=plotTitle, xrange=[
            minWavelength,maxWavelength], xtitle='wavelength (Ang)', ytitle='amplitude (norm)',
            yrange=[0,1.0] ;,linestyle=5,psym=3, SYMSIZE=1
        oplot, koesterwavelengthArray, koesteramplitudeArrayNormalised, COLOR=1
        oplot, tlustywavelengthArray, tlustyamplitudeArrayNormalised, COLOR=2

    endfor
endfor


end
```

A.5.4 Interpolate ASCII Spectroscopic Model Grid and create ASCII model using Best Fit Temperature and Surface Gravity values

This code has been used to create ASCII WD spectroscopic models using the temperature and surface gravity values derived from Xspec modelling and the ASCII model grids.

interpolateTlustyOrKoesterASCIIModelGrid.pro

interpolateTlustyOrKoesterASCIIModelGrid-29Aug10a.zip

```
; Title: interpolate Tlusty Or Koester ASCII Model Grid
; File: interpolateTlustyOrKoesterASCIIModelGrid.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 28 August 10 5:00pm
; Summary:
;
;   Postconditions;
;
;   This program interpolates Tlusty Or Koester ASCII Model Grid for an ideal logg and teff value,
    and outputs the interpolated model grid
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   ngc3532-lls300750.dat [wavelengthArray, amplitudeArray], space delimited
;   da15000_900.dk_tr_rg [wavelengthArray, amplitudeArray], space delimited
;   Usage: gdl
;   .run interpolateTlustyOrKoesterASCIIModelGrid.pro


pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
```

```
                l_blue     : 12,$
                l_red      : 13 $
               }
;         0  1    2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]   ;blue
end


my_colors

;parameters edited by user;
targetTemperature = 16017.8      ;17927.9
targetLogg = 8.51626         ;8.25923
useKoester = 1
    ;NB Tlusty is required for all temperatures > 20000K


if(useKoester eq 1) then begin

    minTemperature = 6000    ;6000
    maxTemperature = 20000
    incrementTemperature = 1000
    numTemperature = ((maxTemperature-minTemperature)/incrementTemperature) + 1
    temperatureArray = dblarr(numTemperature)

    minLogg = 7.00
    maxLogg = 9.50
    incrementLogg = 0.25
    numLogg = ((maxLogg-minLogg)/incrementLogg) + 1
    loggArray = dblarr(numLogg)

    minWavelength = 3500.0
    maxWavelength = 5300.0
    incrementWavelength = 0.25
    numberOfWavelengthPoints = ((maxWavelength-minWavelength)/incrementWavelength) + 1  ;7201

endif else begin

    minTemperature = 15000
    maxTemperature = 47500
    incrementTemperature = 2500
    numTemperature = ((maxTemperature-minTemperature)/incrementTemperature) + 1
    temperatureArray = dblarr(numTemperature)

    minLogg = 7.25
    maxLogg = 8.75
    incrementLogg = 0.25
    numLogg = ((maxLogg-minLogg)/incrementLogg) + 1
    loggArray = dblarr(numLogg)

    minWavelength = 3500.12512
    maxWavelength = 5299.87512
    incrementWavelength = 0.25
    numberOfWavelengthPoints = ((maxWavelength-minWavelength)/incrementWavelength) + 1  ;7200

endelse

;print, "numberOfWavelengthPoints = ", numberOfWavelengthPoints


wavelengthArray = dblarr(numberOfWavelengthPoints)

data = dblarr(numLogg, numTemperature, numberOfWavelengthPoints)


loggIndex = 0

for logg=minLogg, maxLogg, incrementLogg do begin

    temperatureIndex = 0

    for teff=minTemperature, maxTemperature, incrementTemperature do begin

        ;read in koester or lusty model

        teffString = STRING(teff, FORMAT='(I5.5)')
        loggString = STRING(logg, FORMAT='(F0.2)')
        PRINT, "teffString = ", teffString
        PRINT, "loggString = ", loggString

        if(useKoester) then begin
            ;read in koester model
            koesterModelFileName = "da" + teffString + "_" + STRMID(loggString,0,1) + STRMID(
                loggString,2,2) + ".dk_tr_rg"
            PRINT, "koesterModelFileName = ", koesterModelFileName
            readcol64bitvars, koesterModelFileName, wavelengthArray, amplitudeArray, FORMAT='D,D'

            ;graph koester model
            ;plotTitle = 'Logg:' + loggString + ', teff=' + teffString
```

```
            ;plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,
                maxWavelength], xtitle='wavelength (Ang)', ytitle='amplitude (norm)', yrange
                =[0,1.0] ;,linestyle=5,psym=3, SYMSIZE=1


        endif else begin
            ;read in lusty model
            tlustyModelFileName = "ngc3532-lls" + STRMID(teffString,0,3) + STRMID(loggString,0,1) +
                STRMID(loggString,2,2) + ".dat"
            PRINT, "tlustyModelFileName = ", tlustyModelFileName
            readcol64bitvars, tlustyModelFileName, wavelengthArray, amplitudeArray, FORMAT='D,D'

            ;graph in tlusty model
            ;plotTitle = 'Logg:' + loggString + ', teff=' + teffString
            ;plot, wavelengthArray, amplitudeArray, title=plotTitle, xrange=[minWavelength,
                maxWavelength], xtitle='wavelength (Ang)', ytitle='amplitude (norm)', yrange
                =[0,1.0] ;,linestyle=5,psym=3, SYMSIZE=1

        endelse

        print, "amplitudeArray = ", amplitudeArray
        starInputValuesArraySize = SIZE(amplitudeArray)
        numberOfStars = starInputValuesArraySize[1]
        print, "numberOfStars = ", numberOfStars

        data[loggIndex,temperatureIndex,*] = amplitudeArray
        ;PRINT, "data[loggIndex,temperatureIndex,*] = ", data[loggIndex,temperatureIndex,*]

        temperatureArray(temperatureIndex) = teff
        temperatureIndex = temperatureIndex + 1

    endfor

    loggArray(loggIndex) = logg
    loggIndex = loggIndex + 1
endfor

PRINT, "temperatureArray = ", temperatureArray
PRINT, "loggArray = ", loggArray


; method;
; for all points (wavelengths);
;   for all surf grav
;       for all temperatures, find ideal logg value, based on interpolation of T1
;   interpolate this array to find ideal point at surf grav SG1


valueToFindTempArray = double(1)

amplitudeAtClosestTemperatureArray = dblarr(numLogg)
amplitudeAtClosestLoggAndTemperatureArray = dblarr(numberOfWavelengthPoints)

for pt=0, numberOfWavelengthPoints-1, 1 do begin

    wavelength = minWavelength + (pt*incrementWavelength)
    wavelengthArray(pt) = wavelength

    loggIndex = 0

    for logg=minLogg, maxLogg, incrementLogg do begin

        valueToFind = targetTemperature
        valueToFindTempArray(0) = valueToFind

        closestTemperature = valueToFind
        amplitudeAtClosestTemperature = INTERPOL(data[loggIndex,*,pt], temperatureArray,
            valueToFindTempArray) ;, /SPLINE
        amplitudeAtClosestTemperatureArray(loggIndex) = amplitudeAtClosestTemperature

        loggIndex = loggIndex + 1

        print, "amplitudeAtClosestTemperature(loggIndex) = ", amplitudeAtClosestTemperature, "(",
            loggIndex, ")"

    endfor

    valueToFind = targetLogg
    valueToFindTempArray(0) = valueToFind

    closestLogg = valueToFind
    amplitudeAtClosestLoggAndTemperature = INTERPOL(amplitudeAtClosestTemperatureArray, loggArray,
        valueToFindTempArray) ;, /SPLINE
    amplitudeAtClosestLoggAndTemperatureArray(pt) = amplitudeAtClosestLoggAndTemperature
endfor


;graph of interpolated model
plotTitle = 'Logg:' + loggString + ', teff=' + teffString
plot, wavelengthArray, amplitudeAtClosestLoggAndTemperatureArray, title=plotTitle, xrange=[
    minWavelength,maxWavelength], xtitle='wavelength (Ang)', ytitle='amplitude (norm)' ;,
    linestyle=5,psym=3, SYMSIZE=1
```

```
finalData = [wavelengthArray, amplitudeAtClosestLoggAndTemperatureArray]

;outputFileName='interpolatedModel.dat'
;OPENW,1,outputFileName
;PRINTF,1,finalData,FORMAT='(F,1X,F)'

PRINT, "finalData = "
PRINT, finalData

;NOT USED;
;for teff=minTemperature, maxTemperature, incrementTemperature do begin
;endfor


end
```

A.5.5 White Dwarf Parameter Extraction - including gravitational redshift values - NOT USED

This software is otherwise known as the "Fontaine" code. It has been replaced for the most part by the RBB Bergeron grid interpolation code (rbb_wd_dd_param.pro) - except for gravitational redshift and radius determination purposes.

Compilation Instructions

'g77 -c *.f [need to recompile *.f into .o objectfiles]'

'g77 g2m_fbbC.o hunt.o interpralf.o loadvec.o readt1C.o spline.o splt2.o interp.o splint.o splt.o [only C related] [link them all into an executable]'

'g77 g2m_fbbCO.o hunt.o interpralf.o loadvec.o readt1CO.o spline.o splt2.o interp.o splint.o splt.o [only CO related] [link them all into an executable]'

Execution Instructions

./g2m_fbbCO

Inputs

hyad.in [created based upon output of XSPEC] A or B specify name irr irr opttemp low high gopt glow ghigh flag[1,0]-gravitationalredshiftinfo?

Outputs

hyad.out NB GR - Km/s

A.5.6 Flux Recalibration using Best Fit ASCII Spectroscopic Model

This code has been used to recalibrate the flux of a set of magnetic WD candidates before having modelling performed externally by a Magnetic WD expert.

fitPolyToDataModelRatio.pro

fitPolyToDataModelRatio-30Aug10a.zip

```
; Title: fit Poly To Data:Model Ratio
; File: fitPolyToDataModelRatio.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 28 August 10 9:00pm
; Summary:
;
;   Postconditions;
;
;   This program fits a poly to a data model ratio
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   WDModelFile: DD-0Xqmodel.dat [wavelengthArray, amplitudeArray], space delimited
;   WDDataFile: SSWD0Xqw.dat [wavelengthArray, amplitudeArray], space delimited
;       make sure that data file's last amplitude value is correct (not set to 0.0)
;   Usage: gdl
;   .run fitPolyToDataModelRatio.pro


pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
                l_blue    : 12,$
                l_red     : 13 $
               }
;         0  1   2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]    ;blue
end


my_colors

WDModelFileName = "DD-02Amodel.dat" ;DD-04Amodel.dat     ;DD-04Bmodel.dat
WDDataFileName = "SSWD02Aw.dat"     ;SSWD04Aw.dat        ;SSWD04Bw.dat
readcol64bitvars, WDModelFileName, WDModelwavelengthArray, WDModelamplitudeArray, FORMAT='D,D'
readcol64bitvars, WDDataFileName, WDDatawavelengthArray, WDDataamplitudeArray, FORMAT='D,D'

WDModelArraySizeTemp = SIZE(WDModelamplitudeArray)
WDModelArraySize = WDModelArraySizeTemp[1]
WDModelIncrement = 1/0.25

WDDataArraySizeTemp = SIZE(WDDataamplitudeArray)
WDDataArraySize = WDDataArraySizeTemp[1]
WDDataIncrement = 1/1

numberOfWavelengthPoints = WDDataArraySize*WDDataIncrement
    ;[or numberOfWavelengthPoints = WDModelArraySize*WDModelIncrement]

PRINT, "numberOfWavelengthPoints = ", numberOfWavelengthPoints

dataModelRatioArray = dblarr(numberOfWavelengthPoints)
wavelengthArray = dblarr(numberOfWavelengthPoints)
dataModelRatioContinuumArray = dblarr(numberOfWavelengthPoints)
wavelengthContinuumArray = dblarr(numberOfWavelengthPoints)
WDModelamplitudeArrayAligned = dblarr(numberOfWavelengthPoints)
WDDataamplitudeArrayAligned = dblarr(numberOfWavelengthPoints)
WDDataamplitudeArrayAlignedRecalibrated = dblarr(numberOfWavelengthPoints)

WDModelMinWavelength = WDModelwavelengthArray(0)
WDModelmaxWavelength = WDModelwavelengthArray(WDModelArraySize-1)
WDDataMinWavelength = WDDatawavelengthArray(0)
```

```
WDDatamaxWavelength = WDDatawavelengthArray(WDDataArraySize-1)


;perform quick normalisation;
averageWDModelAmplitude = 0.0
averageWDDataAmplitude = 0.0
for pt=0, numberOfWavelengthPoints-1, 1 do begin
    WDModelAmplitude = WDModelamplitudeArray(pt*WDModelIncrement)
    WDDataAmplitude = WDDataamplitudeArray(pt*WDDataIncrement)
    averageWDDataAmplitude = averageWDDataAmplitude + WDDataAmplitude
    averageWDModelAmplitude = averageWDModelAmplitude + WDModelAmplitude
endfor
averageWDDataAmplitude = averageWDDataAmplitude/numberOfWavelengthPoints
averageWDModelAmplitude = averageWDModelAmplitude/numberOfWavelengthPoints


pt = 0
for wavelength=WDDataMinWavelength, WDDatamaxWavelength, 1 do begin
    WDModelamplitudeArrayIndex = ((wavelength-WDModelMinWavelength)*WDModelIncrement)
        ;PRINT, "WDModelamplitudeArrayIndex = ", WDModelamplitudeArrayIndex
    WDModelAmplitude = WDModelamplitudeArray(WDModelamplitudeArrayIndex)
        ;PRINT, "WDModelAmplitude = ", WDModelAmplitude
    WDModelAmplitude = (WDModelAmplitude/averageWDModelAmplitude)*averageWDDataAmplitude
    WDModelamplitudeArrayAligned(pt) = WDModelAmplitude

    WDDataamplitudeArrayIndex = ((wavelength-WDDataMinWavelength)*WDDataIncrement)
        ;PRINT, "WDDataamplitudeArrayIndex = ", WDDataamplitudeArrayIndex
    WDDataAmplitude = WDDataamplitudeArray(WDDataamplitudeArrayIndex)
    WDDataamplitudeArrayAligned(pt) = WDDataAmplitude
        ;PRINT, "WDDataAmplitude = ", WDDataAmplitude
    dataModelRatio = WDModelAmplitude/WDDataAmplitude
    dataModelRatioArray(pt) = dataModelRatio
    wavelengthArray(pt) = wavelength
    pt = pt+1
endfor
initialWavelength = wavelengthArray(0)



;graph of interpolated model
plotTitle = 'Amplitude versus Wavelength'
plot, wavelengthArray, WDDataamplitudeArrayAligned, title=plotTitle, yrange=[0.0, 2.0], xrange=[
    WDDatawavelengthArray(0),WDDatawavelengthArray(numberOfWavelengthPoints-1)], xtitle='wavelength
     (Ang)', ytitle='amplitude (norm)', COLOR=2  ;,linestyle=5,psym=3, SYMSIZE=1
oplot, wavelengthArray, WDModelamplitudeArrayAligned, COLOR=3   ;,linestyle=5,psym=3, SYMSIZE=1
    ;yrange=[0,1.5]


polyFitOrder = 3

; fit using all points (not just continuuum);

dataModelRatioArrayPolyFitParameters = poly_fit(wavelengthArray, dataModelRatioArray, polyFitOrder)

dataModelRatioArrayTheory = poly(wavelengthArray, dataModelRatioArrayPolyFitParameters)

oplot, wavelengthArray, dataModelRatioArray, COLOR=4    ;,linestyle=5,psym=3, SYMSIZE=1
oplot, wavelengthArray, dataModelRatioArrayTheory, COLOR=5  ;,linestyle=5,psym=3, SYMSIZE=1

;now create flux recalibrated data file;
for pt=0, numberOfWavelengthPoints-1, 1 do begin
    WDDataamplitudeArrayAlignedRecalibrated(pt) = WDDataamplitudeArrayAligned(pt)*
        dataModelRatioArrayTheory(pt)
endfor

oplot, wavelengthArray, WDDataamplitudeArrayAlignedRecalibrated, COLOR=5    ;,linestyle=5,psym=3,
    SYMSIZE=1

wavelengthArrayTransposed = transpose(wavelengthArray)
WDDataamplitudeArrayAlignedRecalibratedTransposed = transpose(
    WDDataamplitudeArrayAlignedRecalibrated)

finalData = [wavelengthArrayTransposed, WDDataamplitudeArrayAlignedRecalibratedTransposed]
outputFileName=WDDataFileName + '.FluxRecalibratedUsingModel'
OPENW,1,outputFileName
PRINTF,1,finalData,FORMAT='(F,1X,F)'
CLOSE,1



                ; fit only the continuuum;
                ;
                ;
                ;numberBalmerLines = 6
                ;numberBalmerLinesUsed = 5
                ;balmerLinesLabXPos = dblarr(numberBalmerLines)
                ;;balmerLinesLabXPos[6] = 6562.852  ;H-Alpha - not used
                ;balmerLinesLabXPos[0] = 4861.33       ;H-Beta
                ;balmerLinesLabXPos[1] = 4340.47       ;H-Gamma
                ;balmerLinesLabXPos[2] = 4101.74       ;H-Delta
                ;balmerLinesLabXPos[3] = 3970.072   ;H-Epsilon
                ;balmerLinesLabXPos[4] = 3889.049   ;H-8
```

```
;balmerLinesLabXPos[5] = 3835.384    ;H-..
;initialContinuumPointWavelength = 3850.0
;
;balmerLinesLabXWidth = dblarr(numberBalmerLines)
;;NB left and right normalisation points are always the same distance from the
    centre
;balmerLinesLabXWidth[0] = 120           ;H-Beta
;balmerLinesLabXWidth[1] = 120           ;H-Gamma
;balmerLinesLabXWidth[2] = 70        ;H-Delta
;balmerLinesLabXWidth[3] = 30        ;H-Epsilon
;balmerLinesLabXWidth[4] = 30        ;H-8
;balmerLinesLabXWidth[5] = 20        ;H-..
;
;
;
;continuumArrayIndex = 0
;previousContinuumPointWavelength = initialContinuumPointWavelength
;for balmerLineIndex=0, numberBalmerLines-1, 1 do begin
;    ; fit only the continuum;
;    balmerLineWidth = balmerLinesLabXWidth[balmerLineIndex]
;    balmerLineWavelength = long(balmerLinesLabXPos[balmerLineIndex])
;
;    PRINT, "balmerLineWavelength = ", balmerLineWavelength
;    for wavelength=previousContinuumPointWavelength, balmerLineWavelength-
    balmerLineWidth, 1 do begin
;        dataModelRatio = dataModelRatioArray[wavelength-initialWavelength]
;        dataModelRatioContinuumArray[continuumArrayIndex] = dataModelRatio
;        wavelengthContinuumArray[continuumArrayIndex] = wavelength
;        continuumArrayIndex = continuumArrayIndex+1
;    endfor
;
;    previousContinuumPointWavelength = balmerLineWavelength+balmerLineWidth
;endfor
;
;oplot, wavelengthContinuumArray, dataModelRatioContinuumArray, COLOR=3 ;,linestyle
    =5,psym=3, SYMSIZE=1
;stop
;
;continuumArraySize = continuumArrayIndex
;dataModelRatioContinuumArrayCompacted = dblarr(continuumArraySize)
;wavelengthContinuumArrayCompacted = dblarr(continuumArraySize)
;for continuumIndex=0, continuumArraySize-1, 1 do begin
;    dataModelRatioContinuumArrayCompacted[continuumIndex] =
    dataModelRatioContinuumArray[continuumIndex]
;    wavelengthContinuumArrayCompacted[continuumIndex] = wavelengthContinuumArray[
    continuumIndex]
;endfor
;
;; fit continuum points;
;dataModelRatioContinuumArrayPolyFitParameters = poly_fit(
    wavelengthContinuumArrayCompacted, dataModelRatioContinuumArrayCompacted,
    polyFitOrder)
;
;oplot, wavelengthContinuumArrayCompacted, dataModelRatioContinuumArrayTheory, COLOR
    =3  ;,linestyle=5,psym=3, SYMSIZE=1
;

end
```

A.5.7 Interpolate ASCII Main Sequence Stellar evolutionary model grid using theoretical main sequence mass or age values to calculate a best fit main sequence age or mass

This code has been used during IFMR refinement for both (binary/open cluster) system age calculations (Interpolate ASCII Main Sequence Stellar evolutionary model grid using theoretical main sequence mass to calculate a best fit main sequence age) and initial mass calculations (Interpolate ASCII Main Sequence Stellar evolutionary model grid using theoretical main sequence age value to calculate a best fit main sequence mass).

A.5.8 ASCII Model Grid Creation Software - performed by PDD and Koester

synspec43.tar.gz

tlusty200.tar.gz

Organisation of ASCII Tlusty Model Grid Files

tlustyModelgridOrganise.csh

```
#!/bin/csh
#
foreach teff (14000 15000 17500 20000 22500 25000 27500 30000 32500 35000 37500 40000 42500 45000
     47500)

    cd /home/rich/tasks/atmosphericmodelfitting/working/modelgrid/tlustymodelgrid/LRES/
    /bin/mkdir $teff

    set teffpreceed=""
    if ($teff < 10000) then
        set teffpreceed="0"
    endif
    #echo $teffpreceed

    foreach logg (7.25 7.50 7.75 8.00 8.25 8.50 8.75)

        /bin/mkdir $teff/$logg

        set teffsplita = `echo $teff | awk '{print substr($0,1,3)}'`
        #echo $teffsplita


        set loggsplita = `echo $logg | awk '{print substr($0,1,1)}'`
        set loggsplitb = `echo $logg | awk '{print substr($0,3,4)}'`
        #echo $loggsplita
        #echo $loggsplitb

        set filename = "ngc3532-lls$teffsplita$loggsplita$loggsplitb.dat"
        #echo $filename
        /bin/cp $filename $teff/$logg/
        /bin/cp $filename $teff/$logg/"TLUSTYDA"
    end
end

foreach teff (16000)

    cd /home/rich/tasks/atmosphericmodelfitting/working/modelgrid/tlustymodelgrid/LRES/
    /bin/mkdir $teff

    set teffpreceed=""
    if ($teff < 10000) then
        set teffpreceed="0"
    endif
    #echo $teffpreceed

    foreach logg (7.25 7.50 7.75 8.00 8.25 8.50 8.75)

        /bin/mkdir $teff/$logg

        set teffsplita = `echo $teff | awk '{print substr($0,1,3)}'`
        #echo $teffsplita


        set loggsplita = `echo $logg | awk '{print substr($0,1,1)}'`
        set loggsplitb = `echo $logg | awk '{print substr($0,3,4)}'`
        #echo $loggsplita
        #echo $loggsplitb

        set filename = "T203_BF20_$teffsplita$loggsplita$loggsplitb.dat"
        #echo $filename
        /bin/cp $filename $teff/$logg/
        /bin/cp $filename $teff/$logg/"TLUSTYDA"
    end
end
```

Organisation of ASCII Koester Model Grid Files

koesterModelgridOrganise.csh

```
#!/bin/csh
#
foreach teff (6000 7000 8000 9000 10000 11000 12000 13000 14000 15000 16000 17000 18000 19000 20000)
    cd /home/rich/tasks/atmosphericmodelfitting/working/modelgrid/koestermodelgrid/LRES/
    /bin/mkdir $teff

    set teffpreceed=""
    if ($teff < 10000) then
        set teffpreceed="0"
    endif
    #echo $teffpreceed

    foreach logg (7.00 7.25 7.50 7.75 8.00 8.25 8.50 8.75 9.00 9.25 9.50)

        /bin/mkdir $teff/$logg


        set loggsplita = `echo $logg | awk '{print substr($0,1,1)}'`
        set loggsplitb = `echo $logg | awk '{print substr($0,3,4)}'`
        #echo $loggsplita
        #echo $loggsplitb

        set filename = "da$teffpreceed$teff"_"$loggsplita$loggsplitb.dk_tr_rg"
        /bin/cp $filename $teff/$logg/
        /bin/cp $filename $teff/$logg/"KOESTERDA.dat"
    end
end
```

A.5.9 XSpec Software

xspec11.3.2ag-src.tar.gz

Xspec requires data files and model grid to be converted to Fits.

Conversion of ASCII WD Model Grid to Xspec Fits Format - performed by PDD - sag2xam_mp.pl

PDD's ASCII WD Model Grid to Xspec Fits Format software.

sag2xam_mp.pl

```
#!/usr/bin/perl
##############################################################################
#
# grid of ascii SYNSPEC output --> XSPEC atable model file, OGIP (like) FITS
#
##############################################################################
#
    use Astro::FITS::CFITSIO qw(READWRITE TLOGICAL TSTRING TFLOAT TLONG TUSHORT TINT);
    use Carp;
    use POSIX;
#
    my $a2kev=12.39854; my $erg2kev=1.602177e-9;
    my $cang=2.99793e+18; my $hl=6.62620e-27; my $pi=3.141592654;

#
    my $outtext0=sprintf("\n\n                                                  ------------------------ \n");
    my $outtext1=sprintf("                                              | SAG2XAM -- V1.0mp | \n");
    my $outtext2=sprintf("                                               ------------------------ \n ");
#
    my $outtext3=sprintf("\n\nEnter the name of the model control file          > ");
    my $outtext4=sprintf("\nEnter a name for the model                        > ");
    my $outtext5=sprintf("\n\nYou have entered the following                    > \n\n");
#
    printf $outtext0.$outtext1.$outtext2;
#
    my $ans='n';
    my $infile,$outfile;

#
    while (($ans eq 'n') || ($ans eq 'N')){

        printf $outtext3;
        chomp($infile=<STDIN>);

    printf $outtext4;
    chomp($outfile=<STDIN>);

    my $outtext6=sprintf("INFILE   : %15s \nOUTFILE  : %15s \n\n",$infile,$outfile);
        my $outtext7=sprintf("Is this correct ? (y/n) ");

        printf $outtext5.$outtext6.$outtext7;
```

```
    chomp($ans=<STDIN>);
    }
#
    open(input,$infile) || die "Couldn't open file ".$infile." \n";
#
    my $l1,$l2;

    chomp($l1=<input>);
    my (@noofeachparam)=split(" ",$l1);

    my $nofiles=1;
    my $nostates=1;
    my $nospreg=$noofeachparam[$#noofeachparam];
    my $pc,$pcc;

    for ($pc=0; $pc<=$#noofeachparam; $pc++) {
    $nofiles=$nofiles*$noofeachparam[$pc];
    }

    print "Total files to be read in: ".$nofiles."\n";

    for ($pc=0; $pc<$#noofeachparam; $pc++) {
    $nostates=$nostates*$noofeachparam[$pc];
    }

#   read in the parameter values at each grid point

    my (@nval,@method,@paramnam,@initv,@hminv);
    my (@sminv,@delt,@smaxv,@hmaxv);

    print "Parameters: \n";

    for ($pc=0; $pc<$#noofeachparam; $pc++) {
    my $l2;
    chomp($l2=<input>);
    my @pval=split(" ",$l2);
    $nval[$pc]=$#pval;
    for ($pcc=0; $pcc<=$#pval; $pcc++) {
        $parval[$pc][$pcc]=$pval[$pcc];
    }
    $method[$pc]=0;
    $paramnam[$pc]=$parval[$pc][0];
    print $paramnam[$pc]."\n";
    }

#   length of above arrays and create arrays to feed into write_param_exten call

     my $nparams=$#noofeachparam;
     my $width=44;

     my $nmax=0;
     for ($pc=0; $pc<=$#nval; $pc++) {
     if ($nval[$pc]>$nmax) {
         $nmax=$nval[$pc];
     }
     $initv[$pc]=$parval[$pc][1];
     $hminv[$pc]=$parval[$pc][1];
     $sminv[$pc]=$parval[$pc][1];
     $delt[$pc]=abs($parval[$pc][1]*1.0e-3);
     $smaxv[$pc]=$parval[$pc][$nval[$pc]];
         $hmaxv[$pc]=$parval[$pc][$nval[$pc]];
     }

     $width=$width+4*$nmax;

     my $status;
     my $opfptr1 = Astro::FITS::CFITSIO::create_file($outfile,$status);
     check_status($status);
#
     rewrite_prim_header($opfptr1,$infile,'SYNSPEC',$outfile);
     write_param_header($opfptr1,$width,$nparams,$nmax);
     write_param_exten($opfptr1,\@paramnam,\@method,\@initv,\@hminv,\@sminv,\@delt,\@smaxv,\@hmaxv,\
          @nval,$nparams);

     $felem=1;

#    beware of spurious entries in VALUE array....check that this is not a problem!!!

     for ($pc=0; $pc<$#noofeachparam; $pc++){
     $frow=$pc+1;
     my @ppvals;
     for ($ppc=1; $ppc<=$nval[$pc]; $ppc++) {
         $ppvals[$ppc-1]=$parval[$pc][$ppc];
     }
     $opfptr1->write_col(42,10,$frow,$felem,$nmax,\@ppvals,$status);
     }

     check_status($status);

#    energ ext: read explicit filenames into the array "filename" and close control file

     my $fl,$lf,@filename;
```

```
      for ($fl=0; $fl<$nofiles; $fl++){
          chomp($lf=<input>);
          $filename[$fl]=$lf;
      }
         close(input);

#    setup the energy grid using info in the first group of files.

      my $i=0;
      my (@midangs,@flux,@midkev);

      for ($fl=0; $fl<$nospreg; $fl++){

      open(input,$filename[$fl]) || die "Couldn't open file ".$filename[$fl]." \n";
      while (<input>) {
      chomp;
      ($midangs[$i],$flux[$i])=split " ",$_;
      $midkev[$i]=$a2kev/$midangs[$i];
       $i++;
           }
         $i--;
      close(input);
  }

      my ($ep,@dele,@energ_hi,@energ_lo,@delw);

      for ($ep=0; $ep<$i; $ep++) {

          $dele[$ep]=$midkev[$#midkev-($ep+1)]-$midkev[$#midkev-$ep];
          $energ_hi[$ep]=$midkev[$#midkev-$ep]+0.5*$dele[$ep];

          if ($ep > 0) {
              $energ_lo[$ep]=$energ_hi[$ep-1];
         }
      }

           $energ_hi[$i]=$midkev[0]+0.5*$dele[$i-1];
           $energ_lo[$i]=$energ_hi[$i-1];
           $energ_lo[0]=$midkev[$i]-0.5*$dele[0];

#        need bin widths in angstroms for flux unit conversion

          for ($ep=0; $ep<=$i; $ep++) {
          $delw[$ep]=$a2kev/$energ_lo[$ep]-$a2kev/$energ_hi[$ep];
          }
#
           my $nerows=@energ_hi;
#
           write_energ_header($opfptr1,$nerows);
           write_energ_arrays($opfptr1,\@energ_lo,\@energ_hi);

#         now determine details for spectra extension

          write_spec_header($opfptr1,$nostates,$nerows,$nparams);

#         write paramval array

          $frow=1;
          $felem=1;

          my @pidx;
          for ($pc=0; $pc<$#noofeachparam; $pc++){
          $pidx[$pc]=1;
      }

          while ($frow<=$nostates) {

          for ($pc=$#noofeachparam-1; $pc>=0; $pc--){
          $cpars[$pc]=$parval[$pc][$pidx[$pc]];
          }

          $opfptr1->write_col(42,1,$frow,$felem,$nparams,\@cpars,$status);

          $pidx[$#noofeachparam-1]++;

          for ($pc=$#noofeachparam-1; $pc>0; $pc--){
          if ($pidx[$pc]>$nval[$pc]) {
              $pidx[$pc]=1;
              $pidx[$pc-1]++;
          }
          }
          $frow++;
      }


#         write intpspec array

          my $outer;

          for ($outer=0; $outer<$nostates; $outer++) {
```

```
          my (@mangs,@ergflux,$file,$inner,@ergfluxi,@mkev);
          my $i=0;
          for ($inner=0; $inner<$nospreg; $inner++){
          $file=$nospreg*$outer + $inner;

          open(input,$filename[$file]) || die "Couldn't open file ".$filename[$file]." \n";

          while (<input>) {
              chomp;
              ($mangs[$i],$ergfluxi[$i])=split(" ",$_);
              $i++;
          }
          $i--;
          close(input);
          }

          for ($rord=0; $rord<=$i; $rord++){
          $mkev[$rord]=$a2kev/$mangs[$#mangs-$rord];
          $ergflux[$rord]=$ergfluxi[$#ergfluxi-$rord];
          $photflux[$rord]=4.0/1e20*$pi*$delw[$rord]*$a2kev/$mkev[$rord] * $ergflux[$rord]/($hl *
               $cang);
          }

#     reverse order of flux array (low energy first) and convert flux from ergs to Kev etc

#         for ($ep=0; $ep<$nerows; $ep++) {
#         $photflux[$ep]=4.0/1e20*$pi*$delw[$ep]*$a2kev/$mkev[$ep] * $ergflux[$ep]/($hl * $cang);
#         }

#       and write flux array to file....

          $felem=1;
          $frow=$outer+1;
          $opfptr1->write_col(42,2,$frow,$felem,$nerows,\@photflux,$status);
      }

#         and finally close the file

     $opfptr1->close_file($status);

################################################################################

sub write_param_header{

     my($fptr,$pn1,$pn2,$nm)=@_;
     my($extn,$cls,$cls1,$cls2,$vers,$bits,$pnax,$pc,$gc,$tf,$acc);
     my($colnam,$form,$unit);

#       define a few of the keywords required in extension header

         $bits=8;
         $pnax=2;
         $pc=0;
         $gc=1;
         $tf=10;
         $extn='PARAMETERS';
         $cls='OGIP';
         $cls1='XSPEC TABLE MODEL';
         $cls2='PARAMETERS';
         $vers='1.0.0';
         $acc=$nm.'E';

#        details of the columns in the PARAMETERS extension: names, formats, units

         $colnam=['NAME','METHOD','INITIAL','DELTA','MINIMUM','BOTTOM','TOP','MAXIMUM','NUMBVALS','
              VALUE'];
         $form=['12A','J','1E','1E','1E','1E','1E','1E','J',$acc];
         $unit=['','','','','','','','','','',''];

#        create the PARAMETERS binary table extension

         $fptr->create_tbl(2,0,$tf,$colnam,$form,$unit,$extn,$status);
         check_status($status);

#        write required keywords to the header of this extension

         $fptr->update_key(TINT,    'BITPIX',  $bits,  '8-bit bytes',              $status);
         $fptr->update_key(TINT,    'NAXIS',   $pnax,  '2-dim binary table',       $status);
         $fptr->update_key(TINT,    'NAXIS1',  $pn1,   'width in bytes',           $status);
         $fptr->update_key(TINT,    'NAXIS2',  $pn2,   'no. of table rows',        $status);
         $fptr->update_key(TINT,    'PCOUNT',  $pc,    'size of special data area',$status);
         $fptr->update_key(TINT,    'GCOUNT',  $gc,    'only one data group',      $status);
         $fptr->update_key(TINT,    'TFIELDS', $tf,    'no. of fields in each row',$status);
         $fptr->update_key(TSTRING,'HDUCLASS',$cls,   'Conforms to OGIP standard',$status);
         $fptr->update_key(TSTRING,'HDUCLAS1',$cls1,  'extension contents 1',     $status);
         $fptr->update_key(TSTRING,'HDUCLAS2',$cls2,  'extension contents 2',     $status);
         $fptr->update_key(TSTRING,'HDUVERS1',$vers,  'file format version',      $status);
         $fptr->update_key(TINT,    'NINTPARM',$pn2,   'no. of interp. pars.',    $status);
         $fptr->update_key(TINT,    'NADDPARM',0,      'no. of additional pars.', $status);

#        finished initialising PARAMETERS extension. Now return to main program
```

```
    return;
}

################################################################################

sub write_energ_header{

    my($fptr,$en2)=@_;
    my($extn,$cls1,$cls2,$vers,$bits,$enax,$en1,$pc,$gc,$tf);
    my($colnam,$form,$unit);

#       define a few of the keywords required in extension header

        $bits=8;
        $enax=2;
        $en1=8;
        $pc=0;
        $gc=1;
        $tf=2;
        $extn='ENERGIES';
        $cls1='XSPEC TABLE MODEL';
        $cls2='ENERGIES';
        $vers='1.0.0';

#       details of the columns in the ENERGIES extension: names, formats, units

        $colnam=['ENERG_LO','ENERG_HI'];
        $form=['1E','1E'];
        $unit=['keV','keV'];

#       create the ENERGIES binary table extension

        $fptr->create_tbl(2,0,$tf,$colnam,$form,$unit,$extn,$status);
        check_status($status);

#       write required keywords to the header of this extension

        $fptr->update_key(TINT,    'BITPIX',  $bits,  '8-bit bytes',              $status);
        $fptr->update_key(TINT,    'NAXIS',   $enax,   '2-dim binary table',       $status);
        $fptr->update_key(TINT,    'NAXIS1',  $en1,    'width in bytes',           $status);
        $fptr->update_key(TINT,    'NAXIS2',  $en2,    'no. of table rows',        $status);
        $fptr->update_key(TINT,    'PCOUNT',  $pc,     'size of special data area',$status);
        $fptr->update_key(TINT,    'GCOUNT',  $gc,     'only one data group',      $status);
        $fptr->update_key(TINT,    'TFIELDS', $tf,     'no. of fields in each row',$status);
        $fptr->update_key(TSTRING,'HDUCLAS1',$cls1,   'extension contents 1',     $status);
        $fptr->update_key(TSTRING,'HDUCLAS2',$cls2,   'extension contents 2',     $status);
        $fptr->update_key(TSTRING,'HDUVERS1',$vers,   'file format version',      $status);

#       finished initialising ENERGIES extension. Now return to main program

    return;
}

################################################################################

################################################################################

sub write_spec_header{
#
    my($fptr,$sn2,$sn3,$sn4)=@_;
    my($extn,$cls1,$cls2,$vers,$bits,$snax,$pc,$gc,$tf,$acc1);
    my($colnam,$form,$unit,$small,$sn1);
#
#       define a few of the keywords required in extension header
#

        $bits=8;
        $snax=2;
        $pc=0;
        $gc=1;
        $tf=2;
        $extn='SPECTRA';
        $cls1='XSPEC TABLE MODEL';
        $cls2='MODEL SPECTRA';
        $vers='1.0.0';
        $acc1=$sn3.'E';
        $acc2=$sn4.'E';

          $sn1=4*($sn3+$sn4);
#
#
#       details of the columns in the SPECTRA extension: names, formats, units
#
        $colnam=['PARAMVAL','INTPSPEC'];
        $form=[$acc2,$acc1];
        $unit=[' ','photons/cm^2/s'];
#
#       create the SPECTRA binary table extension
#
        $fptr->create_tbl(2,0,$tf,$colnam,$form,$unit,$extn,$status);
        check_status($status);
#
```

```
#       write required keywords to the header of this extension
#
        $fptr->update_key(TINT,    'BITPIX',  $bits,  '8-bit bytes',                $status);
        $fptr->update_key(TINT,    'NAXIS',   $snax,   '2-dim binary table',        $status);
        $fptr->update_key(TINT,    'NAXIS1',  $sn1,   'width in bytes',             $status);
        $fptr->update_key(TINT,    'NAXIS2',  $sn2,   'no. of table rows',          $status);
        $fptr->update_key(TINT,    'PCOUNT',  $pc,    'size of special data area',$status);
        $fptr->update_key(TINT,    'GCOUNT',  $gc,    'only one data group',        $status);
        $fptr->update_key(TINT,    'TFIELDS', $tf,    'no. of fields in each row',$status);
        $fptr->update_key(TSTRING,'HDUCLAS1',$cls1,  'extension contents 1',       $status);
        $fptr->update_key(TSTRING,'HDUCLAS2',$cls2,  'extension contents 2',       $status);
        $fptr->update_key(TSTRING,'HDUVERS1',$vers,  'file format version',        $status);
#
#       finished initialising SPECTRA extension. Now return to main program
#
    return;
}

###############################################################################

sub check_status {
    my $status = shift;
    if ($status) {
        my $errtxt;
      Astro::FITS::CFITSIO::fits_get_errstatus($status,$errtxt);
        croak("$0: Astro::FITS::CFITSIO error detected, aborting...$errtxt");
    }

    return 1;
}

###############################################################################

sub rewrite_prim_header {

    my($fptr,$fname,$code,$mname)=@_;
    my($cont,$reds,$addmo,$hcls,$hcls1,$hvers,$bits,$extn);

#       $fptr->create_img(8,0,0,$status);

        $cont='MODEL';
        $reds=1;
        $addmo=1;
        $hcls='OGIP';
        $hcls1='XSPEC TABLE MODEL';
        $hvers='1.0.0';
        $bits=8;
#
   #    $fptr->write_key(TSTRING,    'EXTNAME',    $extn,            'extension name',
        $status);
        $fptr->update_key(TLOGICAL, 'SIMPLE',     1,                'fits standard ',
                          $status);
        $fptr->update_key(TINT,      'BITPIX',    $bits,            'bits per pixel',
                          $status);
        $fptr->update_key(TINT,      'NAXIS',     0,                'no. of data axes',
                         $status);
        $fptr->update_key(TLOGICAL, 'EXTEND',     1,                'fits extensions present',
                        $status);
        $fptr->write_key(TSTRING,    'CONTENT',   $cont,            'a table model file',
                       $status);
        $fptr->write_key(TSTRING,    'FILENAME',  $fname,           'name of controlling file',
                      $status);
        $fptr->write_key(TSTRING,    'ORIGIN',    $code,            'origins of this fits file',
                     $status);
        $fptr->write_key(TSTRING,    'MODLNAME',  $mname,           'model name',
                         $status);
        $fptr->write_key(TSTRING,    'MODLUNIT',  'photons/cm^2/s','model units',
                        $status);
        $fptr->write_key(TLOGICAL,   'REDSHIFT',  $reds,            'if T then redshift param included
            ',  $status);
        $fptr->write_key(TLOGICAL,   'ADDMODEL',  $addmo,           'if T then additive model table',
                $status);
        $fptr->write_key(TSTRING,    'HDUCLASS',  $hcls,            'Conforms to OGIP standard',
                      $status);
        $fptr->write_key(TSTRING,    'HDUCLAS1',  $hcls1,           'model spectra for xspec',
                       $status);
        $fptr->write_key(TSTRING,    'HDUVERS1',  $hvers,           'version of file format',
                         $status);

        return;
    }

###############################################################################

sub write_param_exten {

    my($fptr,$pnam,$meth,$iniv,$hmiv,$smiv,$del,$smav,$hmav,$nval,$npar)=@_;
    my($frow,$felem);

    $frow=1;
    $felem=1;
```

```
    $fptr->write_col(16,1,$frow,$felem,$npar,$pnam,$status);
                     check_status($status);

    $fptr->write_col(21,2,$frow,$felem,$npar,$meth,$status);
                     check_status($status);

    $fptr->write_col(42,3,$frow,$felem,$npar,$iniv,$status);
                     check_status($status);


    $fptr->write_col(42,4,$frow,$felem,$npar,$del,$status);
                     check_status($status);


    $fptr->write_col(42,5,$frow,$felem,$npar,$hmiv,$status);
                     check_status($status);


    $fptr->write_col(42,6,$frow,$felem,$npar,$smiv,$status);
                     check_status($status);


    $fptr->write_col(42,7,$frow,$felem,$npar,$smav,$status);
                     check_status($status);

    $fptr->write_col(42,8,$frow,$felem,$npar,$hmav,$status);
                     check_status($status);

    $fptr->write_col(21,9,$frow,$felem,$npar,$nval,$status);
                     check_status($status);

    return;
}

##############################################################################

sub sortit{
    my ($array1,$array2,$array3)=@_;
    my ($n,$i,$j,$index);
    my ($first,$tmp1,$tmp2,$tmp3);

    $n=@$array1;

        for ($i=0;$i<($n-1);$i++){

            $first=$$array1[$i];
            $index=$i;

            for ($j=($i+1); $j<$n; $j++){
                if ($$array1[$j] < $first) {
                    $first=$$array1[$j];
                        $index=$j
                    }
            }

            if ($index != $i) {

                    $tmp1=$$array1[$i];
                    $$array1[$i]=$$array1[$index];
                    $$array1[$index]=$tmp1;

                    $tmp2=$$array2[$i];
                    $$array2[$i]=$$array2[$index];
                    $$array2[$index]=$tmp2;

                    $tmp3=$$array3[$i];

                    $$array3[$i]=$$array3[$index];
                    $$array3[$index]=$tmp3;


                }

        }

    return $array1,$array2,$array3;
}

##############################################################################

sub write_energ_arrays {

    my ($fptr,$enlo,$enhi)=@_;
    my ($frow,$felem,$nchan);
#
#   note that no grouping or quality information is written by this subroutine
#
    $nchan=@$enlo;

    $frow=1;
    $felem=1;
```

```
    $fptr->write_col(42,1,$frow,$felem,$nchan,$enlo,$status);
                    check_status($status);

    $fptr->write_col(42,2,$frow,$felem,$nchan,$enhi,$status);
                    check_status($status);
    return;
}
###############################################################################
sub sortit2{

    my ($array1,$array2)=@_;
    my ($n,$i,$j,$index);
    my ($first,$tmp1,$tmp2);

    $n=@$array1;

        for ($i=0;$i<($n-1);$i++){

            $first=$$array1[$i];
            $index=$i;

            for ($j=($i+1); $j<$n; $j++){
                if ($$array1[$j] < $first) {
                    $first=$$array1[$j];
                        $index=$j
                    }
            }

            if ($index != $i) {

                    $tmp1=$$array1[$i];
                    $$array1[$i]=$$array1[$index];
                    $$array1[$index]=$tmp1;

                    $tmp2=$$array2[$i];
                    $$array2[$i]=$$array2[$index];
                    $$array2[$index]=$tmp2;

                }

        }

    return $array1,$array2;
}

###############################################################################
```

Conversion of ASCII WD Data to Xspec Fits Format - lin2xspB.pl

PDD's ASCII WD data to Xspec fits WD data conversion software.

lin2xspB.pl (based on lin2xsp.pl - but a work around has been applied to compensate for an Astro CFITSIO boolean to string conversion bug)

```
#!/usr/bin/perl
###############################################################################
#
# based on opt2xsp.f and lin2xsp.f
#
# in:  ascii data file with APPROXIMATELY the following form :
#
# lambda (in angs.), flux (ergs cm-2 s-1 ang-1), fluxerror (ergs cm-2 s-1 ang-1).
#
# out: OGIP FITS pha and rsp files suitable for xspec.
#
###############################################################################
#
#   load necessary perl modules
#
    use Astro::FITS::CFITSIO qw(READWRITE TLOGICAL TSTRING TLONG TFLOAT TUSHORT TINT);
    use Carp;
    use POSIX;
#    use PGPLOT;
#
#   a few important values
#
    $a2kev=12.39854; $erg2kev=1.602177e-9; $cang=2.99793e+18;
#
    @lineid= ('1e', '1d', '1g', '1b', 'h9', 'h8',  'he',  'hd',  'hg',  'hb',  'ha',  '43',  '45',
        '46',  '48',  '54',  '65');
    @lambdac
        =('938','950','973','1026','3835.00','3889.048','3970.072','4101.734','4340.462','4861.323','6562.797','4339','451
        ;
    @lambdad=('3',  '6',  '10', '20', '20', '40',  '40',  '75',  '130', '150', '200', '50',  '70',
        '50',  '50',  '50',  '50');
    @slopes= ('0.0','0.0','0.0','0.0','0.0', '0.0', '0.0', '0.0', '0.0', '0.0', '0.0', '0.0', '0.0',
         '0.0', '0.0', '0.0', '0.0');
```

```
#
#    define question and response strings
#
     $outtext0=sprintf("\n\n                                                  ------------------------ \n");
     $outtext1=sprintf("                                              | LIN2XSP Version 1.0 | \n");
     $outtext2=sprintf("                                              ------------------------ \n ");
     $outtext3=sprintf("\n\nEnter the root name of the ascii data file                  > ");
     $outtext35=sprintf("\nIs there a flux error column in this file (y/n) ?          > ");
     $outtext36=sprintf("\nIsolate H-(l)yman, H-(b)almer or H(e)lium line series ?    > ");
     $outtext37=sprintf("\nCorrect slopes of lines (y/n) ?                           > ");
     $outtext38=sprintf("Enter 938,950,973 and 1026 angstrom slopes                 > ");
     $outtext39=sprintf("Enter 3970, 4101, 4340 and 4861 angstrom slopes            > ");
     $outtext40=sprintf("Enter 4339,4542,4680,4860,5400,6560 angstrom slopes        > ");
     $outtext4=sprintf("\nEnter instrument resolution (fwhm channels)               > ");
     $outtext5=sprintf("\nEnter name of source                                      > ");
     $outtext6=sprintf("\n\nYou have entered the following                          > \n\n");
     $outtext11=sprintf("Estimate the fractional flux error on the data             > ");
     $outtext9=sprintf("\n\nData will be output to ".$phaout." with corresponding response file ".
          $rmfout."\n");
#
     printf $outtext0.$outtext1.$outtext2;
#
     $ans='n';
#
     while (($ans eq 'n') || ($ans eq 'N')){

     printf $outtext3;
     chomp($infile=<STDIN>);

     printf $outtext35;
     chomp($dtype=<STDIN>);

     if (($dtype eq 'N') || ($dtype eq 'n')) {
          printf $outtext11;
          chomp($frace=<STDIN>);
     }

     printf $outtext36;
     chomp($lseri=<STDIN>);

     printf $outtext37;
     chomp($corrs=<STDIN>);

     if (($corrs eq 'y') || ($corrs eq 'Y')) {

     if (($lseri eq 'b') || ($lseri eq 'B')){
          printf $outtext39;
          $_=<STDIN>;
          chomp;
          ($slopes[4],$slopes[5],$slopes[6],$slopes[7],$slopes[8],$slopes[9],$slopes[10])=split " ",$_
               ;
            } elsif (($lseri eq 'l') || ($lseri eq 'L')) {

            printf $outtext38;
            $_=<STDIN>;
            chomp;
            ($slopes[0],$slopes[1],$slopes[2],$slopes[3])=split " ",$_;
            }
          else {
          printf $outtext40;
          $_=<STDIN>;
          chomp;
          ($slopes[11],$slopes[12],$slopes[13],$slopes[14],$slopes[15],$slopes[16])=split " ",$_;
          }
     }

     printf $outtext4;
     chomp($instres=<STDIN>);

     printf $outtext5;
     chomp($source=<STDIN>);

     $outtext7=sprintf("FILE     : %15s \nINST RES : %15s \nLINE SER : %15s \nSOURCE   : %15s \n\n",
          $infile,$instres,$lseri,$source);
     $outtext8=sprintf("Is this correct ? (y/n) ");

     printf $outtext6.$outtext7.$outtext8;
     chomp($ans=<STDIN>);
     }
#
     open(input,$infile.".dat") || die "Couldn't find file\n";
     $cnt=0;
#
#    read data which includes a flux error column
#
     if (($dtype eq 'y' ) || ($dtype eq 'Y'))  {

     $cnt=0;
     while(<input>){
     chomp;
     ($lambda[$cnt],$flux[$cnt],$fluxe[$cnt])=split " ",$_;
     $fluxerr[$cnt]=$fluxe[$cnt]/$flux[$cnt];
```

```
        $cnt++;
                }
        $bins=$cnt;
                } else {
#
#   read in data without a flux error column
#
        $cnt=0;

        while(<input>){
        chomp;
        ($lambda[$cnt],$flux[$cnt])=split " ",$_;
        $fluxerr[$cnt]=$frace;
        $cnt++;
    }
    $bins=$cnt;
    }

    close(input);
#
    $dchan=$bins;
#
    printf $outtext9;
#
#   now get to work.....
#
#   determine the channel energies in kev and convert flux to 'per energy' units
#
    for ($cnt=0; $cnt<$dchan; $cnt++){
     $midkev[$cnt]=$a2kev/$lambda[$cnt];
     $flux[$cnt]=$flux[$cnt]*$lambda[$cnt]/($erg2kev*$midkev[$cnt]*$midkev[$cnt]);
    }
#
#   sort arrays into order of increasing energy as required for pha and rsp files
#
   ($ar1r,$ar2r,$ar3r,$ar4r)=sortit(\@midkev,\@lambda,\@flux,\@fluxerr);
#
#   determine a normalised resolution array
#
    $sum=0;
    $rms=abs($instres)/2.36;
    $jres=int(10.0*$rms);

    $jd=2*$jres+1;
    if ($jd > 101) {
    $jd=101;
    }

    for ($rel=0; $rel<$jd; $rel++){
    $reli=$rel-$jres;
    $z=$reli/$rms;
    $resp[$rel]=exp(-0.5*$z*$z);
    if ($resp[$rel] < 1.0e-10){
        $resp[$rel]=1.0e-10;
    }
    $sum=$sum+$resp[$rel];
    }

     for ($rel=0; $rel<$jd; $rel++){
     $resp[$rel]=$resp[$rel]/$sum;
     }
#
#   set things up for a particular line series
#
    if (($lseri eq "L") || ($lseri eq "l")) {
    $offset=0;
    $nolines=4;
    } elsif (($lseri eq "B") || ($lseri eq "b")) {
    $offset=4;
    $nolines=7;
    } else {
    $offset=11;
    $nolines=6;
    }
#
#   begin looping over members of the line series
#
     for ($line=0; $line<$nolines; $line++) {

     $lno=$offset+$line;
#
#      define names of output pha and rsp files
#
     $phaout=$infile.$lineid[$lno].".pha";
         $rmfout=$infile.$lineid[$lno].".rsp";
#
#      determine line limits.....
#
     $iw=0;
     $cnt=0;
     while (($lambda[$cnt]>$lambdac[$lno]) && ($cnt<$dchan)) {
         $iw=$cnt;
```

```
            print $cnt." ".$lambda[$cnt]." ".$lambdac[$lno]."\n";
                $cnt++;
        }

        if ($iw==($dchan-1)) {
            $iw=$iw-1;
        }

#
#           this is for contiguous data
#

        $delw=$lambda[$iw]-$lambda[$iw+1];
#       $iw=int(($lambda[0]-$lambdac[$lno])/$delw);
        $imin=$iw-int($lambdad[$lno]/$delw);
        $imax=$iw+int($lambdad[$lno]/$delw);

        print $lambda[$imin]." ".$lambda[$imax]."\n";

        $lchan=$imax-$imin+1;


#
#   calculate the boundaries of the channels and convert flux to phacnts
#
        $jj=0;

       for ($cnt=$imin; $cnt<$imax; $cnt++){

       $chbnd[$jj+1]=0.5*($$ar1r[$cnt]+$$ar1r[$cnt+1]);
       $jj++;

       }

        $chbnd[0]=$chbnd[1]-($chbnd[2]-$chbnd[1]);
        $chbnd[$lchan]=$chbnd[$lchan-1]+($chbnd[$lchan-1]-$chbnd[$lchan-2]);
#
#   integrate flux over bin width and perform any slope correction
#
        $jj=0;

          for ($cnt=$imin; $cnt<=$imax; $cnt++){

          $rnrm=1.0+$slope[$lno]-2*$slopes[$lno]*$jj/($lchan-1);

       $phacnts[$jj]=$$ar3r[$cnt]*($chbnd[$jj+1]-$chbnd[$jj])*$rnrm;
       $phaerr[$jj]=$$ar4r[$cnt]*$phacnts[$jj]*$rnrm;
       $channels[$jj]=$jj+1;

       $jj++;

       }


#
#
###############################################################################################
#
# open a new fits file for the pha info and dump out relavant information
#
    my $opfptr1 = Astro::FITS::CFITSIO::create_file($phaout,$status);
    check_status($status);
#
    write_pha_info($opfptr1,$source,$lchan);
    write_pha_data($opfptr1,\@channels,\@phacnts,\@phaerr);
#
    $opfptr1->close_file($status);
#
###############################################################################################
#
# open a new fits file for the response matrix and dump out relevant information
#
    my $opfptr2 = Astro::FITS::CFITSIO::create_file($rmfout,$status);
    check_status($status);
#
#   ebound extension first
#
    write_rsp_ebounds_header($opfptr2,'LIN2XSP','OPTICAL',$lchan,$phaout);
    write_rsp_ebounds_array($opfptr2,\@channels,\@chbnd);
#
#   now for matrix extension - some more work first
#
#   extend channel bounds array to account for photons from beyond nominal energy range
#
    $delt1=$chbnd[1]-$chbnd[0];
    $delt2=$chbnd[$lchan]-$chbnd[$lchan-1];

        $enlo=$chbnd[0]-$delt1*$jres;

    for ($cnt=$lchan; $cnt>=0; $cnt--){
    $chbnd[$cnt+$jd]=$chbnd[$cnt];
    }
```

```
    for ($cnt=($jd-1); $cnt>=0; $cnt--){
    $chbnd[$cnt]=$chbnd[$cnt+1]-$delt1;
    if ($chbnd[$cnt] < 0) {

        $chbnd[$cnt]=0;
    }
    }
        for ($cnt=($lchan+$jd+1); $cnt<=($lchan+2*$jd); $cnt++){
    $chbnd[$cnt]=$chbnd[$cnt-1]+$delt2;
    }

    $bins=$lchan+$jd;
#
#   generate lo and hi energy boundaries of response matrix bins
#   and ngrp and fchan arrays - caution required with array index
#
    $energ_lo[0]=$enlo;

    for ($cnt=0; $cnt<($bins-1); $cnt++){

        $idx=$cnt+2+(($jd-1)/2);

        $energ_hi[$cnt]=$chbnd[$idx];
    if ($cnt<($bins-2)) {
        $energ_lo[$cnt+1]=$energ_hi[$cnt];
    }

    $ngrp[$cnt]=1;
    if (($cnt+1)<=$jd){
        $fchan[$cnt]=1;
    } else {
        $fchan[$cnt]=$cnt+2-$jd;
        }
    }
#
#   generate nchan array - the number of channels which contribute
#   photons to the channel we are working with.
#
        for ($cnt=0; $cnt<($bins-1); $cnt++){

        if (($cnt+1)<=$jd) {
        $nchan[$cnt]=$cnt+1;
        }
        if ((($cnt+1)>$jd) && (($cnt+1)<=($bins-$jd))){
        $nchan[$cnt]=$jd;
        }
        if (($cnt+1)>($bins-$jd)) {
        $nchan[$cnt]=$bins-($cnt+1);
        }
    }

     write_rsp_redist_header($opfptr2,$lchan,$phaout,$jd);
     write_rsp_redist_matrix($opfptr2,\@energ_lo,\@energ_hi,\@ngrp,\@fchan,\@nchan);
#
#    generate the matrix
#
     $felem=1;

     for ($cnt=0; $cnt<($bins-1); $cnt++){

          if ($cnt<($jd-1)){
          for ($iff=0; $iff<=$cnt; $iff++){
              $inresp[$iff]=$resp[$jd-$cnt+$iff-1];
          }
          $nmat=$cnt+1;
          }
          if ($cnt>=($jd-1)) {
          for ($iff=0; $iff<$jd; $iff++){
              $inresp[$iff]=$resp[$iff];
          }
          $nmat=$jd;
          }

     $frow=$cnt+1;
         $felem=1;

         $opfptr2->write_col(42,6,$frow,$felem,$nmat,\@inresp,$status);
             check_status($status);

      }
#
#
#
    $opfptr2->close_file($status);
#
#   the end.......bye.
#
     }
                stop;
#
#
```

```
#################################################################################################
#
#   various subroutines for order sorting arrays, writing fits headers and bin table extens.
#
#################################################################################################
#
#
sub sortit{
    my ($array1,$array2,$array3,$array4)=@_;
    my ($n,$i,$j,$index);
    my ($first);

    $n=@$array1;

    for ($i=0;$i<($n-1);$i++){

        $first=$$array1[$i];
            $index=$i;

        for ($j=($i+1); $j<$n; $j++){
        if ($$array1[$j] < $first) {
            $first=$$array1[$j];
            $index=$j
            }
        }

        if ($index != $i) {

            $tmp1=$$array1[$i];
            $$array1[$i]=$$array1[$index];
            $$array1[$index]=$tmp1;

            $tmp2=$$array2[$i];
            $$array2[$i]=$$array2[$index];
            $$array2[$index]=$tmp2;

            $tmp3=$$array3[$i];
            $$array3[$i]=$$array3[$index];
            $$array3[$index]=$tmp3;

            $tmp4=$$array4[$i];
            $$array4[$i]=$$array4[$index];
            $$array4[$index]=$tmp4;

        }

    }

    return $array1,$array2,$array3,$array4;
}
#
#################################################################################################
#
sub check_status {
    my $status = shift;
    if ($status) {
        my $errtxt;
      Astro::FITS::CFITSIO::fits_get_errstatus($status,$errtxt);
        croak("$0: CFITSIO error detected, aborting...$errtxt");
    }

    return 1;
}
#
#################################################################################################
#
sub write_pha_info {

    my ($fptr,$obj,$nchan)=@_;

    $colnam=['CHANNEL','RATE','STAT_ERR'];
    $form=['1I','1E','1E'];
    $unit=[' ','count/s','count/s'];
    $extn='SPECTRUM';
    $tels='N/A';
    $inst='N/A';
    $filt='N/A';
    $vers='1.1.0';
    $cls='OGIP';
    $cls1='SPECTRUM';
    $cls2='NET';
    $cls3='RATE';
    $expt=1.0;
    $area=1.0;
    $backf='NONE';
    $back=1.0;
    $corrf='NONE';
    $corr=0.0;
    $respf='%match%';
    $ancf='';
    $phaver='1992a';
    $chant='UNKNOWN';
```

```
    $pois=F;
    $sys=0;
    $min=1;
    $max=$nchan;

    $fptr->create_tbl(2,0,3,$colnam,$form,$unit,$extn,$status);

        $fptr->update_key(TSTRING,  'OBJECT',  $obj,   'name of source',            $status);
        $fptr->update_key(TSTRING,  'HDUVERS', $vers,  'file format version',       $status);
        $fptr->update_key(TSTRING,  'HDUCLASS',$cls,   'file format',               $status);
        $fptr->update_key(TSTRING,  'HDUCLAS1',$cls1,  'a pha dataset',             $status);
        $fptr->update_key(TSTRING,  'HDUCLAS2',$cls2,  'background substracted',    $status);
        $fptr->update_key(TSTRING,  'HDUCLAS3',$cls3,  'counts or count rate',      $status);
        $fptr->update_key(TSTRING,  'TELESCOP',$tels,  'telescope',                 $status);
        $fptr->update_key(TSTRING,  'INSTRUME',$inst,  'instrument',                $status);
        $fptr->update_key(TSTRING,  'FILTER',  $filt,  'filter',                    $status);
        $fptr->update_key(TFLOAT,   'EXPOSURE',$expt,  'exposure time (secs)',      $status);
        $fptr->update_key(TFLOAT,   'AREASCAL',$area,  'effective area scaling',    $status);
        $fptr->update_key(TSTRING,  'BACKFILE',$backf, 'background file',           $status);
        $fptr->update_key(TFLOAT,   'BACKSCAL',$back,  'background scaling factor', $status);
        $fptr->update_key(TSTRING,  'CORRFILE',$corrf, 'correction file',           $status);
        $fptr->update_key(TFLOAT,   'CORRSCAL',$corr,  'correction scaling factor', $status);
        $fptr->update_key(TSTRING,  'RESPFILE',$respf, 'associated response file',  $status);
        $fptr->update_key(TSTRING,  'ANCRFILE',$ancf,  'associated ancilliary file',$status);
        $fptr->update_key(TSTRING,  'PHAVERSN',$phaver,'apparently obselete',       $status);
        $fptr->update_key(TSTRING,  'CHANTYPE',$chant, 'channel type',              $status);
        $fptr->update_key(TINT,     'DETCHANS',$nchan, 'No. of detector channels',  $status);
        $fptr->update_key(TINT,     'TLMIN1',  $min,   'lowest legal channel no.',  $status);
        $fptr->update_key(TINT,     'TLMAX1',  $max,   'largest legal channel no.', $status);
        $fptr->write_key(TLOGICAL, 'POISSERR', $pois,  'poissonians errors',         $status);
        $fptr->update_key(TINT,     'SYS_ERR', $sys,   'systematic errors',          $status);
        $fptr->update_key(TSTRING,  'HDUVERS', $vers,  'file format version',       $status);

    return;

    }


sub write_pha_data {

    my ($fptr,$chan,$phac,$phace)=@_;
#
#   note that no grouping or quality information is written by this subroutine
#
    $nchan=@$chan;

    $frow=1;
    $felem=1;

    $fptr->write_col(21,1,$frow,$felem,$nchan,$chan,$status);
            check_status($status);

    $fptr->write_col(42,2,$frow,$felem,$nchan,$phac,$status);
            check_status($status);

    $fptr->write_col(42,3,$frow,$felem,$nchan,$phace,$status);
            check_status($status);

    return;
}
#
##################################################################################################
#
sub write_rsp_ebounds_header{
#
    my($fptr,$tels,$inst,$nchan,$phan)=@_;
    my($extn,$filt,$cht,$cls,$cls1,$cls2,$vers);
#
#       define a few of the keywords required in extension header
#
        $extn='EBOUNDS';
        $filt='NONE';
        $cht='PHA';
        $cls='OGIP';
        $cls1='RESPONSE';
        $cls2='EBOUNDS';
        $vers='1.1.0';
#
#       details of the columns in the EBOUND extension: names, formats, units
#
        $colnam=['CHANNEL','E_MIN','E_MAX'];
        $form=['1I','1E','1E'];
        $unit=['none','keV','keV'];
#
#       create the EBOUND binary table extension
#
        $fptr->create_tbl(2,0,3,$colnam,$form,$unit,$extn,$status);
        check_status($status);
#
#       write required keywords to the header of this extension
#
        $fptr->update_key(TSTRING,'TELESCOP',$tels,  'telescope',                $status);
```

```
        $fptr->update_key(TSTRING,'INSTRUME',$inst,   'instrument',                $status);
        $fptr->update_key(TSTRING,'FILTER',  $filt,   'filter',                    $status);
        $fptr->update_key(TSTRING,'CHANTYPE',$cht,    'channel type',             $status);
        $fptr->update_key(TINT,    'DETCHANS',$nchan,'No. of detector channels', $status);
        $fptr->update_key(TSTRING,'HDUCLASS',$cls,    'file format',              $status);
        $fptr->update_key(TSTRING,'HDUCLAS1',$cls1,   'extension contents 1',     $status);
        $fptr->update_key(TSTRING,'HDUCLAS2',$cls2,   'extension contents 2',     $status);
        $fptr->update_key(TSTRING,'HDUVERS', $vers,   'file format version',      $status);
        $fptr->update_key(TSTRING,'PHAFILE', $phan,   'pha file name',            $status);
#
#       finished initialising EBOUNDS extension. return to main program
#
    return;
}
#
###############################################################################################################
#
sub write_rsp_ebounds_array{

    my ($fptr,$chan,$ebound)=@_;
#
#   now have to resolve ebounds into low and high columns
#
    $nchan=@$chan;

    $frow=1;
    $felem=1;

    for ($ct=1; $ct<=$nchan; $ct++){
    $eblo[$ct-1]=$$ebound[$ct-1];
    $ebhi[$ct-1]=$$ebound[$ct];
    }

    $fptr->write_col(21,1,$frow,$felem,$nchan,$chan,$status);
            check_status($status);

    $fptr->write_col(42,2,$frow,$felem,$nchan,(\@eblo),$status);
            check_status($status);

    $fptr->write_col(42,3,$frow,$felem,$nchan,(\@ebhi),$status);
            check_status($status);

    return;

}
#

sub write_rsp_redist_header {

    my ($fptr,$nchan,$phaf,$rmfd)=@_;

    $colnam=['ENERG_LO','ENERG_HI','N_GRP','F_CHAN','N_CHAN','MATRIX'];
    $form=['1E','1E','1I','1I','1I','PE('.$rmfd.')'];
    $unit=['keV','keV','','','',''];
    $extn='SPECRESP MATRIX';
    $tels='N/A';
    $inst='N/A';
    $filt='N/A';
    $chant='UNKNOWN';
    $vers='1.1.0';
    $cls='OGIP';
    $cls1='RESPONSE';
    $cls2='RSP_MATRIX';
    $cls3='FULL';
    $loth=0.0;

    $fptr->create_tbl(2,0,6,$colnam,$form,$unit,$extn,$status);

        $fptr->update_key(TSTRING,  'OBJECT',  $obj,   'name of source',          $status);
        $fptr->update_key(TSTRING,  'HDUVERS', $vers,  'file format version',     $status);
        $fptr->update_key(TSTRING,  'HDUCLASS',$cls,   'file format',             $status);
        $fptr->update_key(TSTRING,  'HDUCLAS1',$cls1,  'a response file',         $status);
        $fptr->update_key(TSTRING,  'HDUCLAS2',$cls2,  'in matrix form',          $status);
        $fptr->update_key(TSTRING,  'HDUCLAS3',$cls3,  'taken into account',      $status);
        $fptr->update_key(TSTRING,  'TELESCOP',$tels,  'telescope',               $status);
        $fptr->update_key(TSTRING,  'INSTRUME',$inst,  'instrument',              $status);
        $fptr->update_key(TSTRING,  'FILTER',  $filt,  'filter',                  $status);
        $fptr->update_key(TSTRING,  'PHAFILE', $phaf,  'associated pha file',     $status);
        $fptr->update_key(TSTRING,  'CHANTYPE',$chant, 'channel type',            $status);
        $fptr->update_key(TINT,     'DETCHANS',$nchan, 'No. of detector channels',$status);
        $fptr->update_key(TINT,     'TLMIN1',  $min,   'lowest legal channel no.',$status);

    return;
}
#
#
#
sub write_rsp_redist_matrix {

    my ($fptr,$elo,$ehi,$grpn,$chanf,$chann)=@_;

    $nchan=@$elo;
```

```
    $frow=1;
    $felem=1;

    $fptr->write_col(42,1,$frow,$felem,$nchan,$elo,$status);
            check_status($status);

    $fptr->write_col(42,2,$frow,$felem,$nchan,$ehi,$status);
            check_status($status);

    $fptr->write_col(21,3,$frow,$felem,$nchan,$grpn,$status);
            check_status($status);

    $fptr->write_col(21,4,$frow,$felem,$nchan,$chanf,$status);
            check_status($status);

     $fptr->write_col(21,5,$frow,$felem,$nchan,$chann,$status);
            check_status($status);

    return;
}

#
#
##############################################################################################
```

Note for reversion back to version A in the event the bug in Astro CFITSIO is corrected sometime in the future;

diff lin2xspB.pl lin2xsp.pl gives;

'515c515'

'¡ $fptr-¿write_key(TLOGICAL, 'POISSERR', $pois, 'poissonians errors', $status);

'__'

'¿ $fptr-¿update_key(TLOGICAL, 'POISSERR',$pois, 'poissonian errors', $status);'

lin2xspB.pl prerequisities

CFITSIO Astro-FITS-CFITSIO-1.05.tar.gz

Astro-FITS-CFITSIO Astro-FITS-CFITSIO-1.05.tar.gz

Pgplot

Pgplot is difficult to compile and install on a modern Linux System, so it is recommended that a precompiled version of pgplot from starlink is used (precompiled and tested for one's system).

A.5.10 PDD IDL ASCII Model Interpolation Software Prerequisites - NOT USED

IDL Astro Lib

astroLib - astron.zip

A.5.11 Model Grids

Koester Low-Mid Temperature WD Spectoscopic Model Grid (T=6000K - 20000K)

kmod.tar.gz (ASCII)

KOESTERDA (Xspec Fits)

Intervals logg interval = 0.25
Teff interval = 1000K

File Name Format ""da" + teffString + "_" + STRMID(loggString,0,1) + STRMID(loggString,2,2) + ".dk_tr_rg"'
(Eg da16000_775.dk_tr_rg)

Format Wavelength Flux

PDD Mid-High Temperature WD Spectoscopic Model Grid (T=15000K - 47500K)

tlmod.tar.gz (ASCII)
TLUSTY10v2 (Xspec Fits)

Intervals logg interval = 0.25
Teff interval = 2500K

File Name Format tlustyModelFileName = "ngc3532-lls" + STRMID(teffString,0,3) + STRMID(loggString,0,1) + STRMID(loggString,2,2) + ".dat"
(Eg ngc3532-lls160775.dat)

Format Wavelength Flux

A.5.12 Xspec Model Files

xspecWorkingFolder-29July11a.zip

Flux Calibrated Data Files

WDname.dat (Eg SSWD01A.dat)

Xspec workspace files

WDname.xcm (Eg SSWD01A.xcm)

```
cpd /xw
data 1:1 "SSWD01Bwhb.pha"
data 2:2 "SSWD01Bwhg.pha"
data 3:3 "SSWD01Bwhd.pha"
data 4:4 "SSWD01Bwhe.pha"
data 5:5 "SSWD01Bwh8.pha"
 statistic chi
 abund angr
 xsect bcmc
 xset forcecalc off
 cosmo    70.000    0.000    0.730
model atable{/home/rich/tasks/atmosphericmodelfitting/working/modelgrid/KOESTERDA}
   7.9689       7.00000E-03   7.0000        7.0000        9.5000        9.5000
```

```
   19217.        6.0000        6000.0        6000.0        20000.        20000.
  6.44005E-05  1.00000E-06 -1.00000E-03 -1.00000E-03  1.00000E-03  1.00000E-03
   320.75        5.0000        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   306.38        5.0000        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   293.99        5.0000        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   287.75        5.0000        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   282.88        5.0000        0.0000        0.0000       1.00000E+24  1.00000E+24
fit
ipl
```

```
cpd /xw
data 1:1 "SSWD01Bwhb.pha"
data 2:2 "SSWD01Bwhg.pha"
data 3:3 "SSWD01Bwhd.pha"
data 4:4 "SSWD01Bwhe.pha"
data 5:5 "SSWD01Bwh8.pha"
 statistic chi
 abund angr
 xsect bcmc
 xset forcecalc off
 cosmo    70.000    0.000    0.730
model atable{/home/rich/tasks/atmosphericmodelfitting/working/modelgrid/TLUSTY10v2}
   19217.        6.0000        15000.        15000.        45000.        45000.
   7.9689       7.00000E-03    7.2500        7.2500        8.7500        8.7500
  6.44005E-05  1.00000E-06 -1.00000E-03 -1.00000E-03  1.00000E-03  1.00000E-03
   320.75        1.0E+8        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   306.38        1.0E+8        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   293.99        1.0E+8        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   287.75        1.0E+8        0.0000        0.0000       1.00000E+24  1.00000E+24
 =    1
 =    2
 =    3
   282.88        1.0E+8        0.0000        0.0000       1.00000E+24  1.00000E+24
```

'Fits Model Files generated using lin2xspec.pl'

- WDnameha.pha (Eg SSWD01Aha.pha)
- WDnameha.rsp
- WDnamehb.pha
- WDnamehb.rsp
- WDnamehg.pha
- WDnamehg.rsp
- WDnamehd.pha
- WDnamehd.rsp
- WDnamehe.pha
- WDnamehe.rsp
- WDnameh8.pha

- WDnameh8.rsp
- WDnameh9.pha
- WDnameh9.rsp

A.5.13 Operational Manuals

All data processing procedures have been documented in operational manuals.

Spectroscopic Modelling Instructions

```
//do: WD model fitting software installation and usage instructions;

1. 1 WD Model Grid Creation utils [PDD generates models]

    A. synspec43
    cannot compile on linux

    B. tlusty200
    cd tlusty200
    g77 -O3 -static -fno-automatic -o tlusty200 tlusty200.f

2. astrolib
    install astrofits idl/gdl library
        - if idl on server, copy pro files to working directory? use gftp.
        - if gdl, copy pro files to /usr/share/gdl/

3. IDL normalisation using model grid - IDL Software [IDL preliminary T and log g model grid fitting
     process: Normalises Mel111-WD.dat data using model grid producing spectralfile0/Mel111-WD.
    dat_m/norm (first two columns of this data may be used by lin2xsp.pl/xspec), and calculates
    basic estimate of g and Teff using the model grid]
    1. place MODELS folder somewhere, nedit loadmodarr0.pro,
            (eg in IDL model grid working folder, somewhere = /home/rich/tasks/
                atmosphericmodelfitting/utils/1modelgridgenerationUsingIDL/home/pdd/IDL-WD/)
        and change
            modfilename='/MODELS/'+resstring+'/'+tstring+'/'+gstring+'/'+'H090717.dat'
        to;
                modfilename='somewhere/MODELS/'+resstring+'/'+tstring+'/'+gstring+'/'+'H090717.dat'
    2. nedit in_0.FRC and change dat file [eg Mel111-WD.dat]
        filename must be <= 8 characters.
       nedit in_1.dat, and set min/max temp and logg parameters
       nedit loadmodarr0.pro
       set the following;
        ntmax= number of temperature folders in model grid LRES folder
        nlmax= number of logg folders in model grid LRES/temp folder
        teffs=[all temp folder names in model grid LRES folder]
        logg=[all logg folder names in model grid LRES/temp folder]
        modfilename='/disks/etna/home/science/staff/rbaxter/modelgrid/
             koestermodelgridORtlustymodelgridOR/'+resstring+'/'+tstring+'/'+gstring+'/'+'TLUSTYDA.
             dat'
    3. use idl [NOT gdl; too slow]
        or to use IDL;
        ssh rbaxter@137.111.88.18 -X
        cd idl
        idl

    {/home/rich/tasks/atmosphericmodelfitting/}

    .run loadmodarr0.pro
    .run read_in
    read_in [to perform model grid fitting using idl, and to produce a continuum normalised output
         file with extension .dat_m]

    .run inter.pro
    inter,0 [to calc values without producing a poly/continuum normalised output file with extension
         n.dat, for a single redshift value]
    filename
    1
    5
    poly

4. cfitsio [install CFITSIO]
    su rich
    cd utils/cfitsio          [v3.x]
    ./configure --prefix=/usr/local
    make
    su root
    make install
```

```
5. Astro-FITS-CFITSIO-1.05 [Perl Modules - Astro:Fits:CFITSIO (and required or fortan I, libraries)]
    su rich
    cd utils/Astro-FITS-CFITSIO-1.05     [v1.05]
    nedit Makefile.PL, and set -I/usr/local/include and -L/usr/local/lib -lcfitsio -lm
    perl Makefile.PL OPTIMIZE=-O
    make
    make test
    su root
    make install

6. lin2xsp [lin2xsp.pl - what does lin2xsp do? generates pha/phm files used by xspec (eg referenced
    in COMA-WD.xcm)]
        [must use lin2xsp.pl version B {with hack/work around} when using buggy cfitsio v1.05]
    cuts the lines out and creates separate files.
        [rsp = response files - uses rsp files to blur the model to match the resolution of the data
            ]

    lin2xsp;
        Enter the root name of the ascii data file                  > COMA-WDn
        Is there a flux error column in this file (y/n) ?           > n
        Estimate the fractional flux error on the data              > 0.02 {signal to noise 50:1 [so
            insert 0.02]}
        Isolate H-(l)yman, H-(b)almer or H(e)lium line series ?     > b
        Correct slopes of lines (y/n) ?                             > n
        Enter instrument resolution (fwhm channels)                 > 2.5        {fwhm in channels?
            deltaLambda/RBBfitsfileresl[1.0A] = }

            Coma WD:
            [NB COMA =  0.6arc with R300B grating, most likely has been binned every 1.0 arc seconds
                - use WHT blue arm website - PDD had matched the slit width and the seeing on the
                night - ; as 1" will project to 4pixels, therefore 0.6" will project to 2.4 pixels,
                 then * 0.86 gives ~2.064 fwhm]
            divide spec res by image res, so if image res is 1.7 [should be 1.0 with myGemini/myWHT
                data], then enter into Lw2xsp 2.0/1.7 = 2.5
            instrument resolution = spec res / image res

            http://en.wikipedia.org/wiki/Spectral_resolution
            spectral resolution (R) = lambda[4400] / delta lambda
            delta lambda = lambda[4400] / spectral resolution[R]


            Gemini GMOS:
            http://www.gemini.edu/?q=node/10375

        Enter name of source                                        > coma   {??????}


7. pgplot
        pgplot 5.x insallation instructions;
            install starlink [http://starlink.jach.hawaii.edu/starlink]
                [create a link in /usr/bin to /home/rich/star/manifests/pgplot]
            su root
            ln -s /home/rich/star/manifests/pgplot /usr/bin/pgplot

8. XSPEC (HEASARC) version 11.3 [what does xspec do? Performs formal T and log g model grid fitting.
     Creates a file similar to hyad.in for fontaine?]

    what xpsec does? reads in model grid [dont need to smooth the model]

        instructions to install XSPEC 11;
            download xspecv11.3 source, xspec11.3.2ag-src.tar   [MB I had to obtain this in person
                from Bryan Irby (irby@milkyway.gsfc.nasa.gov) as only v12 is available online at
                the moment]
            cd xspecv11.3/heasoft-6.6.3/BUILD_DIR
            ./configure
            make
            su root
            make install

        instructions to run XSPEC 11;
            https://astrophysics.gsfc.nasa.gov/XSPECwiki/XSPECPage
            http://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/index.html
                http://heasarc.gsfc.nasa.gov/docs/xanadu/index.html#quicklink
            http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/install.html

            su rich
            cd /home/rich/tasks/atmosphericmodelfitting/working/xspec
            HEADAS=/home/rich/tasks/atmosphericmodelfitting/utils/xspecv11.3/heasoft-6.6.3/x86_64-
                unknown-linux-gnu-libc2.5
            export HEADAS
            alias heainit=". $HEADAS/headas-init.sh"
            heainit
            xspec11

                    manual operations (without using pre-existing xcm file);

                        configure plot device;

                            set plotting device to pgplot (http://heasarc.gsfc.nasa.gov/docs/xanadu/
                                xspec/manual/XScpd.html);
```

```
                        A?
                        cpd /GIF
                        exit

                        B?
                        [pl]
                        ipl
                        dev /xw

                    configure balmer line models

                        da 1:1 W2T-WAV-FLX-Ahb 2:2 W2T-WAV-FLX-Ahg 3:3 W2T-WAV-FLX-Ahd 4:4 W2T-
                            WAV-FLX-Ahe 5:5 W2T-WAV-FLX-Ah8

                what it does;
                        findsthe optimum fit
                        test it out on a dataset already analysed
                            PDD supply a anormalised COMA-WDn.dat file

        modify SSWD08Bv.xcm with appropriate file names (including model grid)
            model atable{/home/rich/tasks/atmosphericmodelfitting/modelgrid/TLUSTY10}
            OR
            model atable{/home/rich/tasks/atmosphericmodelfitting/modelgrid/BAXTERDA}
        xspec11

        EITHER; (if xcm file exists);

            @SSWD08Bv.xcm        [or @COMA-WD.xcm]
            thaw 3  {I don't know why this is required, but for some reason the redshift always
                happens to be frozen upon load of the xcm file}
            ipl [only required if initial estimate not good enought and xcm fit reference
                triggered interactive fit autoenabled]
            fit [only required if initial estimate not good enought and xcm fit reference
                triggered interactive fit autoenabled]
            ign 5:1-** [only required if balmer line 8 gives bad fit]
            error 3.5 stopat 200000,,1,2

        OR; (if xcm file does not exist);

            data 1:1 "SSWD08Bvhb.pha" 2:2 "SSWD08Bvhg.pha" 3:3 "SSWD08Bvhd.pha" 4:4 "SSWD08Bvhe.
                pha" 5:5 "SSWD08Bvh8.pha"
            fit
            ipl
            thaw 3 [unfreeze redshift]
            new 3
                1e-4 1e-6 -1e-3 -1e-3 1e-3 1e-3
                1e-6 1e-6 -1e-3 -1e-3 1e-3 1e-3
            untie 4 8 12 16 20
            ign 5:1-** [only required if balmer line 8 gives bad fit]
            fit
            error 3.5 stopat 200000,,1,2
                [reduced ch sq <= 2.0]
                [delta chi sq ~= 3.5]
            exit
            save all SSWD08Bv


        reduce chi squared to ~3.5;
        if >>1 , then under estimate error, in lin2xspec increase the fraction
        if <<1 , then over estimate error , in lin2xspec reduce the fraction

        As for the fractional error....you will likely need to make a couple of
        iterations of the lin2xsp.pl -> fitting process to tune this to give a
        reduced chi^2 of ~1.0

                                ask Quentin about it g - XSPEC


9. fontaine [Matt Wood Software; calculates WD cooling times and masses]

    http://astro.fit.edu/wood/wd.html
    [in folder called "Fontaine" - feb 92; bergeron tracks differ from MW tracks; very hottest/
        coolest[crystalise] end differences]
    rpm -i compat-gcc-34-g77-3.4.6-4.x86_64.rpm
    g77 -c *.f       [need to recompile *.f into .o objectfiles]
    g77 g2m_fbbC.o hunt.o interpralf.o loadvec.o readt1C.o spline.o splt2.o interp.o splint.o splt.o
                [only C related]     [link them all into an executable]
    g77 g2m_fbbCO.o hunt.o interpralf.o loadvec.o readt1CO.o spline.o splt2.o interp.o splint.o splt
        .o  [only CO related]    [link them all into an executable]

    yum install libg2c.so.0
    ./g2m_fbbCO
        Note the following files;
            sdssdd.in    [created based upon output of XSPEC]     {A or B specify}
                name     irr irr opttemp low high     gopt     glow     ghigh    flag[1,0]-
                    gravitationalredshiftinfo?
            sdssdd.out
                GR - Km/s

10. bergeronPDDcode [calculates absolute magnitudes using WD atmospheres - in g/v band? [need to
    modify this such that it outputs in r, u, g/v bands also]
```

```
await code from PD (Bergeron)

bergeron code (pda_wd_Vmag.f, fort.55, tefflogg.dat);

g77 -fno-second-underscore -c pda_wd_Vmag.f
g77 pda_wd_Vmag.o /home/rich/star/lib/libpda.a -o pda_wd_Vmag

ie. you'll need starlink to link with...
fort.55 is the bergeron input data,
tefflogg.dat is your temps and gravs...
answers output to fort.21
```

A.6 Data Analysis

A.6.1 Calculate Main Sequence Ages

This software is used to calculate the main sequence age of a given main sequence mass (or vice versa). It is also used to plot system age confirmation bar graphs (across a range of published IFMRs). It is also used to perform a statistical IFMR derivation (m/gradient and c/y-intersection parameters of a linear IFMR form) based upon system age difference optimisation (minimisation). It is also used to plot the initial-final mass of components of each system against a known IFMR form.

findMainSequenceCoolingTimeOrMassInGirardiModel.pro

mainSequenceStellarEvolutionModelCalculations-01Aug11a.zip

```
; Title: find Main Sequence Cooling Time Or Mass In Girardi Model
; File: findMainSequenceCoolingTimeOrMassInGirardiModel.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 11 August 10 6:00pm (Updated 31 July 11)
; Summary:
;
;   Postconditions;
;
;   This program uses Girardi 2000 main sequence evolution data to either a) find mass based upon
    age or b) find age based upon mass
;
;   Preconditions;
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   mainSequenceStellarAges-SolarMetallicity.txt [recno, Overshoot, Z yr, age solMass, Mini solMass,
      Mact [solLum], logL, logTeff [K], logg [cm/s2], VMAG mag, U-B mag, B-V mag, V-I mag, Stage],
    tab delimited
;   1. initialMassInput.txt [value], 2. initialAgeInput.txt [value], or 3. finalAgeInput.txt,
    finalMassInput.txt [value, valueMin, valueMax] {NB 3:0 requires initialMassInput.txt,
    initialAgeInput.txt [value, valueMin, valueMax] also}
;   for usage 3:0. only: systemAgeConstraints.txt [known IFMR estimated System Age, known IFMR
    estimated max System Age, known IFMR estimated min System Age], tab delimited (in Myrs)
;   for usage 3:1 and 3:2 only: systemAgeCommon.txt [whether to use the DD system for final
    calculations]
;
;   Only modes still used [all other modes have been replaced with PDD/
    RBBcompareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS and SDSSDDsystemAgeConfirmation
     >= 31 July 11a];
;   3:0 and 3:1 [ie 1, 2, and 3:2 are no longer used]
;
;   Usage GDL [linear interpolation only];
;
;   gdl
;   .run findMainSequenceCoolingTimeOrMassInGirardiModel.pro
```

```
;
;    Usage IDL:
;    mkdir /home/aaossz/rbaxter/idllib/astron
;    cd /home/aaossz/rbaxter/idllib/astron
;    tar -xvf astron.tar.gz
;    idl
;    !PATH = Expand_Path('+/home/aaossz/rbaxter/idllib/astron/pro') + ':' + !PATH
;    PRINT, !PATH
;    .run findMainSequenceCoolingTimeOrMassInGirardiModel.pro
;

pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
                l_blue    : 12,$
                l_red     : 13 $
               }
;         0  1    2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]    ;blue
end


my_colors



;********************************************** START Parameters
     **********************************************

useGDL = 0
searchTypeSelectedBasic = 3      ;1 = find age based upon mass, 2 = find mass based upon initial age,
     3 = see searchTypeSelectedIFMR below for operational modes
plotToFile = 1
maxStarIndex = 28

;PRINT, "systemAgeEstimate = ", systemAgeEstimate

if((searchTypeSelectedBasic eq 1) or (searchTypeSelectedBasic eq 2)) then begin

    PRINT, "illegal mode detected: use PDD/
         RBBcompareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS >= 31July11a instead"
    exit

    numberofextremes = 3     ;norm, min, max

    ;IDL only
    numberOfCombinations = 6
    stageSelectedArray = STRARR(numberOfCombinations)
    interpolationTypeArray = STRARR(numberOfCombinations)
    stageSelectedArray[0] = "AGBt"
    stageSelectedArray[1] = "AGBt"
    stageSelectedArray[2] = "EHeb"
    stageSelectedArray[3] = "EHeb"
    stageSelectedArray[4] = "TO"
    stageSelectedArray[5] = "TO"
    interpolationTypeArray[0] = "linear"
    interpolationTypeArray[1] = "spline"
    interpolationTypeArray[2] = "linear"
    interpolationTypeArray[3] = "spline"
    interpolationTypeArray[4] = "linear"
    interpolationTypeArray[5] = "spline"
    combinationResultsArray = DBLARR(numberOfCombinations, numberofextremes, maxStarIndex)

endif else if(searchTypeSelectedBasic eq 3) then begin

    numberofextremes = 3     ;norm, min, max

    numberOfCombinations = 1
    stageSelectedArray = STRARR(numberOfCombinations)
    interpolationTypeArray = STRARR(numberOfCombinations)
    stageSelectedArray[0] = "AGBt" ;EHeb    ;TO
    if(useGDL eq 1) then begin
           interpolationTypeArray[0] = "linear"
           charSizeDefault = 1.0
    endif else begin
           interpolationTypeArray[0] = "spline"
           charSizeDefault = 1.0
```

```
endelse
combinationResultsArray = DBLARR(numberOfCombinations, numberofextremes, maxStarIndex)

minIFMRmInitial = 0
maxIFMRmInitial = 8
minIFMRmFinal = 0
maxIFMRmFinal = 2

useRangeOfSystemAges = 1
WDxResultsArrayNumberOfResults = 5
if(searchTypeSelectedBasic eq 3) then begin

    ;Dobbie 06;
    IFMRm = 0.133
    IFMRc = 0.289
    IFMRmErr = 0.015
    IFMRcErr = 0.051

    ;Casewell 09;
    ;IFMRm = 0.1197
    ;IFMRc = 0.3569
    ;IFMRmErr = 0.0074
    ;IFMRcErr = 0.0220

    IFMRMInitialMax = 8.0
    IFMRMInitialMin = 0.0

    searchTypeSelectedIFMR = 1 ;0 = Plot IFMR; 1 = system age optimisation of IFMR, 2 = system
        age comparison bar graph (for different published IFMRs)
    if(searchTypeSelectedIFMR eq 0) then begin
        searchTypeSelectedIFMRPlot = 2  ;0 = Mi:Mi plot, 1=IFMR plot using theoretical IFMR, 2=
            IFMR plot using system age of alternate component
        useSharedEstimatedSystemAgeOfSystem = 1
        systemAgeConstraintsFileName = "systemAgeConstraints.txt"
        PRINT, "systemAgeConstraintsFileName = ", systemAgeConstraintsFileName
        readcol64bitvars, systemAgeConstraintsFileName, systemAgeEstimateNorm,
            systemAgeEstimateMin, systemAgeEstimateMax, FORMAT='D,D,D', SKIPLINE=1
        systemAgeEstimateNorm = systemAgeEstimateNorm*1e6
        systemAgeEstimateMin = systemAgeEstimateMin*1e6
        systemAgeEstimateMax = systemAgeEstimateMax*1e6
        systemAgeEstimate = [transpose(systemAgeEstimateNorm), transpose(systemAgeEstimateMin),
            transpose(systemAgeEstimateMax)]
        systemAgeIncrement = double(10000000.0) ;million years
        maxNumberOfSystemAgesIncrements = 1000000
        systemAgeRangeResultsArray = DBLARR(maxStarIndex, numberofextremes,
            maxNumberOfSystemAgesIncrements, WDxResultsArrayNumberOfResults)          ;massFinal
              ageFinal    systemAge    massInitial ageInitial
        systemAgeRangeSizeArray = LONARR(maxStarIndex)
    endif else if(searchTypeSelectedIFMR eq 1) then begin
        overlaySystemAgeIndiciesForDebug = 0
        systemAgeOptimisationFast = 0    ;default:0
        if(systemAgeOptimisationFast eq 1) then begin
            systemAgeOptimisationFastTest = 1
        endif
        IFMRParameterOptimisationMinM = 0.05
        IFMRParameterOptimisationMaxM = 0.55
        ;IFMRParameterOptimisationMinC = IFMRc-(0.004)   ;TEMPORARY
        ;IFMRParameterOptimisationMaxC = IFMRc+(0.004)   ;TEMPORARY
        IFMRParameterOptimisationMinC = 0.0 ;0.0      ;0.0      ;NORMAL
        IFMRParameterOptimisationMaxC = 1.0 ;1.0      ;0.5      ;NORMAL
        IFMRParameterOptimisationIncrementM = 0.002 ;0.002   ;NORMAL ;0.005   ;TEMPORARY
        IFMRParameterOptimisationIncrementC = 0.002 ;0.002   ;NORMAL ;0.005   ;TEMPORARY
            ;IFMRoptimisationOverkill = 3.0
            ;IFMRParameterOptimisationMinM = IFMRm-(IFMRmErr*IFMRoptimisationOverkill)
            ;IFMRParameterOptimisationMaxM = IFMRm+(IFMRmErr*IFMRoptimisationOverkill)
            ;IFMRParameterOptimisationMinC = IFMRc-(IFMRcErr*IFMRoptimisationOverkill)
            ;IFMRParameterOptimisationMaxC = IFMRc+(IFMRcErr*IFMRoptimisationOverkill)
            ;IFMRParameterOptimisationIncrementM = 0.001
            ;IFMRParameterOptimisationIncrementC = 0.001
        numberOfMIndicies = long((IFMRParameterOptimisationMaxM-IFMRParameterOptimisationMinM)/
            IFMRParameterOptimisationIncrementM)
        numberOfCIndicies = long((IFMRParameterOptimisationMaxC-IFMRParameterOptimisationMinC)/
            IFMRParameterOptimisationIncrementC)
        PRINT, "numberOfMIndicies =", numberOfMIndicies
        PRINT, "numberOfCIndicies =", numberOfCIndicies
        ;PRINT, "IFMRParameterOptimisationMinM = ", IFMRParameterOptimisationMinM
        ;PRINT, "IFMRParameterOptimisationMaxM = ", IFMRParameterOptimisationMaxM
        ;PRINT, "IFMRParameterOptimisationMinC = ", IFMRParameterOptimisationMinC
        ;PRINT, "IFMRParameterOptimisationMaxC = ", IFMRParameterOptimisationMaxC
        ;exit

        ;PRINT, "maxStarIndex =", maxStarIndex
        ;PRINT, "WDxResultsArrayNumberOfResults =", WDxResultsArrayNumberOfResults
        systemAgeRangeArrayForIFMRParameters = DBLARR(maxStarIndex, numberofextremes,
            numberOfMIndicies, numberOfCIndicies, WDxResultsArrayNumberOfResults)
        ;PRINT, "he2"

        numberOfIFMRparameters = 2
        idealIFMRpropertiesForStarArray = IntArr(maxStarIndex, numberofextremes,
            numberOfIFMRparameters)
        idealSystemAgeForStarArray = DBLARR(maxStarIndex, numberofextremes) ;record system age
```

```
            systemAgeCommonFileName = "systemAgeCommon.txt"
            PRINT, "systemAgeCommonFileName = ", systemAgeCommonFileName
            readcol64bitvars, systemAgeCommonFileName, systemAgeCommonArray, FORMAT='I', SKIPLINE=1

            onlyUseSystemsWithCommonSystemAgeDuringIFMRparameterOptimisation = 1
        endif else if(searchTypeSelectedIFMR eq 2) then begin
            PRINT, "illegal mode detected - use SDSSDDsystemAgeConfirmation >= 31July11a instead"
            exit

            useErrorsInPublishedIFMRdefinitions = 1
            systemAgePlotType = 1    ;0: plot linear, 1: plot log

            numberOfIFMRparameters = 2
            numberOfPublshedIFMRs = 5
            publshedIFMRparamterArray = DBLARR(numberOfPublshedIFMRs, numberOfIFMRparameters)
            publshedIFMRparamterArrayErr = DBLARR(numberOfPublshedIFMRs, numberOfIFMRparameters)

            systemAgeRangeArrayForPublishedIFMRs = DBLARR(maxStarIndex, numberofextremes,
                numberOfPublshedIFMRs, WDxResultsArrayNumberOfResults)

            publshedIFMRparamterArray(0, 0) = 0.133      ;dobbie_new_2006
            publshedIFMRparamterArray(0, 1) = 0.289      ;dobbie_new_2006
            publshedIFMRparamterArray(1, 0) = 0.109      ;kalirai_initial-final_2008
            publshedIFMRparamterArray(1, 1) = 0.394      ;kalirai_initial-final_2008
            publshedIFMRparamterArray(2, 0) = 0.084      ;salaris_semi-empirical_2009
            publshedIFMRparamterArray(2, 1) = 0.466      ;salaris_semi-empirical_2009
            publshedIFMRparamterArray(3, 0) = 0.1197     ;casewell_high-resolution_2009
            publshedIFMRparamterArray(3, 1) = 0.3569     ;casewell_high-resolution_2009
            publshedIFMRparamterArray(4, 0) = 0.129      ;williams_probinglower_2009
            publshedIFMRparamterArray(4, 1) = 0.339      ;williams_probinglower_2009

            publshedIFMRparamterArrayErr(0, 0) = 0.015
            publshedIFMRparamterArrayErr(0, 1) = 0.051
            publshedIFMRparamterArrayErr(1, 0) = 0.007
            publshedIFMRparamterArrayErr(1, 1) = 0.025
            publshedIFMRparamterArrayErr(2, 0) = 0.015   ;unspecified
            publshedIFMRparamterArrayErr(2, 1) = 0.051   ;unspecified
            publshedIFMRparamterArrayErr(3, 0) = 0.0074
            publshedIFMRparamterArrayErr(3, 1) = 0.0220
            publshedIFMRparamterArrayErr(4, 0) = 0.004
            publshedIFMRparamterArrayErr(4, 1) = 0.015

            systemAgeCommonFileName = "systemAgeCommon.txt"
            PRINT, "systemAgeCommonFileName = ", systemAgeCommonFileName
            readcol64bitvars, systemAgeCommonFileName, systemAgeCommonArray, FORMAT='I', SKIPLINE=1
        endif else begin
            PRINT, "searchTypeSelectedIFMR = invalid"
            exit
        endelse
    endif
endif else begin
    PRINT, "searchTypeSelectedBasic = invalid"
    exit
endelse

;********************************************** END Parameters
     **********************************************


tempXArray = dblarr(2)
tempYArray = dblarr(2)
tempXArray = [-100, -100]
tempYArray = [-100, -100]


for combinationIndex=0, numberOfCombinations-1, 1 do begin

    ;input parameters;
    stageSelected = stageSelectedArray[combinationIndex]    ;use end of horizontal branch   ;TO,
        RGBb, L-M, L-m, RGBt, BHeb, Te-m, Te-M, EHeb, L-M, L-m, 1TP, AGBt

    mainSequenceModelFileName = "mainSequenceStellarAges-SolarMetallicity.txt"
    starValuesFileNameInitialMass = "initialMassInput.txt"  ;for searchTypeSelectedBasic 1 only
    starValuesFileNameInitialAge = "initialAgeInput.txt"    ;for searchTypeSelectedBasic 2 only
    starValuesFileNameFinalAge = "finalAgeInput.txt"    ;for searchTypeSelectedBasic 3 only
    starValuesFileNameFinalMass = "finalMassInput.txt"  ;for searchTypeSelectedBasic 3 only - for
        display/graph purposes only

    PRINT, "mainSequenceModelFileName = ", mainSequenceModelFileName
    readcol64bitvars, mainSequenceModelFileName, recno, Overshoot, Zyr, age, Mini, Mact, logL,
        logTeff, logg, VMAGmag, UBmag, BVmag, VImag, Stage, FORMAT='D,D,D,D,D,D,D,D,D,D,D,D,A',
        SKIPLINE=3

    modelArraySize = SIZE(recno)
    numberOfStages = 10

    modelGridSize = modelArraySize[1]
    PRINT, "modelGridSize = ", modelGridSize

    if (searchTypeSelectedBasic eq 1) then begin

        PRINT, "starValuesFileNameInitialMass = ", starValuesFileNameInitialMass
```

```
    readcol64bitvars, starValuesFileNameInitialMass, initialMassNormArray, initialMassMinArray,
        initialMassMaxArray, FORMAT='D,D,D', SKIPLINE=1
    initialMassArray = [transpose(initialMassNormArray), transpose(initialMassMinArray),
        transpose(initialMassMaxArray)]
    ;readcol64bitvars, starValuesFileNameInitialMass, initialMassNormArray, FORMAT='D', SKIPLINE
        =1
    ;initialMassArray = transpose(initialMassNormArray)

    PRINT, "searchTypeSelectedBasic = find initial age based upon initial/MS mass"
    PRINT, "isochrone stageSelected = ", stageSelected

    starInputValuesArraySize = SIZE(initialMassNormArray)
    numberOfStars = starInputValuesArraySize[1]

endif else if(searchTypeSelectedBasic eq 2) then begin

    PRINT, "starValuesFileNameInitialAge = ", starValuesFileNameInitialAge
    readcol64bitvars, starValuesFileNameInitialAge, initialAgeNormArray, initialAgeMinArray,
        initialAgeMaxArray, FORMAT='D,D,D', SKIPLINE=1
    initialAgeNormArray=initialAgeNormArray*1e6
    initialAgeMinArray=initialAgeMinArray*1e6
    initialAgeMaxArray=initialAgeMaxArray*1e6
    initialAgeArray = [transpose(initialAgeNormArray), transpose(initialAgeMinArray), transpose(
        initialAgeMaxArray)]
    ;readcol64bitvars, starValuesFileNameInitialAge, initialAgeNormArray, FORMAT='D', SKIPLINE=1
    ;initialAgeArray = transpose(initialAgeNormArray)


    PRINT, "searchTypeSelectedBasic = find initial mass based upon initial/MS age"
    PRINT, "isochrone stageSelected = ", stageSelected

    starInputValuesArraySize = SIZE(initialAgeNormArray)
    numberOfStars = starInputValuesArraySize[1]

endif else if(searchTypeSelectedBasic eq 3) then begin

    PRINT, "starValuesFileNameFinalAge = ", starValuesFileNameFinalAge
    readcol64bitvars, starValuesFileNameFinalAge, finalAgeNormArray, finalAgeMinArray,
        finalAgeMaxArray, FORMAT='D,D,D', SKIPLINE=1
    finalAgeNormArray=finalAgeNormArray*1e6
    finalAgeMinArray=finalAgeMinArray*1e6
    finalAgeMaxArray=finalAgeMaxArray*1e6
    finalAgeArray = [transpose(finalAgeNormArray), transpose(finalAgeMinArray), transpose(
        finalAgeMaxArray)]


    PRINT, "searchTypeSelectedBasic = find initial mass based upon final/WD age and system age
        constraints"
    PRINT, "isochrone stageSelected = ", stageSelected

    PRINT, "starValuesFileNameFinalMass = ", starValuesFileNameFinalMass
    readcol64bitvars, starValuesFileNameFinalMass, finalMassNormArray, finalMassMinArray,
        finalMassMaxArray, FORMAT='D,D,D', SKIPLINE=1
    finalMassArray = [transpose(finalMassNormArray), transpose(finalMassMinArray), transpose(
        finalMassMaxArray)]

    if(searchTypeSelectedIFMR eq 0) then begin
        ;used to take into account errors in IFMR form during plotting [added 27 June 11a]

        PRINT, "starValuesFileNameInitialAge = ", starValuesFileNameInitialAge
        readcol64bitvars, starValuesFileNameInitialAge, initialAgeNormArray, initialAgeMinArray,
            initialAgeMaxArray, FORMAT='D,D,D', SKIPLINE=1
        initialAgeNormArray=initialAgeNormArray*1e6
        initialAgeMinArray=initialAgeMinArray*1e6
        initialAgeMaxArray=initialAgeMaxArray*1e6
        initialAgeArray = [transpose(initialAgeNormArray), transpose(initialAgeMinArray),
            transpose(initialAgeMaxArray)]


        PRINT, "searchTypeSelectedBasic = find initial mass based upon initial/MS age"
        PRINT, "isochrone stageSelected = ", stageSelected

        PRINT, "starValuesFileNameInitialMass = ", starValuesFileNameInitialMass
        readcol64bitvars, starValuesFileNameInitialMass, initialMassNormArray,
            initialMassMinArray, initialMassMaxArray, FORMAT='D,D,D', SKIPLINE=1
        initialMassArray = [transpose(initialMassNormArray), transpose(initialMassMinArray),
            transpose(initialMassMaxArray)]

        PRINT, "searchTypeSelectedBasic = find initial age based upon initial/MS mass"
        PRINT, "isochrone stageSelected = ", stageSelected

    endif

    starInputValuesArraySize = SIZE(finalAgeNormArray)
    numberOfStars = starInputValuesArraySize[1]

endif else begin

    PRINT, "searchTypeSelectedBasic = invalid"
    exit

endelse
```

```
;PRINT, "numberOfStars = ", numberOfStars
;PRINT, "values to find = "
;for starIndex=0, numberOfStars-1, 1 do begin
;
;   PRINT, valuesToFind[starIndex]
;endfor


valueToFindTempArray = double(1)
MiniFiltered = DblArr(modelGridSize)
ageFiltered = DblArr(modelGridSize)

; create arrays using stageSelected;

modelIndexFiltered = 0

for modelIndex=0, modelGridSize-1, 1 do begin

    if(Stage[modelIndex] eq stageSelected) then begin

        MiniFiltered[modelIndexFiltered] = Mini[modelIndex]
        ageFiltered[modelIndexFiltered] = age[modelIndex]
        modelIndexFiltered = modelIndexFiltered + 1
    endif

endfor

modelGridSizeFiltered = modelIndexFiltered

MiniFiltered2 = DblArr(modelGridSizeFiltered)
ageFiltered2 = DblArr(modelGridSizeFiltered)

for modelIndexFiltered=0, modelGridSizeFiltered-1, 1 do begin

    MiniFiltered2[modelIndexFiltered] = MiniFiltered[modelIndexFiltered]
    ageFiltered2[modelIndexFiltered] = ageFiltered[modelIndexFiltered]

endfor


idealSystemAgeIndexForStarArray = IntArr(maxStarIndex, numberofextremes)



; perform match;
for extremeIndex=0, numberOfExtremes-1, 1 do begin

    ;PRINT, "extremeIndex = ", extremeIndex

    for starIndex=0, numberOfStars-1, 1 do begin

        ;PRINT, "starIndex = ", starIndex

        if(searchTypeSelectedBasic eq 3) then begin
            if(searchTypeSelectedIFMR eq 0) then begin
                if(useRangeOfSystemAges eq 1) then begin
                    if(useSharedEstimatedSystemAgeOfSystem eq 1) then begin

                        starIndexOnTwo = LONG(starIndex/2)
                        estimatedSystemAgeMinForDDSystemA = min([systemAgeEstimateMin[
                            starIndexOnTwo*2], systemAgeEstimateMax[starIndexOnTwo*2]])
                        estimatedSystemAgeMinForDDSystemB = min([systemAgeEstimateMin[
                            starIndexOnTwo*2+1], systemAgeEstimateMax[starIndexOnTwo*2+1]])
                        estimatedSystemAgeMaxForDDSystemA = max([systemAgeEstimateMax[
                            starIndexOnTwo*2], systemAgeEstimateMin[starIndexOnTwo*2]])
                        estimatedSystemAgeMaxForDDSystemB = max([systemAgeEstimateMax[
                            starIndexOnTwo*2+1], systemAgeEstimateMin[starIndexOnTwo*2+1]])

                        ;PRINT, "estimatedSystemAge1ForDDSystemA = ", systemAgeEstimateMin[
                            starIndexOnTwo*2]
                        ;PRINT, "estimatedSystemAge1ForDDSystemB = ", systemAgeEstimateMin[
                            starIndexOnTwo*2+1]
                        ;PRINT, "estimatedSystemAge2ForDDSystemA = ", systemAgeEstimateMax[
                            starIndexOnTwo*2]
                        ;PRINT, "estimatedSystemAge2ForDDSystemB = ", systemAgeEstimateMax[
                            starIndexOnTwo*2+1]

                        ;PRINT, "estimatedSystemAgeMinForDDSystemA = ",
                            estimatedSystemAgeMinForDDSystemA
                        ;PRINT, "estimatedSystemAgeMinForDDSystemB = ",
                            estimatedSystemAgeMinForDDSystemB
                        ;PRINT, "estimatedSystemAgeMaxForDDSystemA = ",
                            estimatedSystemAgeMaxForDDSystemA
                        ;PRINT, "estimatedSystemAgeMaxForDDSystemB = ",
                            estimatedSystemAgeMaxForDDSystemB

                        estimatedSystemAgeMinForDDSystem = min([
                            estimatedSystemAgeMinForDDSystemA,
                            estimatedSystemAgeMinForDDSystemB])
```

```
                estimatedSystemAgeMaxForDDSystem = max([
                    estimatedSystemAgeMaxForDDSystemA,
                    estimatedSystemAgeMaxForDDSystemB])

                ;estimatedSystemAgeMinForDDSystem = min([systemAgeEstimateMin[
                    starIndexOnTwo*2], systemAgeEstimateMin[starIndexOnTwo*2+1]])
                ;estimatedSystemAgeMaxForDDSystem = max([systemAgeEstimateMax[
                    starIndexOnTwo*2], systemAgeEstimateMax[starIndexOnTwo*2+1]])

            endif else begin
                estimatedSystemAgeMinForDDSystem = min([systemAgeEstimateMax[starIndex],
                    systemAgeEstimateMin[starIndex]])
                estimatedSystemAgeMaxForDDSystem = max([systemAgeEstimateMax[starIndex],
                    systemAgeEstimateMin[starIndex]])

                ;estimatedSystemAgeMinForDDSystem = systemAgeEstimateMin[starIndex]
                ;estimatedSystemAgeMaxForDDSystem = systemAgeEstimateMax[starIndex]
            endelse
            ;PRINT, "estimatedSystemAgeMinForDDSystem = ",
                estimatedSystemAgeMinForDDSystem
            ;PRINT, "estimatedSystemAgeMaxForDDSystem = ",
                estimatedSystemAgeMaxForDDSystem

            numberOfSystemAgeEstimates = (estimatedSystemAgeMaxForDDSystem-
                estimatedSystemAgeMinForDDSystem) / systemAgeIncrement
            ;PRINT, "numberOfSystemAgeEstimates = ", numberOfSystemAgeEstimates

        endif else begin
            numberOfSystemAgeEstimates = 1
        endelse
        systemAgeRangeSizeArray(starIndex) = numberOfSystemAgeEstimates
    endif else begin
        numberOfSystemAgeEstimates = 1
    endelse
endif else begin
    numberOfSystemAgeEstimates = 1
endelse


if (searchTypeSelectedBasic eq 1) then begin

    ;PRINT, "ageFiltered2 = ", ageFiltered2
    ;PRINT, "MiniFiltered2 = ", MiniFiltered2
    ;PRINT, "valueToFindTempArray = ", valueToFindTempArray

    closestMassInitial = initialMassArray[extremeIndex, starIndex]
    ;PRINT, "closestMassInitial = ", closestMassInitial

    valueToFindTempArray(0) = closestMassInitial

    if(interpolationTypeArray[combinationIndex] eq "linear") then begin
        closestAgeInitial = INTERPOL(ageFiltered2, MiniFiltered2, valueToFindTempArray)
    endif else if(interpolationTypeArray[combinationIndex] eq "spline") then begin
        closestAgeInitial = INTERPOL(ageFiltered2, MiniFiltered2, valueToFindTempArray,
            SPLINE=1)
        ;closestAgeInitial = SPLINE(REVERSE(MiniFiltered2), REVERSE(ageFiltered2),
            valueToFindTempArray)
    endif

    ;PRINT, "closestAgeInitial = ", closestAgeInitial
    combinationResultsArray[combinationIndex, extremeIndex, starIndex] =
        closestAgeInitial

endif else if(searchTypeSelectedBasic eq 2) then begin

    closestAgeInitial = initialAgeArray[extremeIndex, starIndex]
    valueToFindTempArray(0) = closestAgeInitial

    if(interpolationTypeArray[combinationIndex] eq "linear") then begin
        closestMassInitial = INTERPOL(MiniFiltered2, ageFiltered2, valueToFindTempArray)
    endif else if(interpolationTypeArray[combinationIndex] eq "spline") then begin
        closestMassInitial = INTERPOL(MiniFiltered2, ageFiltered2, valueToFindTempArray,
            SPLINE=1)
        ;closestMassInitial = SPLINE(REVERSE(ageFiltered2), REVERSE(MiniFiltered2),
            valueToFindTempArray)
    endif

    ;PRINT, "closestMassInitial = ", closestMassInitial
    combinationResultsArray[combinationIndex, extremeIndex, starIndex] =
        closestMassInitial

endif else if(searchTypeSelectedBasic eq 3) then begin

    if(searchTypeSelectedIFMR eq 0) then begin
        currentClosestSystemAgeDiffTemp = dblarr(3)
        for systemAgeExtremeIndex=0, numberOfExtremes-1, 1 do begin
            currentClosestSystemAgeDiffTemp(systemAgeExtremeIndex) = 10e20
        endfor

        ;PRINT, "systemAgeEstimate[starIndex, 0] = ", systemAgeEstimate[starIndex, 0]
        for systemAgeIndex=0L, numberOfSystemAgeEstimates-1L, 1L do begin
```

```
;PRINT, "hello0"

for systemAgeExtremeIndex=0, numberOfExtremes-1, 1 do begin ;
     systemAgeExtremeIndex > 0 not used at the moment: only
     systemAgeExtremeNorm is currently used...
     ;PRINT, "systemAgeExtremeIndex = ", systemAgeExtremeIndex
          ;PRINT, "systemAgeEstimate[systemAgeExtremeIndex, starIndex]  = ",
               systemAgeEstimate[systemAgeExtremeIndex, starIndex]
          ;PRINT, "currentSystemAgeTemp = ", currentSystemAgeTemp
          ;PRINT, "currentSystemAgeDiff = ", currentSystemAgeDiff
          ;PRINT, "systemAgeIndex = ", systemAgeIndex
     currentSystemAgeTemp = (estimatedSystemAgeMinForDDSystem+(systemAgeIndex
          *systemAgeIncrement))
     currentSystemAgeDiff = abs(systemAgeEstimate[systemAgeExtremeIndex,
          starIndex] - currentSystemAgeTemp)
     if(currentSystemAgeDiff lt currentClosestSystemAgeDiffTemp[
          systemAgeExtremeIndex]) then begin
          currentClosestSystemAgeDiffTemp[systemAgeExtremeIndex] =
               currentSystemAgeDiff
          idealSystemAgeIndexForStarArray[starIndex, systemAgeExtremeIndex] =
               systemAgeIndex
               ;PRINT, "currentClosestSystemAgeDiffTemp[systemAgeExtremeIndex]
                    = ", currentClosestSystemAgeDiffTemp[systemAgeExtremeIndex]
               ;PRINT, "idealSystemAgeIndexForStarArray[starIndex,
                    systemAgeExtremeIndex] = ", idealSystemAgeIndexForStarArray
                    [starIndex, systemAgeExtremeIndex]
     endif
endfor

;PRINT, "hello1"

closestMassFinal = finalMassArray[extremeIndex, starIndex]
closestAgeFinal = finalAgeArray[extremeIndex, starIndex]
if(useRangeOfSystemAges eq 1) then begin
     systemAge = (estimatedSystemAgeMinForDDSystem+(systemAgeIndex*
          systemAgeIncrement))
endif else begin
     systemAge = (systemAgeEstimateNorm[starIndex])
endelse
closestAgeInitial = systemAge - closestAgeFinal
;PRINT, "closestAgeFinal = ", closestAgeFinal
;PRINT, "closestAgeInitial = ", closestAgeInitial
valueToFindTempArray(0) = closestAgeInitial

if(interpolationTypeArray[combinationIndex] eq "linear") then begin
     closestMassInitial = INTERPOL(MiniFiltered2, ageFiltered2,
          valueToFindTempArray)
endif else if(interpolationTypeArray[combinationIndex] eq "spline") then
     begin
     closestMassInitial = INTERPOL(MiniFiltered2, ageFiltered2,
          valueToFindTempArray, SPLINE=1)
     ;closestMassInitial = SPLINE(REVERSE(ageFiltered2), REVERSE(
          MiniFiltered2), valueToFindTempArray)
endif

if(useRangeOfSystemAges eq 1) then begin
     if(systemAgeIndex eq 0) then begin  ;NB the first system age is declared
          as the best estimate system age (based upon the known IFMR)
          systemAge = systemAgeEstimate[extremeIndex, starIndex]       ;
               irrelevant [not used]
          ;used to take into account errors in IFMR form during plotting [
               added 27 June 11a]
          closestMassInitial = initialMassArray[extremeIndex, starIndex]
          closestAgeInitial = initialAgeArray[extremeIndex, starIndex]
     endif
endif

;PRINT, "closestMassInitial = ", closestMassInitial
if(useRangeOfSystemAges eq 1) then begin
     ;PRINT, "maxNumberOfSystemAgesIncrements = ",
          maxNumberOfSystemAgesIncrements
     ;PRINT, "systemAgeIndex = ", systemAgeIndex
     ;PRINT, "numberOfSystemAgeEstimates = ", numberOfSystemAgeEstimates
     systemAgeRangeResultsArray(starIndex, extremeIndex, systemAgeIndex, 0) =
          closestMassFinal
     systemAgeRangeResultsArray(starIndex, extremeIndex, systemAgeIndex, 1) =
          closestAgeFinal
     systemAgeRangeResultsArray(starIndex, extremeIndex, systemAgeIndex, 2) =
          systemAge
     systemAgeRangeResultsArray(starIndex, extremeIndex, systemAgeIndex, 3) =
          closestMassInitial
     systemAgeRangeResultsArray(starIndex, extremeIndex, systemAgeIndex, 4) =
          closestAgeInitial
endif else begin
     combinationResultsArray[combinationIndex, extremeIndex, starIndex] =
          closestMassInitial
endelse

;PRINT, "hello2"

  endfor
endif else if(searchTypeSelectedIFMR eq 1) then begin
```

```
if(systemAgeOptimisationFast eq 0) then begin

    for mIndex=0, numberOfMIndicies-1, 1 do begin
        for cIndex=0, numberOfCIndicies-1, 1 do begin

            IFMRmTemp = IFMRParameterOptimisationMinM + double(mIndex)*
                IFMRParameterOptimisationIncrementM
            IFMRcTemp = IFMRParameterOptimisationMinC + double(cIndex)*
                IFMRParameterOptimisationIncrementC
            ;PRINT, "mTemp = ", IFMRmTemp
            ;PRINT, "cTemp = ", IFMRcTemp

            finalMassTheoretical =  finalMassArray[extremeIndex, starIndex]
            initialMassTheoretical = (finalMassTheoretical-IFMRcTemp)/IFMRmTemp

            valueToFindTempArray(0) = initialMassTheoretical
            if(interpolationTypeArray[combinationIndex] eq "linear") then begin
                closestAgeInitial = INTERPOL(ageFiltered2, MiniFiltered2,
                    valueToFindTempArray)
            endif else if(interpolationTypeArray[combinationIndex] eq "spline")
                then begin
                closestAgeInitial = INTERPOL(ageFiltered2, MiniFiltered2,
                    valueToFindTempArray, SPLINE=1)
                ;closestAgeInitial = SPLINE(REVERSE(MiniFiltered2), REVERSE(
                    ageFiltered2), valueToFindTempArray)
            endif

            ;PRINT, "finalMassTheoretical = ", finalMassTheoretical
            ;PRINT, "coolingAge = ", coolingAge
            ;PRINT, "systemAge = ", systemAge
            ;PRINT, "initialMassTheoretical = ", initialMassTheoretical
            ;PRINT, "mainSequenceAge = ", mainSequenceAge

            mainSequenceAge = closestAgeInitial
            coolingAge =  finalAgeArray[extremeIndex, starIndex]
            systemAge = coolingAge + mainSequenceAge

            systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex, mIndex
                , cIndex, 0] = finalMassTheoretical
            systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex, mIndex
                , cIndex, 1] = coolingAge
            systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex, mIndex
                , cIndex, 2] = systemAge
            systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex, mIndex
                , cIndex, 3] = initialMassTheoretical
            systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex, mIndex
                , cIndex, 4] = mainSequenceAge

            ;PRINT, "systemAgeRangeArrayForIFMRParameters[starIndex,
                extremeIndex, mIndex, cIndex, 0] = ",
                systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex,
                mIndex, cIndex, 0]
        endfor
    endfor



    ;if(DDIndex eq 11) then begin
    ;   PRINT, "WDxResultsArrayNorm = "
    ;   PRINT, WDxResultsArrayNorm
    ;endif

    if(extremeIndex eq 0) then begin

        DDIndex = (starIndex/2)+1
        DDIndexString = STRTRIM(DDIndex, 2)
        DDComponentString = ''
        if((starIndex mod 2) ne 0) then begin
            DDComponentString = 'B'
        endif else begin
            DDComponentString = 'A'
        endelse

        outputFileNameTxt= 'DD-' + DDIndexString + DDComponentString + '
            SystemAgeOptimisation' + '.txt'

        OPENW,1,outputFileNameTxt
        PRINTF,1,transpose(systemAgeRangeArrayForIFMRParameters[starIndex,
            extremeIndex, *, *, *]), FORMAT='(e11.5,1X,e11.5,1X,e11.5,1X,e11
            .5,1X,e11.5)'
        ;PRINTF,1,systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex,
            *, *, 2], FORMAT='(e11.5)'
        CLOSE,1

        ;if(DDIndex eq 1) then begin
        ;   PRINT, "systemAgeRangeArrayForIFMRParameters = "
        ;   PRINT, transpose(systemAgeRangeArrayForIFMRParameters[starIndex,
            extremeIndex, *, *, *])
        ;endif
```

```
            ;PRINT, "systemAgeRangeArrayForIFMRParameters system ages, for starindex
                = ", starIndex
            ;PRINT, systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex, *,
                *, 2]
        endif
    endif else begin
        ;NB systemAgeOptimisationFast = 1
        if((starIndex mod 2) ne 0) then begin

            if(systemAgeOptimisationFastTest eq 1) then begin
                numberOfGridSizes = 1
                IFMRParameterOptimisationIncrementMForGridSize =
                    IFMRParameterOptimisationIncrementM
                IFMRParameterOptimisationIncrementCForGridSize =
                    IFMRParameterOptimisationIncrementC

            endif else begin
                numberOfGridSizes = 3    ;coarse, medium, fine
                IFMRParameterOptimisationIncrementMForGridSize = 0.1     ;NO: some
                    function of IFMRParameterOptimisationIncrementM
                IFMRParameterOptimisationIncrementCForGridSize = 0.1     ;NO: some
                    function of IFMRParameterOptimisationIncrementC
            endelse
            IFMRParameterOptimisationRangeMForGridSize =
                IFMRParameterOptimisationMaxM-IFMRParameterOptimisationMinM
            IFMRParameterOptimisationRangeCForGridSize =
                IFMRParameterOptimisationMaxC-IFMRParameterOptimisationMinC
            IFMRParameterOptimisationCentreMForGridSize =
                IFMRParameterOptimisationRangeMForGridSize/2 +
                IFMRParameterOptimisationMinM
            IFMRParameterOptimisationCentreCForGridSize =
                IFMRParameterOptimisationRangeCForGridSize/2 +
                IFMRParameterOptimisationMinC

            for gridSize=0, numberOfGridSizes-1, 1 do begin

                IFMRParameterOptimisationMinMForGridSize = max([
                    IFMRParameterOptimisationCentreMForGridSize-(
                    IFMRParameterOptimisationRangeMForGridSize/2), 0.0])
                IFMRParameterOptimisationMaxMForGridSize = min([
                    IFMRParameterOptimisationCentreMForGridSize+(
                    IFMRParameterOptimisationRangeMForGridSize/2),
                    IFMRParameterOptimisationMaxM])
                IFMRParameterOptimisationMinCForGridSize = max([
                    IFMRParameterOptimisationCentreCForGridSize-(
                    IFMRParameterOptimisationRangeCForGridSize/2), 0.0])
                IFMRParameterOptimisationMaxCForGridSize = min([
                    IFMRParameterOptimisationCentreCForGridSize+(
                    IFMRParameterOptimisationRangeCForGridSize/2),
                    IFMRParameterOptimisationMaxC])
                numberOfMIndiciesForGridSize = long((
                    IFMRParameterOptimisationMaxMForGridSize-
                    IFMRParameterOptimisationMinMForGridSize)/
                    IFMRParameterOptimisationIncrementMForGridSize)
                numberOfCIndiciesForGridSize = long((
                    IFMRParameterOptimisationMaxCForGridSize-
                    IFMRParameterOptimisationMinCForGridSize)/
                    IFMRParameterOptimisationIncrementCForGridSize)

                if(systemAgeOptimisationFastTest eq 1) then begin
                    currentSystemAgeDiffArray = dblarr(numberOfMIndiciesForGridSize
                        * numberOfCIndiciesForGridSize)
                    ;currentSystemAgeDiffArrayX = dblarr(
                        numberOfMIndiciesForGridSize)
                    ;currentSystemAgeDiffArrayY = dblarr(
                        numberOfCIndiciesForGridSize)
                    currentSystemAgeDiffArrayM = dblarr(numberOfMIndiciesForGridSize
                        * numberOfCIndiciesForGridSize)
                    currentSystemAgeDiffArrayC = dblarr(numberOfMIndiciesForGridSize
                        * numberOfCIndiciesForGridSize)
                endif

                PRINT, "gridSize = ", gridSize
                PRINT, "IFMRParameterOptimisationCentreMForGridSize = ",
                    IFMRParameterOptimisationCentreMForGridSize
                PRINT, "IFMRParameterOptimisationRangeMForGridSize = ",
                    IFMRParameterOptimisationRangeMForGridSize
                PRINT, "IFMRParameterOptimisationIncrementMForGridSize = ",
                    IFMRParameterOptimisationIncrementMForGridSize
                PRINT, "IFMRParameterOptimisationMinMForGridSize = ",
                    IFMRParameterOptimisationMinMForGridSize
                PRINT, "IFMRParameterOptimisationMaxMForGridSize = ",
                    IFMRParameterOptimisationMaxMForGridSize
                PRINT, "numberOfMIndiciesForGridSize = ",
                    numberOfMIndiciesForGridSize
                PRINT, "IFMRParameterOptimisationCentreCForGridSize = ",
                    IFMRParameterOptimisationCentreCForGridSize
                PRINT, "IFMRParameterOptimisationRangeCForGridSize = ",
                    IFMRParameterOptimisationRangeCForGridSize
                PRINT, "IFMRParameterOptimisationIncrementCForGridSize = ",
                    IFMRParameterOptimisationIncrementCForGridSize
```

```
PRINT, "IFMRParameterOptimisationMinCForGridSize = ",
    IFMRParameterOptimisationMinCForGridSize
PRINT, "IFMRParameterOptimisationMaxCForGridSize = ",
    IFMRParameterOptimisationMaxCForGridSize
PRINT, "numberOfCIndiciesForGridSize = ",
    numberOfCIndiciesForGridSize

idealMIndexForGridSize = -1
idealCIndexForGridSize = -1
idealMForGridSize = 0.0
idealCForGridSize = 0.0
idealSystemAgeForGridSize = 0.0
;idealSystemPropertiesForGridSize = dblarr(
    WDxResultsArrayNumberOfResults)
currentClosestSystemAgeDiffTemp = 10e20


for mIndex=0, numberOfMIndiciesForGridSize-1, 1 do begin
    for cIndex=0, numberOfCIndiciesForGridSize-1, 1 do begin

        IFMRmTemp = IFMRParameterOptimisationMinMForGridSize +
            double(mIndex)*
            IFMRParameterOptimisationIncrementMForGridSize
        IFMRcTemp = IFMRParameterOptimisationMinCForGridSize +
            double(cIndex)*
            IFMRParameterOptimisationIncrementCForGridSize
        ;PRINT, "mTemp = ", IFMRmTemp
        ;PRINT, "cTemp = ", IFMRcTemp

        AfinalMassTheoretical =  finalMassArray[extremeIndex,
            starIndex-1]
        AinitialMassTheoretical = (AfinalMassTheoretical-IFMRcTemp)/
            IFMRmTemp
        BfinalMassTheoretical =  finalMassArray[extremeIndex,
            starIndex]
        BinitialMassTheoretical = (BfinalMassTheoretical-IFMRcTemp)/
            IFMRmTemp

        valueToFindTempArray(0) = AinitialMassTheoretical
        if(interpolationTypeArray[combinationIndex] eq "linear")
            then begin
            AclosestAgeInitial = INTERPOL(ageFiltered2,
                MiniFiltered2, valueToFindTempArray)
        endif else if(interpolationTypeArray[combinationIndex] eq "
            spline") then begin
            AclosestAgeInitial = INTERPOL(ageFiltered2,
                MiniFiltered2, valueToFindTempArray, SPLINE=1)
            ;AclosestAgeInitial = SPLINE(REVERSE(MiniFiltered2),
                REVERSE(ageFiltered2), valueToFindTempArray)
        endif
        valueToFindTempArray(0) = BinitialMassTheoretical
        if(interpolationTypeArray[combinationIndex] eq "linear")
            then begin
            BclosestAgeInitial = INTERPOL(ageFiltered2,
                MiniFiltered2, valueToFindTempArray)
        endif else if(interpolationTypeArray[combinationIndex] eq "
            spline") then begin
            BclosestAgeInitial = INTERPOL(ageFiltered2,
                MiniFiltered2, valueToFindTempArray, SPLINE=1)
            ;BclosestAgeInitial = SPLINE(REVERSE(MiniFiltered2),
                REVERSE(ageFiltered2), valueToFindTempArray)
        endif

        AmainSequenceAge = AclosestAgeInitial
        AcoolingAge =  finalAgeArray[extremeIndex, starIndex-1]
        AsystemAge = AcoolingAge + AmainSequenceAge
        BmainSequenceAge = BclosestAgeInitial
        BcoolingAge =  finalAgeArray[extremeIndex, starIndex]
        BsystemAge = BcoolingAge + BmainSequenceAge

        ;PRINT, "AfinalMassTheoretical = ", AfinalMassTheoretical
        ;PRINT, "AcoolingAge = ", AcoolingAge
        ;PRINT, "AsystemAge = ", AsystemAge
        ;PRINT, "AinitialMassTheoretical = ",
            AinitialMassTheoretical
        ;PRINT, "AmainSequenceAge = ", AmainSequenceAge
        ;PRINT, "BfinalMassTheoretical = ", BfinalMassTheoretical
        ;PRINT, "BcoolingAge = ", BcoolingAge
        ;PRINT, "BsystemAge = ", BsystemAge
        ;PRINT, "BinitialMassTheoretical = ",
            BinitialMassTheoretical
        ;PRINT, "BmainSequenceAge = ", BmainSequenceAge

        currentSystemAgeDiff = abs(AsystemAge - BsystemAge)

        if(currentSystemAgeDiff lt currentClosestSystemAgeDiffTemp)
            then begin
            currentClosestSystemAgeDiffTemp = currentSystemAgeDiff
            idealMIndexForGridSize = mIndex
            idealCIndexForGridSize = cIndex
            idealMForGridSize = IFMRmTemp
```

```
                                idealCForGridSize = IFMRcTemp
                                idealSystemAgeForGridSize = AsystemAge
                                ;PRINT, "currentClosestSystemAgeDiffTemp= ",
                                    currentClosestSystemAgeDiffTemp
                            endif

                            if(systemAgeOptimisationFastTest eq 1) then begin
                                currentSystemAgeDiffArray[mIndex*
                                    numberOfCIndiciesForGridSize + cIndex] =
                                    currentSystemAgeDiff
                                currentSystemAgeDiffArrayM[mIndex*
                                    numberOfCIndiciesForGridSize + cIndex] = IFMRmTemp
                                currentSystemAgeDiffArrayC[mIndex*
                                    numberOfCIndiciesForGridSize + cIndex] = IFMRcTemp
                                ;currentSystemAgeDiffArrayX[mIndex] = IFMRmTemp
                                ;currentSystemAgeDiffArrayY[cIndex] = IFMRcTemp
                            endif


                        endfor
                    endfor

                    if(systemAgeOptimisationFastTest eq 1) then begin

                        if(extremeIndex eq 0) then begin
                            DDIndex = (starIndex/2)+1
                            DDIndexString = STRTRIM(DDIndex, 2)
                            currentSystemAgeDiffArrayFileName = "
                                currentSystemAgeDiffArrayDD" + DDIndexString + ".txt"
                            OPENW,1,currentSystemAgeDiffArrayFileName
                            currentSystemAgeDiffArrayTransposed = [transpose(
                                currentSystemAgeDiffArrayM), transpose(
                                currentSystemAgeDiffArrayC), transpose(
                                currentSystemAgeDiffArray)]
                            PRINTF,1,currentSystemAgeDiffArrayTransposed, FORMAT='(e11
                                .5, 1X, e11.5, 1X, e11.5)'
                            CLOSE,1
                            ;if(DDIndex eq 1) then begin
                            ;   PRINT, "currentSystemAgeDiffArray = "
                            ;   PRINT, (currentSystemAgeDiffArray)  ;alog10(
                                currentSystemAgeDiffArray)
                            ;   PRINT, "currentSystemAgeDiffArray[*,0] = "
                            ;   PRINT, currentSystemAgeDiffArray[*,0]
                            ;   ;so currentSystemAgeDiffArrayFileName is printed as c=0
                                all ms, c=1 all ms, etc
                            ;endif

                            ;surface plot doesnt work with gdl
                            ;surface, alog10(currentSystemAgeDiffArray),
                                currentSystemAgeDiffArrayX, currentSystemAgeDiffArrayY,
                                 background=255,color=0
                            ;wait, 5
                            ;as
                                ;wait, 5
                                ;plot, tempXArray, tempYArray, title='2D plot of sys age
                                     diff', xtitle='m', ytitle='c', xrange=[
                                    IFMRParameterOptimisationMinMForGridSize,
                                    IFMRParameterOptimisationMaxMForGridSize], yrange=[
                                    IFMRParameterOptimisationMinCForGridSize,
                                    IFMRParameterOptimisationMaxCForGridSize]
                                ;oplot, IFMRmTemp, IFMRcTemp
                            endif
                        endif

                        IFMRParameterOptimisationCentreMForGridSize = idealMForGridSize
                        IFMRParameterOptimisationCentreCForGridSize = idealCForGridSize
                        IFMRParameterOptimisationRangeMForGridSize =
                            IFMRParameterOptimisationIncrementMForGridSize*3
                        IFMRParameterOptimisationRangeCForGridSize =
                            IFMRParameterOptimisationIncrementCForGridSize*3
                        IFMRParameterOptimisationIncrementMForGridSize =
                            IFMRParameterOptimisationIncrementMForGridSize/10
                        IFMRParameterOptimisationIncrementCForGridSize =
                            IFMRParameterOptimisationIncrementCForGridSize/10
                    endfor

                    ;PRINT, "idealMIndexForGridSize" = ", idealMIndexForGridSize
                    ;PRINT, "idealCIndexForGridSize" = ", idealCIndexForGridSize
                    PRINT, "idealMForGridSize = ", idealMForGridSize
                    PRINT, "idealCForGridSize = ", idealCForGridSize
                    idealIFMRpropertiesForStarArray[starIndex, extremeIndex, 0] =
                        idealMIndexForGridSize
                    idealIFMRpropertiesForStarArray[starIndex, extremeIndex, 1] =
                        idealCIndexForGridSize
                    idealSystemAgeForStarArray[starIndex, extremeIndex] =
                        idealSystemAgeForGridSize

                endif
            endelse
        endif else if(searchTypeSelectedIFMR eq 2) then begin
            for IFMRIndex=0, numberOfPublshedIFMRs-1, 1 do begin
```

```
                        IFMRmTemp = publshedIFMRparamterArray(IFMRIndex, 0)
                        IFMRcTemp = publshedIFMRparamterArray(IFMRIndex, 1)

                        if(useErrorsInPublishedIFMRdefinitions eq 1) then begin
                            ;When calculating System Age comparison bar graph, take into account
                                published errors in IFMR linear form definitions
                            IFMRmErrTemp = publshedIFMRparamterArrayErr(IFMRIndex, 0)
                            IFMRcErrTemp = publshedIFMRparamterArrayErr(IFMRIndex, 1)
                            if(extremeIndex eq 1) then begin
                                IFMRmTemp = IFMRmTemp + IFMRmErrTemp
                                IFMRcTemp = IFMRcTemp + IFMRcErrTemp
                            endif else if(extremeIndex eq 2) then begin
                                IFMRmTemp = IFMRmTemp - IFMRmErrTemp
                                IFMRcTemp = IFMRcTemp - IFMRcErrTemp
                            endif else begin

                            endelse
                        endif


                        ;PRINT, "IFMRmTemp = ", IFMRmTemp
                        ;PRINT, "IFMRcTemp = ", IFMRcTemp

                        ;PRINT, "mTemp = ", IFMRmTemp
                        ;PRINT, "cTemp = ", IFMRcTemp

                        finalMassTheoretical =  finalMassArray[extremeIndex, starIndex]
                        initialMassTheoretical = (finalMassTheoretical-IFMRcTemp)/IFMRmTemp

                        valueToFindTempArray(0) = initialMassTheoretical
                        if(interpolationTypeArray[combinationIndex] eq "linear") then begin
                            closestAgeInitial = INTERPOL(ageFiltered2, MiniFiltered2,
                                valueToFindTempArray)
                        endif else if(interpolationTypeArray[combinationIndex] eq "spline") then
                             begin
                            closestAgeInitial = INTERPOL(ageFiltered2, MiniFiltered2,
                                valueToFindTempArray, SPLINE=1)
                            ;closestAgeInitial = SPLINE(REVERSE(MiniFiltered2), REVERSE(ageFiltered2
                                ), valueToFindTempArray)
                        endif

                        mainSequenceAge = closestAgeInitial
                        coolingAge =  finalAgeArray[extremeIndex, starIndex]
                        systemAge = coolingAge + mainSequenceAge

                        ;if((starIndex eq 0) and (IFMRIndex eq 3)) then begin
                        ;   PRINT, ""
                        ;   PRINT, "extremeIndex = ", extremeIndex
                        ;   PRINT, "starIndex = ", starIndex
                        ;   PRINT, "finalMassTheoretical = ", finalMassTheoretical
                        ;   PRINT, "coolingAge = ", coolingAge
                        ;   PRINT, "systemAge = ", systemAge
                        ;   PRINT, "initialMassTheoretical = ", initialMassTheoretical
                        ;   PRINT, "mainSequenceAge = ", mainSequenceAge
                        ;endif

                        systemAgeRangeArrayForPublishedIFMRs[starIndex, extremeIndex, IFMRIndex, 0]
                            = finalMassTheoretical
                        systemAgeRangeArrayForPublishedIFMRs[starIndex, extremeIndex, IFMRIndex, 1]
                            = coolingAge
                        systemAgeRangeArrayForPublishedIFMRs[starIndex, extremeIndex, IFMRIndex, 2]
                            = systemAge
                        systemAgeRangeArrayForPublishedIFMRs[starIndex, extremeIndex, IFMRIndex, 3]
                            = initialMassTheoretical
                        systemAgeRangeArrayForPublishedIFMRs[starIndex, extremeIndex, IFMRIndex, 4]
                            = mainSequenceAge

                        ;PRINT, "systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex,
                            mIndex, cIndex, 0] = ", systemAgeRangeArrayForIFMRParameters[starIndex,
                             extremeIndex, mIndex, cIndex, 0]
                    endfor
                endif

            endif else begin
                PRINT, "searchTypeSelectedBasic = invalid"
                exit
            endelse



        endfor
    endfor

    ;exit
endfor



if (searchTypeSelectedBasic eq 1) then begin
    PRINT, "closestAgeInitial = "
    combinationResultsArrayNorm = combinationResultsArray[*,0,*]
    combinationResultsArrayMin = combinationResultsArray[*,1,*]
```

```
    combinationResultsArrayMax = combinationResultsArray[*,2,*]
    ;PRINT, [combinationResultsArrayNorm, combinationResultsArrayMin, combinationResultsArrayMax]
    PRINT, "combinationResultsArrayNorm = "
    PRINT, combinationResultsArrayNorm
    PRINT, "combinationResultsArrayMin = "
    PRINT, combinationResultsArrayMin
    PRINT, "combinationResultsArrayMax = "
    PRINT, combinationResultsArrayMax
endif else if(searchTypeSelectedBasic eq 2) then begin
    PRINT, "closestMassInitial = "
    combinationResultsArrayNorm = combinationResultsArray[*,0,*]
    combinationResultsArrayMin = combinationResultsArray[*,1,*]
    combinationResultsArrayMax = combinationResultsArray[*,2,*]
    ;PRINT, [combinationResultsArrayNorm, combinationResultsArrayMin, combinationResultsArrayMax]
    PRINT, "combinationResultsArrayNorm = "
    PRINT, combinationResultsArrayNorm
    PRINT, "combinationResultsArrayMin = "
    PRINT, combinationResultsArrayMin
    PRINT, "combinationResultsArrayMax = "
    PRINT, combinationResultsArrayMax
endif else if(searchTypeSelectedBasic eq 3) then begin
    if(searchTypeSelectedIFMR eq 0) then begin
        if(useRangeOfSystemAges eq 1) then begin
            PRINT, "systemAge    ageFinal    ageInitial  massInitial = "
            for starIndex=0, numberOfStars-1, 1 do begin
                ;PRINT, "starIndex = ", starIndex
                numberOfSystemAgeEstimates = systemAgeRangeSizeArray(starIndex)
                    ;PRINT, "numberOfSystemAgeEstimates = ", numberOfSystemAgeEstimates
                    ;PRINT, "numberOfExtremes = ", numberOfExtremes
                    ;PRINT, "WDxResultsArrayNumberOfResults = ", WDxResultsArrayNumberOfResults
                WDxResultsArray = DBLARR(numberOfExtremes, numberOfSystemAgeEstimates,
                    WDxResultsArrayNumberOfResults)


                ;PRINT, systemAgeRangeResultsArray(3, *, *, *)
                ;exit

                for extremeIndex=0, numberOfExtremes-1, 1 do begin
                    for systemAgeIndex=0L, numberOfSystemAgeEstimates-1L, 1 do begin

                        ;PRINT, "extremeIndex = ", extremeIndex
                        ;PRINT, "systemAgeIndex = ", systemAgeIndex

                            ;PRINT, "systemAgeRangeResultsArray(starIndex, extremeIndex,
                                systemAgeIndex, *) = ", systemAgeRangeResultsArray(starIndex,
                                extremeIndex, systemAgeIndex, *)
                            ;PRINT, "WDxResultsArray(extremeIndex, systemAgeIndex, *) = ",
                                WDxResultsArray(extremeIndex, systemAgeIndex, *)

                        WDxResultsArray(extremeIndex, systemAgeIndex, *) =
                            systemAgeRangeResultsArray(starIndex, extremeIndex, systemAgeIndex, *)
                                ;tempArray2

                        tempArray = systemAgeRangeResultsArray(starIndex, extremeIndex,
                            systemAgeIndex, *)
                        tempArray2 = transpose(tempArray)
                        ;PRINT, "tempArray2 = ", tempArray2

                    endfor
                endfor

                WDxResultsArrayNorm = transpose(WDxResultsArray[0, *, *])
                WDxResultsArrayMin = transpose(WDxResultsArray[1, *, *])
                WDxResultsArrayMax = transpose(WDxResultsArray[2, *, *])

                if(starIndex eq 0) then begin
                    WDxResultsArrayA = WDxResultsArray
                    WDxResultsArrayANorm = WDxResultsArrayNorm
                    WDxResultsArrayAMin = WDxResultsArrayMin
                    WDxResultsArrayAMax = WDxResultsArrayMax
                endif

                DDIndex = (starIndex/2)+1
                DDIndexString = STRTRIM(DDIndex, 2)
                DDComponentString = ''
                if((starIndex mod 2) ne 0) then begin
                    DDComponentString = 'B'
                endif else begin
                    DDComponentString = 'A'
                endelse

                outputFileNameTxt= 'DD-' + DDIndexString + DDComponentString + 'plotIFMR' + '.txt'
                outputFileNamePDF= 'DD-' + DDIndexString + 'plotIFMR' + '.pdf'
                outputFileNamePS= 'DD-' + DDIndexString + 'plotIFMR' + '.ps'



                ;PRINT, "massFinal, ageFinal, systemAge, massInitial, ageInitial, massFinal,
                    ageFinal (x3: ageFinal set to norm, min, max)"
                ;PRINT, "WDxResultsArray = ", WDxResultsArray
```

```
;PRINT, "WDxResultsArrayNorm[*, 0] = ", [WDxResultsArrayBNorm[*, 0],
    WDxResultsArrayANorm[*, 0]]
;PRINT, "WDxResultsArrayMin[*, 0] = ", [WDxResultsArrayBMin[*, 0],
    WDxResultsArrayAMin[*, 0]]
;PRINT, "WDxResultsArrayMax[*, 0] = ", [WDxResultsArrayBMax[*, 0],
    WDxResultsArrayAMax[*, 0]]



;if(DDIndex eq 11) then begin
;   PRINT, "WDxResultsArrayNorm = "
;   PRINT, WDxResultsArrayNorm
;endif

OPENW,1,outputFileNameTxt
PRINTF,1,[WDxResultsArrayNorm, WDxResultsArrayMin, WDxResultsArrayMax], FORMAT='(e11
    .5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,
    e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5,1X,e11.5)'
CLOSE,1



if((starIndex mod 2) ne 0) then begin

    WDxResultsArrayB = WDxResultsArray
    WDxResultsArrayBNorm = WDxResultsArrayNorm
    WDxResultsArrayBMin = WDxResultsArrayMin
    WDxResultsArrayBMax = WDxResultsArrayMax

    ;PRINT, "starIndex2 = ", starIndex

    ;creates plot of Mi A versus Mi B for a given system age (from minimum estimated
        system age to maximum estimated system age, for m_f norm, m_f min, and m_f
        max)
    ;it also overlays a plot of MiA versus MiB for their individual ideal system
        ages

    if(searchTypeSelectedIFMRPlot eq 0) then begin
        ;3.0

        if(plotToFile eq 0) then begin

            plot, tempXArray, tempYArray, title='', xtitle='M initial A', ytitle='M
                initial B', xrange=[minIFMRmInitial, maxIFMRmInitial], yrange=[
                minIFMRmInitial, maxIFMRmInitial]

        endif else begin

            set_plot,'ps'
            device,filename=outputFileNamePS, /LANDSCAPE     ;/PORTRAIT /LANDSCAPE
                ;SET_RESOLUTION=[320,200]    ;SET_CHARACTER_SIZE=[200,40]
            if(useGDL) then begin
                plot, tempXArray, tempYArray, title='', xtitle='M initial A', ytitle
                    ='M initial B', xrange=[minIFMRmInitial, maxIFMRmInitial],
                    yrange=[minIFMRmInitial, maxIFMRmInitial], BACKGROUND=1,
                    CHARSIZE=2.0,XMARGIN=32,YMARGIN=18
            endif else begin
                plot, tempXArray, tempYArray, title='', xtitle='M initial A', ytitle
                    ='M initial B', xrange=[minIFMRmInitial, maxIFMRmInitial],
                    yrange=[minIFMRmInitial, maxIFMRmInitial]
            endelse
        endelse

            ;massFinal  ageFinal    systemAge   massInitial ageInitial

        ;Mi:Mi plot;
        oplot, WDxResultsArrayANorm[3, 1:*], WDxResultsArrayBNorm[3, 1:*]
        oplot, WDxResultsArrayAMin[3, 1:*], WDxResultsArrayBMin[3, 1:*]
        oplot, WDxResultsArrayAMax[3, 1:*], WDxResultsArrayBMax[3, 1:*]

        WDxResultsArrayOptSysAgeIdeal = [WDxResultsArrayBMin[3, 0],
            WDxResultsArrayBNorm[3, 0], WDxResultsArrayBMax[3, 0]]  ;Mi for ideal
            sys age
        WDxResultsArrayAOptSysAgeIdeal = [WDxResultsArrayAMin[3, 0],
            WDxResultsArrayANorm[3, 0], WDxResultsArrayAMax[3, 0]]     ;Mi for
            ideal sys age
        PRINT, "WDxResultsArrayAOptSysAgeIdeal = ", WDxResultsArrayAOptSysAgeIdeal
        PRINT, "WDxResultsArrayOptSysAgeIdeal = ", WDxResultsArrayOptSysAgeIdeal


        oplot, WDxResultsArrayAOptSysAgeIdeal, WDxResultsArrayOptSysAgeIdeal, THICK
            =3, psym=-1
    endif else if((searchTypeSelectedIFMRPlot eq 1) or (searchTypeSelectedIFMRPlot
        eq 2)) then begin

        if(plotToFile eq 0) then begin

            plot, tempXArray, tempYArray, title='', xtitle='M initial', ytitle='M
                final', xrange=[minIFMRmInitial, maxIFMRmInitial], yrange=[
                minIFMRmFinal, maxIFMRmFinal]
```

```
endif else begin

    set_plot,'ps'
    device,filename=outputFileNamePS, /LANDSCAPE     ;/PORTRAIT  /LANDSCAPE
            ;SET_RESOLUTION=[320,200]     ;SET_CHARACTER_SIZE=[200,40]

    if(useGDL) then begin
        plot, tempXArray, tempYArray, title='', xtitle='M initial', ytitle='
            M final', xrange=[minIFMRmInitial, maxIFMRmInitial], yrange=[
            minIFMRmFinal, maxIFMRmFinal], BACKGROUND=1, CHARSIZE=2.0,
            XMARGIN=32,YMARGIN=18
    endif else begin
        plot, tempXArray, tempYArray, title='', xtitle='M initial', ytitle='
            M final', xrange=[minIFMRmInitial, maxIFMRmInitial], yrange=[
            minIFMRmFinal, maxIFMRmFinal]
    endelse
endelse

;IFMR Plot;
IFMRincrement = 0.01
numberOfIFMRPoints = (IFMRMInitialMax-IFMRMInitialMin)/IFMRincrement
IFMRtheoreticalArray = dblarr(numberOfIFMRPoints, 4)
;PRINT, "numberOfIFMRPoints = ", numberOfIFMRPoints
for IFMRIndex=0, numberOfIFMRPoints-1, 1 do begin
    InitialMassTemp = IFMRIndex*IFMRincrement + IFMRMInitialMin
    FinalMassTheoretical = IFMRm * InitialMassTemp + IFMRc
    FinalMassTheoreticalMin = (IFMRm-IFMRmErr) * InitialMassTemp + (IFMRc-
        IFMRcErr)
    FinalMassTheoreticalMax = (IFMRm+IFMRmErr) * InitialMassTemp + (IFMRc+
        IFMRcErr)
    IFMRtheoreticalArray(IFMRIndex, 0) = InitialMassTemp
    IFMRtheoreticalArray(IFMRIndex, 1) = FinalMassTheoretical
    IFMRtheoreticalArray(IFMRIndex, 2) = FinalMassTheoreticalMin
    IFMRtheoreticalArray(IFMRIndex, 3) = FinalMassTheoreticalMax
endfor

oplot, IFMRtheoreticalArray(*,0), IFMRtheoreticalArray(*,1)
oplot, IFMRtheoreticalArray(*,0), IFMRtheoreticalArray(*,2), linestyle=2
oplot, IFMRtheoreticalArray(*,0), IFMRtheoreticalArray(*,3), linestyle=2

    ;3.1/3.2

        ;massFinal  ageFinal    systemAge   massInitial ageInitial

for extremeIndex=0, numberOfExtremes-1, 1 do begin
    ;PRINT, "WDxResultsArrayA(1, 0, 2) = ", WDxResultsArrayA(1, 0, 2)
    ;PRINT, "WDxResultsArrayA = ", transpose(WDxResultsArrayA[0, *, *])
    ;PRINT, "WDxResultsArrayB = ", transpose(WDxResultsArrayB[0, *, *])

    ;PRINT, "as"

    WDxResultsArrayAOptSysAgeIdeal = [WDxResultsArrayA[extremeIndex, 0, 3],
        WDxResultsArrayA[extremeIndex, 0, 0]]
    WDxResultsArrayBOptSysAgeIdeal = [WDxResultsArrayB[extremeIndex, 0, 3],
        WDxResultsArrayB[extremeIndex, 0, 0]]

    ;PRINT, "WDxResultsArrayAOptSysAgeIdeal = ",
        WDxResultsArrayAOptSysAgeIdeal
    ;PRINT, "WDxResultsArrayBOptSysAgeIdeal = ",
        WDxResultsArrayBOptSysAgeIdeal


    if(extremeIndex eq 0) then begin
        linestyletemp = 0    ;norm

        ;if(DDIndex lt 10) then begin
        ;    DDnameformat = '(I1)'
        ;    DDindexStringFormal = '0' + STRING(DDIndex, FORMAT=DDnameformat)
        ;endif else begin
        ;    DDnameformat = '(I2)'
        ;    DDindexStringFormal = STRING(DDIndex, FORMAT=DDnameformat)
        ;endelse
        ;textA =  'DD-' + DDindexStringFormal + 'A' + ' (using theor. sys
            age)'
        ;textB =  'DD-' + DDindexStringFormal + 'B' + ' (using theor. sys
            age)'
        textA = 'A'
        textB = 'B'
        XYOUTS, WDxResultsArrayAOptSysAgeIdeal(0),
            WDxResultsArrayAOptSysAgeIdeal(1), textA, CHARSIZE=
            charSizeDefault
        XYOUTS, WDxResultsArrayBOptSysAgeIdeal(0),
            WDxResultsArrayBOptSysAgeIdeal(1), textB, CHARSIZE=
            charSizeDefault

        if(DDIndex lt 10) then begin
            DDnameformat = '(I1)'
            DDindexStringFormal = '0' + STRING(DDIndex, FORMAT=
                DDnameformat)
        endif else begin
            DDnameformat = '(I2)'
            DDindexStringFormal = STRING(DDIndex, FORMAT=DDnameformat)
```

```
            endelse
            textDD =  'DD-' + DDindexStringFormal
            XYOUTS, [6.5], [1.8], textDD, CHARSIZE=charSizeDefault*3


        endif else begin
            linestyletemp = 1    ;min/max
        endelse

        ;PRINT, "[WDxResultsArrayAOptSysAgeIdeal(0),
             WDxResultsArrayBOptSysAgeIdeal(0)] = ", [
             WDxResultsArrayAOptSysAgeIdeal(0), WDxResultsArrayBOptSysAgeIdeal
             (0)]
        ;PRINT, "[WDxResultsArrayAOptSysAgeIdeal(1),
             WDxResultsArrayBOptSysAgeIdeal(1)] = ", [
             WDxResultsArrayAOptSysAgeIdeal(1), WDxResultsArrayBOptSysAgeIdeal
             (1)]
        oplot, [WDxResultsArrayAOptSysAgeIdeal(0),
             WDxResultsArrayBOptSysAgeIdeal(0)], [WDxResultsArrayAOptSysAgeIdeal
             (1), WDxResultsArrayBOptSysAgeIdeal(1)], THICK=3, linestyle=
             linestyletemp, psym=-4, symsize=0.5

    endfor



    if(searchTypeSelectedIFMRPlot eq 2) then begin
        ;NB searchTypeSelectedIFMRPlot eq 2
        ;3.2, Part B: create IFMR graph of B based upon sys age A, and vice
             versa

        idealSystemAgeIndexForComponentB = idealSystemAgeIndexForStarArray[
             starIndex, 0]
        idealSystemAgeIndexForComponentA = idealSystemAgeIndexForStarArray[
             starIndex-1, 0]

        ;PRINT, "starIndex = ", starIndex
        ;PRINT, "idealSystemAgeIndexForComponentA = ",
             idealSystemAgeIndexForComponentA
        ;PRINT, "idealSystemAgeIndexForComponentB = ",
             idealSystemAgeIndexForComponentB

        for extremeIndex=0, numberOfExtremes-1, 1 do begin
            ;take into account min/max final mass extremes in calculations, not
                 just ideal mass

            ;PRINT, "systemAgeRangeResultsArray(starIndex, extremeIndex,
                 idealSystemAgeIndexForComponentB, *) = ",
                 systemAgeRangeResultsArray(starIndex, extremeIndex,
                 idealSystemAgeIndexForComponentB, *)

            idealSystemPropertiesForAlternateComponentB =
                 systemAgeRangeResultsArray(starIndex, extremeIndex,
                 idealSystemAgeIndexForComponentA, *)
            idealSystemPropertiesForAlternateComponentA =
                 systemAgeRangeResultsArray(starIndex-1, extremeIndex,
                 idealSystemAgeIndexForComponentB, *)

            ;PRINT, "idealSystemPropertiesForAlternateComponentA = ",
                 idealSystemPropertiesForAlternateComponentA
            ;PRINT, "idealSystemPropertiesForAlternateComponentB = ",
                 idealSystemPropertiesForAlternateComponentB

            if(extremeIndex eq 0) then begin
                linestyletemp = 0    ;norm

                ;if(DDIndex lt 10) then begin
                ;    DDnameformat = '(I1)'
                ;    DDindexStringFormal = '0' + STRING(DDIndex, FORMAT=
                     DDnameformat)
                ;endif else begin
                ;    DDnameformat = '(I2)'
                ;    DDindexStringFormal = STRING(DDIndex, FORMAT=DDnameformat)
                ;endelse
                ;textA =  'DD-' + DDindexStringFormal + 'A' + ' (using theor.
                     sys age of alt. component)'
                ;textB =  'DD-' + DDindexStringFormal + 'B' + ' (using theor.
                     sys age of alt. component)'
                textA = ' A(alt)'
                textB = ' B(alt)'
                XYOUTS, idealSystemPropertiesForAlternateComponentA(3),
                     idealSystemPropertiesForAlternateComponentA(0), textA,
                     CHARSIZE=charSizeDefault
                XYOUTS, idealSystemPropertiesForAlternateComponentB(3),
                     idealSystemPropertiesForAlternateComponentB(0), textB,
                     CHARSIZE=charSizeDefault

            endif else begin
                linestyletemp = 1    ;min/max
            endelse
```

```
                                oplot, [idealSystemPropertiesForAlternateComponentA(3),
                                    idealSystemPropertiesForAlternateComponentB(3)], [
                                    idealSystemPropertiesForAlternateComponentA(0),
                                    idealSystemPropertiesForAlternateComponentB(0)], linestyle=
                                    linestyletemp, psym=-5, symsize=0.5
                            endfor

                            ;3.2, Part A: create IFMR graph using a range of system ages [and
                                signify ideal system age, ie as determined based upon theoretical
                                IFMR]
                            ;already done (see above)

                                    ;oplot, WDxResultsArrayANorm[3, 1:*], WDxResultsArrayANorm
                                        [0, 1:*], linestyle=0
                                    ;oplot, WDxResultsArrayAMin[3, 1:*], WDxResultsArrayAMin[0,
                                        1:*], linestyle=2
                                    ;oplot, WDxResultsArrayAMax[3, 1:*], WDxResultsArrayAMax[0,
                                        1:*], linestyle=2

                                    ;oplot, WDxResultsArrayBNorm[3, 1:*], WDxResultsArrayBNorm
                                        [0, 1:*], linestyle=0
                                    ;oplot, WDxResultsArrayBMin[3, 1:*], WDxResultsArrayBMin[0,
                                        1:*], linestyle=2
                                    ;oplot, WDxResultsArrayBMax[3, 1:*], WDxResultsArrayBMax[0,
                                        1:*], linestyle=2


                                    ;WDxResultsArrayOptSysAgeIdealMi = [WDxResultsArrayBMin[3,
                                        0], WDxResultsArrayBNorm[3, 0], WDxResultsArrayBMax[3,
                                        0]]
                                    ;WDxResultsArrayAOptSysAgeIdealMf = [WDxResultsArrayAMin[0,
                                        0], WDxResultsArrayANorm[0, 0], WDxResultsArrayAMax[0,
                                        0]]
                                    ;PRINT, "WDxResultsArrayOptSysAgeIdealMi = ",
                                        WDxResultsArrayOptSysAgeIdealMi
                                    ;PRINT, "WDxResultsArrayAOptSysAgeIdealMf = ",
                                        WDxResultsArrayAOptSysAgeIdealMf

                                    ;oplot, WDxResultsArrayOptSysAgeIdealMi,
                                        WDxResultsArrayAOptSysAgeIdealMf, THICK=3, linestyle=0,
                                         psym=-1

                            endif

                        endif else begin
                            PRINT, "error: invalid searchTypeSelectedIFMRPlot option"
                            exit
                        endelse


                        if(plotToFile eq 0) then begin

                        endif else begin

                            device,/close
                            ;convert ps to pdf;
                            convertPStoPDFcommand = "convert " + outputFileNamePS + " " +
                                outputFileNamePDF
                            spawn,convertPStoPDFcommand,result

                        endelse

                        ;PRINT, "as2"
                        WDxResultsArrayA = WDxResultsArrayB
                        WDxResultsArrayANorm = WDxResultsArrayBNorm
                        WDxResultsArrayAMin = WDxResultsArrayBMin
                        WDxResultsArrayAMax = WDxResultsArrayBMax
                        ;PRINT, "as3"
                    endif

                endfor
            endif else begin
                PRINT, "closestMassInitial = "
                combinationResultsArrayNorm = combinationResultsArray[*,0,*]
                combinationResultsArrayMin = combinationResultsArray[*,1,*]
                combinationResultsArrayMax = combinationResultsArray[*,2,*]
                PRINT, [combinationResultsArrayNorm, combinationResultsArrayMin,
                    combinationResultsArrayMax]
            endelse
        endif else if(searchTypeSelectedIFMR eq 1) then begin
            if(systemAgeOptimisationFast eq 0) then begin
                for starIndex=0, numberOfStars-1, 1 do begin
                    if((starIndex mod 2) ne 0) then begin
                        for extremeIndex=0, numberOfExtremes-1, 1 do begin
                            currentClosestSystemAgeDiffTemp = 10e20

                            for mIndex=0, numberOfMIndicies-1, 1 do begin
                                for cIndex=0, numberOfCIndicies-1, 1 do begin

                                    idealSystemPropertiesForComponentA =
                                        systemAgeRangeArrayForIFMRParameters[starIndex-1, extremeIndex,
                                         mIndex, cIndex, *]
```

```
                                idealSystemPropertiesForComponentB =
                                    systemAgeRangeArrayForIFMRParameters[starIndex, extremeIndex,
                                    mIndex, cIndex, *]

                                ;PRINT, "idealSystemPropertiesForComponentA = ",
                                    idealSystemPropertiesForComponentA
                                ;PRINT, "idealSystemPropertiesForComponentB = ",
                                    idealSystemPropertiesForComponentB
                                ;PRINT, "idealSystemPropertiesForComponentA[2] = ",
                                    idealSystemPropertiesForComponentA[2]
                                ;PRINT, "idealSystemPropertiesForComponentB[2] = ",
                                    idealSystemPropertiesForComponentB[2]
                                currentSystemAgeDiff = abs(idealSystemPropertiesForComponentA[2] -
                                    idealSystemPropertiesForComponentB[2])
                                ;PRINT, "currentSystemAgeDiff = ", currentSystemAgeDiff

                                if(currentSystemAgeDiff lt currentClosestSystemAgeDiffTemp) then
                                    begin
                                    currentClosestSystemAgeDiffTemp = currentSystemAgeDiff
                                    idealIFMRpropertiesForStarArray[starIndex, extremeIndex, 0] =
                                        mIndex
                                    idealIFMRpropertiesForStarArray[starIndex, extremeIndex, 1] =
                                        cIndex
                                    idealSystemAgeForStarArray[starIndex, extremeIndex] =
                                        idealSystemPropertiesForComponentA[2]      ; record system
                                        age for error checking only
                                    ;PRINT, "currentClosestSystemAgeDiffTemp= ",
                                        currentClosestSystemAgeDiffTemp
                                endif

                                ;temp work out of c convergence is independent of m convergence [no
                                    they are dependent, ie c will converge at a different spot,
                                    depending upon m {which is expected}]:
                                    ;so try to achieve convergence by using a coarse grid, then fine
                                        grid?
                                ;DDIndex = (starIndex/2)+1
                                ;if((extremeIndex eq 0) and (mIndex eq 0) and (DDIndex eq 11)) then
                                    begin
                                ;   currentIFMRPropertiesIndexTemp = (mIndex*numberOfCIndicies +
                                    cIndex)
                                ;   cIdealTemp = IFMRParameterOptimisationMinC + cIndex*
                                    IFMRParameterOptimisationIncrementC
                                ;   PRINT, "[cIndex, currentSystemAgeDiff,
                                    currentIFMRPropertiesIndexTemp, cIdealTemp] = "
                                ;   PRINT, [cIndex, currentSystemAgeDiff,
                                    currentIFMRPropertiesIndexTemp, cIdealTemp]
                                ;endif
                            endfor
                        endfor

                        ;PRINT, "currentClosestSystemAgeDiffTemp = ",
                            currentClosestSystemAgeDiffTemp

                        ;For PRINT/display purposes only
                        if((systemAgeCommonArray[starIndex] eq 1) or (
                            onlyUseSystemsWithCommonSystemAgeDuringIFMRparameterOptimisation eq 0))
                             then begin
                            if(extremeIndex eq 0) then begin
                                mIndexIdeal = idealIFMRpropertiesForStarArray[starIndex,
                                    extremeIndex, 0]
                                cIndexIdeal = idealIFMRpropertiesForStarArray[starIndex,
                                    extremeIndex, 1]
                                mIdeal = IFMRParameterOptimisationMinM + mIndexIdeal*
                                    IFMRParameterOptimisationIncrementM
                                cIdeal = IFMRParameterOptimisationMinC + cIndexIdeal*
                                    IFMRParameterOptimisationIncrementC

                                    ;PRINT, "[starIndex, extremeIndex] =", [starIndex, extremeIndex]
                                ;PRINT, "[mIndexIdeal, cIndexIdeal, mIndexIdeal, cIndexIdeal] = "
                                PRINT, [mIndexIdeal, cIndexIdeal, mIdeal, cIdeal]
                            endif
                        endif

                    endfor
                endif
            endfor
        endif
endif else if(searchTypeSelectedIFMR eq 2) then begin


    if(systemAgePlotType eq 0) then begin
        fileNameBase1 = 'SDSSDDsystemAgeConfirmationForPublishedIFMR'
        xRangeMax = 15000
        if(useGDL) then begin
            xtitleTemp = 't sys (My)'    ;GDL only
        endif else begin
            xtitleTemp = 't!Dsys!N (My)'     ;IDL only
        endelse
    endif else begin
        fileNameBase1 = 'SDSSDDsystemAgeConfirmationForPublishedIFMR-log'
        xRangeMax = 12
        if(useGDL) then begin
```

```
        xtitleTemp = 'log(t sys)'    ;GDL only
    endif else begin
        xtitleTemp = 'log(t!Dsys!N)'     ;GDL only
    endelse
endelse

derivedCommonSystemAgeArray = lonarr(numberOfPublshedIFMRs, numberOfStars)
derivedCommonSystemAgeTotal = lonarr(numberOfPublshedIFMRs)
derivedCommonSystemAgeTotalExcludingAssumedDiscrepancies = lonarr(numberOfPublshedIFMRs)

for IFMRIndex=0, numberOfPublshedIFMRs-1, 1 do begin

    derivedCommonSystemAgeTotal[IFMRIndex] = 0
    derivedCommonSystemAgeTotalExcludingAssumedDiscrepancies[IFMRIndex] = 0
    numberOfAssumedSystemAgeDiscrepancies = 0

    IFMRIndexString = STRTRIM(IFMRIndex, 2)

    if(plotToFile eq 0) then begin

        plot, tempXArray, tempYArray, title=plotTitle, xtitle=xtitleTemp, ytitle='SDSS DD
            system (1 to 12 respectively)', xrange=[0, xRangeMax], yrange=[((numberOfStars
            +1)/2.0)+0.5, 0.0]

    endif else begin


        psGraphfileNameExtension = '.ps'
        pdfGraphfileNameExtension = '.pdf'
        psGraphfileName = fileNameBase1 + IFMRIndexString + psGraphfileNameExtension
        pdfGraphfileName = fileNameBase1 + IFMRIndexString + pdfGraphfileNameExtension

        set_plot,'ps'
        device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE      ;
            SET_RESOLUTION=[320,200]    ;SET_CHARACTER_SIZE=[200,40]

        if(useGDL) then begin
            ;GDL;
            plot, tempXArray, tempYArray, title='', xtitle=xtitleTemp, ytitle='SDSS DD
                system (1 to 12, PG+0922, HS2240)', xrange=[0, xRangeMax], yrange=[((
                numberOfStars+1)/2.0)+0.5, 0.0], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=32,
                YMARGIN=18
        endif else begin
            ;IDL old;
            plot, tempXArray, tempYArray, title='', xtitle=xtitleTemp, ytitle='SDSS DD
                system (1 to 12, PG+0922, HS2240)', xrange=[0, xRangeMax], yrange=[((
                numberOfStars+1)/2.0)+0.5, 0.0]
        endelse
    endelse


    sysAgePrevious = dblarr(3)
    for starIndex=0L, numberOfStars-1, 1L do begin

            sysAge = dblarr(3)
            y = dblarr(3)

        sysAge[1] = systemAgeRangeArrayForPublishedIFMRs[starIndex, 0, IFMRIndex, 2]
        sysAgeMinTemp = systemAgeRangeArrayForPublishedIFMRs[starIndex, 1, IFMRIndex, 2]
        sysAgeMaxTemp = systemAgeRangeArrayForPublishedIFMRs[starIndex, 2, IFMRIndex, 2]
        sysAge[0] = MIN([sysAgeMinTemp, sysAgeMaxTemp])
        sysAge[2] = MAX([sysAgeMinTemp, sysAgeMaxTemp])

        ;PRINT, "sysAge = ", sysAge[1]

        if((starIndex mod 2) eq 0) then begin
            y[1] = (starIndex/2.0)+1.0-0.05
            colour = 0
        endif else begin
            y[1] = (starIndex/2.0)+0.7-0.05
            colour = 1
        endelse

        y[0] = y[1]
        y[2] = y[1]

        ;PRINT, "sysAge = ", sysAge
        ;PRINT, "y = ", y

        if(systemAgePlotType eq 0) then begin
            sysAgeArrayForPlot = sysAge/1e6
        endif else begin
            sysAgeArrayForPlot = alog10(sysAge/1e6)
        endelse

        ;PRINT, "sysAgeArrayForPlot = ", sysAgeArrayForPlot

        oplot, sysAgeArrayForPlot, y, thick=2
            ;if GDL only; COLOR=1

        ;PRINT common system ages
        if((starIndex mod 2) ne 0) then begin
```

```
                    if(sysAgePrevious[0] lt sysAge[0]) then begin
                        if(sysAgePrevious[2] gt sysAge[0]) then begin
                            commonSystemAge = 1
                        endif else begin
                            commonSystemAge = 0
                        endelse
                    endif else begin
                        if(sysAge[2] gt sysAgePrevious[0]) then begin
                            commonSystemAge = 1
                        endif else begin
                            commonSystemAge = 0
                        endelse
                    endelse

                    if(systemAgeCommonArray[starIndex] eq 1) then begin
                        derivedCommonSystemAgeTotalExcludingAssumedDiscrepancies[IFMRIndex] =
                             derivedCommonSystemAgeTotalExcludingAssumedDiscrepancies[IFMRIndex] +
                             1
                    endif else begin
                        numberOfAssumedSystemAgeDiscrepancies =
                             numberOfAssumedSystemAgeDiscrepancies + 1
                    endelse

                    derivedCommonSystemAgeTotal[IFMRIndex] = derivedCommonSystemAgeTotal[IFMRIndex]
                         + commonSystemAge
                    derivedCommonSystemAgeArray[IFMRIndex, starIndex] = commonSystemAge
                    derivedCommonSystemAgeArray[IFMRIndex, starIndex-1] = commonSystemAge
                endif

                sysAgePrevious[0] = sysAge[0]
                sysAgePrevious[1] = sysAge[1]
                sysAgePrevious[2] = sysAge[2]

            endfor

            if(plotToFile eq 0) then begin

            endif else begin

                device,/close
                ;convert ps to pdf;
                convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
                spawn,convertPStoPDFcommand,result

            endelse
        endfor

        PRINT, "derivedCommonSystemAgeArray = "
        PRINT, derivedCommonSystemAgeArray
        PRINT, "derivedCommonSystemAgeTotal = "
        PRINT, derivedCommonSystemAgeTotal
        PRINT, "numberOfAssumedSystemAgeDiscrepancies = "
        PRINT, numberOfAssumedSystemAgeDiscrepancies
        PRINT, "derivedCommonSystemAgeTotalExcludingAssumedDiscrepancies = "
        PRINT, derivedCommonSystemAgeTotalExcludingAssumedDiscrepancies

    endif else begin
        PRINT, "searchTypeSelectedBasic = invalid"
        exit
    endelse
endif else begin
    PRINT, "searchTypeSelectedBasic = invalid"
    exit
endelse

if(searchTypeSelectedBasic eq 3) then begin
    if(searchTypeSelectedIFMR eq 1) then begin
        IFMRaverageIdealPropertiesArray = DBLARR(3, 2)

        outputFileNamePDF= 'DDsystemAgeOptimisation' + '.pdf'
        outputFileNamePS= 'DDsystemAgeOptimisation' + '.ps'

        if(plotToFile eq 0) then begin

            plot, tempXArray, tempYArray, title='', xtitle='M initial', ytitle='M final', xrange=[
                 minIFMRmInitial, maxIFMRmInitial], yrange=[minIFMRmFinal, maxIFMRmFinal]

        endif else begin

            set_plot,'ps'
            device,filename=outputFileNamePS, /LANDSCAPE ;/PORTRAIT /LANDSCAPE ;
                 SET_RESOLUTION=[320,200] ;SET_CHARACTER_SIZE=[200,40]

            if(useGDL) then begin
                plot, tempXArray, tempYArray, title='', xtitle='M initial', ytitle='M final', xrange
                     =[minIFMRmInitial, maxIFMRmInitial], yrange=[minIFMRmFinal, maxIFMRmFinal],
                     BACKGROUND=1, CHARSIZE=2.0,XMARGIN=32,YMARGIN=18
            endif else begin
                plot, tempXArray, tempYArray, title='', xtitle='M initial', ytitle='M final', xrange
                     =[minIFMRmInitial, maxIFMRmInitial], yrange=[minIFMRmFinal, maxIFMRmFinal]
            endelse
```

```
endelse


;maxStarIndex, mIndexMax, cIndexMax, WDxResultsArrayNumberOfResults
for extremeIndex=0, numberOfExtremes-1, 1 do begin
    ;PRINT, "extremeIndex = ", extremeIndex
    mIdealTotal = 0.0
    cIdealTotal = 0.0
    numberOfSystemComponentsWithCommonSystemAge = 0
    for starIndex=0, numberOfStars-1, 1 do begin
        if((starIndex mod 2) ne 0) then begin
            if((systemAgeCommonArray[starIndex] eq 1) or (
                onlyUseSystemsWithCommonSystemAgeDuringIFMRparameterOptimisation eq 0))
                then begin

                mIndexIdeal = idealIFMRpropertiesForStarArray[starIndex, extremeIndex, 0]
                cIndexIdeal = idealIFMRpropertiesForStarArray[starIndex, extremeIndex, 1]
                mIdeal = IFMRParameterOptimisationMinM + mIndexIdeal*
                    IFMRParameterOptimisationIncrementM
                cIdeal = IFMRParameterOptimisationMinC + cIndexIdeal*
                    IFMRParameterOptimisationIncrementC
                PRINT, "[mIdeal, cIdeal] = ", [mIdeal, cIdeal]
                mIdealTotal = mIdealTotal+mIdeal
                cIdealTotal = cIdealTotal+cIdeal

                if(extremeIndex eq 0) then begin
                    linestyletemp = 0    ;norm
                    charsizetemp = charSizeDefault*0.75
                endif else begin
                    linestyletemp = 1    ;min/max
                    charsizetemp = charSizeDefault*0.5
                endelse

                mFinalTempDDA  = finalMassArray[extremeIndex, starIndex-1]
                mInitialTempDDA = (mFinalTempDDA-cIdeal)/mIdeal
                mFinalTempDDB = finalMassArray[extremeIndex, starIndex]
                mInitialTempDDB = (mFinalTempDDB-cIdeal)/mIdeal
                oplot, [mInitialTempDDA, mInitialTempDDB] , [mFinalTempDDA, mFinalTempDDB],
                    linestyle=linestyletemp


                DDIndex = (starIndex/2)+1
                if(DDIndex lt 10) then begin
                    DDnameformat = '(I1)'
                    DDindexStringFormal = '0' + STRING(DDIndex, FORMAT=DDnameformat)
                endif else begin
                    DDnameformat = '(I2)'
                    DDindexStringFormal = STRING(DDIndex, FORMAT=DDnameformat)
                endelse

                if(((mInitialTempDDA gt minIFMRmInitial) and (mInitialTempDDB gt
                    minIFMRmInitial)) and $
                ((mInitialTempDDA lt maxIFMRmInitial) and (mInitialTempDDB lt
                    maxIFMRmInitial)) and $
                ((mFinalTempDDA gt minIFMRmFinal) and (mFinalTempDDB gt minIFMRmFinal)) and
                    $
                ((mFinalTempDDA lt maxIFMRmFinal) and (mFinalTempDDB lt maxIFMRmFinal)))
                    then begin
                    plotRangeOK = 1
                endif else begin
                    plotRangeOK = 0
                endelse

                if(plotRangeOK) then begin

                    if(extremeIndex eq 0) then begin        ;Optional

                        textA =  DDindexStringFormal + 'A'  ;'DD-' +
                        textB =  DDindexStringFormal + 'B'  ;'DD-' +
                        XYOUTS, mInitialTempDDA, mFinalTempDDA, textA, CHARSIZE=charsizetemp
                        XYOUTS, mInitialTempDDB, mFinalTempDDB, textB, CHARSIZE=charsizetemp

                        if(overlaySystemAgeIndiciesForDebug eq 1) then begin

                            ;print system age - for error checking purposes only;
                            IFMRPropertiesIndexIdeal = (mIndexIdeal*numberOfCIndicies +
                                cIndexIdeal)
                            IFMRPropertiesIndexIdealString = STRING(IFMRPropertiesIndexIdeal
                                , FORMAT='(I5)')
                            IFMRPropertiesCIdealString = STRING(cIdeal, FORMAT='(F10.4)')
                            ;IFMRPropertiesMIdealString = STRING(mIdeal, FORMAT='(e11.5)')
                            ;IFMRPropertiesCIndexIdealString = STRING(cIndexIdeal, FORMAT='(
                                I10)')
                            ;IFMRPropertiesMIndexIdealString = STRING(mIndexIdeal, FORMAT='(
                                I10)')
                            ;systemAgeIdeal = idealSystemAgeForStarArray[starIndex,
                                extremeIndex]
                            ;systemAgeIdealString = STRING(systemAgeIdeal, FORMAT='(e11.5)')

                            centralDisplayString = IFMRPropertiesIndexIdealString + ' ' +
                                IFMRPropertiesCIdealString    ;
                                IFMRPropertiesIndexIdealString ;systemAgeIdealString
```

```
                                ;May be temporarily removed:
                                XYOUTS, (mInitialTempDDA+mInitialTempDDB)/2.0 - 0.5, (
                                    mFinalTempDDA+mFinalTempDDB)/2.0, centralDisplayString,
                                    CHARSIZE=charsizetemp

                            endif

                        endif else begin
                            ;May be temporarily removed:
                            XYOUTS, (mInitialTempDDA+mInitialTempDDB)/2.0, (mFinalTempDDA+
                                mFinalTempDDB)/2.0, DDindexStringFormal, CHARSIZE=charsizetemp
                        endelse

                    endif


                    numberOfSystemComponentsWithCommonSystemAge =
                        numberOfSystemComponentsWithCommonSystemAge + 1
                endif
            endif
        endfor
        mIdealAverage = mIdealTotal/numberOfSystemComponentsWithCommonSystemAge
        cIdealAverage = cIdealTotal/numberOfSystemComponentsWithCommonSystemAge

        IFMRaverageIdealPropertiesArray[extremeIndex, 0] = mIdealAverage
        IFMRaverageIdealPropertiesArray[extremeIndex, 1] = cIdealAverage

        ;PRINT, "ideal IFMR parameters to achieve common system ages of DD components, extreme
            index =", extremeIndex
        ;PRINT, "mAverage = ", mAverage
        ;PRINT, "cAverage = ", cAverage
    endfor

    ;for starIndex=0, numberOfStars-1, 1 do begin
    ;    if((starIndex mod 2) ne 0) then begin
    ;        if((systemAgeCommonArray[starIndex] eq 1) or (
        onlyUseSystemsWithCommonSystemAgeDuringIFMRparameterOptimisation eq 0)) then begin
    ;
    ;        endif
    ;    endif
    ;endif



    ;IFMR Plot;
    IFMRincrement = 0.01
    numberOfIFMRPoints = (IFMRMInitialMax-IFMRMInitialMin)/IFMRincrement
    IFMRtheoreticalArray = dblarr(numberOfIFMRPoints, 4)
    IFMRBayesianArray = dblarr(numberOfIFMRPoints, 4)
    ;PRINT, "numberOfIFMRPoints = ", numberOfIFMRPoints
    for IFMRIndex=0, numberOfIFMRPoints-1, 1 do begin
        InitialMassTemp = IFMRIndex*IFMRincrement + IFMRMInitialMin

        FinalMassTheoretical = IFMRm * InitialMassTemp + IFMRc
        FinalMassTheoreticalMin = (IFMRm-IFMRmErr) * InitialMassTemp + (IFMRc-IFMRcErr)
        FinalMassTheoreticalMax = (IFMRm+IFMRmErr) * InitialMassTemp + (IFMRc+IFMRcErr)
        IFMRtheoreticalArray(IFMRIndex, 0) = InitialMassTemp
        IFMRtheoreticalArray(IFMRIndex, 1) = FinalMassTheoretical
        IFMRtheoreticalArray(IFMRIndex, 2) = FinalMassTheoreticalMin
        IFMRtheoreticalArray(IFMRIndex, 3) = FinalMassTheoreticalMax

        FinalMassBayesian = IFMRaverageIdealPropertiesArray[0, 0] * InitialMassTemp +
            IFMRaverageIdealPropertiesArray[0, 1]
        FinalMassBayesianMin = IFMRaverageIdealPropertiesArray[1, 0] * InitialMassTemp +
            IFMRaverageIdealPropertiesArray[1, 1]
        FinalMassBayesianMax = IFMRaverageIdealPropertiesArray[2, 0] * InitialMassTemp +
            IFMRaverageIdealPropertiesArray[2, 1]
        IFMRBayesianArray(IFMRIndex, 0) = InitialMassTemp
        IFMRBayesianArray(IFMRIndex, 1) = FinalMassBayesian
        IFMRBayesianArray(IFMRIndex, 2) = FinalMassBayesianMin
        IFMRBayesianArray(IFMRIndex, 3) = FinalMassBayesianMax

    endfor

    oplot, IFMRtheoreticalArray(*,0), IFMRtheoreticalArray(*,1)
    oplot, IFMRtheoreticalArray(*,0), IFMRtheoreticalArray(*,2), linestyle=2
    oplot, IFMRtheoreticalArray(*,0), IFMRtheoreticalArray(*,3), linestyle=2

    oplot, IFMRBayesianArray(*,0), IFMRBayesianArray(*,1)
    oplot, IFMRBayesianArray(*,0), IFMRBayesianArray(*,2), linestyle=2
    oplot, IFMRBayesianArray(*,0), IFMRBayesianArray(*,3), linestyle=2


    PRINT, "IFMRaverageIdealPropertiesArray = "
    PRINT, IFMRaverageIdealPropertiesArray
    ;PRINT, transpose(IFMRaverageIdealPropertiesArray)

    if(plotToFile eq 0) then begin

    endif else begin
```

```
            device,/close
            ;convert ps to pdf;
            convertPStoPDFcommand = "convert " + outputFileNamePS + " " + outputFileNamePDF
            spawn,convertPStoPDFcommand,result

        endelse
    endif
endif


end
```

A.6.2 compare DD System Age Difference Distribution Across Published IFMRs

This software is used to compare the DD System Age Difference Distribution Across Published IFMRs (and creates monte carlo distributions of expected component [WD/-final and MS/initial] combinations for each DD system based on final mass/age errors).

compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS.pro

compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS-01Aug11a.zip

```
pro compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS


; Title: compare DD System Age Difference Distribution Across Published IFMRs
; File: compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 09 July 11 10:00am [updated 14 July 11a 1am]
; Summary:
;
;   Postconditions;
;
;   This program compares the DD System Age Difference Distribution Across Published IFMRs (and
    creates monte carlo distributions of expected component [WD/final and MS/initial] combinations
    for each DD system based on final mass/age errors)
;
;   Preconditions;
;
;   compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs.pro has been executed
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   WDinputParameters.txt [name logg    logg error  temp    temp error   observed u_mag  observed
    g_mag  observed r_mag  observed i_mag  observed z_mag  observed u_mag error   observed g_mag
    error     observed r_mag error    observed i_mag error    observed z_mag error   Special Case (
    for the purpose of mass/age SW calculations)]
;
;   Usage GDL [linear interpolation only];
;
;   gdl
;   .comp compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS.pro
;   compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS
;
;   Usage IDL:
;   mkdir /home/aaossz/rbaxter/idllib/astron
;   cd /home/aaossz/rbaxter/idllib/astron
;   tar -xvf astron.tar.gz
;   idl
;   !PATH = Expand_Path('+/home/aaossz/rbaxter/idllib/astron/pro') + ':' + !PATH
;   PRINT, !PATH
;   .comp compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS.pro
;   compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS
;
```

```
common dataLanguage,useGDL

common wd_masses,object_mass,object_cool          ;overdeclared - should be replaced with below
     variable names '...final...'
common wd2_mass_and_cool,companion_mass,companion_cool  ;overdeclared - should be replaced with
     below variable names '...final...'

common object_1_final_mass_and_cool,object_final_mass_1,object_final_age_1
common object_2_final_mass_and_cool,object_final_mass_2,object_final_age_2
common object_1_initial_mass_and_cool,object_initial_mass_1,object_initial_age_1
common object_2_initial_mass_and_cool,object_initial_mass_2,object_initial_age_2
common object_commom_system_age_diff,object_system_age_diff

runTestOfCodeOnly = 0

useGDL = 0

WDinputParametersFileName = "WDinputParameters.txt"
PRINT, "WDinputParametersFileName = ", WDinputParametersFileName
readcol64bitvars, WDinputParametersFileName, Name, surface_gravity, surface_gravity_err,
     effective_temp, effective_temp_err, observed_u_magnitude, observed_g_magnitude,
     observed_r_magnitude, observed_i_magnitude, observed_z_magnitude, observed_u_magnitude_err,
     observed_g_magnitude_err, observed_r_magnitude_err, observed_i_magnitude_err,
     observed_z_magnitude_err, specialCase, FORMAT='A,D,D,D,D,D,D,D,D,D,D,D,D,D,A', SKIPLINE=1

inputTableSize = SIZE(surface_gravity)
if(runTestOfCodeOnly eq 1) then begin
    numberOfWDs = 2      ;inputTableSize[1]       ;2
    numberOfPublishedIFMRs = 1
    realisations=2501
endif else begin
    numberOfWDs = inputTableSize[1]
    numberOfPublishedIFMRs = 3
    realisations=25001   ;temp: 2500
endelse

numberOfOutputParameters=5  ;final mass and age (WD cooling time), initial mass and age, and system
     age diff
WDparametersOutput=dblarr(numberOfPublishedIFMRs, numberOfWDs, realisations,
     numberOfOutputParameters)

observed_magnitude_band='g'
observed_magnitude = observed_g_magnitude
observed_magnitude_err = observed_g_magnitude_err
pb=1
CASE observed_magnitude_band OF
'V':pb=7
'g':pb=14
'r':pb=15
'i':pb=16
ENDCASE


for WDindex=0, numberOfWDs-1, 1 do begin

    if((WDindex mod 2) ne 0) then begin

        systemHasANoLoggComponent = -1
        indexOfNormComponent = -1
        indexOfNormComponentB = -1
        indexOfNoLoggComponent = -1
        indexOfComponent1 = -1
        indexOfComponent2 = -1
        if(specialCase[WDindex] eq 'nologg') then begin
            if(specialCase[WDindex-1] eq 'norm') then begin
                indexOfNormComponent = WDindex-1
                indexOfNoLoggComponent = WDindex
                systemHasANoLoggComponent = 1
                indexOfComponent1 = indexOfNormComponent
                indexOfComponent2 = indexOfNoLoggComponent
            endif else begin
                PRINT, "invalid (NB specialCase[WDindex] = nologg) specialCase[WDindex-1] = ",
                     specialCase[WDindex-1]
                exit
            endelse
        endif else if(specialCase[WDindex-1] eq 'nologg') then begin
            if(specialCase[WDindex] eq 'norm') then begin
                indexOfNormComponent = WDindex
                indexOfNoLoggComponent = WDindex-1
                systemHasANoLoggComponent = 1
                indexOfComponent1 = indexOfNormComponent
                indexOfComponent2 = indexOfNoLoggComponent
            endif else begin
                PRINT, "invalid specialCase[WDindex]: ", specialCase[WDindex]
                exit
            endelse
        endif else if(specialCase[WDindex] eq 'norm') then begin
            if(specialCase[WDindex-1] eq 'norm') then begin
                indexOfNormComponent = WDindex-1
                indexOfNormComponentB = WDindex
```

```
        systemHasANoLoggComponent = 0
        indexOfComponent1 = indexOfNormComponent
        indexOfComponent2 = indexOfNormComponentB
    endif else begin
        PRINT, "invalid specialCase[WDindex-1]:", specialCase[WDindex-1]
        exit
    endelse
endif else begin
    PRINT, "invalid specialCase[WDindex]:", specialCase[WDindex]
    exit
endelse

SET_PLOT, 'PS'

;Set the PostScript device to *8* bits per color, not 24:

DEVICE, FILE='24bit.ps'

    ;wd1507={binary}
    ;wd1507.realisations=25000
    ;wd1507.observed_magnitude_band='g'
    ;wd1507.observed_magnitude_DA=          16.91
    ;wd1507.observed_magnitude_err_DA=       0.04
    ;wd1507.effective_temp_DA=           17622.0
    ;wd1507.effective_temp_err_DA=         406.0
    ;wd1507.surface_gravity_DA=            8.13
    ;wd1507.surface_gravity_err_DA=        0.07
    ;wd1507.effective_temp_DAH=          18000.0
    ;wd1507.effective_temp_err_DAH=       1000.0
    ;wd1507.approx_surf_gravity_DAH=        8.75
    ;wd1507.observed_r_magnitude_DAH=      18.06
    ;wd1507.observed_r_magnitude_err_DAH=  0.03
    ;wd1507.observed_i_magnitude_DAH=      18.33
    ;wd1507.observed_i_magnitude_err_DAH=  0.03
    ;wd1507.observed_z_magnitude_DAH=      18.55
    ;wd1507.observed_z_magnitude_err_DAH=  0.04

;effective_temp_A=           effective_temp[indexOfNormComponent]
;effective_temp_err_A=         effective_temp_err[indexOfNormComponent]
;surface_gravity_A=             surface_gravity[indexOfNormComponent]
;surface_gravity_err_A=         surface_gravity_err[indexOfNormComponent]
;observed_g_magnitude_A=      observed_g_magnitude[indexOfNormComponent]
;observed_g_magnitude_err_A=   observed_g_magnitude_err[indexOfNormComponent]

; produce two arrays, containing normally distributed teff and logg

generate_wds,realisations,effective_temp[indexOfNormComponent], $
effective_temp_err[indexOfNormComponent],surface_gravity[indexOfNormComponent], $
surface_gravity_err[indexOfNormComponent]

; determine the distance modulus distribution

abmag_dist,realisations,observed_magnitude[indexOfNormComponent], $
observed_magnitude_err[indexOfNormComponent],pb

; determine the mass and cooling time of the primary WD

massandcool,realisations

object_final_mass_1 = object_mass
object_final_age_1 = object_cool

if(systemHasANoLoggComponent eq 0) then begin

    ;effective_temp_B=           effective_temp[indexOfNormComponentB]
    ;effective_temp_err_B=        effective_temp_err[indexOfNormComponentB]
    ;surface_gravity_B=              surface_gravity[indexOfNormComponentB]
    ;surface_gravity_err_B=          surface_gravity_err[indexOfNormComponentB]
    ;observed_g_magnitude_B=      observed_g_magnitude[indexOfNormComponentB]
    ;observed_g_magnitude_err_B=   observed_g_magnitude_err[indexOfNormComponentB]

    ; produce two arrays, containing normally distributed teff and logg

    generate_wds,realisations,effective_temp[indexOfNormComponentB], $
    effective_temp_err[indexOfNormComponentB],surface_gravity[indexOfNormComponentB], $
    surface_gravity_err[indexOfNormComponentB]

    ; determine the distance modulus distribution

    abmag_dist,realisations,observed_magnitude[indexOfNormComponentB], $
    observed_magnitude_err[indexOfNormComponentB],pb

    ; determine the mass and cooling time of the primary WD

    massandcool,realisations

    object_final_mass_2 = object_mass
    object_final_age_2 = object_cool

endif else begin

    ;effective_temp_B=           effective_temp[indexOfNoLoggComponent]
```

```
            ;effective_temp_err_B=       effective_temp_err[indexOfNoLoggComponent]
            ;approx_surf_gravity_B=        surface_gravity[indexOfNoLoggComponent]
            ;observed_r_magnitude_B=      observed_r_magnitude[indexOfNoLoggComponent]
            ;observed_r_magnitude_err_B=  observed_r_magnitude_err[indexOfNoLoggComponent]
            ;observed_i_magnitude_B=      observed_i_magnitude[indexOfNoLoggComponent]
            ;observed_i_magnitude_err_B=  observed_i_magnitude_err[indexOfNoLoggComponent]
            ;observed_z_magnitude_B=      observed_z_magnitude[indexOfNoLoggComponent]
            ;observed_z_magnitude_err_B=  observed_z_magnitude_err[indexOfNoLoggComponent]

            ; estimate the radius of a companion white dwarf based on distance,
            ; observed magnitude and spectroscopic effective temperature

            radius_dist,realisations,effective_temp[indexOfNoLoggComponent], $
            effective_temp_err[indexOfNoLoggComponent], $
            surface_gravity[indexOfNoLoggComponent], $
            observed_r_magnitude[indexOfNoLoggComponent], $
            observed_r_magnitude_err[indexOfNoLoggComponent], $
            observed_i_magnitude[indexOfNoLoggComponent], $
            observed_i_magnitude_err[indexOfNoLoggComponent], $
            observed_z_magnitude[indexOfNoLoggComponent], $
            observed_z_magnitude_err[indexOfNoLoggComponent]

            ; determine the mass from the radius and the mass-radius relations of
            ; Fontaine et al. (2001)

            mass_from_radius,realisations

            ; determine the cooling time based on the evolutionary models of Fontaine et
            ; al. (2001)

            cool_from_mass,realisations

            ;
            ;  determine the initial masses of the primary and secondary
            ;  components assuming canonical single star evolution and various
            ;  IFMRs from the recent literature
            ;

            object_final_mass_2 = companion_mass
            object_final_age_2 = companion_cool

        endelse

        for IFMRindex=0, numberOfPublishedIFMRs-1, 1 do begin

            total_lifetimes,realisations,IFMRindex

            WDparametersOutput[IFMRindex, indexOfComponent1, *, 0] = object_final_mass_1
            WDparametersOutput[IFMRindex, indexOfComponent1, *, 1] = object_final_age_1
            WDparametersOutput[IFMRindex, indexOfComponent1, *, 2] = object_initial_mass_1
            WDparametersOutput[IFMRindex, indexOfComponent1, *, 3] = object_initial_age_1
            WDparametersOutput[IFMRindex, indexOfComponent1, *, 4] = object_system_age_diff

            WDparametersOutput[IFMRindex, indexOfComponent2, *, 0] = object_final_mass_2
            WDparametersOutput[IFMRindex, indexOfComponent2, *, 1] = object_final_age_2
            WDparametersOutput[IFMRindex, indexOfComponent2, *, 2] = object_initial_mass_2
            WDparametersOutput[IFMRindex, indexOfComponent2, *, 3] = object_initial_age_2
            WDparametersOutput[IFMRindex, indexOfComponent2, *, 4] = object_system_age_diff
        endfor

        DEVICE, /CLOSE

    endif
endfor



;PRINT, "WDparametersOutput[0,*,*,0] = ", WDparametersOutput[0,*,*,0]
;PRINT, "WDparametersOutput[0,*,*,1] = ", WDparametersOutput[0,*,*,1]
;PRINT, "WDparametersOutput[0,*,*,2] = ", WDparametersOutput[0,*,*,2]
;PRINT, "WDparametersOutput[0,*,*,3] = ", WDparametersOutput[0,*,*,3]
;PRINT, "WDparametersOutput[0,*,*,4] = ", WDparametersOutput[0,*,*,4]

for IFMRindex=0, numberOfPublishedIFMRs-1, 1 do begin

    IFMRindexString = STRTRIM(IFMRindex, 2)

    WDparametersOutputFileNameFinalMass = "IFMR" + IFMRindexString + "WDoutputParameterFinalMass.txt
        "
    WDparametersOutputFileNameFinalAge = "IFMR" + IFMRindexString + "WDoutputParameterFinalAge.txt"
    WDparametersOutputFileNameInitialMass = "IFMR" + IFMRindexString + "WDoutputParameterInitialMass
        .txt"
    WDparametersOutputFileNameInitialAge = "IFMR" + IFMRindexString + "WDoutputParameterInitialAge.
        txt"
    WDparametersOutputFileNameSystemAgeDiff = "IFMR" + IFMRindexString + "
        WDoutputParameterSystemAgeDiff.txt"

    PRINT, "WDparametersOutputFileNameFinalMass = ", WDparametersOutputFileNameFinalMass
    PRINT, "WDparametersOutputFileNameFinalAge = ", WDparametersOutputFileNameFinalAge
    PRINT, "WDparametersOutputFileNameInitialMass = ", WDparametersOutputFileNameInitialMass
    PRINT, "WDparametersOutputFileNameInitialAge = ", WDparametersOutputFileNameInitialAge
    PRINT, "WDparametersOutputFileNameSystemAgeDiff = ", WDparametersOutputFileNameSystemAgeDiff
```

```
if(useGDL eq 1) then begin

    if(runTestOfCodeOnly eq 1) then begin
        OPENW,1,WDparametersOutputFileNameFinalMass
        PRINTF,1,WDparametersOutput[IFMRindex,*,*,0],FORMAT='(E10.3, 1X, E10.3)'
        CLOSE,1
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,0] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,0]

        OPENW,2,WDparametersOutputFileNameFinalAge
        PRINTF,2,WDparametersOutput[IFMRindex,*,*,1],FORMAT='(E10.3, 1X, E10.3)'
        CLOSE,2
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,1] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,1]

        OPENW,3,WDparametersOutputFileNameInitialMass
        PRINTF,3,WDparametersOutput[IFMRindex,*,*,2],FORMAT='(E10.3, 1X, E10.3)'
        CLOSE,3
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,2] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,2]

        OPENW,4,WDparametersOutputFileNameInitialAge
        PRINTF,4,WDparametersOutput[IFMRindex,*,*,3],FORMAT='(E10.3, 1X, E10.3)'
        CLOSE,4
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,3] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,3]

        OPENW,5,WDparametersOutputFileNameSystemAgeDiff
        PRINTF,5,WDparametersOutput[IFMRindex,*,*,4],FORMAT='(E10.3, 1X, E10.3)'
        CLOSE,5
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,4] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,4]

    endif else begin
        OPENW,1,WDparametersOutputFileNameFinalMass
        PRINTF,1,WDparametersOutput[IFMRindex,*,*,0],FORMAT='(E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3)'
        CLOSE,1
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,0] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,0]

        OPENW,2,WDparametersOutputFileNameFinalAge
        PRINTF,2,WDparametersOutput[IFMRindex,*,*,1],FORMAT='(E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3)'
        CLOSE,2
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,1] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,1]

        OPENW,3,WDparametersOutputFileNameInitialMass
        PRINTF,3,WDparametersOutput[IFMRindex,*,*,2],FORMAT='(E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3)'
        CLOSE,3
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,2] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,2]

        OPENW,4,WDparametersOutputFileNameInitialAge
        PRINTF,4,WDparametersOutput[IFMRindex,*,*,3],FORMAT='(E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3)'
        CLOSE,4
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,3] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,3]

        OPENW,5,WDparametersOutputFileNameSystemAgeDiff
        PRINTF,5,WDparametersOutput[IFMRindex,*,*,4],FORMAT='(E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X,
              E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
                1X, E10.3, 1X, E10.3)'
        CLOSE,5
        ;PRINT, "WDparametersOutput[IFMRindex,*,*,4] = "
        ;PRINT, WDparametersOutput[IFMRindex,*,*,4]

    endelse


endif else begin

    numberOfWDsString = STRTRIM(numberOfWDs, 2)
    formatString= '(' + numberOfWDsString + '(E10.3,","))'  ;F10.4 works
```

```
        OPENW,1,WDparametersOutputFileNameFinalMass
        PRINTF,1,WDparametersOutput[IFMRindex,*,*,0],FORMAT=formatString
        CLOSE,1

        OPENW,2,WDparametersOutputFileNameFinalAge
        PRINTF,2,WDparametersOutput[IFMRindex,*,*,1],FORMAT=formatString
        CLOSE,2

        OPENW,3,WDparametersOutputFileNameInitialMass
        PRINTF,3,WDparametersOutput[IFMRindex,*,*,2],FORMAT=formatString
        CLOSE,3

        OPENW,4,WDparametersOutputFileNameInitialAge
        PRINTF,4,WDparametersOutput[IFMRindex,*,*,3],FORMAT=formatString
        CLOSE,4

        OPENW,5,WDparametersOutputFileNameSystemAgeDiff
        PRINTF,5,WDparametersOutput[IFMRindex,*,*,4],FORMAT=formatString
        CLOSE,5
    endelse
endfor

end


pro plotDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS

; Title: plot DD System Age Difference Distribution Across Published IFMRs
; File: plotDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 09 July 11 10:00am [updated 31 July 11a]
; Summary:
;
;   Postconditions;
;
;   This program plots the DD System Age Difference Distribution Across Published IFMRs
;
;   Preconditions;
;
;   compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs.pro has been executed
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   IFMR0WDoutputParameterSystemAgeDiff.txt [sysAgeDiff] for n WDs
;
;   Usage GDL [linear interpolation only];
;
;   gdl
;   .comp plotDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs.pro
;   plotDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs
;
;   Usage IDL:
;   mkdir /home/aaossz/rbaxter/idllib/astron
;   cd /home/aaossz/rbaxter/idllib/astron
;   tar -xvf astron.tar.gz
;   idl
;   !PATH = Expand_Path('+/home/aaossz/rbaxter/idllib/astron/pro') + ':' + !PATH
;   PRINT, !PATH
;   .comp plotDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs.pro
;   plotDDSystemAgeDifferenceDistributionAcrossPublishedIFMRs
;

useGDL = 0
plotToFile = 1
runTestOfCodeOnly = 0

if(runTestOfCodeOnly eq 1) then begin
    numberOfWDs = 2
    numberOfPublishedIFMRs = 1
    numberOfRealisations = 2501
    numberOfRealisationsUsed = numberOfRealisations-1
endif else begin
    numberOfWDs = 28
    numberOfPublishedIFMRs = 3  ;default 3
    numberOfRealisations = 25001
    numberOfRealisationsUsed = numberOfRealisations-1
endelse

sysAgeDiffMedianArray = DBLARR(numberOfWDs, numberOfPublishedIFMRs) ;artefact
;sysAgeDiffArray = DBLARR(numberOfWDs, numberOfRealisationsUsed, numberOfPublishedIFMRs)
;initialMassArray = DBLARR(numberOfWDs, numberOfRealisationsUsed, numberOfPublishedIFMRs)
;initialAgeArray = DBLARR(numberOfWDs, numberOfRealisationsUsed, numberOfPublishedIFMRs)
```

```
numberOfValues = 5
numberOfErrors = 3
oneSigmaErrorArray = DBLARR(numberOfErrors, numberOfValues, numberOfWDs, numberOfPublishedIFMRs)

cullFraction = 0.158     ;0.158  ;16%
loRealisationIndex=fix(cullFraction*numberOfRealisationsUsed)
hiRealisationIndex=fix((1.0-cullFraction)*numberOfRealisationsUsed)
;loRealisationIndex=fix(0.158*numberOfRealisationsUsed)
;hiRealisationIndex=fix(0.842*numberOfRealisationsUsed)



for valueIndex=0, numberOfValues-1, 1 do begin

    valueArray = DBLARR(numberOfWDs, numberOfRealisationsUsed, numberOfPublishedIFMRs)
    valueArrayIdeal = DBLARR(numberOfWDs, numberOfPublishedIFMRs)

    histogramNumberBins = 50
    valuePlotMinY = 0
    valuePlotMaxY = 10000

    if(valueIndex eq 0) then begin
        inputFileNameEnd = "WDoutputParameterFinalMass.txt"
        finalMassMin = 0.0
        finalMassMax = 20.0
        valuePlotMin = finalMassMin
        valuePlotMax = finalMassMax
        valueXtitle='WD Mass of DD component (M_sol)'
        fileNameBase1 = "finalMassDistribution"
        valueWDindexIncrement = 1
        histogramBinSize = 0.01
    endif else if(valueIndex eq 1) then  begin
        inputFileNameEnd = "WDoutputParameterFinalAge.txt"
        finalAgeMin = 0.0
        finalAgeMax = 10000
        valuePlotMin = finalAgeMin
        valuePlotMax = finalAgeMax
        valueXtitle='WD Cooling Age of DD component (Myrs)'
        fileNameBase1 = "finalAgeDistribution"
        valueWDindexIncrement = 1
        histogramBinSize = 50
    endif else if(valueIndex eq 2) then  begin
        inputFileNameEnd = "WDoutputParameterInitialMass.txt"
        initialMassMin = 0.0
        initialMassMax = 20.0
        valuePlotMin = initialMassMin
        valuePlotMax = initialMassMax
        valueXtitle='MS Progenitor Mass of DD component (M_sol)'
        fileNameBase1 = "initialMassDistribution"
        valueWDindexIncrement = 1
        histogramBinSize = 0.1
    endif else if(valueIndex eq 3) then  begin
        inputFileNameEnd = "WDoutputParameterInitialAge.txt"
        initialAgeMin = 0.0
        initialAgeMax = 10000
        valuePlotMin = initialAgeMin
        valuePlotMax = initialAgeMax
        valueXtitle='MS Progenitor Age of DD component (Myrs)'
        fileNameBase1 = "initialAgeDistribution"
        valueWDindexIncrement = 1
        histogramBinSize = 50
    endif else if(valueIndex eq 4) then  begin
        inputFileNameEnd = "WDoutputParameterSystemAgeDiff.txt"
        initialAgeMin = -2000
        initialAgeMax = 2000
        valuePlotMin = initialAgeMin
        valuePlotMax = initialAgeMax
        valueXtitle='System Age Difference between DD components (Myrs)'
        fileNameBase1 = "systemAgeDifferenceDistribution"
        valueWDindexIncrement = 2
        histogramBinSize = 50
    endif else begin
        PRINT, "valueIndex = invalid"
        exit
    endelse

    for IFMRindex=0, numberOfPublishedIFMRs-1, 1 do begin
        PRINT, "IFMRindex = ", IFMRindex

        IFMRindexString = STRTRIM(IFMRindex, 2)

        WDparametersOutputFileName = "IFMR" + IFMRindexString + inputFileNameEnd
        PRINT, "WDparametersOutputFileName = ", WDparametersOutputFileName
        if(runTestOfCodeOnly eq 1) then begin
              readcol64bitvars, WDparametersOutputFileName, parameter_WD1, parameter_WD2, FORMAT='D,
                  D', SKIPLINE=0
              valueArrayTemp = [transpose(parameter_WD1), transpose(parameter_WD2)]
        endif else begin
```

```
        readcol64bitvars, WDparametersOutputFileName, parameter_WD1, parameter_WD2,
            parameter_WD3, parameter_WD4, parameter_WD5, parameter_WD6, parameter_WD7,
            parameter_WD8, parameter_WD9, parameter_WD10, parameter_WD11, parameter_WD12,
            parameter_WD13, parameter_WD14, parameter_WD15, parameter_WD16, parameter_WD17,
            parameter_WD18, parameter_WD19, parameter_WD20, parameter_WD21, parameter_WD22,
            parameter_WD23, parameter_WD24, parameter_WD25, parameter_WD26, parameter_WD27,
            parameter_WD28, FORMAT='D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,D',
             SKIPLINE=0
        valueArrayTemp = [transpose(parameter_WD1), transpose(parameter_WD2), transpose(
            parameter_WD3), transpose(parameter_WD4), transpose(parameter_WD5), transpose(
            parameter_WD6), transpose(parameter_WD7), transpose(parameter_WD8), transpose(
            parameter_WD9), transpose(parameter_WD10), transpose(parameter_WD11), transpose(
            parameter_WD12), transpose(parameter_WD13), transpose(parameter_WD14), transpose(
            parameter_WD15), transpose(parameter_WD16), transpose(parameter_WD17), transpose(
            parameter_WD18), transpose(parameter_WD19), transpose(parameter_WD20), transpose(
            parameter_WD21), transpose(parameter_WD22), transpose(parameter_WD23), transpose(
            parameter_WD24), transpose(parameter_WD25), transpose(parameter_WD26), transpose(
            parameter_WD27), transpose(parameter_WD28)]
    endelse

    valueArray[*,*,IFMRindex] = valueArrayTemp[*,0:numberOfRealisationsUsed-1]
    valueArrayIdeal[*,IFMRindex] = valueArrayTemp[*,numberOfRealisationsUsed]

    if(valueIndex eq 4) then begin
        ;should not be required;
        for WDindex=0, numberOfWDs-2, 1 do begin

            ;calculate median for statistics;

            ;PRINT, "valueArray[0,*] = ", valueArray[0,*]
            ;PRINT, "valueArray[2,*] = ", valueArray[2,*]

            sysAgeDiffAverage = mean(valueArray[WDindex,*,IFMRindex])
            sysAgeDiffMedian = median(valueArray[WDindex,*,IFMRindex])
            sysAgeDiffLow = valueArray[WDindex,loRealisationIndex,IFMRindex]
            sysAgeDiffHigh = valueArray[WDindex,hiRealisationIndex,IFMRindex]

            ;PRINT, "sysAgeDiffAverage = ", sysAgeDiffAverage
            ;PRINT, "sysAgeDiffMedian = ", sysAgeDiffMedian
            ;PRINT, "sysAgeDiffLow = ", sysAgeDiffLow
            ;PRINT, "sysAgeDiffHigh= ", sysAgeDiffHigh

            sysAgeDiffMedianArray(WDindex, IFMRindex) = sysAgeDiffMedian

        endfor
    endif
endfor

if(valueIndex eq 4) then begin
    for WDindex=0, numberOfWDs-1, 2 do begin
        lowestSysAgeDiffMedian = 10e12
        indexOfLowestSysAgeDiffMedian = -1
        for IFMRindex=0, numberOfPublishedIFMRs-1, 1 do begin
            currentSysAgeDiffMedian = sysAgeDiffMedianArray(WDindex, IFMRindex)
            if(currentSysAgeDiffMedian lt lowestSysAgeDiffMedian) then begin
                lowestSysAgeDiffMedian = currentSysAgeDiffMedian
                indexOfLowestSysAgeDiffMedian = IFMRindex
            endif
        endfor

        PRINT, "WDindex = ", WDindex
        PRINT, "lowestSysAgeDiffMedian = ", lowestSysAgeDiffMedian
        PRINT, "indexOfLowestSysAgeDiffMedian (Published IFMR) = ",
             indexOfLowestSysAgeDiffMedian
    endfor


    for IFMRindex=0, numberOfPublishedIFMRs-1, 1 do begin
        averageMedianSysAgeDiff = 0
        for WDindex=0, numberOfWDs-1, 2 do begin
            averageMedianSysAgeDiff = averageMedianSysAgeDiff +  abs(sysAgeDiffMedianArray(
                WDindex, IFMRindex))
        endfor
        averageMedianSysAgeDiff = averageMedianSysAgeDiff/(numberOfWDs/2)

        PRINT, "IFMRindex = ", IFMRindex
        PRINT, "Published IFMR averageMedianSysAgeDiff = ", averageMedianSysAgeDiff
    endfor
endif



for WDindex=0, numberOfWDs-1, valueWDindexIncrement do begin

    DDIndex = (WDindex/2)+1
    DDIndexString = STRTRIM(DDIndex, 2)
    if(DDIndex lt 10) then begin
           DDnameformat = '(I1)'
           DDindexStringFormal = 'DD-0' + STRING(DDIndex, FORMAT=DDnameformat)
    endif else begin
           DDnameformat = '(I2)'
           DDindexStringFormal = 'DD-' + STRING(DDIndex, FORMAT=DDnameformat)
```

```
endelse

WDIndexString = STRTRIM(WDIndex, 2)
WDComponentString = ''
if((WDindex mod 2) eq 0) then begin
    WDComponentString = 'A'
endif else begin
    WDComponentString = 'B'
endelse



if(plotToFile eq 0) then begin
    plotTitle = 'Simulated Distribution of ' + valueXtitle + ' based upon parameters errors'
    plot, [-10e6], [-10e6], title=plotTitle, xtitle=valueXtitle, ytitle='number of
        realisations', xrange=[valuePlotMin, valuePlotMax], yrange=[valuePlotMinY,
        valuePlotMaxY]    ;yrange=[((numberOfStars+1)/2.0)+0.5, 0.0]
endif else begin
    psGraphfileNameExtension = '.ps'
    pdfGraphfileNameExtension = '.pdf'
    psGraphfileName = fileNameBase1 + "DD" + DDIndexString + WDComponentString +
        psGraphfileNameExtension
    pdfGraphfileName = fileNameBase1 + "DD" + DDIndexString + WDComponentString +
        pdfGraphfileNameExtension

    set_plot,'ps'
    device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE       ;
        SET_RESOLUTION=[320,200]    ;SET_CHARACTER_SIZE=[200,40]

    if(useGDL) then begin
        ;GDL;
        plot, [-10e6], [-10e6], title=plotTitle, xtitle=valueXtitle, ytitle='number of
            realisations', xrange=[valuePlotMin, valuePlotMax], yrange=[valuePlotMinY,
            valuePlotMaxY], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=32,YMARGIN=18  ;yrange=[((
            numberOfStars+1)/2.0)+0.5, 0.0]
    endif else begin
        ;IDL old;
        plot, [-10e6], [-10e6], title=plotTitle, xtitle=valueXtitle, ytitle='number of
            realisations', xrange=[valuePlotMin, valuePlotMax], yrange=[valuePlotMinY,
            valuePlotMaxY]    ;yrange=[((numberOfStars+1)/2.0)+0.5, 0.0]
    endelse

    oplot, [0, 0], [valuePlotMinY, valuePlotMaxY], THICK=2.0

    ;XYOUTS, [valuePlotMax - (valuePlotMax - valuePlotMin)/5.0], [valuePlotMaxY*0.8],
        IFMRIndexString, CHARSIZE=3

endelse

for IFMRindex=0, numberOfPublishedIFMRs-1 , 1 do begin      ;numberOfPublishedIFMRs-1

    ;IFMRIndexString = STRTRIM(IFMRIndex, 2)

    ;calculate histogram [semi-automatic];

     ;PRINT, "loRealisationIndex = ", loRealisationIndex
    ;PRINT, "hiRealisationIndex = ", hiRealisationIndex
    ;PRINT, "WDindex = ", WDindex
    ;PRINT, "IFMRindex = ", IFMRindex

    ;valueLow = valueArray[WDindex,loRealisationIndex,IFMRindex]
    ;valueHigh = valueArray[WDindex,hiRealisationIndex,IFMRindex]

    valueInterval = (valuePlotMax - valuePlotMin) / histogramNumberBins
    valueArraySingleWDSingleIFMRIndex = valueArray[WDindex, *, IFMRindex]
    valueDistributionArrayY = histogram(valueArraySingleWDSingleIFMRIndex, MAX=
        valuePlotMax, MIN=valuePlotMin, NBINS=histogramNumberBins)    ;NBINS=
        histogramNumberBins  ;BINSIZE=histogramBinSize
    valueDistributionArrayX = DBLARR(histogramNumberBins)
    ;valuePlotMinY = 0
    ;valuePlotMaxY = max(valueDistributionArrayY)

   ;realisationsCounts = 0
    ;lowErrorFound = 0
    ;highErrorFound = 0
    for currentBinIndex=0, histogramNumberBins-1, 1 do begin
        valueMinForBin = valuePlotMin + (valueInterval*currentBinIndex)
        valueMaxForBin = valueMinForBin + valueInterval
        valueDistributionArrayX[currentBinIndex] = ((valueMaxForBin+valueMinForBin)/2.0)

     ;realisationsCounts = realisationsCounts + valueDistributionArrayY[currentBinIndex]
     ;if(lowErrorFound eq 0) then begin
     ;    if(realisationsCounts gt loRealisationIndex) then begin
     ;        lowErrorFound = 1
     ;        lowError = valueMinForBin
     ;    endif
     ;endif
     ;if(highErrorFound eq 0) then begin
     ;    if(realisationsCounts gt hiRealisationIndex) then begin
     ;        highErrorFound = 1
     ;        highError = valueMaxForBin
     ;    endif
```

```
            ;endif
            endfor

        valueArraySingleWDSingleIFMRIndexS=sort(valueArraySingleWDSingleIFMRIndex)
        lowError = valueArraySingleWDSingleIFMRIndex(valueArraySingleWDSingleIFMRIndexS[
            loRealisationIndex])
        highError = valueArraySingleWDSingleIFMRIndex(valueArraySingleWDSingleIFMRIndexS[
            hiRealisationIndex])
        lowErrorDiff = lowError - valueArrayIdeal[WDindex, IFMRindex]
        highErrorDiff = highError - valueArrayIdeal[WDindex, IFMRindex]
        oneSigmaErrorArray[0, valueIndex, WDindex, IFMRindex] = valueArrayIdeal[WDindex,
            IFMRindex]
        oneSigmaErrorArray[1, valueIndex, WDindex, IFMRindex] = lowErrorDiff
        oneSigmaErrorArray[2, valueIndex, WDindex, IFMRindex] = highErrorDiff
        ;PRINT, "WDindex = ", WDindex
        ;PRINT, "IFMRindex = ", IFMRindex
        ;PRINT, "lowError = ", lowError
        ;PRINT, "highError = ", highError
        if(IFMRindex eq 0) then begin
            oplot, [lowError, lowError], [valuePlotMinY, valuePlotMaxY], linestyle=IFMRindex
                ;THICK=1.0,
            oplot, [highError, highError], [valuePlotMinY, valuePlotMaxY], linestyle=IFMRindex
                ;THICK=1.0,
        endif

            ;PRINT, "value[WDindex, *] = ", value[WDindex, *]
            ;PRINT, "valueDistributionArrayX = ", transpose(valueDistributionArrayX)
            ;PRINT, "valueDistributionArrayY = ", transpose(valueDistributionArrayY)
            ;PRINT, "valuePlotMaxY = ", valuePlotMaxY
            ;as

            if(plotToFile eq 0) then begin
                oplot, valueDistributionArrayX, valueDistributionArrayY, linestyle=IFMRindex
            endif else begin
                oplot, valueDistributionArrayX, valueDistributionArrayY, linestyle=IFMRindex
            endelse

        endfor

        if(plotToFile eq 0) then begin

        endif else begin

            device,/close
            ;convert ps to pdf;
            convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
            spawn,convertPStoPDFcommand,result

        endelse

endfor


for IFMRindex=0, numberOfPublishedIFMRs-1 , 1 do begin

    IFMRindexString = STRTRIM(IFMRindex, 2)



    if(plotToFile eq 0) then begin
        plotTitle = 'Simulated Distribution of ' + valueXtitle + ' based upon parameters errors'
        plot, [-10e6], [-10e6], title=plotTitle, xtitle=valueXtitle, ytitle='number of
            realisations', xrange=[valuePlotMin, valuePlotMax], yrange=[valuePlotMinY,
            valuePlotMaxY]     ;yrange=[((numberOfStars+1)/2.0)+0.5, 0.0]
    endif else begin
        psGraphfileNameExtension = '.ps'
        pdfGraphfileNameExtension = '.pdf'
        psGraphfileName = fileNameBase1 + "PublishedIFMR" + IFMRindexString +
            psGraphfileNameExtension
        pdfGraphfileName = fileNameBase1 + "PublishedIFMR" + IFMRindexString +
            pdfGraphfileNameExtension

        set_plot,'ps'
        device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE       ;
            SET_RESOLUTION=[320,200]    ;SET_CHARACTER_SIZE=[200,40]

        if(useGDL) then begin
            ;GDL;
            plot, [-10e6], [-10e6], title=plotTitle, xtitle=valueXtitle, ytitle='number of
                realisations', xrange=[valuePlotMin, valuePlotMax], yrange=[valuePlotMinY,
                valuePlotMaxY], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=32,YMARGIN=18   ;yrange=[((
                numberOfStars+1)/2.0)+0.5, 0.0]
        endif else begin
            ;IDL old;
            plot, [-10e6], [-10e6], title=plotTitle, xtitle=valueXtitle, ytitle='number of
                realisations', xrange=[valuePlotMin, valuePlotMax], yrange=[valuePlotMinY,
                valuePlotMaxY]     ;yrange=[((numberOfStars+1)/2.0)+0.5, 0.0]
        endelse

        oplot, [0, 0], [valuePlotMinY, valuePlotMaxY], THICK=2.0
```

```
    ;XYOUTS, [valuePlotMax - (valuePlotMax - valuePlotMin)/5.0], [valuePlotMaxY*0.8],
        IFMRIndexString, CHARSIZE=3

endelse

for WDindex=0, numberOfWDs-1, valueWDindexIncrement do begin

    ;IFMRIndexString = STRTRIM(IFMRIndex, 2)

    ;calculate histogram [semi-automatic];

     ;PRINT, "loRealisationIndex = ", loRealisationIndex
    ;PRINT, "hiRealisationIndex = ", hiRealisationIndex
    ;PRINT, "WDindex = ", WDindex
    ;PRINT, "IFMRindex = ", IFMRindex

    ;valueLow = valueArray[WDindex,loRealisationIndex,IFMRindex]
    ;valueHigh = valueArray[WDindex,hiRealisationIndex,IFMRindex]

    histogramNumberBins = 50
    ;binSize = 50    ;50Myrs
    valueInterval = (valuePlotMax - valuePlotMin) / histogramNumberBins
    valueArraySingleWDSingleIFMRIndex = valueArray[WDindex, *, IFMRindex]
    valueDistributionArrayY = histogram(valueArraySingleWDSingleIFMRIndex, MAX=
        valuePlotMax, MIN=valuePlotMin, NBINS=histogramNumberBins)
    valueDistributionArrayX = DBLARR(histogramNumberBins)
    ;valuePlotMinY = 0
    ;valuePlotMaxY = max(valueDistributionArrayY)

   ; realisationsCounts = 0
    ; lowErrorFound = 0
   ; highErrorFound = 0
    for currentBinIndex=0, histogramNumberBins-1, 1 do begin
        valueMinForBin = valuePlotMin + (valueInterval*currentBinIndex)
        valueMaxForBin = valueMinForBin + valueInterval
        valueDistributionArrayX[currentBinIndex] = ((valueMaxForBin+valueMinForBin)/2.0)

     ;realisationsCounts = realisationsCounts + valueDistributionArrayY[currentBinIndex]
     ;if(lowErrorFound eq 0) then begin
     ;   if(realisationsCounts gt loRealisationIndex) then begin
     ;       lowErrorFound = 1
     ;       lowError = valueMinForBin
     ;   endif
     ;endif
     ;if(highErrorFound eq 0) then begin
     ;   if(realisationsCounts gt hiRealisationIndex) then begin
     ;       highErrorFound = 1
     ;       highError = valueMinForBin
     ;   endif
        ;endif
    endfor

  valueArraySingleWDSingleIFMRIndexS=sort(valueArraySingleWDSingleIFMRIndex)
  lowError = valueArraySingleWDSingleIFMRIndex(valueArraySingleWDSingleIFMRIndexS[
      loRealisationIndex])
  highError = valueArraySingleWDSingleIFMRIndex(valueArraySingleWDSingleIFMRIndexS[
      hiRealisationIndex])
  lowErrorDiff = lowError - valueArrayIdeal[WDindex, IFMRindex]
  highErrorDiff = highError - valueArrayIdeal[WDindex, IFMRindex]
  ;if(WDindex eq 0) then begin
  ;   oplot, [lowError, lowError], [valuePlotMinY, valuePlotMaxY], linestyle=IFMRindex
      ;THICK=1.0,
  ;   oplot, [highError, highError], [valuePlotMinY, valuePlotMaxY], linestyle=IFMRindex
      ;THICK=1.0,
  ;endif



    ;PRINT, "value[WDindex, *] = ", value[WDindex, *]
    ;PRINT, "valueDistributionArrayX = ", transpose(valueDistributionArrayX)
    ;PRINT, "valueDistributionArrayY = ", transpose(valueDistributionArrayY)
    ;PRINT, "valuePlotMaxY = ", valuePlotMaxY
    ;as

    if(plotToFile eq 0) then begin
        oplot, valueDistributionArrayX, valueDistributionArrayY, linestyle=0
    endif else begin
        oplot, valueDistributionArrayX, valueDistributionArrayY, linestyle=0
    endelse

endfor

if(plotToFile eq 0) then begin

endif else begin

    device,/close
    ;convert ps to pdf;
    convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
    spawn,convertPStoPDFcommand,result
```

```
            endelse

    endfor
endfor


IFMRspecificDatafirstColumnIndex = 6
IFMRcolumnIncrement = 9
numberOfColumns = (IFMRspecificDatafirstColumnIndex + numberOfPublishedIFMRs*IFMRcolumnIncrement)

resultsArrayForEnterIntoObjectTable = dblarr(numberOfWDs, numberOfColumns)
for WDindex=0, numberOfWDs-1, 1 do begin
    ;PRINT, "WDindex, ", WDindex
    DDIndex = (WDindex/2)+1
    ;PRINT, "DDIndex = ", DDIndex
        ;PRINT, "sigmaErrorArray for valueIndex (Mf/Agef/Mi/Agei/SysAgeDiff), ", valueIndex
    ;PRINT, [oneSigmaErrorArray[*, 0, WDindex, 0], oneSigmaErrorArray[*, 1, WDindex, 0]]
    resultsArrayForEnterIntoObjectTable[WDindex, 0:IFMRspecificDatafirstColumnIndex-1] = [
         oneSigmaErrorArray[0, 0, WDindex, 0], oneSigmaErrorArray[2, 0, WDindex, 0],
         oneSigmaErrorArray[1, 0, WDindex, 0], oneSigmaErrorArray[0, 1, WDindex, 0],
         oneSigmaErrorArray[2, 1, WDindex, 0],oneSigmaErrorArray[1, 1, WDindex, 0]]
    resultsArrayColumnIndex = IFMRspecificDatafirstColumnIndex
    for IFMRindex=0, numberOfPublishedIFMRs-1 , 1 do begin
        ;PRINT, [oneSigmaErrorArray[*, 2, WDindex, IFMRindex], oneSigmaErrorArray[*, 3, WDindex,
            IFMRindex], oneSigmaErrorArray[*, 4, WDindex, IFMRindex]]
        resultsArrayForEnterIntoObjectTable[WDindex, resultsArrayColumnIndex:resultsArrayColumnIndex
            +IFMRcolumnIncrement-1] = [oneSigmaErrorArray[0, 2, WDindex, IFMRindex],
            oneSigmaErrorArray[2, 2, WDindex, IFMRindex], oneSigmaErrorArray[1, 2, WDindex,
            IFMRindex], oneSigmaErrorArray[0, 3, WDindex, IFMRindex], oneSigmaErrorArray[2, 3,
            WDindex, IFMRindex],oneSigmaErrorArray[1, 3, WDindex, IFMRindex], oneSigmaErrorArray[0,
             4, WDindex, IFMRindex], oneSigmaErrorArray[2, 4, WDindex, IFMRindex],
            oneSigmaErrorArray[1, 4, WDindex, IFMRindex]]
        resultsArrayColumnIndex = resultsArrayColumnIndex+IFMRcolumnIncrement
    endfor
endfor

;32 columns
numberOfColumnsString = STRTRIM(numberOfColumns, 2)
formatString= '(' + numberOfColumnsString + '(E10.3,","))'  ;F10.4 works

resultsArrayForEnterIntoObjectTableFileName = "resultsArrayForEnterIntoObjectTableFileName.txt"

resultsArrayForEnterIntoObjectTable = transpose(resultsArrayForEnterIntoObjectTable)

;PRINT, "resultsArrayForEnterIntoObjectTable = "
;PRINT, resultsArrayForEnterIntoObjectTable

OPENW,1,resultsArrayForEnterIntoObjectTableFileName
PRINTF,1,resultsArrayForEnterIntoObjectTable,FORMAT=formatString
CLOSE,1

;GDL...
;OPENW,5,resultsArrayForEnterIntoObjectTableFileName
;PRINTF,1,resultsArrayForEnterIntoObjectTable,FORMAT='(E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X,
     E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
     1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10
     .3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3, 1X, E10.3,
      E10.3, 1X, E10.3)'
;CLOSE,5


end
```

A.6.3 Distance Moduli Comparison of DD components

This software is used to plot the distance moduli of double degenerate WD components based upon their observed apparent magnitudes and theoretical absolute magnitudes (for u, g, r, bands), to confirm that they are associated and that the modelling processes have provided accurate absolute magnitude values.

plotToFileDistanceModulus.pro

SDSSDDdistancemoduliconfirmation-29July11a.zip

```
; Title: Plot Distance Modulus
; File: plotDistanceModulus.pro
```

```
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 1 July 10 10pm
; Summary:
;
;   Postconditions;
;
;   This program plots g vs g-r for each blue object pair
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   DDphotometricBands.txt [DD_NUMBER, G app psf, R app psf, U app psf, G abs, R abs, U abs,
     Temperature, G app psf error, R app psf error, U app psf error, delta M g, delta M r, delta M u
     , G m-M error, R m-M error, U m-M error]     for X blue objects Pairs, tab delimited

pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black     : 0, $
                white     : 1, $
                red       : 2, $
                green     : 3, $
                blue      : 4, $
                yellow    : 5, $
                magenta   : 6, $
                cyan      : 7, $
                orange    : 8, $
                l_green   : 9, $
                violet    : 10,$
                turquoise : 11,$
                l_blue    : 12,$
                l_red     : 13 $
               }
;         0  1    2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]    ;blue
end


my_colors


useGDL = 1
numberOfTables = 8

for tableIndex=0L, numberOfTables-1, 1L do begin

    xRangeMax = 12.0
    xRangeMin = 2.0

    useAbsoluteMagnitudes = 1

    colourMatchWithUMinusGorGMinusR = 2 ;   2 = colourMatchWithUMinusR, 1 = colourMatchWithUMinusG,
        0 = colourMatchWithGMinusR

    tableIndexString = STRTRIM(tableIndex, 2)
    database1NameOfFilePart1 = 'DDphotometricBands' + tableIndexString + '.txt'

        ;long precision is required for ID
    readcol64bitvars, database1NameOfFilePart1, DD_NUMBER, G_APP, R_APP, U_APP, G_ABS, R_ABS, U_ABS,
         TEMPERATURE, G_APP_ERR, R_APP_ERR, U_APP_ERR, G_DELTA_ABS, R_DELTA_ABS, U_DELTA_ABS,
        G_DM_ERR, R_DM_ERR, U_DM_ERR, SPECIAL_CASE, FORMAT='D,D,D,D,D,D,D,D,D,D,D,D,D,D,D,A'



    database1fieldNumberOfStars = (size(DD_NUMBER))[1]

    plotTitle = 'Distance Moduli (m - M) for SDSS DD systems (u, g, r) [circles/crosses=reduced
        component A/B]'

    tempXArray = dblarr(3)
    tempYArray = dblarr(3)
    tempXArray = [0, 0, 0]
    tempYArray = [0, 0, 0]

    ;PRINT, "sdf = ", ((database1fieldNumberOfStars+1)/2.0)+0.5



    plotToFile = 1
    if(plotToFile eq 0) then begin
```

```
    plot, tempXArray, tempYArray, title=plotTitle, xtitle='m-M', ytitle='SDSS DD system (1 to 12
          respectively, for u, g, and r respectively)', xrange=[xRangeMin, xRangeMax], yrange
         =[((database1fieldNumberOfStars+1)/2.0)+0.5, 0.0]

endif else begin

    psGraphfileNameExtension = '.ps'
    pdfGraphfileNameExtension = '.pdf'
    fileNameBase1 = 'SDSSDDdistancemoduliconfirmation' + tableIndexString
    psGraphfileName = fileNameBase1 + psGraphfileNameExtension
    pdfGraphfileName = fileNameBase1 + pdfGraphfileNameExtension

    set_plot,'ps'
    if(useGDL eq 1) then begin
        device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE       ;
            SET_RESOLUTION=[320,200]    ;SET_CHARACTER_SIZE=[200,40]
    endif else begin
        device,filename=psGraphfileName,/color,bits_per_pixel=24, /LANDSCAPE    ;/PORTRAIT  /
            LANDSCAPE       ;SET_RESOLUTION=[320,200]    ;SET_CHARACTER_SIZE=[200,40]
    endelse
    plot, tempXArray, tempYArray, xtitle='m-M', ytitle='SDSS DD system (A and B, u, g, and r,
         respectively)', xrange=[xRangeMin, xRangeMax], yrange=[((database1fieldNumberOfStars+1)
         /2.0)+0.5, 0.0], BACKGROUND=1, CHARSIZE=2.0,XMARGIN=32,YMARGIN=18

endelse


PRINT, "database1fieldNumberOfStars = ", database1fieldNumberOfStars


for database1starIndex=0L, database1fieldNumberOfStars-1, 1L do begin

        uminusU = dblarr(3)
        gminusG = dblarr(3)
        rminusR = dblarr(3)
        yuminusU = dblarr(3)
        ygminusG = dblarr(3)
        yrminusR = dblarr(3)

    uminusU[1] = U_APP[database1starIndex] - U_ABS[database1starIndex]
    gminusG[1] = G_APP[database1starIndex] - G_ABS[database1starIndex]
    rminusR[1] = R_APP[database1starIndex] - R_ABS[database1starIndex]
    uminusU[0] = uminusU[1] - U_DM_ERR[database1starIndex]
    gminusG[0] = gminusG[1] - G_DM_ERR[database1starIndex]
    rminusR[0] = rminusR[1] - R_DM_ERR[database1starIndex]
    uminusU[2] = uminusU[1] + U_DM_ERR[database1starIndex]
    gminusG[2] = gminusG[1] + G_DM_ERR[database1starIndex]
    rminusR[2] = rminusR[1] + R_DM_ERR[database1starIndex]

    ;PRINT, "uminusU[1] = ", uminusU[1]
    ;PRINT, "gminusG[1] = ", gminusG[1]
    ;PRINT, "rminusR[1] = ", rminusR[1]
    ;PRINT, "U_DM_ERR[database1starIndex] = ", U_DM_ERR[database1starIndex]
    ;PRINT, "G_DM_ERR[database1starIndex] = ", G_DM_ERR[database1starIndex]
    ;PRINT, "R_DM_ERR[database1starIndex] = ", R_DM_ERR[database1starIndex]

    if((database1starIndex mod 2) eq 0) then begin
        base = database1starIndex/2
        ;PRINT, "base = ", base
        ;baseDouble = DOUBLE(base)
        yuminusU[1] = (database1starIndex/2.0)+1.0-0.05
        ygminusG[1] = (database1starIndex/2.0)+1.0
        yrminusR[1] = (database1starIndex/2.0)+1.0+0.05
    endif else begin
        yuminusU[1] = (database1starIndex/2.0)+0.7-0.05
        ygminusG[1] = (database1starIndex/2.0)+0.7
        yrminusR[1] = (database1starIndex/2.0)+0.7+0.05

    endelse
        yuminusU[0] = yuminusU[1]
        ygminusG[0] = ygminusG[1]
        yrminusR[0] = yrminusR[1]
        yuminusU[2] = yuminusU[1]
        ygminusG[2] = ygminusG[1]
        yrminusR[2] = yrminusR[1]

    oplot, uminusU, yuminusU, COLOR=4
    oplot, gminusG, ygminusG, COLOR=3
    oplot, rminusR, yrminusR, COLOR=2



    DDIndex = long(DD_NUMBER[database1starIndex])
    DDIndexString = STRTRIM(DDIndex, 2)
    if(DDIndex lt 10) then begin
            DDnameformat = '(I1)'
            DDindexStringFormal = '0' + STRING(DDIndex, FORMAT=DDnameformat)
    endif else begin
            DDnameformat = '(I2)'
```

```
            DDindexStringFormal = STRING(DDIndex, FORMAT=DDnameformat)
        endelse
        systemNameString= 'DD-'+DDindexStringFormal

        DDComponentString = ''
        if((database1starIndex mod 2) ne 0) then begin
            DDComponentString = 'B'
        endif else begin
            DDComponentString = 'A'
        endelse

        if((database1starIndex mod 2) eq 0) then begin
            systemNameStringXPos = xRangeMax - (xRangeMax/10)
            XYOUTS, [systemNameStringXPos, systemNameStringXPos], [ygminusG[1], ygminusG[1]],
                systemNameString, CHARSIZE=2
        endif

        systemNameStringXPos = xRangeMax - (xRangeMax/6)
        specialCaseString = DDComponentString + ' ' + SPECIAL_CASE[database1starIndex]
        XYOUTS, [systemNameStringXPos, systemNameStringXPos], [ygminusG[1], ygminusG[1]],
            specialCaseString, CHARSIZE=1

    endfor


    if(plotToFile eq 0) then begin

    endif else begin

        device,/close
        ;convert ps to pdf;
        convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
        spawn,convertPStoPDFcommand,result

    endelse
endfor

end
```

A.6.4 DD Mass Distribution Comparison with Field WD Survey

This software is used to apply a K-S test to my DD Data and a known field WD survey mass distribution.

cmd.pro

compareMassDistributions-29July11b.zip

```
; Title: CMD: Compare Mass Distributions (DD systems versus SDSS field WDs)
; File: cmd.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 24 Feb 11 8:00pm (updated 7 Mar 11a)
; Summary:
;
;   Postconditions;
;
;   This program compare Mass Distributions of DD systems versus SDSS field WDs
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   kstwo.pro (NASA)
;   PROB_KS.pro (NASA)
;   METHOD1; WDmassDistributionPGsurveyVmaxCorrectedWithCoarseBinning.txt   [NB this data has been
    rebinned to match binning of DD mass Distribution]
;   METHOD1; DDmassDistribution.txt
```

```
;   METHOD2; (SW generated from predefined gaussians) [uses WDmassDistributionKepler.txt for visual
     comparison]
;   METHOD2; DDmasses.txt
;   METHOD4; WDmassDistributionPGsurveyVmaxCorrected.txt
;   METHOD4; DDmasses.txt
;   METHOD5; (SW generated from predefined gaussians) [uses WDmassDistributionPGsurveyVmaxCorrected.
     txt for visual comparison]
;   METHOD5; DDmasses.txt
;   METHOD6; (SW generated from predefined gaussians) [uses WDmassDistributionVmaxCorrectedKepler.
     txt for visual comparison]
;   METHOD6; DDmasses.txt

;
; Note; should use for verification - http://www.physics.csbsju.edu/stats/KS-test.n.plot_form.html


pro my_colors

defsysv,'!col',EXISTS = col_exists
if not col_exists then $
defsysv,'!col',{black      : 0, $
                white      : 1, $
                red        : 2, $
                green      : 3, $
                blue       : 4, $
                yellow     : 5, $
                magenta    : 6, $
                cyan       : 7, $
                orange     : 8, $
                l_green    : 9, $
                violet     : 10,$
                turquoise  : 11,$
                l_blue     : 12,$
                l_red      : 13 $
               }
;         0  1    2   3   4   5   6   7   8   9   10  11  12  13
tvlct,[  0,255, 255,  0,  0,255,255,  0,255,125,125,  0,  0,255],$ ;red
      [  0,255,   0,255,  0,255,  0,255,125,255,  0,255,125,  0],$ ;green
      [  0,255,   0,  0,255,  0,255,255,  0,  0,255,125,255,125]    ;blue
end


my_colors


; KS Test METHOD 1; Compare DD mass Distribution with PG Survey Vmax corrected mass distribution
;PRINT, "KS Test METHOD 1; Compare DD mass Distribution with PG Survey mass distribution"

database1NameOfFile = 'DDmassDistribution.txt'
database2NameOfFile = 'WDmassDistributionPGsurveyVmaxCorrectedWithCoarseBinning.txt'

readcol64bitvars, database1NameOfFile, MASS_BIN_DB1, FREQUENCY_DB1, FORMAT='D,D', SKIPLINE=1
readcol64bitvars, database2NameOfFile, MASS_BIN_DB2, FREQUENCY_DB2, FORMAT='D,D', SKIPLINE=1

;PRINT, "FREQUENCY_DB1 = ", FREQUENCY_DB1
;PRINT, "FREQUENCY_DB2 = ", FREQUENCY_DB2

kstwo, FREQUENCY_DB1, FREQUENCY_DB2, PG_MASSDISTRIBUTION_D, PG_MASSDISTRIBUTION_prob
PRINT," Method 1: "
PRINT," PG_MASSDISTRIBUTION_D = ", PG_MASSDISTRIBUTION_D
PRINT," PG_MASSDISTRIBUTION_prob = ", PG_MASSDISTRIBUTION_prob




; KS Test METHOD 2; Compare DD masses with Kepler theoretical/modeled SDSS WD masses
;PRINT, "KS Test METHOD 2; Compare DD masses with Kepler theoretical/modeled SDSS WD masses"


;generate model/function based upon SDSS Kepler 2007 survey (magnitude limited, >12000K)
;  Evaulate a Gaussian centered at x=0, with sigma=1, and a peak value
;       of 10 at the points 0.5 and 1.5.   Also compute the derivative
;       IDL> f = gaussian( [0.5,1.5], [10,0,1], DERIV )
; 1733 stars brighter than g=19, T>=12000K


; WDmassDistributionKepler.txt is only used for print out visual comparison (not used for method2,
    as method2 relies upon kepler's predetermined gaussian fit)
database6NameOfFile = 'WDmassDistributionKepler.txt'

readcol64bitvars, database6NameOfFile, MASS_BIN_DB6, FREQUENCY_DB6, FORMAT='D,D', SKIPLINE=1

;PRINT, "FREQUENCY_DB6 = ", FREQUENCY_DB6

combinedSDSSgaussianArrayRealTotalNumberOfObjects = 1733
SDSSgaussianArrayXIncrement = 0.01
useKSwebForm = 1
if(useKSwebForm eq 1) then begin
    useMethod2KSwebFormSDSSKeplerModelMasses = 1
    useMethod3KSwebFormSDSSKeplerModelMassDistribution = 0
    if(useMethod2KSwebFormSDSSKeplerModelMasses eq 1) then begin
```

```
        combinedSDSSgaussianArrayRealTotalNumberOfObjects = 1000     ;max number of objects supported
            by webform
    endif
    if(useMethod3KSwebFormSDSSKeplerModelMassDistribution eq 1) then begin
        combinedSDSSgaussianArrayRealTotalNumberOfObjects = 28   ;DD number of objects
        SDSSgaussianArrayXIncrement = 0.1            ;DD number of objects

    endif
endif

SDSSmaximumMass = 1.5
SDSSgaussianArrayXCurrent = 0.0
SDSSgaussianArrayNumberOfPoints = SDSSmaximumMass/SDSSgaussianArrayXIncrement
SDSSgaussianArrayX = dblarr(SDSSgaussianArrayNumberOfPoints)
;PRINT, "SDSSgaussianArrayNumberOfPoints = ", SDSSgaussianArrayNumberOfPoints

for SDSSgaussianArrayIndex=0, SDSSgaussianArrayNumberOfPoints-1, 1 do begin
    SDSSgaussianArrayXCurrent = SDSSgaussianArrayXCurrent + SDSSgaussianArrayXIncrement
    SDSSgaussianArrayX[SDSSgaussianArrayIndex] = SDSSgaussianArrayXCurrent
endfor

SDSSgaussianArrayComponent1sigma = 0.047
SDSSgaussianArrayComponent2sigma = 0.149
SDSSgaussianArrayComponent3sigma = 0.050
SDSSgaussianArrayComponent4sigma = 0.143

SDSSgaussianArrayComponent1magnitude = 264.8
SDSSgaussianArrayComponent2magnitude = 27.8
SDSSgaussianArrayComponent3magnitude = 27
SDSSgaussianArrayComponent4magnitude = 3.0

SDSSgaussianArrayComponent1centre = 0.578
SDSSgaussianArrayComponent2centre = 0.658
SDSSgaussianArrayComponent3centre = 0.381
SDSSgaussianArrayComponent4centre = 1.081

SDSSgaussianArrayComponent1fraction = 69
SDSSgaussianArrayComponent2fraction = 23
SDSSgaussianArrayComponent3fraction = 7
SDSSgaussianArrayComponent4fraction = 1

SDSSgaussianArrayComponent1 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent1magnitude,
    SDSSgaussianArrayComponent1centre,SDSSgaussianArrayComponent1sigma], DERIV )
SDSSgaussianArrayComponent2 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent2magnitude,
    SDSSgaussianArrayComponent2centre,SDSSgaussianArrayComponent2sigma], DERIV )
SDSSgaussianArrayComponent3 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent3magnitude,
    SDSSgaussianArrayComponent3centre,SDSSgaussianArrayComponent3sigma], DERIV )
SDSSgaussianArrayComponent4 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent4magnitude,
    SDSSgaussianArrayComponent4centre,SDSSgaussianArrayComponent4sigma], DERIV )

combinedSDSSgaussianArray = SDSSgaussianArrayComponent1 + SDSSgaussianArrayComponent2 +
    SDSSgaussianArrayComponent3 + SDSSgaussianArrayComponent4


plot, MASS_BIN_DB6, FREQUENCY_DB6
oplot, SDSSgaussianArrayX, combinedSDSSgaussianArray
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent1
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent2
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent3
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent4

;wait, 5

; now extract number of components at each mass interval/bin
;...
;now integrate over the curve to obtain the total area, and divide by the total number of stars in
    this survey

    ;find total area
totalAreaUnderCombinedSDSSgaussianArray = 0
for SDSSgaussianArrayIndex=0, SDSSgaussianArrayNumberOfPoints-1, 1 do begin
    totalAreaUnderCombinedSDSSgaussianArray = totalAreaUnderCombinedSDSSgaussianArray +
        combinedSDSSgaussianArray[SDSSgaussianArrayIndex]
endfor
    ;now normalise
for SDSSgaussianArrayIndex=0, SDSSgaussianArrayNumberOfPoints-1, 1 do begin
    combinedSDSSgaussianArray[SDSSgaussianArrayIndex] = combinedSDSSgaussianArray[
        SDSSgaussianArrayIndex]/totalAreaUnderCombinedSDSSgaussianArray*
        combinedSDSSgaussianArrayRealTotalNumberOfObjects
endfor
;oplot, SDSSgaussianArrayX, combinedSDSSgaussianArray


    ;now generate SDSSmassesArray
SDSSmassesArray = dblarr(combinedSDSSgaussianArrayRealTotalNumberOfObjects)
SDSSgaussianArrayIndex = 0
combinedSDSSgaussianArrayTemp = combinedSDSSgaussianArray
for SDSSmassesArrayIndex=0, combinedSDSSgaussianArrayRealTotalNumberOfObjects-1, 0 do begin
    ;PRINT, "combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex] = ",
        combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex]
    if(combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex] lt 0.45) then begin
        SDSSgaussianArrayIndex = SDSSgaussianArrayIndex + 1
```

```
        ;PRINT, "here2"
    endif else begin
        combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex] = combinedSDSSgaussianArrayTemp[
            SDSSgaussianArrayIndex] - 1
        SDSSmassesArray[SDSSmassesArrayIndex] = SDSSgaussianArrayIndex/
            SDSSgaussianArrayNumberOfPoints*SDSSmaximumMass
        SDSSmassesArrayIndex = SDSSmassesArrayIndex + 1
        ;PRINT, "here"
    endelse
endfor

database3NameOfFile = 'DDmasses.txt'
readcol64bitvars, database3NameOfFile, MASS_DB3, FORMAT='D', SKIPLINE=1

;PRINT, "MASS_DB3 = ", MASS_DB3

kstwo, SDSSmassesArray, MASS_DB3, SDSS_MASSES_D, SDSS_MASSES_prob
PRINT,"Method 2:"
PRINT,"SDSS_MASSES_D = ", SDSS_MASSES_D
PRINT,"SDSS_MASSES_prob = ", SDSS_MASSES_prob

outputFileName2a='METHOD2SDSSKeplerModelMasses.txt'
OPENW,1,outputFileName2a
PRINTF,1,SDSSmassesArray,FORMAT='(F)'
CLOSE,1

outputFileName2b='METHOD2SDSSKeplerModelMassDistribution.txt'
OPENW,1,outputFileName2b
PRINTF,1,combinedSDSSgaussianArray,FORMAT='(F)'
CLOSE,1



; KS Test METHOD 3; Compare DD masses with PG Survey Vmax corrected masses
;PRINT, "KS Test METHOD 3; Compare DD masses with PG Survey Vmax corrected masses"
;
;database4NameOfFile = 'WDmassesAndVmaxPGsurveyWithKnownDDsRemoved.txt'
;
;readcol64bitvars, database4NameOfFile, MASS_DB4, VMAX_DB4, FORMAT='D,D', SKIPLINE=1
;
;PGTotalNumberObjects = (size(MASS_DB4))[1]
;PRINT, "PGTotalNumberObjects = ", PGTotalNumberObjects
;VmaxModifierResolution = 1000L ;1000:1
;PRINT, "PGTotalNumberObjects*VmaxModifierResolution = ", PGTotalNumberObjects*
    VmaxModifierResolution
;artificialPGSurveyMassesInAVolumeLimitedSample = dblarr(PGTotalNumberObjects*VmaxModifierResolution
    )
;artificialPGSurveyMassesInAVolumeLimitedSampleIndex = 0L
;for PGmassesArrayIndex=0L, PGTotalNumberObjects-1, 1L do begin
;   PGnumberOfArtificalObjectsToCreateBasedUponVmaxModifier = VMAX_DB4[PGmassesArrayIndex] * 10e8
;   PRINT, "PGnumberOfArtificalObjectsToCreateBasedUponVmaxModifier = ",
    PGnumberOfArtificalObjectsToCreateBasedUponVmaxModifier
;   PRINT, "artificialPGSurveyMassesInAVolumeLimitedSampleIndex = ",
    artificialPGSurveyMassesInAVolumeLimitedSampleIndex
;   for PGnumberOfArtificalObjectsToCreateBasedUponVmaxModifierIndex=0L,
    PGnumberOfArtificalObjectsToCreateBasedUponVmaxModifier-1, 1L do begin
;       artificialPGSurveyMassesInAVolumeLimitedSample[
    artificialPGSurveyMassesInAVolumeLimitedSampleIndex] = MASS_DB4[PGmassesArrayIndex]
;       artificialPGSurveyMassesInAVolumeLimitedSampleIndex =
    artificialPGSurveyMassesInAVolumeLimitedSampleIndex+1
;       ;PRINT, "artificialPGSurveyMassesInAVolumeLimitedSampleIndex = ",
    artificialPGSurveyMassesInAVolumeLimitedSampleIndex
;   endfor
;endfor
;
;
;PRINT, "MASS_DB4 = ", MASS_DB4
;PRINT, "VMAX_DB4 = ", VMAX_DB4
;
;kstwo, MASS_DB3, artificialPGSurveyMassesInAVolumeLimitedSample, PG_MASSES_D, PG_MASSES_prob
;PRINT,"PG_MASSES_D = ", PG_MASSES_D
;PRINT,"PG_MASSES_prob = ", PG_MASSES_prob



; KS Test METHOD 4; Compare DD masses with re-generated PG Survey Vmax corrected masses
;PRINT, "KS Test METHOD 4; Compare DD masses with re-generated PG Survey Vmax corrected masses"

database5NameOfFile = 'WDmassDistributionPGsurveyVmaxCorrected.txt'

readcol64bitvars, database5NameOfFile, MASS_BIN_DB5, FREQUENCY_DB5, FORMAT='D,D', SKIPLINE=1

PGMassDistributionTotalNumberBins = (size(MASS_BIN_DB5))[1]

;PRINT, "MASS_BIN_DB5 = ", MASS_BIN_DB5
;PRINT, "FREQUENCY_DB5 = ", FREQUENCY_DB5

artificialPGSurveyMassesInAVolumeLimitedSampleTotalNumberObjects = 1000
artificialPGSurveyMassesInAVolumeLimitedSample = dblarr(
    artificialPGSurveyMassesInAVolumeLimitedSampleTotalNumberObjects)
```

```
artificialPGSurveyMassesInAVolumeLimitedSampleIndex = 0

    ;now generate artificialPGSurveyMassesInAVolumeLimitedSample
for PGMassDistributionBinIndex=0, PGMassDistributionTotalNumberBins-1, 1 do begin
    ;PRINT, "FREQUENCY_DB5[artificialPGSurveyMassesInAVolumeLimitedSampleIndex] = ", FREQUENCY_DB5[
        artificialPGSurveyMassesInAVolumeLimitedSampleIndex]

    numberOfObjectsAtGivenMass = FREQUENCY_DB5[PGMassDistributionBinIndex]*(
        artificialPGSurveyMassesInAVolumeLimitedSampleTotalNumberObjects/2)
    ;PRINT, "numberOfObjectsAtGivenMass = ", numberOfObjectsAtGivenMass
    for numberOfObjectsAtGivenMassIndex=0, numberOfObjectsAtGivenMass-1, 1 do begin
        artificialPGSurveyMassesInAVolumeLimitedSample[
            artificialPGSurveyMassesInAVolumeLimitedSampleIndex] = MASS_BIN_DB5[
            PGMassDistributionBinIndex]
        artificialPGSurveyMassesInAVolumeLimitedSampleIndex =
            artificialPGSurveyMassesInAVolumeLimitedSampleIndex + 1
    endfor
endfor

;PRINT, "MASS_DB3 = ", MASS_DB3
;PRINT, "artificialPGSurveyMassesInAVolumeLimitedSample = ",
    artificialPGSurveyMassesInAVolumeLimitedSample

kstwo, MASS_DB3, artificialPGSurveyMassesInAVolumeLimitedSample, PG_MASSES_D, PG_MASSES_prob
PRINT,"Method 4: "
PRINT,"PG_MASSES_D = ", PG_MASSES_D
PRINT,"PG_MASSES_prob = ", PG_MASSES_prob

outputFileName4='METHOD4PGSurveyVMaxCorrectedModelMasses.txt'
OPENW,1,outputFileName4
PRINTF,1,artificialPGSurveyMassesInAVolumeLimitedSample,FORMAT='(F)'
CLOSE,1
```

```
; KS Test METHOD 5; Compare DD masses with regenerated masses obtained from theoretical/modeled PG
    Survey Vmax corrected mass distribution
;PRINT, "KS Test METHOD 5; Compare DD masses with regenerated masses obtained from theoretical/
    modeled PG Survey Vmax corrected mass distribution"

;PRINT,"Method 5: "
;PRINT, "MASS_BIN_DB5 = ", MASS_BIN_DB5
;PRINT, "FREQUENCY_DB5 = ", FREQUENCY_DB5

;PGSurveyVmaxCorrectedMassDistributionPolyFitArrayCoefficients = POLY_FIT(MASS_BIN_DB5,
    FREQUENCY_DB5, 10, MEASURE_ERRORS=measure_errors, SIGMA=sigma)

        ;this section of code is from METHOD 2
method5PGSurveyUseSameNumberOfObjectsAsMethod2SDSSSurvey = 0
if(method5PGSurveyUseSameNumberOfObjectsAsMethod2SDSSSurvey eq 1) then begin
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayRealTotalNumberOfObjects =
        combinedSDSSgaussianArrayRealTotalNumberOfObjects  ;eg 1733
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayXIncrement = SDSSgaussianArrayXIncrement  ;eg
        0.01
    PGSurveyVmaxCorrectedMassDistributionGaussianArraymaximumMass = SDSSmaximumMass
endif else begin
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayRealTotalNumberOfObjects = 347
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayXIncrement = 0.01
    PGSurveyVmaxCorrectedMassDistributionGaussianArraymaximumMass = 1.5
endelse
PGSurveyVmaxCorrectedMassDistributionGaussianArrayXCurrent = 0.0
PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints =
    PGSurveyVmaxCorrectedMassDistributionGaussianArraymaximumMass/
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayXIncrement
PGSurveyVmaxCorrectedMassDistributionGaussianArrayX = dblarr(
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints)
;PRINT, "PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints = ",
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints

for PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex=0,
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints-1, 1 do begin
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayXCurrent =
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayXCurrent +
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayXIncrement
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayX[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex] =
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayXCurrent
endfor


PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0sigma = 0.03
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1sigma = 0.060
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2sigma = 0.1
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3sigma = 0.04
```

```
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4sigma = 0.1
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5sigma = 0.02

PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0magnitude = 0.015 ;8.0
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1magnitude = 0.34 ;8.0
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2magnitude = 0.07 ;1.5
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3magnitude = 0.10 ;2.5
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4magnitude = 0.015 ;0.5
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5magnitude = 0.04 ;1.0

PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0centre = 0.400
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1centre = 0.565
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2centre = 0.725
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3centre = 0.925
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4centre = 1.125
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5centre = 1.325

PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0 = gaussian(
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayX, [
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0magnitude,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0centre,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0sigma], DERIV )
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1 = gaussian(
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayX, [
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1magnitude,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1centre,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1sigma], DERIV )
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2 = gaussian(
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayX, [
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2magnitude,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2centre,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2sigma], DERIV )
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3 = gaussian(
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayX, [
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3magnitude,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3centre,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3sigma], DERIV )
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4 = gaussian(
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayX, [
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4magnitude,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4centre,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4sigma], DERIV )
PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5 = gaussian(
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayX, [
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5magnitude,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5centre,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5sigma], DERIV )

PGSurveyVmaxCorrectedMassDistributionGaussianArray =
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0 +
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1 +
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2 +
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3 +
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4 +
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5

plot, MASS_BIN_DB5, FREQUENCY_DB5
oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArray
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent0
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent1
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent2
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent3
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent4
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayComponent5


;wait, 5

;PGSurveyVmaxCorrectedMassDistributionPolyFitArray = poly(
     PGSurveyVmaxCorrectedMassDistributionPolyFitArrayX,
     PGSurveyVmaxCorrectedMassDistributionPolyFitArrayCoefficients)
;plot, MASS_BIN_DB5, FREQUENCY_DB5
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
     PGSurveyVmaxCorrectedMassDistributionGaussianArray
;wait, 5


        ;this section of code is from METHOD 2
    ;find total area
totalAreaUnderPGSurveyVmaxCorrectedMassDistributionGaussianArray = 0
for PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex=0,
     PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints-1, 1 do begin
```

```
    totalAreaUnderPGSurveyVmaxCorrectedMassDistributionGaussianArray =
        totalAreaUnderPGSurveyVmaxCorrectedMassDistributionGaussianArray +
        PGSurveyVmaxCorrectedMassDistributionGaussianArray[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex]
endfor
    ;now normalise
for PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex=0,
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints-1, 1 do begin
    PGSurveyVmaxCorrectedMassDistributionGaussianArray[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex] =
        PGSurveyVmaxCorrectedMassDistributionGaussianArray[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex]/
        totalAreaUnderPGSurveyVmaxCorrectedMassDistributionGaussianArray*
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayRealTotalNumberOfObjects
endfor
;oplot, PGSurveyVmaxCorrectedMassDistributionGaussianArrayX,
    PGSurveyVmaxCorrectedMassDistributionGaussianArray
    ;now generate PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArray
PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArray = dblarr(
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayRealTotalNumberOfObjects)
PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex = 0
PGSurveyVmaxCorrectedMassDistributionGaussianArrayTemp =
    PGSurveyVmaxCorrectedMassDistributionGaussianArray
for PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArrayIndex=0,
    PGSurveyVmaxCorrectedMassDistributionGaussianArrayRealTotalNumberOfObjects-1, 0 do begin
    ;PRINT, "PGSurveyVmaxCorrectedMassDistributionGaussianArrayTemp[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex] = ",
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayTemp[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex]
    if(PGSurveyVmaxCorrectedMassDistributionGaussianArrayTemp[
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex] lt 0.45) then begin
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex =
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex + 1
        ;PRINT, "here2"
    endif else begin
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayTemp[
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex] =
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayTemp[
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex] - 1
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArray[
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArrayIndex] =
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayIndex/
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayNumberOfPoints*
            PGSurveyVmaxCorrectedMassDistributionGaussianArraymaximumMass
        PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArrayIndex =
            PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArrayIndex + 1
        ;PRINT, "here"
    endelse
endfor


kstwo, PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArray, MASS_DB3, PGSURVEY_MASSES_D,
    PGSURVEY_MASSES_prob
PRINT,"Method 5: "
PRINT,"PGSURVEY_MASSES_D = ", PGSURVEY_MASSES_D
PRINT,"PGSURVEY_MASSES_prob = ", PGSURVEY_MASSES_prob

outputFileName5a='METHOD5PGSurveyVMaxCorrectedModelMasses.txt'
OPENW,1,outputFileName5a
PRINTF,1,PGSurveyVmaxCorrectedMassDistributionGaussianArrayMassesArray,FORMAT='(F)'
CLOSE,1

outputFileName5b='METHOD5PGSurveyVMaxCorrectedModelMassDistribution.txt'
OPENW,1,outputFileName5b
PRINTF,1,PGSurveyVmaxCorrectedMassDistributionGaussianArray,FORMAT='(F)'
CLOSE,1









; KS Test METHOD 6;Compare DD masses with Kepler Vmax Corrected  theoretical/modeled SDSS WD masses
;PRINT, "KS Test METHOD 6; Compare DD masses with Kepler Vmax Corrected theoretical/modeled SDSS WD
    masses"


;WDmassDistributionKeplerVmaxCorrected.txt is only used for print out visual comparison (not used
    for method8, as method8 relies upon kepler's predetermined gaussian fit)
database8NameOfFile = 'WDmassDistributionKeplerVmaxCorrected.txt'

readcol64bitvars, database8NameOfFile, MASS_BIN_DB8, FREQUENCY_DB8, FORMAT='D,D', SKIPLINE=1

;PRINT, "FREQUENCY_DB8 = ", FREQUENCY_DB8


; SAME AS METHOD2;
;combinedSDSSgaussianArrayRealTotalNumberOfObjects = 1733
```

```
;SDSSgaussianArrayXIncrement = 0.01
;useKSwebForm = 1
;if(useKSwebForm eq 1) then begin
;    useMethod2KSwebFormSDSSKeplerModelMasses = 1
;    useMethod3KSwebFormSDSSKeplerModelMassDistribution = 0
;    if(useMethod2KSwebFormSDSSKeplerModelMasses eq 1) then begin
;        combinedSDSSgaussianArrayRealTotalNumberOfObjects = 1000     ;max number of objects supported
      by webform
;    endif
;    if(useMethod3KSwebFormSDSSKeplerModelMassDistribution eq 1) then begin
;        combinedSDSSgaussianArrayRealTotalNumberOfObjects = 28  ;DD number of objects
;        SDSSgaussianArrayXIncrement = 0.1            ;DD number of objects
;
;    endif
;endif

SDSSmaximumMass = 1.5
SDSSgaussianArrayXCurrent = 0.0
SDSSgaussianArrayNumberOfPoints = SDSSmaximumMass/SDSSgaussianArrayXIncrement
SDSSgaussianArrayX = dblarr(SDSSgaussianArrayNumberOfPoints)
;PRINT, "SDSSgaussianArrayNumberOfPoints = ", SDSSgaussianArrayNumberOfPoints

for SDSSgaussianArrayIndex=0, SDSSgaussianArrayNumberOfPoints-1, 1 do begin
    SDSSgaussianArrayXCurrent = SDSSgaussianArrayXCurrent + SDSSgaussianArrayXIncrement
    SDSSgaussianArrayX[SDSSgaussianArrayIndex] = SDSSgaussianArrayXCurrent
endfor

SDSSgaussianArrayComponent1sigma = 0.081
SDSSgaussianArrayComponent2sigma = 0.034
SDSSgaussianArrayComponent3sigma = 0.201
SDSSgaussianArrayComponent4sigma = 0.076
SDSSgaussianArrayComponent5sigma = 0.037

SDSSgaussianArrayComponent1magnitude = 5.965e-6
SDSSgaussianArrayComponent2magnitude = 1.203e-5
SDSSgaussianArrayComponent3magnitude = 1.165e-6
SDSSgaussianArrayComponent4magnitude = 1.455e-6
SDSSgaussianArrayComponent5magnitude = 8.305e-7

SDSSgaussianArrayComponent1centre = 0.603
SDSSgaussianArrayComponent2centre = 0.571
SDSSgaussianArrayComponent3centre = 0.775
SDSSgaussianArrayComponent4centre = 1.175
SDSSgaussianArrayComponent5centre = 0.358

SDSSgaussianArrayComponent1fraction = 38
SDSSgaussianArrayComponent2fraction = 32
SDSSgaussianArrayComponent3fraction = 19
SDSSgaussianArrayComponent4fraction = 9
SDSSgaussianArrayComponent5fraction = 2

SDSSgaussianArrayComponent1 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent1magnitude,
    SDSSgaussianArrayComponent1centre,SDSSgaussianArrayComponent1sigma], DERIV )
SDSSgaussianArrayComponent2 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent2magnitude,
    SDSSgaussianArrayComponent2centre,SDSSgaussianArrayComponent2sigma], DERIV )
SDSSgaussianArrayComponent3 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent3magnitude,
    SDSSgaussianArrayComponent3centre,SDSSgaussianArrayComponent3sigma], DERIV )
SDSSgaussianArrayComponent4 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent4magnitude,
    SDSSgaussianArrayComponent4centre,SDSSgaussianArrayComponent4sigma], DERIV )
SDSSgaussianArrayComponent5 = gaussian(SDSSgaussianArrayX, [SDSSgaussianArrayComponent5magnitude,
    SDSSgaussianArrayComponent5centre,SDSSgaussianArrayComponent5sigma], DERIV )

combinedSDSSgaussianArray = SDSSgaussianArrayComponent1 + SDSSgaussianArrayComponent2 +
    SDSSgaussianArrayComponent3 + SDSSgaussianArrayComponent4 + SDSSgaussianArrayComponent5


;plot, MASS_BIN_DB8, FREQUENCY_DB8
plot, SDSSgaussianArrayX, combinedSDSSgaussianArray
oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent1
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent2
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent3
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent4
;oplot, SDSSgaussianArrayX, SDSSgaussianArrayComponent5

;wait, 5

; now extract number of components at each mass interval/bin
;...
;now integrate over the curve to obtain the total area, and divide by the total number of stars in
    this survey

    ;find total area
totalAreaUnderCombinedSDSSgaussianArray = 0
for SDSSgaussianArrayIndex=0, SDSSgaussianArrayNumberOfPoints-1, 1 do begin
    totalAreaUnderCombinedSDSSgaussianArray = totalAreaUnderCombinedSDSSgaussianArray +
        combinedSDSSgaussianArray[SDSSgaussianArrayIndex]
endfor
    ;now normalise
for SDSSgaussianArrayIndex=0, SDSSgaussianArrayNumberOfPoints-1, 1 do begin
    combinedSDSSgaussianArray[SDSSgaussianArrayIndex] = combinedSDSSgaussianArray[
        SDSSgaussianArrayIndex]/totalAreaUnderCombinedSDSSgaussianArray*
        combinedSDSSgaussianArrayRealTotalNumberOfObjects
```

```
endfor
;oplot, SDSSgaussianArrayX, combinedSDSSgaussianArray


    ;now generate SDSSmassesArray
SDSSmassesArray = dblarr(combinedSDSSgaussianArrayRealTotalNumberOfObjects)
SDSSgaussianArrayIndex = 0
combinedSDSSgaussianArrayTemp = combinedSDSSgaussianArray
for SDSSmassesArrayIndex=0, combinedSDSSgaussianArrayRealTotalNumberOfObjects-1, 0 do begin
    ;PRINT, "combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex] = ",
        combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex]
    if(combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex] lt 0.45) then begin
        SDSSgaussianArrayIndex = SDSSgaussianArrayIndex + 1
        ;PRINT, "here2"
    endif else begin
        combinedSDSSgaussianArrayTemp[SDSSgaussianArrayIndex] = combinedSDSSgaussianArrayTemp[
            SDSSgaussianArrayIndex] - 1
        SDSSmassesArray[SDSSmassesArrayIndex] = SDSSgaussianArrayIndex/
            SDSSgaussianArrayNumberOfPoints*SDSSmaximumMass
        SDSSmassesArrayIndex = SDSSmassesArrayIndex + 1
        ;PRINT, "here"
    endelse
endfor

kstwo, SDSSmassesArray, MASS_DB3, SDSS_MASSES_D, SDSS_MASSES_prob
PRINT,"Method 6: "
PRINT,"SDSS_MASSES_D = ", SDSS_MASSES_D
PRINT,"SDSS_MASSES_prob = ", SDSS_MASSES_prob

outputFileName6a='METHOD6DSSVmaxCorrectedKeplerModelMasses.txt'
OPENW,1,outputFileName6a
PRINTF,1,SDSSmassesArray,FORMAT='(F)'
CLOSE,1

outputFileName6b='METHOD6SDSSVmaxCorrectedKeplerModelMassDistribution.txt'
OPENW,1,outputFileName6b
PRINTF,1,combinedSDSSgaussianArray,FORMAT='(F)'
CLOSE,1




plotTitle = 'Mass Distribution Compaison'
plotXAxisTitle = 'WD mass'
plotYAxisTitle = 'frequency'
plotXmin = 0.0
plotXmax = 1.5
plotYmin = 0.0
plotYmax = 20.0
fileNameBase1 = 'WDversusDDmassDistribution'
fileNameBase2 = ''
asciiTablefileNameExtension = '.txt'
psGraphfileNameExtension = '.ps'
pdfGraphfileNameExtension = '.pdf'
numPlots = 2
lineStyleY = intarr(numPlots)
lineStyleY[0] = 3
lineStyleY[1] = 0
psymY = intarr(numPlots)
psymY[0]=-0
psymY[1]=-0
normalise = 1
specifyYRange = 1

useGDL = 1
xMarginDefault = 0
yMarginDefault = 0
xPlotDefault = 0.0
yPlotDefault = 0.0
if(useGDL eq 1) then begin
    xMarginDefault = 32
    yMarginDefault = 18
    xPlotDefault = 0
    yPlotDefault = 0
endif else begin
    xMarginDefault = 3
    yMarginDefault = 1
    xPlotDefault = [1e10,1e10]
    yPlotDefault = [1e10,1e10]
endelse


; print normalised pg_survey vmax corrected mass distribution and DD mass distribution
;PRINT, "print normalised pg_survey vmax corrected mass distribution and DD mass distribution"

;applying the normalisation method as defined in SDSSDDobjectTable-07Mar11b.ods (NB normalisation
    was used in the formation of WDmassDistributionPGsurveyVmaxCorrectedWithCoarseBinning.txt, but
    not WDmassDistributionPGsurveyVmaxCorrected.txt);
summedDDSurveyFrequency = TOTAL(FREQUENCY_DB1)/(SIZE(FREQUENCY_DB1))[1]
summedFieldFrequency = TOTAL(FREQUENCY_DB5)/(SIZE(FREQUENCY_DB5))[1]
maxDDSurveyFrequency = MAX(FREQUENCY_DB1)
```

```
maxFieldFrequency = MAX(FREQUENCY_DB5)
FieldFrequencyNormalised = FREQUENCY_DB5/summedFieldFrequency*summedDDSurveyFrequency
    ;FieldFrequencyNormalised2 = FREQUENCY_DB5/maxFieldFrequency*summedDDSurveyFrequency/2
    ;FieldFrequencyNormalised3 = FREQUENCY_DB5/maxFieldFrequency*maxDDSurveyFrequency
    ;maxFieldFrequencyNormalised = MAX(FieldFrequencyNormalised)
    ;plotYmax = maxFieldFrequencyNormalised
FieldFrequencyNormalisedDB5 = FieldFrequencyNormalised

; print normalised kepler mass distribution and DD mass distribution
;PRINT, "print normalised kepler mass distribution and DD mass distribution"

;applying the normalisation method as defined in SDSSDDobjectTable-07Mar11b.ods (NB normalisation
    was used in the formation of WDmassDistributionPGsurveyVmaxCorrectedWithCoarseBinning.txt, but
    not WDmassDistributionPGsurveyVmaxCorrected.txt);
summedDDSurveyFrequency = TOTAL(FREQUENCY_DB1)/(SIZE(FREQUENCY_DB1))[1]
summedFieldFrequency = TOTAL(FREQUENCY_DB6)/(SIZE(FREQUENCY_DB6))[1]
maxDDSurveyFrequency = MAX(FREQUENCY_DB1)
maxFieldFrequency = MAX(FREQUENCY_DB6)
FieldFrequencyNormalised = FREQUENCY_DB6/summedFieldFrequency*summedDDSurveyFrequency
    ;FieldFrequencyNormalised2 = FREQUENCY_DB6/maxFieldFrequency*summedDDSurveyFrequency/2
    ;FieldFrequencyNormalised3 = FREQUENCY_DB6/maxFieldFrequency*maxDDSurveyFrequency
    ;maxFieldFrequencyNormalised = MAX(FieldFrequencyNormalised)
    ;plotYmax = maxFieldFrequencyNormalised
FieldFrequencyNormalisedDB6 = FieldFrequencyNormalised


; print normalised kepler vmax corrected mass distribution and DD mass distribution
;PRINT, "print normalised kepler vmax corrected mass distribution and DD mass distribution"

;applying the normalisation method as defined in SDSSDDobjectTable-07Mar11b.ods (NB normalisation
    was used in the formation of WDmassDistributionPGsurveyVmaxCorrectedWithCoarseBinning.txt, but
    not WDmassDistributionPGsurveyVmaxCorrected.txt);
summedDDSurveyFrequency = TOTAL(FREQUENCY_DB1)/(SIZE(FREQUENCY_DB1))[1]
summedFieldFrequency = TOTAL(FREQUENCY_DB8)/(SIZE(FREQUENCY_DB8))[1]
maxDDSurveyFrequency = MAX(FREQUENCY_DB1)
maxFieldFrequency = MAX(FREQUENCY_DB8)
FieldFrequencyNormalised = FREQUENCY_DB8/summedFieldFrequency*summedDDSurveyFrequency
    ;FieldFrequencyNormalised2 = FREQUENCY_DB8/maxFieldFrequency*summedDDSurveyFrequency/2
    ;FieldFrequencyNormalised3 = FREQUENCY_DB8/maxFieldFrequency*maxDDSurveyFrequency
    ;maxFieldFrequencyNormalised = MAX(FieldFrequencyNormalised)
    ;plotYmax = maxFieldFrequencyNormalised
FieldFrequencyNormalisedDB8 = FieldFrequencyNormalised


plotToFile = 1
if(plotToFile eq 0) then begin
endif else begin
    fileNameBase1 = 'WDversusDDmassDistribution'
    psGraphfileName = fileNameBase1 + psGraphfileNameExtension
    pdfGraphfileName = fileNameBase1 + pdfGraphfileNameExtension

    set_plot,'ps'
    device,filename=psGraphfileName, /LANDSCAPE      ;/PORTRAIT  /LANDSCAPE       ;SET_RESOLUTION
        =[320,200]    ;SET_CHARACTER_SIZE=[200,40]
endelse

plot, MASS_BIN_DB1, FREQUENCY_DB1, psym=-0, LINESTYLE=0, COLOR=0, xtitle=plotXAxisTitle, ytitle=
    plotYAxisTitle, xrange=[plotXmin, plotXmax], yrange=[plotYmin, plotYmax], BACKGROUND=1,
    CHARSIZE=2.0,XMARGIN=xMarginDefault,YMARGIN=yMarginDefault
oplot, MASS_BIN_DB5, FieldFrequencyNormalisedDB5, psym=-0, LINESTYLE=1, COLOR=0
oplot, MASS_BIN_DB6, FieldFrequencyNormalisedDB6, psym=-0, LINESTYLE=2, COLOR=0
oplot, MASS_BIN_DB8, FieldFrequencyNormalisedDB8, psym=-0, LINESTYLE=3, COLOR=0

if(plotToFile eq 0) then begin
endif else begin
    device,/close
    ;convert ps to pdf;
    convertPStoPDFcommand = "convert " + psGraphfileName + " " + pdfGraphfileName
    spawn,convertPStoPDFcommand,result
endelse


wait, 1



end
```

A.7 List of Documentation

A.7.1 Object Table

All WD parameters from the project are stored in the object table spreadsheet (also serving as a data model process guideline);

SDSSDDobjectTable.ods

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	**SDSS Data**						http://skyserver.sdss3.org/dr8/en/tools/crossid/crossid.asp				CHECKS									
2	Survey Paper DD Name		SDSS Name (SDSS Jx)	SDSS ID	RA (decimal)	Dec (decimal)	DR8 SDSS ID	DR8 RA (decimal)	DR8 Dec (decima	Survey Paper DD No	Reduction h	igher flux at u	SDSS spectra	SDSS Link				SDSS g field g	SDSS i field ga	separation (arc
3	System nam	WD name									is reis table A	brighter than	table B?							
4																				
5	="DD-" & J5="DD-" & J5&M5		005212.73+135301.1	587724231565246516	13.05304438	13.88365572	1237653650766757964	13.05299637	13.88363857	01	0	0	A	=J5	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D5	3.320		0.00188254
6	="DD-" & J6="DD-" & J6&M6		005212.26+135302.0	587724231565246515	13.05112233	13.88390554	1237653650766757962	13.05107155	13.88389105	01	0	0	B	=J6	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D6	3.320		0.00188254
7	="DD-" & J7="DD-" & J7&M7		150746.48+521002.1	587733604259332157	226.94369130	52.16725612	1237659325484236937	226.94366050	52.16732740	02	1	1	A	=J7	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D7	3.845		0.00140200
8	="DD-" & J8="DD-" & J8&M8		150746.80+520958.0	587733604259332160	226.94503382	52.16612142	1237659325484236938	226.94500347	52.16619274	02	1	1	B	=J8	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D8	3.845		0.00140200
9	="DD-" & J9="DD-" & J9&M9		170355.91+330438.4	587729752747475228	255.98296977	33.07733943	1237655501887963481	255.98297339	33.07733460	03	0	0	A	=J9	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D9	4.050		0.00310129
10	="DD-" & J1="DD-" & J10&M10		170356.77+330435.7	587729752747475236	255.98656627	33.07660731	1237655501887963483	255.98657446	33.07660246	03	0	0	B	=J10	h	y	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D10	4.050		0.00310129
11	="DD-" & J1="DD-" & J11&M11		225932.74+140444.2	587727221943959713	344.88642274	14.07895147	1237652943168864403	344.88642937	14.07895109	04	0	0	A	=J11	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D11	3.845		0.00253760
12	="DD-" & J1="DD-" & J12&M12		225932.21+140439.2	587727221943959712	344.88422825	14.07756995	1237652943168864402	344.88423489	14.07756995	04	0	0	B	=J12	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D12	3.845		0.00253760
13	="DD-" & J1="DD-" & J13&M13		222301.62+220131.3	587740712426537180	335.75675515	22.02537688	1237666433651441913	335.75675463	22.02537842	05	1	1	A	=J13	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D13	3.845		0.00182340
14	="DD-" & J1="DD-" & J14&M14		222301.72+220124.9	587740712426537181	335.75717854	22.02359622	1237666433651441914	335.75717804	22.02359775	05	1	1	B	=J14	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D14	3.845		0.00182340
15	="DD-" & J1="DD-" & J15&M15		021131.52+171428.3	758881524261781580	32.88134033	17.24121308	1237670459646607418	32.88133929	17.24120166	06	1	1	A	=J15	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D15	4.035	4.760	0.00056563
16	="DD-" & J1="DD-" & J16&M16		021131.51+171430.4	758881524261781581	32.88133036	17.24177863	1237670459646607419	32.88133195	17.24179269	06	1	1	B	=J16	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D16	4.035	4.760	0.00056563
17	="DD-" & J1="DD-" & J17&M17		033236.86-004936.9	588015508218904600	53.15359480	-0.82692479	1237660239778152487	53.15358969	-0.82693267	07	1	1	A	=J17	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D17	3.855		0.00525253
18	="DD-" & J1="DD-" & J18&M18		033236.60-004918.4	588015508218904597	53.15251716	-0.82178397	1237660239778152484	53.15251139	-0.82178964	07	1	1	B	=J18	l	y	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D18	3.855		0.00525253
19	="DD-" & J1="DD-" & J19&M19		092646.88+132134.5	587745403612430400	141.69533369	13.35958843	1237671124837335126	141.69532655	13.35959067	08	1	1	A	=J19	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D19	4.050		0.00120818
20	="DD-" & J2="DD-" & J20&M20		092647.00+132138.4	587745403612430401	141.69584285	13.36069038	1237671124837335127	141.69583638	13.36069656	08	1	1	B	=J20	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D20	4.050		0.00120818
21	="DD-" & J2="DD-" & J21&M21		115937.81+134413.9	587735346962432035	179.90758042	13.73720478	1237661068187336754	179.90757041	13.73721126	09	0	1	A	=J21	l	y	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D21	3.320		0.00143782
22	="DD-" & J2="DD-" & J22&M22		115937.82+134408.7	587735346962432036	179.90760447	13.73576715	1237661068187336755	179.90759441	13.73577356	09	0	1	B	=J22	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D22	3.320		0.00143782
23	="DD-" & J2="DD-" & J23&M23		131332.14+203039.6	587742578058395699	198.38394049	20.51100050	1237668299283300391	198.38393928	20.51100405	10	0	0	A	=J23	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D23	4.035		0.00164622
24	="DD-" & J2="DD-" & J24&M24		131332.56+203039.3	587742578058395698	198.38569698	20.51094081	1237668299283300390	198.38569599	20.51094406	10	0	0	B	=J24	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D24	4.035		0.00164622
25	="DD-" & J2="DD-" & J25&M25		132814.28+163151.5	587742774031024218	202.05953933	16.53098677	1237668589730201656	202.05953613	16.53098515	11	1	1	A	=J25	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D25	3.845	4.860	0.00036759
26	="DD-" & J2="DD-" & J26&M26		132814.36+163150.9	587742774031024219	202.05987454	16.53080830	1237668589730201657	202.05987535	16.53077256	11	1	1	B	=J26	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D26	3.845	4.860	0.00036759
27	="DD-" & J2="DD-" & J27&M27		222236.30-082808.0	587726878883381389	335.65128196	-8.46888943	1237652600108286093	335.65127612	-8.46887632	12	1	1	A	=J27	h	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D27	3.995		0.00119192
28	="DD-" & J2="DD-" & J28&M28		222236.56-082806.0	587726878883381390	335.65235447	-8.46834597	1237652600108286094	335.65234858	-8.46833311	12	1	1	B	=J28	l	n	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D28	3.995		0.00119192
29																	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D29			
30																	="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D30			
31																				
32		**Third Party WDs for mass distribution**																		
33		PG0922+162A		587742013534175308	141.30492201	16.02928106								13			="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D33	6 r= 4.895		
34		PG0922+162B		587742013534175307	141.30617216	16.02893043								13			="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D34	11 r = 4.72		
35		HS2240+125A		587727220868382858	340.62976052	12.83470668								14			="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D35			
36		HS2240+125B		587727220868382859	340.62638401	12.83399075								14			="http://cas.sdss.org/astro/en/tools/explore/obj.asp?id=" & D36			

	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI
1		extracted	extracted	what software program is this for?											
2	separation (arc seconds)	probability	probability	Survey Paper DD Name					POSS-II (Blue) positions (incompl		SDSS Positions		SDSS Positions		SDSS Positions combined
3				name	ra	dec	gain_r [2011D]	gain_g [2011C]	ra	dec	ra	dec	ra	dec	
4															
5	=MROUND(T5*60*60; 0.01)	98.02	99.05	="DD-" & J5&M5	=E5	=F5	4.710000		=R5 0 52 12.711	+13 53 1.56	00 52 12.73	+13 53 01.1	00:52:12.73	+13:53:01.1	=AE5 & " " & AF5
6	=MROUND(T6*60*60; 0.01)	98.02	99.05				4.710000		=R6 0 52 12.253	+13 53 2.58	00 52 12.26	+13 53 02.0	00:52:12.26	+13:53:02.0	=AE6 & " " & AF6
7	=MROUND(T7*60*60; 0.01)	98.28	97.03	="DD-" & J7&M7	=E7	=F7	4.720000		=R7		15 07 46.48	+52 10 02.1	15:07:46.48	+52:10:02.1	=AE7 & " " & AF7
8	=MROUND(T8*60*60; 0.01)	98.28	97.03				4.720000		=R8		15 07 46.80	+52 09 58.0	15:07:46.80	+52:09:58.0	=AE8 & " " & AF8
9	=MROUND(T9*60*60; 0.01)	96.23	91.09	="DD-" & J9&M9	=E9	=F9	4.725000		=R9 17 3 55.950	+33 4 39.40	17 03 55.91	+33 04 38.4	17:03:55.91	+33:04:38.4	=AE9 & " " & AF9
10	=MROUND(T10*60*60; 0.01)	96.23	91.09				4.725000		=R10 17 3 56.802	+33 4 36.71	17 03 56.77	+33 04 35.7	17:03:56.77	+33:04:35.7	=AE10 & " " & AF10
11	=MROUND(T11*60*60; 0.01)	97.25	98	="DD-" & J11&M11	=E11	=F11	4.720000		=R11 22 59 32.742	+14 4 44.55	22 59 32.74	+14 04 44.2	22:59:32.74	+14:04:44.2	=AE11 & " " & AF11
12	=MROUND(T12*60*60; 0.01)	97.25	98				4.720000		=R12 22 59 32.210	+14 4 39.54	22 59 32.21	+14 04 39.2	22:59:32.21	+14:04:39.2	=AE12 & " " & AF12
13	=MROUND(T13*60*60; 0.01)	98.07	98.84	="DD-" & J13&M13	=E13	=F13	4.720000		=R13		22 23 01.62	+22 01 31.3	22:23:01.62	+22:01:31.3	=AE13 & " " & AF13
14	=MROUND(T14*60*60; 0.01)	98.07	98.84				4.720000		=R14		22 23 01.72	+22 01 24.9	22:23:01.72	+22:01:24.9	=AE14 & " " & AF14
15	=MROUND(T15*60*60; 0.01)	97.94	96.54	="DD-" & J15&M15	=E15	=F15	4.895000		=R15 2 11 31.461	+17 14 29.22	02 11 31.52	+17 14 28.3	02:11:31.52	+17:14:28.3	=AE15 & " " & AF15
16	=MROUND(T16*60*60; 0.01)	97.94	96.54				4.895000		=R16 2 11 31.461	+17 14 29.22	02 11 31.51	+17 14 30.4	02:11:31.51	+17:14:30.4	=AE16 & " " & AF16
17	=MROUND(T17*60*60; 0.01)	90.14	25.92	="DD-" & J17&M17	=E17	=F17	4.600000		=R17		03 32 36.86	-00 49 36.9	03:32:36.86	-00:49:36.9	=AE17 & " " & AF17
18	=MROUND(T18*60*60; 0.01)	90.14	25.92				4.600000		=R18		03 32 36.60	-00 49 18.4	03:32:36.60	-00:49:18.4	=AE18 & " " & AF18
19	=MROUND(T19*60*60; 0.01)	98.28	96.19	="DD-" & J19&M19	=E19	=F19	4.725000		=R19		09 26 46.88	+13 21 34.5	09:26:46.88	+13:21:34.5	=AE19 & " " & AF19
20	=MROUND(T20*60*60; 0.01)	98.28	96.19				4.725000		=R20		09 26 47.00	+13 21 38.4	09:26:47.00	+13:21:38.4	=AE20 & " " & AF20
21	=MROUND(T21*60*60; 0.01)	98.26	97.2	="DD-" & J21&M21	=E21	=F21	4.710000		=R21 11 59 37.845	+13 44 14.31	11 59 37.81	+13 44 13.9	11:59:37.81	+13:44:13.9	=AE21 & " " & AF21
22	=MROUND(T22*60*60; 0.01)	98.26	97.2				4.710000		=R22 11 59 37.848	+13 44 9.79	11 59 37.82	+13 44 08.7	11:59:37.82	+13:44:08.7	=AE22 & " " & AF22
23	=MROUND(T23*60*60; 0.01)	98.19	98.21	="DD-" & J23&M23	=E23	=F23	4.895000		=R23		13 13 32.14	+20 30 39.6	13:13:32.14	+20:30:39.6	=AE23 & " " & AF23
24	=MROUND(T24*60*60; 0.01)	98.19	98.21				4.895000		=R24		13 13 32.56	+20 30 39.3	13:13:32.56	+20:30:39.3	=AE24 & " " & AF24
25	=MROUND(T25*60*60; 0.01)	97.69	99.05	="DD-" & J25&M25	=E25	=F25	4.720000		=R25		13 28 14.28	+16 31 51.5	13:28:14.28	+16:31:51.5	=AE25 & " " & AF25
26	=MROUND(T26*60*60; 0.01)	97.69	99.05				4.720000		=R26		13 28 14.36	+16 31 50.9	13:28:14.36	+16:31:50.9	=AE26 & " " & AF26
27	=MROUND(T27*60*60; 0.01)	98.28	96.12	="DD-" & J27&M27	=E27	=F27	4.76000000		=R27		22 22 36.30	-08 28 08.0	22:22:36.30	-08:28:08.0	=AE27 & " " & AF27
28	=MROUND(T28*60*60; 0.01)	98.28	96.12				4.76000000		=R28		22 22 36.56	-08 28 06.0	22:22:36.56	-08:28:06.0	=AE28 & " " & AF28
29	=MROUND(T29*60*60; 0.01)														
30	=MROUND(T30*60*60; 0.01)														
31															
32															
33															
34															
35															
36				replace;											

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	**Observation Data**								\delta \lambda (FWHM in pixels/channels) [com			**Instrument Details;** see observingInstrumentDetails.ods for derivation												
2	WD name	Exposure Time(s)	Slit Width (arc	Grating	Observation R	Observatory	Instrument	Date of observati	instrument re	Arc Lamp Type	DC WD for flu	Instrument	Central Wavel	Central Wavel	Wavelength ra	dispersion	(A/mm	dispersion	(A/\delta \lambda (FW	\delta \lambda	\delta \lambda	pixelwidth (um	Slit Width (as)	resolution (lambd	resolution (lambd	slit width
3																										
4																										
5	=SDSS.B5	5x2.4 ks	1.0	R300B	SDSS0052	WHT 4.2m	Long Slit	[25/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M5	3539	64	0.86	=N5/X5/Z5	=R5/Q5	3.000	13.500	1.0	=M5/(4*Q5)	=W5*V5/Y5	1
6	=SDSS.B6	5x2.4 ks	1.0	R300B	SDSS0052	WHT 4.2m	Long Slit	[25/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M6	3539	64	0.86	=N6/X6/Z6	=R6/Q6	3.000	13.500	1.0	=M6/(4*Q6)	=W6*V6/Y6	1
7	=SDSS.B7	3x1.8 ks	1.0	R300B	SDSS1507	WHT 4.2m	Long Slit	[24/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M7	3539	64	0.86	=N7/X7/Z7	=R7/Q7	3.000	13.500	1.0	=M7/(4*Q7)	=W7*V7/Y7	1
8	=SDSS.B8	3x1.8 ks	1.0	R300B	SDSS1507	WHT 4.2m	Long Slit	[24/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M8	3539	64	0.86	=N8/X8/Z8	=R8/Q8	3.000	13.500	1.0	=M8/(4*Q8)	=W8*V8/Y8	1
9	=SDSS.B9	4x1.8+3x1.8 ks	1.0	R300B	SDSS1703	WHT 4.2m	Long Slit	[24+25/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M9	3539	64	0.86	=N9/X9/Z9	=R9/Q9	3.000	13.500	1.0	=M9/(4*Q9)	=W9*V9/Y9	1
10	=SDSS.B10	4x1.8+3x1.8 ks	1.0	R300B	SDSS1703	WHT 4.2m	Long Slit	[24+25/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M10	3539	64	0.86	=N10/X10/Z10	=R10/Q10	3.000	13.500	1.0	=M10/(4*Q10)	=W10*V10/Y10	1
11	=SDSS.B11	4x1.8+1.8+1.7+1.2 ks	1.0	R300B	SDSS2259	WHT 4.2m	Long Slit	[24+25/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M11	3539	64	0.86	=N11/X11/Z11	=R11/Q11	3.000	13.500	1.0	=M11/(4*Q11)	=W11*V11/Y11	1
12	=SDSS.B12	4x1.8+1.8+1.7+1.2 ks	1.0	R300B	SDSS2259	WHT 4.2m	Long Slit	[24+25/07/2008]	3	Cu-Ne + Cu-Ar	WD1918+38	IRIS blue	4503	=M12	3539	64	0.86	=N12/X12/Z12	=R12/Q12	3.000	13.500	1.0	=M12/(4*Q12)	=W12*V12/Y12	1
13	=SDSS.B13	3x0.75 ks	2.0	B600	DD-08A+B	Gemini North	Long Slit	[20/07/2009]	4.5	Cu-Ar	WD1918+38	GMOS-N	4100	4610	2760	=Q13*1000/U13	0.5	=N13/X13/Z13	=R13/Q13	4.500	13.500	0.5	1688	=W13*V13/Y13	2
14	=SDSS.B14	3x0.75 ks	2.0	B600	DD-08A+B	Gemini North	Long Slit	[20/07/2009]	4.5	Cu-Ar	WD1918+38	GMOS-N	4100	4610	2760	=Q14*1000/U14	0.5	=N14/X14/Z14	=R14/Q14	4.500	13.500	0.5	1688	=W14*V14/Y14	2
15	=SDSS.B15	3x2.0 ks	2.0	B600	DD-20A+B	Gemini North	Long Slit	[20+25/07/2009]	4.5	Cu-Ar	WD1918+38	GMOS-N	4100	4610	2760	=Q15*1000/U15	0.5	=N15/X15/Z15	=R15/Q15	4.500	13.500	0.5	1688	=W15*V15/Y15	2
16	=SDSS.B16	3x2.0 ks	2.0	B600	DD-20A+B	Gemini North	Long Slit	[20+25/07/2009]	4.5	Cu-Ar	WD1918+38	GMOS-N	4100	4610	2760	=Q16*1000/U16	0.5	=N16/X16/Z16	=R16/Q16	4.500	13.500	0.5	1688	=W16*V16/Y16	2
17	=SDSS.B17	2x0.6 ks	1.3	600B	WDB02	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M17	3300 - 6210	50	0.75	=N17/X17/Z17	=R17/Q17	3.500	15.000	1.0	780	=W17*V17/Y17	1.3
18	=SDSS.B18	2x0.6 ks	1.3	600B	WDB02	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M18	3300 - 6210	50	0.75	=N18/X18/Z18	=R18/Q18	3.500	15.000	1.0	780	=W18*V18/Y18	1.3
19	=SDSS.B19	1x0.5+1x0.36 ks	1.3	600B	WDB04	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M19	3300 - 6210	50	0.75	=N19/X19/Z19	=R19/Q19	3.500	15.000	1.0	780	=W19*V19/Y19	1.3
20	=SDSS.B20	1x0.5+1x0.36 ks	1.3	600B	WDB04	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M20	3300 - 6210	50	0.75	=N20/X20/Z20	=R20/Q20	3.500	15.000	1.0	780	=W20*V20/Y20	1.3
21	=SDSS.B21	2x0.6 ks	1.3	600B	WDB10	VLT-ANTU	LSS	[02/06/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M21	3300 - 6210	50	0.75	=N21/X21/Z21	=R21/Q21	3.500	15.000	1.0	780	=W21*V21/Y21	1.3
22	=SDSS.B22	2x0.6 ks	1.3	600B	WDB10	VLT-ANTU	LSS	[02/06/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M22	3300 - 6210	50	0.75	=N22/X22/Z22	=R22/Q22	3.500	15.000	1.0	780	=W22*V22/Y22	1.3
23	=SDSS.B23	2x0.3 ks	1.3	600B	WDB11	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M23	3300 - 6210	50	0.75	=N23/X23/Z23	=R23/Q23	3.500	15.000	1.0	780	=W23*V23/Y23	1.3
24	=SDSS.B24	2x0.3 ks	1.3	600B	WDB11	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M24	3300 - 6210	50	0.75	=N24/X24/Z24	=R24/Q24	3.500	15.000	1.0	780	=W24*V24/Y24	1.3
25	=SDSS.B25	2x0.3 ks	1.3	600B	WDB12	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M25	3300 - 6210	50	0.75	=N25/X25/Z25	=R25/Q25	3.500	15.000	1.0	780	=W25*V25/Y25	1.3
26	=SDSS.B26	2x0.3 ks	1.3	600B	WDB12	VLT-ANTU	LSS	[02/07/2010]	3.5	He-HgCd	LHS-2333	FORS2 with E	4650	=M26	3300 - 6210	50	0.75	=N26/X26/Z26	=R26/Q26	3.500	15.000	1.0	780	=W26*V26/Y26	1.3
27	=SDSS.B27	3x1.8 ks	2.0	B600	DD-07A+B	Gemini South	Long Slit	[11/05/2010]	4.5	Cu-Ar	WD0000+345	GMOS-S	4100	4610	2760	=Q27*1000/U27	0.5	=N27/X27/Z27	=R27/Q27	4.500	13.500	0.5	1688	=W27*V27/Y27	2
28	=SDSS.B28	3x1.8 ks	2.0	B600	DD-07A+B	Gemini South	Long Slit	[11/05/2010]	4.5	Cu-Ar	WD0000+345	GMOS-S	4100	4610	2760	=Q28*1000/U28	0.5	=N28/X28/Z28	=R28/Q28	4.500	13.500	0.5	1688	=W28*V28/Y28	2

	z
1	
2	binning irrelev
3	
4	
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	4
14	4
15	4
16	4
17	2
18	2
19	2
20	2
21	2
22	2
23	2
24	2
25	2
26	2
27	4
28	4

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	**Proper Motior**	**DDproperMotion**	**WCScalibr**	**(used asDDproperMotions.txt)**						**calcAveragePMofDDNearest150MDwarfObjects.p**			**calcAveragePMofDDNearest150WDwarfObje**			**from SDSS skyserver properm**						**from calculateRBBprobabilityOfChanceA**		
2		GAIA/Topcat	Proper motions	RA (decimal)	Dec (decimal)	GAIA/Topcat Proper motions				average PM of 150	Nearest M Dwarfs	probability Of	average PM of 150	Nearest W D	probability Of CI	SDSS-USNOB	SDSS-USNOB proper motions			DR8 SDSS-USN	(DR8 SDSS-USN	probability Of Chance Common Proper Moti			
3	WD name	pixels X	pixels Y			ra (mas/year)	dec (mas/year)	ra error (mas/y	dec error (mas/	ra (mas/year)	dec (mas/year)		ra (mas/year)	dec (mas/year)		ra (mas/year)	dec (mas/year)	ra error (mas/y	eadec error (mas/y	era (mas/year)	dec (mas/year)			
4																								
5	=SDSS.B5	3.8547	-0.3523	=SDSS.E5	=SDSS.F5	1.73850968	-36.55117513	8.01673132	5.96091002	-4.38000000	6.86000000	0.0000	19.2600	-15.7800	0.0133	18.24740000	-39.65219900	3.541020	3.541020	13.04112800	-39.43949900	0.2317		
6	=SDSS.B6	3.6437	0.2144	=SDSS.E6	=SDSS.F6	-3.56125584	-34.34100014	8.01075132	5.99895734							-2.97924000	-38.46889900	3.138230	3.138230	-7.82823600	-38.02074400			
7	=SDSS.B7	1.5526	2.7554	=SDSS.E7	=SDSS.F7	-28.01043587	2.91258914	19.58713680	1.54068467	-6.23000000	-1.64000000	0.0203	-11.1400	-3.0100	0.0037	-73.93920100	66.73200200	3.741530	3.741530	-74.30173500	67.70958700	0.6170		
8	=SDSS.B8	1.1961	3.1457	=SDSS.E8	=SDSS.F8	-29.00871149	7.51166090	19.59161602	1.60871703							0.00000000	0.00000000	0.000000	0.000000	0.00000000	0.00000000			
9	=SDSS.B9	2.5872	-4.8598	=SDSS.E9	=SDSS.F9	4.29303812	-50.20194740	9.45516579	1.82325458	-2.76000000	-3.60000000	0.0000	-8.0300	-2.3100	0.0000	-1.79195000	-51.16529800	3.405780	3.405780	0.10466600	-52.62974200	0.0786		
10	=SDSS.B10	2.5567	-5.0004	=SDSS.E10	=SDSS.F10	5.21682022	-51.12809890	9.45424969	1.70476319							0.31886500	-50.52290000	3.125070	3.125070	2.43021200	-51.82735100			
11	=SDSS.B11	1.3458	-2.6787	=SDSS.E11	=SDSS.F11	25.84051367	-10.30078321	4.21281200	4.91090898	2.75000000	-7.28000000	0.0368	9.9300	-8.9600	0.0000	28.09149900	-14.31910000	3.261640	3.261640	28.21968100	-14.00891500	0.1331		
12	=SDSS.B12	1.3266	-2.9730	=SDSS.E12	=SDSS.F12	28.53798700	-9.88933159	4.21850840	4.86216497							28.56500100	-16.31900000	2.752450	2.752450	28.68944500	-15.98317100			
13	=SDSS.B13	11.4193	-4.8810	=SDSS.E13	=SDSS.F13	60.85083005	-85.85138561	3.75552974	5.82300661	0.88000000	-1.71000000	0.0000	6.0700	-7.9100	0.0000	0.00000000	0.00000000	0.000000	0.000000	0.00000000	0.00000000	0.1221		
14	=SDSS.B14	8.3707	-4.7616	=SDSS.E14	=SDSS.F14	54.34549561	-60.85826582	3.74658428	5.82240223							23.89970000	-5.00742000	2.442280	2.442280	0.00000000	0.00000000			
15	=SDSS.B15	-0.5320	0.3016	=SDSS.E15	=SDSS.F15	18.56449018	-33.65875086	8.44047499	13.08278355	4.25000000	-4.62000000	0.0345	13.0300	-10.5300	0.0000	0.00000000	0.00000000	0.000000	0.000000	0.00000000	0.00000000	0.3395		
16	=SDSS.B16	-0.5757	0.2284	=SDSS.E16	=SDSS.F16	13.91580963	-36.32882012	9.04425590	13.50366262							0.00000000	0.00000000	0.000000	0.000000	24.01131600	-4.19816600			
17	=SDSS.B17	3.5187	5.7835	=SDSS.E17	=SDSS.F17	-51.96504081	30.75167300	9.69625565	7.42312490	6.20000000	-5.57000000	0.0000	5.4600	-10.2500	0.0000	-30.91420000	-23.30710000	4.475600	4.475600	-31.18883500	-23.28346800	0.0979		
18	=SDSS.B18	3.1108	5.8023	=SDSS.E18	=SDSS.F18	-52.09240421	27.15594858	9.67524106	7.40441898							-24.78809900	-23.63470100	5.386570	5.386570	-25.08047700	-23.48458900			
19	=SDSS.B19	8.3146	4.1688	=SDSS.E19	=SDSS.F19	-19.41259959	-74.43296313	0.70125205	5.29171835	-5.02000000	-4.53000000	0.0000	-7.6100	-10.2500	0.0000	-50.89450100	-142.98700000	3.881730	3.881730	-50.33237100	-142.28422500	0.0750		
20	=SDSS.B20	7.0938	2.1542	=SDSS.E20	=SDSS.F20	-5.27312166	-61.05242065	0.88365208	5.28271915							11.46060000	-5.05102000	2.677330	2.677330	11.60775800	-4.09162100			
21	=SDSS.B21	5.3160	2.1437	=SDSS.E21	=SDSS.F21	-17.80664125	-47.26580448	5.64621682	3.17333589	-8.88000000	-7.47000000	0.0038	-9.8800	-9.6000	0.0000	-22.41750000	-26.25429900	2.894380	2.894380	-23.08503900	-25.93358200	0.1155		
22	=SDSS.B22	4.5708	1.5048	=SDSS.E22	=SDSS.F22	-12.33708142	-40.56560845	5.67927102	3.19667604							-19.23170100	-77.06710100	2.868640	2.868640	-19.90151600	-76.77650500			
23	=SDSS.B23	-3.5421	1.4438	=SDSS.E23	=SDSS.F23	-9.76087409	30.55954415	5.60242936	5.94215840	-12.21000000	-3.70000000	0.0000	-21.3100	-13.6300	0.0000	-37.71830000	34.73820100	3.990630	3.990630	-37.43351400	34.64875800	0.2207		
24	=SDSS.B24	-3.2085	2.2588	=SDSS.E24	=SDSS.F24	-16.79247234	28.26307432	5.67187333	5.92569531							-32.05040000	24.88180000	4.655690	4.655690	-32.12209700	24.68237300			
25	=SDSS.B25	0.1770	-1.4138	=SDSS.E25	=SDSS.F25	-82.19857206	11.44932777	12.71053777	13.26234086	-6.28000000	-4.40000000	0.0000	-19.3900	-10.5100	0.0000	-84.00509600	23.29100000	2.323570	2.323570	-83.58644100	23.18941700	0.1749		
26	=SDSS.B26	0.2598	-1.1625	=SDSS.E26	=SDSS.F26	-67.49863589	16.00481630	12.75832778	13.31264510							0.00000000	0.00000000	0.000000	0.000000	0.00000000	0.00000000			
27	=SDSS.B27	3.4697	-0.8088	=SDSS.E27	=SDSS.F27	4.70231735	32.59634915	4.15278414	5.06828761	5.58000000	-9.62000000	0.0000	5.6100	-10.9700	0.0000	0.00000000	0.00000000	0.000000	0.000000	0.00000000	0.00000000	0.1678		
28	=SDSS.B28	4.3171	-0.6499	=SDSS.E28	=SDSS.F28	2.55623395	40.26453104	4.14035506	5.07008516							0.00000000	0.00000000	0.000000	0.000000	0.00000000	0.00000000			
29																								
30																								
31																								
32											**key:**					requires update (and all cells relying upon these cells also)								
33																wrong/not used								
34																warning								

	Y
1	**Alignment.pro**
2	on (using RBB
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	
34	

	A	B	C	D	E	F
1	**Signal to Noi***scalculateSignalToNoise* results					
2			Signal to noise of reduced data			
3	WD name	SN	Visually Calcul	Primary Model	Secondary Model Grid – after xspe	
4						
5	=SDSS.B5	12.15	=20/540	=C5/2	=C5*1.5	
6	=SDSS.B6	32.28	=70/630	0.060		
7	=SDSS.B7	64.3	=13/540	=C7/1.7		
8	=SDSS.B8	37.78	=15/600			
9	=SDSS.B9	26.83	=15/630	=C9/1.2		
10	=SDSS.B10	42	=30/640	=C10/1.5		
11	=SDSS.B11	31.76	=9/385	=C11/1.45		
12	=SDSS.B12	105.27	=20/540	=C12		
13	=SDSS.B13	136.93	=2.5/215	=C13/1.5		
14	=SDSS.B14	133.62	=3.5/320	=C14		
15	=SDSS.B15	148.28	=5/210	=C15/2.5		
16	=SDSS.B16	90.18	=5/260	=C16/1.6		
17	=SDSS.B17	89.46	=4/150	=C17/4		
18	=SDSS.B18	26.89	=15/350	=C18/1.2		
19	=SDSS.B19	32.07	=10/250			
20	=SDSS.B20	45.67	=10/340	=C20/1.7		
21	=SDSS.B21	23.44	=20/480	=C21/1.55		
22	=SDSS.B22	33.09	=10/290	=C22/1.7		
23	=SDSS.B23	43.22	=7.5/320	=C23/1.3		
24	=SDSS.B24	57.16	=5/370	=C24/0.85		
25	=SDSS.B25	81.16	=5/260	=C25/1.8		
26	=SDSS.B26	55.8	=7.5/340	=C26/1.8		
27	=SDSS.B27	99.72		0.018		
28	=SDSS.B28	94.65		0.018		

	A	B	C	D	E	F	G	H	I	J	K
1	**SDSS Photometric Data - original (before A/B switching derivations)**										
2	name	psfMag_u	psfMag_g	psfMag_r	psfMag_i	psfmag_z	psfMagErr_u	psfMagErr_g	psfMagErr_r	psfMagErr_i	psfMagErr_z
3											
4											
5	=SDSS.B5	19.35	18.89	18.92	19.05	19.27	0.03	0.03	0.02	0.02	0.08
6	=SDSS.B6	17.79	17.71	17.98	18.24	18.58	0.02	0.03	0.02	0.02	0.04
7	=SDSS.B7	17.14	16.91	17.29	17.55	17.84	0.02	0.03	0.01	0.02	0.02
8	=SDSS.B8	17.98	17.76	18.06	18.33	18.55	0.02	0.03	0.01	0.02	0.03
9	=SDSS.B9	19.16	18.81	18.86	18.97	19.11	0.02	0.01	0.01	0.02	0.04
10	=SDSS.B10	18.48	18.16	18.27	18.42	18.66	0.02	0.01	0.01	0.02	0.03
11	=SDSS.B11	19.02	18.57	18.68	18.85	19.08	0.03	0.02	0.01	0.02	0.05
12	=SDSS.B12	16.16	16.36	16.78	17.12	17.43	0.02	0.01	0.01	0.01	0.02
13	=SDSS.B13	15.66	15.6	15.91	16.21	16.48	0.01	0.02	0.01	0.01	0.02
14	=SDSS.B14	16.37	16.01	16.2	16.46	16.68	0.01	0.03	0.03	0.01	0.02
15	=SDSS.B15	16.69	16.76	17.19	17.46	17.7	0.03	0.03	0.03	0.03	0.03
16	=SDSS.B16	17.36	17.26	17.65	17.87	18.14	0.12	0.09	0.07	0.08	0.07
17	=SDSS.B17	15.32	15.64	16.09	16.41	16.75	0.01	0.02	0.02	0.02	0.02
18	=SDSS.B18	18.64	18.2	18.3	18.45	18.64	0.03	0.02	0.02	0.02	0.03
19	=SDSS.B19	18.46	18.34	18.39	18.5	18.6	0.02	0.02	0.02	0.02	0.03
20	=SDSS.B20	18.74	18.4	18.46	18.6	18.79	0.03	0.03	0.05	0.04	0.03
21	=SDSS.B21	18.45	18.07	18.12	18.16	18.3	0.03	0.02	0.02	0.02	0.03
22	=SDSS.B22	18.42	18.28	18.52	18.75	18.93	0.03	0.02	0.02	0.03	0.05
23	=SDSS.B23	18.13	17.8	17.98	18.19	18.43	0.02	0.02	0.01	0.02	0.03
24	=SDSS.B24	17.86	17.48	17.69	17.91	18.14	0.02	0.02	0.01	0.02	0.03
25	=SDSS.B25	16.34	16.27	16.63	16.99	17.21	0.02	0.02	0.02	0.02	0.03
26	=SDSS.B26	17.75	17.65	17.74	17.84	18.53	0.27	0.23	0.19	0.15	0.4
27	=SDSS.B27	16.68	16.41	16.67	16.92	17.19	0.02	0.02	0.03	0.03	0.02
28	=SDSS.B28	17.56	17.11	17.3	17.47	17.67	0.03	0.07	0.07	0.06	0.03
29											
30											
31											
32											
33	PG0922+162A	16.05	16.13	16.55	16.87	17.17	0.02	0.02	0.02	0.02	0.02
34	PG0922+162B	17.05	17.13	17.51	17.83	18.11	0.08	0.07	0.1	0.07	0.03
35	HS2240+125A	16.48	16.23	16.5	16.74	17.01	0.01	0.02	0.01	0.02	0.02
36	HS2240+125B	16.83	16.5	16.75	16.97	17.19	0.01	0.02	0.01	0.02	0.02

s

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	**Spectroscopic Data**			**./lin2xspB.pl**	recently updat	requires updat	**RBB xspec formal fit 19 May 10a – 3 June 10a – primary model grid – with lin2xpB + TLUSTY10v2**												**RBB xspec formal fit 3 June 10a – alternate model gri**			
2	name		telesxspec project name	model fit file name (8 characters) [.dat]	=SN.C3	=obs.T2	logg	logg error min	logg error max	Teff	Teff error min	Teff error max	redhift	reduced chi sq	comments	xspec name	model fit file name (8 char	Primary model	logg	logg error min	logg error max	Teff
3																						
4																						
5	=SDSS.B5	w	="@" & $D5 &".xcm"	="SSWD" & SDSS.N6 & SDSS.M5 & B6	=SN.C5	=obs.T5	8.11050	8.03067	8.19123	10648.0	10503.6	10804.4	0.00006440	0.89953630	OK	="@" & $D5 &".xcm"	=$D5	koester				
6	=SDSS.B6	w	="@" & $D6 &".xcm"	="SSWD" & SDSS.N5 & SDSS.M6 & B5	=SN.C6	=obs.T6	7.97101	7.94273	7.99969	19198.5	19058.5	19342.7	0.00006222	1.06777300	OK	="@" & $D6 &".xcm"	=$D6	koester	8.062060	crash	crash	18730.8
7	=SDSS.B7	w	="@" & $D7 &".xcm"	="SSWD" & SDSS.N7 & SDSS.M7 & B7	=SN.C7	=obs.T7	8.13190	8.07430	8.19023	17285.0	17057.7	17529.3	0.00006440	1.06150900	OK	="@" & $D7 &".xcm"	=$D7	koester	8.344020	8.307150	8.380920	17100.1
8	=SDSS.B8	w	="@" & $D8 &".xcm"	="SSWD" & SDSS.N8 & SDSS.M8 & B8	=SN.C8	=obs.T8	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	OK	="@" & $D8 &".xcm"	=$D8					
9	=SDSS.B9	w	="@" & $D9 &".xcm"	="SSWD" & SDSS.N10 & SDSS.M9 & B10	=SN.C9	=obs.T9	8.44889	8.34424	8.44451	10008.1	9797.1	9948.4	0.00008835	0.89841020	Suggest that	y="@" & $D9 &".xcm"	=$D9	koester				
10	=SDSS.B10	w	="@" & $D10 &".xcm"	="SSWD" & SDSS.N9 & SDSS.M10 & B9	=SN.C10	=obs.T10	8.25309	8.22635	8.27697	11222.3	11150.4	11300.4	0.00016228	0.99392900	Suggest that	y="@" & $D10 &".xcm"	=$D10	koester				
11	=SDSS.B11	w	="@" & $D11 &".xcm"	="SSWD" & SDSS.N12 & SDSS.M11 & B12	=SN.C11	=obs.T11	8.15810	8.08910	8.22893	13987.0	13648.1	14157.9	-0.00029886	1.01026500	Suggest that	y="@" & $D11 &".xcm"	=$D11	koester	8.540750	8.492650	8.589960	15870.4
12	=SDSS.B12	w	="@" & $D12 &".xcm"	="SSWD" & SDSS.N11 & SDSS.M12 & B11	=SN.C12	=obs.T12	8.44644	8.42278	8.46963	28066.6	27974.1	28170.9	0.00022332	0.89599010	Suggest that	y="@" & $D12 &".xcm"	=$D12	tlusty				
13	=SDSS.B13	g	="@" & $D13 &".xcm"	="SSWD" & SDSS.N13 & SDSS.M13 & B13	=SN.C13	=obs.T13	8.36914	8.35873	8.37967	19019.6	18961.7	19079.5	-1.385079E-04	0.82387340	all balmer line	="@" & $D13 & ".redshiftuntied" &".xcm"	=$D13	koester	8.441220	8.435680	8.450010	18384.1
14	=SDSS.B14	g	="@" & $D14 &".xcm"	="SSWD" & SDSS.N14 & SDSS.M14 & B14	=SN.C14	=obs.T14	8.07238	8.06164	8.08858	13950.4	13974.7	14008.8	-1.814109E-04	1.07660500	all balmer line	="@" & $D14 & ".redshiftuntied" &".xcm"	=$D14	koester	8.109860	8.096750	8.119270	13976.9
15	=SDSS.B15	g	="@" & $D15 &".xcm"	="SSWD" & SDSS.N15 & SDSS.M15 & B15	=SN.C15	=obs.T15	8.06434	8.04818	8.07882	24788.3	24667.9	24863.6	9.83E-005	1.15427700	all balmer line	="@" & $D15 & ".redshiftuntied" & ".xcm"	=$D15 & ".redshiftuntied"	tlusty				
16	=SDSS.B16	g	="@" & $D16 &".xcm"	="SSWD" & SDSS.N16 & SDSS.M16 & B16	=SN.C16	=obs.T16	8.05804	8.04344	8.07366	18502.2	18430	18562.4	8.71E-005	1.07799600	all balmer line	="@" & $D16 & ".redshiftuntied" &".xcm"	=$D16 & ".redshiftuntied"	tlusty				
17	=SDSS.B17	v	="@" & $D17 &".xcm"	="SSWD" & SDSS.N17 & SDSS.M17 & B17	=SN.C17	=obs.T17	7.99583	7.976140	8.014640	33443.8	33361.9	33493.4	0.00032910	0.91474710	Suggest that	y="@" & $D17 &".xcm"	=$D17	tlusty				
18	=SDSS.B18	v	="@" & $D18 &".xcm"	="SSWD" & SDSS.N18 & SDSS.M18 & B18	=SN.C18	=obs.T18	8.31067	8.24935	8.37166	10392.9	10293.9	10497.6	0.00035260	0.88246760	Suggest that	y="@" & $D18 &".xcm"	=$D18	koester				
19	=SDSS.B19	v	="@" & $D19 &".xcm"	="SSWD" & SDSS.N19 & SDSS.M19 & B19	=SN.C19	=obs.T19	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	DAH WD	OK	="@" & $D19 &".xcm"	=$D19					
20	=SDSS.B20	v	="@" & $D20 &".xcm"	="SSWD" & SDSS.N20 & SDSS.M20 & B20	=SN.C20	=obs.T20	8.56641	8.53670	8.59351	10492.0	10436.4	10536.1	0.00015657	1.06409100	OK	="@" & $D20 &".xcm"	=$D20	koester				
21	=SDSS.B21	v	="@" & $D21 &".xcm"	="SSWD" & SDSS.N21 & SDSS.M21 & B21	=SN.C21	=obs.T21	8.16630	8.14363	8.21118	11543.0	11423.6	11695.3	0.00029051	1.09586500	Suggest that	y="@" & $D21 &".xcm"	=$D21	koester				
22	=SDSS.B22	v	="@" & $D22 &".xcm"	="SSWD" & SDSS.N22 & SDSS.M22 & B22	=SN.C22	=obs.T22	8.93744	8.90574	8.96889	16037.2	15906.8	16347.2	0.00060184	1.09280000	Suggest that	y="@" & $D22 &".xcm"	=$D22	koester	8.750000	8.747540	8.750000	16215.2
23	=SDSS.B23	v	="@" & $D23 &".xcm"	="SSWD" & SDSS.N23 & SDSS.M23 & B23	=SN.C23	=obs.T23	8.42667	8.39340	8.45964	13072.2	12911.5	13236.1	0.00007636	0.91582360	Suggest that	y="@" & $D23 &".xcm"	=$D23	koester				
24	=SDSS.B24	v	="@" & $D24 &".xcm"	="SSWD" & SDSS.N24 & SDSS.M24 & B24	=SN.C24	=obs.T24	8.24378	8.21357	8.27763	12757.9	12604.5	12904.1	0.00001072	1.02139900	Suggest that	y="@" & $D24 &".xcm"	=$D24	koester				
25	=SDSS.B25	v	="@" & $D25 &".xcm"	="SSWD" & SDSS.N25 & SDSS.M25 & B25	=SN.C25	=obs.T25	8.19118	8.17301	8.21147	20000.3	19862.7	20086.9	0.00011470	0.84372430	Suggest that	y="@" & $D25 &".xcm"	=$D25	tlusty	8.101210	8.082290	8.126280	20000.0
26	=SDSS.B26	v	="@" & $D26 &".xcm"	="SSWD" & SDSS.N26 & SDSS.M26 & B26	=SN.C26	=obs.T26	8.44544	8.42631	8.46616	14036.7	13877.4	14166.2	0.00023544	1.11728200	Suggest that	y="@" & $D26 &".xcm"	=$D26	koester	8.446660	8.424380	8.466140	14024.1
27	=SDSS.B27	g	="@" & $D27 &".xcm"	="SSWD" & SDSS.N27 & SDSS.M27 & B27	=SN.C27	=obs.T27	8.01893	7.98992	8.04610	15834.2	15686.9	16011.4	-0.00003383	1.08781100	Suggest that	y="@" & $D27 & ".redshiftuntied" &".xcm"	=$D27	koester				
28	=SDSS.B28	g	="@" & $D28 &".xcm"	="SSWD" & SDSS.N28 & SDSS.M28 & B28	=SN.C28	=obs.T28	8.29001	8.26720	8.31314	11452.5	11379.1	11529.6	-0.00003275	0.85509630	Suggest that	y="@" & $D28 & ".redshiftuntied" &".xcm"	=$D28	koester				
29																						
30																						
31																						
32																						
33																						
34																						

s

	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW
1	**d – with lin2xpB + TLUSTY10v2**								**Magnetic?**	OLD WRONG	**RBB xspec primary grid fit versus alternate grid fit comparison – with lin2xpB + TLUSTY10v2**												**WD spec parm**				
2	Teff error min	Teff error max	redhift	reduced chi sq	comments	xspec name	model fit file n	Secondary m	odel grid (koester/tlusty)		prim logg	altern logg	prim teff	altern teff	prim redshift	altern redshift	comments	xspec name	model fit file n	Primary model	original name		name	Teff	Teff error	logg	logg error
3																											
4																											
5											=G5	=U5	=J5	=V5	=M5	=Y6	=O5	=P5	=Q5	=R5	=obs.E6		=SDSS.B5		=s.J5	=AT5*$AS$34	=s.G5	=($AT$33)
6	18583.5	18879.1	0.00006365	1.096	OK (except for		###	=#ref!.$#REF!#	tlusty		=G6	=U6	=J6	=V6	=M6	=Y7	=O6	=P6	=Q6	=R6	=obs.E5		=SDSS.B6		=s.J6	=AT6*$AS$34	=s.G6	=($AT$33)
7	16830.2	17387.8	0.00006440	0.913	OK		###	=#ref!.$#REF!#	tlusty	Magnetic	(magnetic WD		=G7	=U7	=J7	=V7	=M7	=Y8	=O7	=P7	=Q7	=R7	=obs.E7		=SDSS.B7		=s.J7	=AT7*$AS$34	=s.G7	=($AT$33)
8										Magnetic	(magnetic WD	=G8		=U8	=J8		=V8	=M8		=Y9	=O8	=P8	=Q8	=R8	=obs.E8		=SDSS.B8		18000.0		4000	0.00000	=($AT$33)
9											=G9		=U9	=J9		=V9	=M9			=Y10	=O9	=P9	=Q9	=R9	=obs.E10		=SDSS.B9		=s.J9	=AT9*$AS$34	=s.G9	=($AT$33)
10											=G10		=U10	=J10		=V10	=M10			=Y11	=O10	=P10	=Q10	=R10	=obs.E9		=SDSS.B10		=s.J10	=AT10*$AS$34	=s.G10	=($AT$33)
11	15387.3	16035.0	-0.00027281	1.072	Minimization n		###	=#ref!.$#REF!#	tlusty	Magnetic	(magnetic WD		=G11	=U11	=J11	=V11	=M11	=Y12	=O11	=P11	=Q11	=R11	=obs.E12		=SDSS.B11		=s.J11	=AT11*$AS$34	=s.G11	=($AT$33)
12											(magnetic WD		=G12	=U12	=J12	=V12	=M12	=Y13	=O12	=P12	=Q12	=R12	=obs.E11		=SDSS.B12		=s.J12	=AT12*$AS$34	=s.G12	=($AT$33)
13	18339.5	18421.5	-0.00014497	0.682	Suggest that y		###	=#ref!.$#REF!#	tlusty				=G13	=U13	=J13	=V13	=M13		=Y14	=O13	=P13	=Q13	=R13	=obs.E13		=SDSS.B13		=s.J13	=AT13*$AS$34	=s.G13	=($AT$33)
14	13990.8	14003.5	-0.00019095	1.393	Suggest that y		###	=#ref!.$#REF!#	tlusty				=G14	=U14	=J14	=V14	=M14		=Y15	=O14	=P14	=Q14	=R14	=obs.E14		=SDSS.B14		=s.J14	=AT14*$AS$34	=s.G14	=($AT$33)
15													=G15	=U15	=J15	=V15	=M15		=Y16	=O15	=P15	=Q15	=R15	=obs.E15		=SDSS.B15		=s.J15	=AT15*$AS$34	=s.G15	=($AT$33)
16													=G16	=U16	=J16	=V16	=M16		=Y17	=O16	=P16	=Q16	=R16	=obs.E16		=SDSS.B16		=s.J16	=AT16*$AS$34	=s.G16	=($AT$33)
17													=G17	=U17	=J17	=V17	=M17		=Y18	=O17	=P17	=Q17	=R17	=obs.E17		=SDSS.B17		=s.J17	=AT17*$AS$34	=s.G17	=($AT$33)
18													=G18	=U18	=J18	=V18	=M18		=Y19	=O18	=P18	=Q18	=R18	=obs.E18		=SDSS.B18		=s.J18	=AT18*$AS$34	=s.G18	=($AT$33)
19									Magnetic		=G19		=U19	=J19		=V19	=M19		=Y20	=O19	=P19	=Q19	=R19	=obs.E19		=SDSS.B19		9500.0		500.0	0.00000	=($AT$33)
20												=G20	=U20	=J20	=V20	=M20		=Y21	=O20	=P20	=Q20	=R20	=obs.E20		=SDSS.B20		=s.J20	=AT20*$AS$34	=s.G20	=($AT$33)
21												=G21	=U21	=J21	=V21	=M21		=Y22	=O21	=P21	=Q21	=R21	=obs.E21		=SDSS.B21		=s.J21	=AT21*$AS$34	=s.G21	=($AT$33)
22	15962.5	16076.6	0.00059240	1.298	Too massive fi		###	=#ref!.$#REF!#	tlusty			=G22	=U22	=J22	=V22	=M22		=Y23	=O22	=P22	=Q22	=R22	=obs.E22		=SDSS.B22		=s.J22	=AT22*$AS$34	=s.G22	=($AT$33)
23												=G23	=U23	=J23	=V23	=M23		=Y24	=O23	=P23	=Q23	=R23	=obs.E23		=SDSS.B23		=s.J23	=AT23*$AS$34	=s.G23	=($AT$33)
24												=G24	=U24	=J24	=V24	=M24		=Y25	=O24	=P24	=Q24	=R24	=obs.E24		=SDSS.B24		=s.J24	=AT24*$AS$34	=s.G24	=($AT$33)
25	19979.3	20000.0	0.00012575	0.816			###	=#ref!.$#REF!#	koester – just	out of range		=G25	=U25	=J25	=V25	=M25		=Y26	=O25	=P25	=Q25	=R25	=obs.E25		=SDSS.B25		=s.J25	=AT25*$AS$34	=s.G25	=($AT$33)
26	13978.3	14153.2	0.00022263	1.045	Suggest that y		###	=#ref!.$#REF!#	tlusty			=G26	=U26	=J26	=V26	=M26		=Y27	=O26	=P26	=Q26	=R26	=obs.E26		=SDSS.B26		=s.J26	=AT26*$AS$34	=s.G26	=($AT$33)
27												=G27	=U27	=J27	=V27	=M27		=Y28	=O27	=P27	=Q27	=R27	=obs.E27		=SDSS.B27		=s.J27	=AT27*$AS$34	=s.G27	=($AT$33)
28												=G28	=U28	=J28	=V28	=M28		=Y29	=O28	=P28	=Q28	=R28	=obs.E28		=SDSS.B28		=s.J28	=AT28*$AS$34	=s.G28	=($AT$33)
29																											
30																											
31																											
32																							Max temp errc	Max logg error in models			
33																							2.300	0.07			
34																							=0.01*AS33				

p

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	**Photometric D**	**Distance Check (compare absolute mag differences and relative magnitude differences to confirm DD system comp**						**Absolute Magnitude Difference Between Components**						
2	name	**Obs name**	**Paper name**	**obs**	**psfMag_u diff**	**psfMag_g diff**	**psfMag_r diff**	**abs mag (u) diff**	**abs mag (g) diff**	**abs mag (r) diff**	**u (app diff – ag (app diff – *ar* (app diff – a**			**Visual check**
3														
4					BAD – before switch						BAD – before switch			
5	=SDSS.B5	=obs.E6	="DD-"&SDSS=obs.F6		=pOrig.B5-pOrig.B6	=pOrig.C5-pOrig.C6	=pOrig.D5-pOrig.D6	=WDnorm.W5-WDnorm.W6	=WDnorm.X5-WDnorm.X6	=WDnorm.Y5-WDnorm.Y6	=E5-H5	=F5-I5	=G5-J5	0
6	=SDSS.B6													=N5
7	=SDSS.B7	=obs.E7	="DD-"&SDSS=obs.F7		=pOrig.B7-pOrig.B8	=pOrig.C7-pOrig.C8	=pOrig.D7-pOrig.D8	B is DC WD	B is DC WD	B is DC WD	N/A	N/A	N/A	0
8	=SDSS.B8													=N7
9	=SDSS.B9	=obs.E10	="DD-"&SDSS=obs.F10		=pOrig.B9-pOrig.B10	=pOrig.C9-pOrig.C10	=pOrig.D9-pOrig.D10	=WDnorm.W9-WDnorm.W10	=WDnorm.X9-WDnorm.X10	=WDnorm.Y9-WDnorm.Y10	=E9-H9	=F9-I9	=G9-J9	0
10	=SDSS.B10													=N9
11	=SDSS.B11	=obs.E12	="DD-"&SDSS=obs.F12		=pOrig.B11-pOrig.B12	=pOrig.C11-pOrig.C12	=pOrig.D11-pOrig.D12	=WDnorm.W11-WDnorm.W12	=WDnorm.X11-WDnorm.X12	=WDnorm.Y11-WDnorm.Y12	=E11-H11	=F11-I11	=G11-J11	0
12	=SDSS.B12													=N11
13	=SDSS.B13	=obs.E13	="DD-"&SDSS=obs.F13		=pOrig.B13-pOrig.B14	=pOrig.C13-pOrig.C14	=pOrig.D13-pOrig.D14	=WDnorm.W13-WDnorm.W14	=WDnorm.X13-WDnorm.X14	=WDnorm.Y13-WDnorm.Y14	=E13-H13	=F13-I13	=G13-J13	0
14	=SDSS.B14													=N13
15	=SDSS.B15	=obs.E15	="DD-"&SDSS=obs.F15		=pOrig.B15-pOrig.B16	=pOrig.C15-pOrig.C16	=pOrig.D15-pOrig.D16	=WDnorm.W15-WDnorm.W16	=WDnorm.X15-WDnorm.X16	=WDnorm.Y15-WDnorm.Y16	=E15-H15	=F15-I15	=G15-J15	0
16	=SDSS.B16													=N15
17	=SDSS.B17	=obs.E17	="DD-"&SDSS=obs.F17		=pOrig.B17-pOrig.B18	=pOrig.C17-pOrig.C18	=pOrig.D17-pOrig.D18	=WDnorm.W17-WDnorm.W18	=WDnorm.X17-WDnorm.X18	=WDnorm.Y17-WDnorm.Y18	=E17-H17	=F17-I17	=G17-J17	0
18	=SDSS.B18													=N17
19	=SDSS.B19	=obs.E19	="DD-"&SDSS=obs.F19		=pOrig.B19-pOrig.B20	=pOrig.C19-pOrig.C20	=pOrig.D19-pOrig.D20	A is DC WD	A is DC WD	A is DC WD	N/A	N/A	N/A	0
20	=SDSS.B20													0
21	=SDSS.B21	=obs.E21	="DD-"&SDSS=obs.F21		=pOrig.B21-pOrig.B22	=pOrig.C21-pOrig.C22	=pOrig.D21-pOrig.D22	=WDnorm.W21-WDnorm.W22	=WDnorm.X21-WDnorm.X22	=WDnorm.Y21-WDnorm.Y22	=E21-H21	=F21-I21	=G21-J21	0
22	=SDSS.B22													=N21
23	=SDSS.B23	=obs.E23	="DD-"&SDSS=obs.F23		=pOrig.B23-pOrig.B24	=pOrig.C23-pOrig.C24	=pOrig.D23-pOrig.D24	=WDnorm.W23-WDnorm.W24	=WDnorm.X23-WDnorm.X24	=WDnorm.Y23-WDnorm.Y24	=E23-H23	=F23-I23	=G23-J23	0
24	=SDSS.B24													=N23
25	=SDSS.B25	=obs.E25	="DD-"&SDSS=obs.F25		=pOrig.B25-pOrig.B26	=pOrig.C25-pOrig.C26	=pOrig.D25-pOrig.D26	=WDnorm.W25-WDnorm.W26	=WDnorm.X25-WDnorm.X26	=WDnorm.Y25-WDnorm.Y26	=E25-H25	=F25-I25	=G25-J25	0
26	=SDSS.B26													=N25
27	=SDSS.B27	=obs.E27	="DD-"&SDSS=obs.F27		=pOrig.B27-pOrig.B28	=pOrig.C27-pOrig.C28	=pOrig.D27-pOrig.D28	=WDnorm.W27-WDnorm.W28	=WDnorm.X27-WDnorm.X28	=WDnorm.Y27-WDnorm.Y28	=E27-H27	=F27-I27	=G27-J27	0
28	=SDSS.B28													=N27
29														
30														
31														
32														
33														=N32
34														=N33
35														=N34
36														=N35

p

	O	P	Q	R	S	T	U	V	W	X	Y	Z
1			Apparent Magnitude Difference Between Components			Difference Between Absolute Magnitude (S			SDSS Photometric Data - A/B component order Corrected to match spectroscopic components			
2	**Theoretical A/B component naming switch required?**	**Reason for co**	**psfMag_g diff**	**psfMag_r diff**	**psfMag_u diff**	**g (app diff – ar**	**(app diff – alu**	**(app diff – a**	name	psfMag_u	psfMag_g	psfMag_r
3												
4			GOOD – after switch									
5	=IF((((SDSS.K5=1) AND (SDSS.L5=1)) OR ((SDSS.K5=0) AND (SDSS.L5=0))); 0; 1)	makes sense	=IF($N5=1;-F5;F5)	=IF($N5=1;-G5;G5)	=IF($N5=1;-E5;E5)	=Q5-I5	=R5-J5	=S5-H5	=SDSS.B5	=IF($N5=0;pOrig.B5;pOrig.B6)	=IF($N5=0;pOrig.C5;pOrig.C6)	=IF($N5=0;pOrig.D5;pOrig.D6)
6		=O5							=SDSS.B6	=IF($N6=0;pOrig.B6;pOrig.B5)	=IF($N6=0;pOrig.C6;pOrig.C5)	=IF($N6=0;pOrig.D6;pOrig.D5)
7	=IF((((SDSS.K7=1) AND (SDSS.L7=1)) OR ((SDSS.K7=0) AND (SDSS.L7=0))); 0; 1)	DC WD – hard	=IF($N7=1;-F7;F7)	=IF($N7=1;-G7;G7)	=IF($N7=1;-E7;E7)				=SDSS.B7	=IF($N7=0;pOrig.B7;pOrig.B8)	=IF($N7=0;pOrig.C7;pOrig.C8)	=IF($N7=0;pOrig.D7;pOrig.D8)
8		=O7							=SDSS.B8	=IF($N8=0;pOrig.B8;pOrig.B7)	=IF($N8=0;pOrig.C8;pOrig.C7)	=IF($N8=0;pOrig.D8;pOrig.D7)
9	=IF((((SDSS.K9=1) AND (SDSS.L9=1)) OR ((SDSS.K9=0) AND (SDSS.L9=0))); 0; 1)	makes sense	=IF($N9=1;-F9;F9)	=IF($N9=1;-G9;G9)	=IF($N9=1;-E9;E9)	=Q9-I9	=R9-J9	=S9-H9	=SDSS.B9	=IF($N9=0;pOrig.B9;pOrig.B10)	=IF($N9=0;pOrig.C9;pOrig.C10)	=IF($N9=0;pOrig.D9;pOrig.D10)
10		=O9							=SDSS.B10	=IF($N10=0;pOrig.B10;pOrig.B9)	=IF($N10=0;pOrig.C10;pOrig.C9)	=IF($N10=0;pOrig.D10;pOrig.D9)
11		### makes sense	=IF($N11=1;-F11;F11)	=IF($N11=1;-G11;G11)	=IF($N11=1;-E11;E11)	=Q11-I11	=R11-J11	=S11-H11	=SDSS.B11	=IF($N11=0;pOrig.B11;pOrig.B12)	=IF($N11=0;pOrig.C11;pOrig.C12)	=IF($N11=0;pOrig.D11;pOrig.D12)
12		=O11							=SDSS.B12	=IF($N12=0;pOrig.B12;pOrig.B11)	=IF($N12=0;pOrig.C12;pOrig.C11)	=IF($N12=0;pOrig.D12;pOrig.D11)
13		### makes sense	=IF($N13=1;-F13;F13)	=IF($N13=1;-G13;G13)	=IF($N13=1;-E13;E13)	=Q13-I13	=R13-J13	=S13-H13	=SDSS.B13	=IF($N13=0;pOrig.B13;pOrig.B14)	=IF($N13=0;pOrig.C13;pOrig.C14)	=IF($N13=0;pOrig.D13;pOrig.D14)
14		=O13							=SDSS.B14	=IF($N14=0;pOrig.B14;pOrig.B13)	=IF($N14=0;pOrig.C14;pOrig.C13)	=IF($N14=0;pOrig.D14;pOrig.D13)
15		### makes sense	=IF($N15=1;-F15;F15)	=IF($N15=1;-G15;G15)	=IF($N15=1;-E15;E15)	=Q15-I15	=R15-J15	=S15-H15	=SDSS.B15	=IF($N15=0;pOrig.B15;pOrig.B16)	=IF($N15=0;pOrig.C15;pOrig.C16)	=IF($N15=0;pOrig.D15;pOrig.D16)
16		=O15							=SDSS.B16	=IF($N16=0;pOrig.B16;pOrig.B15)	=IF($N16=0;pOrig.C16;pOrig.C15)	=IF($N16=0;pOrig.D16;pOrig.D15)
17		### makes sense	=IF($N17=1;-F17;F17)	=IF($N17=1;-G17;G17)	=IF($N17=1;-E17;E17)	=Q17-I17	=R17-J17	=S17-H17	=SDSS.B17	=IF($N17=0;pOrig.B17;pOrig.B18)	=IF($N17=0;pOrig.C17;pOrig.C18)	=IF($N17=0;pOrig.D17;pOrig.D18)
18		=O17							=SDSS.B18	=IF($N18=0;pOrig.B18;pOrig.B17)	=IF($N18=0;pOrig.C18;pOrig.C17)	=IF($N18=0;pOrig.D18;pOrig.D17)
19		### DC WD – hard	=IF($N19=1;-F19;F19)	=IF($N19=1;-G19;G19)	=IF($N19=1;-E19;E19)				=SDSS.B19	=IF($N19=0;pOrig.B19;pOrig.B20)	=IF($N19=0;pOrig.C19;pOrig.C20)	=IF($N19=0;pOrig.D19;pOrig.D20)
20		0							=SDSS.B20	=IF($N20=0;pOrig.B20;pOrig.B19)	=IF($N20=0;pOrig.C20;pOrig.C19)	=IF($N20=0;pOrig.D20;pOrig.D19)
21		### makes sense	=IF($N21=1;-F21;F21)	=IF($N21=1;-G21;G21)	=IF($N21=1;-E21;E21)	=Q21-I21	=R21-J21	=S21-H21	=SDSS.B21	=IF($N21=0;pOrig.B21;pOrig.B22)	=IF($N21=0;pOrig.C21;pOrig.C22)	=IF($N21=0;pOrig.D21;pOrig.D22)
22		=O21							=SDSS.B22	=IF($N22=0;pOrig.B22;pOrig.B21)	=IF($N22=0;pOrig.C22;pOrig.C21)	=IF($N22=0;pOrig.D22;pOrig.D21)
23		### makes sense	=IF($N23=1;-F23;F23)	=IF($N23=1;-G23;G23)	=IF($N23=1;-E23;E23)	=Q23-I23	=R23-J23	=S23-H23	=SDSS.B23	=IF($N23=0;pOrig.B23;pOrig.B24)	=IF($N23=0;pOrig.C23;pOrig.C24)	=IF($N23=0;pOrig.D23;pOrig.D24)
24		=O23							=SDSS.B24	=IF($N24=0;pOrig.B24;pOrig.B23)	=IF($N24=0;pOrig.C24;pOrig.C23)	=IF($N24=0;pOrig.D24;pOrig.D23)
25		### makes sense	=IF($N25=1;-F25;F25)	=IF($N25=1;-G25;G25)	=IF($N25=1;-E25;E25)	=Q25-I25	=R25-J25	=S25-H25	=SDSS.B25	=IF($N25=0;pOrig.B25;pOrig.B26)	=IF($N25=0;pOrig.C25;pOrig.C26)	=IF($N25=0;pOrig.D25;pOrig.D26)
26		=O25							=SDSS.B26	=IF($N26=0;pOrig.B26;pOrig.B25)	=IF($N26=0;pOrig.C26;pOrig.C25)	=IF($N26=0;pOrig.D26;pOrig.D25)
27		### makes sense	=IF($N27=1;-F27;F27)	=IF($N27=1;-G27;G27)	=IF($N27=1;-E27;E27)	=Q27-I27	=R27-J27	=S27-H27	=SDSS.B27	=IF($N27=0;pOrig.B27;pOrig.B28)	=IF($N27=0;pOrig.C27;pOrig.C28)	=IF($N27=0;pOrig.D27;pOrig.D28)
28		=O27							=SDSS.B28	=IF($N28=0;pOrig.B28;pOrig.B27)	=IF($N28=0;pOrig.C28;pOrig.C27)	=IF($N28=0;pOrig.D28;pOrig.D27)
29												
30												
31												
32												
33		=O32							=SDSS.B33	=IF($N32=0;pOrig.B33;pOrig.B34)	=IF($N32=0;pOrig.C33;pOrig.C34)	=IF($N32=0;pOrig.D33;pOrig.D34)
34		=O33							=SDSS.B34	=IF($N33=0;pOrig.B34;pOrig.B35)	=IF($N33=0;pOrig.C34;pOrig.C35)	=IF($N33=0;pOrig.D34;pOrig.D35)
35		=O34							=SDSS.B35	=IF($N34=0;pOrig.B35;pOrig.B36)	=IF($N34=0;pOrig.C35;pOrig.C36)	=IF($N34=0;pOrig.D35;pOrig.D36)
36		=O35							=SDSS.B36	=IF($N35=0;pOrig.B36;pOrig.B36)	=IF($N35=0;pOrig.C36;pOrig.C36)	=IF($N35=0;pOrig.D36;pOrig.D36)

p

	AA	AB	AC	AD	AE	AF	AG
1							
2	psfMag_i	psfMag_z	psfMagErr_u	psfMagErr_g	psfMagErr_r	psfMagErr_i	psfMagErr_z
3							
4							
5	=IF($N5=0;pOrig.E5;pOrig.E6)	=IF($N5=0;pOrig.F5;pOrig.F6)	=IF($N5=0;pOrig.G5;pOrig.G6)	=IF($N5=0;pOrig.H5;pOrig.H6)	=IF($N5=0;pOrig.I5;pOrig.I6)	=IF($N5=0;pOrig.J5;pOrig.J6)	=IF($N5=0;pOrig.K5;pOrig.K6)
6	=IF($N6=0;pOrig.E6;pOrig.E5)	=IF($N6=0;pOrig.F6;pOrig.F5)	=IF($N6=0;pOrig.G6;pOrig.G5)	=IF($N6=0;pOrig.H6;pOrig.H5)	=IF($N6=0;pOrig.I6;pOrig.I5)	=IF($N6=0;pOrig.J6;pOrig.J5)	=IF($N6=0;pOrig.K6;pOrig.K5)
7	=IF($N7=0;pOrig.E7;pOrig.E8)	=IF($N7=0;pOrig.F7;pOrig.F8)	=IF($N7=0;pOrig.G7;pOrig.G8)	=IF($N7=0;pOrig.H7;pOrig.H8)	=IF($N7=0;pOrig.I7;pOrig.I8)	=IF($N7=0;pOrig.J7;pOrig.J8)	=IF($N7=0;pOrig.K7;pOrig.K8)
8	=IF($N8=0;pOrig.E8;pOrig.E7)	=IF($N8=0;pOrig.F8;pOrig.F7)	=IF($N8=0;pOrig.G8;pOrig.G7)	=IF($N8=0;pOrig.H8;pOrig.H7)	=IF($N8=0;pOrig.I8;pOrig.I7)	=IF($N8=0;pOrig.J8;pOrig.J7)	=IF($N8=0;pOrig.K8;pOrig.K7)
9	=IF($N9=0;pOrig.E9;pOrig.E10)	=IF($N9=0;pOrig.F9;pOrig.F10)	=IF($N9=0;pOrig.G9;pOrig.G10)	=IF($N9=0;pOrig.H9;pOrig.H10)	=IF($N9=0;pOrig.I9;pOrig.I10)	=IF($N9=0;pOrig.J9;pOrig.J10)	=IF($N9=0;pOrig.K9;pOrig.K10)
10	=IF($N10=0;pOrig.E10;pOrig.E9)	=IF($N10=0;pOrig.F10;pOrig.F9)	=IF($N10=0;pOrig.G10;pOrig.G9)	=IF($N10=0;pOrig.H10;pOrig.H9)	=IF($N10=0;pOrig.I10;pOrig.I9)	=IF($N10=0;pOrig.J10;pOrig.J9)	=IF($N10=0;pOrig.K10;pOrig.K9)
11	=IF($N11=0;pOrig.E11;pOrig.E12)	=IF($N11=0;pOrig.F11;pOrig.F12)	=IF($N11=0;pOrig.G11;pOrig.G12)	=IF($N11=0;pOrig.H11;pOrig.H12)	=IF($N11=0;pOrig.I11;pOrig.I12)	=IF($N11=0;pOrig.J11;pOrig.J12)	=IF($N11=0;pOrig.K11;pOrig.K12)
12	=IF($N12=0;pOrig.E12;pOrig.E11)	=IF($N12=0;pOrig.F12;pOrig.F11)	=IF($N12=0;pOrig.G12;pOrig.G11)	=IF($N12=0;pOrig.H12;pOrig.H11)	=IF($N12=0;pOrig.I12;pOrig.I11)	=IF($N12=0;pOrig.J12;pOrig.J11)	=IF($N12=0;pOrig.K12;pOrig.K11)
13	=IF($N13=0;pOrig.E13;pOrig.E14)	=IF($N13=0;pOrig.F13;pOrig.F14)	=IF($N13=0;pOrig.G13;pOrig.G14)	=IF($N13=0;pOrig.H13;pOrig.H14)	=IF($N13=0;pOrig.I13;pOrig.I14)	=IF($N13=0;pOrig.J13;pOrig.J14)	=IF($N13=0;pOrig.K13;pOrig.K14)
14	=IF($N14=0;pOrig.E14;pOrig.E13)	=IF($N14=0;pOrig.F14;pOrig.F13)	=IF($N14=0;pOrig.G14;pOrig.G13)	=IF($N14=0;pOrig.H14;pOrig.H13)	=IF($N14=0;pOrig.I14;pOrig.I13)	=IF($N14=0;pOrig.J14;pOrig.J13)	=IF($N14=0;pOrig.K14;pOrig.K13)
15	=IF($N15=0;pOrig.E15;pOrig.E16)	=IF($N15=0;pOrig.F15;pOrig.F16)	=IF($N15=0;pOrig.G15;pOrig.G16)	=IF($N15=0;pOrig.H15;pOrig.H16)	=IF($N15=0;pOrig.I15;pOrig.I16)	=IF($N15=0;pOrig.J15;pOrig.J16)	=IF($N15=0;pOrig.K15;pOrig.K16)
16	=IF($N16=0;pOrig.E16;pOrig.E15)	=IF($N16=0;pOrig.F16;pOrig.F15)	=IF($N16=0;pOrig.G16;pOrig.G15)	=IF($N16=0;pOrig.H16;pOrig.H15)	=IF($N16=0;pOrig.I16;pOrig.I15)	=IF($N16=0;pOrig.J16;pOrig.J15)	=IF($N16=0;pOrig.K16;pOrig.K15)
17	=IF($N17=0;pOrig.E17;pOrig.E18)	=IF($N17=0;pOrig.F17;pOrig.F18)	=IF($N17=0;pOrig.G17;pOrig.G18)	=IF($N17=0;pOrig.H17;pOrig.H18)	=IF($N17=0;pOrig.I17;pOrig.I18)	=IF($N17=0;pOrig.J17;pOrig.J18)	=IF($N17=0;pOrig.K17;pOrig.K18)
18	=IF($N18=0;pOrig.E18;pOrig.E17)	=IF($N18=0;pOrig.F18;pOrig.F17)	=IF($N18=0;pOrig.G18;pOrig.G17)	=IF($N18=0;pOrig.H18;pOrig.H17)	=IF($N18=0;pOrig.I18;pOrig.I17)	=IF($N18=0;pOrig.J18;pOrig.J17)	=IF($N18=0;pOrig.K18;pOrig.K17)
19	=IF($N19=0;pOrig.E19;pOrig.E20)	=IF($N19=0;pOrig.F19;pOrig.F20)	=IF($N19=0;pOrig.G19;pOrig.G20)	=IF($N19=0;pOrig.H19;pOrig.H20)	=IF($N19=0;pOrig.I19;pOrig.I20)	=IF($N19=0;pOrig.J19;pOrig.J20)	=IF($N19=0;pOrig.K19;pOrig.K20)
20	=IF($N20=0;pOrig.E20;pOrig.E19)	=IF($N20=0;pOrig.F20;pOrig.F19)	=IF($N20=0;pOrig.G20;pOrig.G19)	=IF($N20=0;pOrig.H20;pOrig.H19)	=IF($N20=0;pOrig.I20;pOrig.I19)	=IF($N20=0;pOrig.J20;pOrig.J19)	=IF($N20=0;pOrig.K20;pOrig.K19)
21	=IF($N21=0;pOrig.E21;pOrig.E22)	=IF($N21=0;pOrig.F21;pOrig.F22)	=IF($N21=0;pOrig.G21;pOrig.G22)	=IF($N21=0;pOrig.H21;pOrig.H22)	=IF($N21=0;pOrig.I21;pOrig.I22)	=IF($N21=0;pOrig.J21;pOrig.J22)	=IF($N21=0;pOrig.K21;pOrig.K22)
22	=IF($N22=0;pOrig.E22;pOrig.E21)	=IF($N22=0;pOrig.F22;pOrig.F21)	=IF($N22=0;pOrig.G22;pOrig.G21)	=IF($N22=0;pOrig.H22;pOrig.H21)	=IF($N22=0;pOrig.I22;pOrig.I21)	=IF($N22=0;pOrig.J22;pOrig.J21)	=IF($N22=0;pOrig.K22;pOrig.K21)
23	=IF($N23=0;pOrig.E23;pOrig.E24)	=IF($N23=0;pOrig.F23;pOrig.F24)	=IF($N23=0;pOrig.G23;pOrig.G24)	=IF($N23=0;pOrig.H23;pOrig.H24)	=IF($N23=0;pOrig.I23;pOrig.I24)	=IF($N23=0;pOrig.J23;pOrig.J24)	=IF($N23=0;pOrig.K23;pOrig.K24)
24	=IF($N24=0;pOrig.E24;pOrig.E23)	=IF($N24=0;pOrig.F24;pOrig.F23)	=IF($N24=0;pOrig.G24;pOrig.G25)	=IF($N24=0;pOrig.H24;pOrig.H25)	=IF($N24=0;pOrig.I24;pOrig.I25)	=IF($N24=0;pOrig.J24;pOrig.J25)	=IF($N24=0;pOrig.K24;pOrig.K25)
25	=IF($N25=0;pOrig.E25;pOrig.E26)	=IF($N25=0;pOrig.F25;pOrig.F26)	=IF($N25=0;pOrig.G25;pOrig.G26)	=IF($N25=0;pOrig.H25;pOrig.H26)	=IF($N25=0;pOrig.I25;pOrig.I26)	=IF($N25=0;pOrig.J25;pOrig.J26)	=IF($N25=0;pOrig.K25;pOrig.K26)
26	=IF($N26=0;pOrig.E26;pOrig.E25)	=IF($N26=0;pOrig.F26;pOrig.F25)	=IF($N26=0;pOrig.G26;pOrig.G27)	=IF($N26=0;pOrig.H26;pOrig.H27)	=IF($N26=0;pOrig.I26;pOrig.I27)	=IF($N26=0;pOrig.J26;pOrig.J27)	=IF($N26=0;pOrig.K26;pOrig.K27)
27	=IF($N27=0;pOrig.E27;pOrig.E28)	=IF($N27=0;pOrig.F27;pOrig.F28)	=IF($N27=0;pOrig.G27;pOrig.G28)	=IF($N27=0;pOrig.H27;pOrig.H28)	=IF($N27=0;pOrig.I27;pOrig.I28)	=IF($N27=0;pOrig.J27;pOrig.J28)	=IF($N27=0;pOrig.K27;pOrig.K28)
28	=IF($N28=0;pOrig.E28;pOrig.E27)	=IF($N28=0;pOrig.F28;pOrig.F27)	=IF($N28=0;pOrig.G28;pOrig.K29)	=IF($N28=0;pOrig.H28;pOrig.I29)	=IF($N28=0;pOrig.I28;pOrig.J29)	=IF($N28=0;pOrig.J28;pOrig.L29)	=IF($N28=0;pOrig.K28;pOrig.M29)
29							
30							
31							
32							
33	=IF($N32=0;pOrig.E33;pOrig.E31)	=IF($N32=0;pOrig.F33;pOrig.F31)	=IF($N32=0;pOrig.G33;pOrig.K34)	=IF($N32=0;pOrig.H33;pOrig.I34)	=IF($N32=0;pOrig.I33;pOrig.J34)	=IF($N32=0;pOrig.J33;pOrig.L33)	=IF($N32=0;pOrig.K33;pOrig.M33)
34	=IF($N33=0;pOrig.E34;pOrig.E33)	=IF($N33=0;pOrig.F34;pOrig.F33)	=IF($N33=0;pOrig.G34;pOrig.K35)	=IF($N33=0;pOrig.H34;pOrig.I35)	=IF($N33=0;pOrig.I34;pOrig.J35)	=IF($N33=0;pOrig.J34;pOrig.L34)	=IF($N33=0;pOrig.K34;pOrig.M34)
35	=IF($N34=0;pOrig.E35;pOrig.E34)	=IF($N34=0;pOrig.F35;pOrig.F34)	=IF($N34=0;pOrig.G35;pOrig.K36)	=IF($N34=0;pOrig.H35;pOrig.I36)	=IF($N34=0;pOrig.I35;pOrig.J36)	=IF($N34=0;pOrig.J35;pOrig.L35)	=IF($N34=0;pOrig.K35;pOrig.M35)
36	=IF($N35=0;pOrig.E36;pOrig.E35)	=IF($N35=0;pOrig.F36;pOrig.F35)	=IF($N35=0;pOrig.G36;pOrig.K36)	=IF($N35=0;pOrig.H36;pOrig.I36)	=IF($N35=0;pOrig.I36;pOrig.J36)	=IF($N35=0;pOrig.J36;pOrig.L36)	=IF($N35=0;pOrig.K36;pOrig.M36)

dNorm

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	**Distance**	DDphotometricBands.txt (after switch)			.run plotDistanceModulus.pro																
2	Paper name	DD_NUMBER	psfMag_g	psfMag_r	psfMag_u	abs mag (g [vl])	abs mag (r)	abs mag (u)	Temperature	psfMagErr_g	psfMagErr_r	psfMagErr_u	delta M (g)	delta M (r)	delta M (u)	g (m – M) error	r (m – M) error	u (m – M) error	Special Case	distance modu	distance modu
3																					
4																					
5	=SDSS.B5	=SDSS.N5	=p.Y5	=p.Z5	=p.X5	=WDnorm.X5	=WDnorm.Y5	=WDnorm.W5	=WDnorm.D5	=p.AD5	=p.AE5	=p.AC5	=WDnorm.AC5	=WDnorm.AD5	=WDnorm.AB5	=SQRT(J5^2 + M5^2)	=SQRT(K5^2 + N5^2)	=SQRT(L5^2 + O5^2)	=fin.B5	=C5-F5	=T5-T6
6	=SDSS.B6	=SDSS.N6	=p.Y6	=p.Z6	=p.X6	=WDnorm.X6	=WDnorm.Y6	=WDnorm.W6	=WDnorm.D6	=p.AD6	=p.AE6	=p.AC6	=WDnorm.AC6	=WDnorm.AD6	=WDnorm.AB6	=SQRT(J6^2 + M6^2)	=SQRT(K6^2 + N6^2)	=SQRT(L6^2 + O6^2)	=fin.B6	=C6-F6	
7	=SDSS.B7	=SDSS.N7	=p.Y7	=p.Z7	=p.X7	=WDnorm.X7	=WDnorm.Y7	=WDnorm.W7	=WDnorm.D7	=p.AD7	=p.AE7	=p.AC7	=WDnorm.AC7	=WDnorm.AD7	=WDnorm.AB7	=SQRT(J7^2 + M7^2)	=SQRT(K7^2 + N7^2)	=SQRT(L7^2 + O7^2)	=fin.B7	=C7-F7	=T7-T8
8	=SDSS.B8	=SDSS.N8	=p.Y8	=p.Z8	=p.X8	=WDnorm.X8	=WDnorm.Y8	=WDnorm.W8	=WDnorm.D8	=p.AD8	=p.AE8	=p.AC8	=WDnorm.AC8	=WDnorm.AD8	=WDnorm.AB8	=SQRT(J8^2 + M8^2)	=SQRT(K8^2 + N8^2)	=SQRT(L8^2 + O8^2)	=fin.B8	NaN	
9	=SDSS.B9	=SDSS.N9	=p.Y9	=p.Z9	=p.X9	=WDnorm.X9	=WDnorm.Y9	=WDnorm.W9	=WDnorm.D9	=p.AD9	=p.AE9	=p.AC9	=WDnorm.AC9	=WDnorm.AD9	=WDnorm.AB9	=SQRT(J9^2 + M9^2)	=SQRT(K9^2 + N9^2)	=SQRT(L9^2 + O9^2)	=fin.B9	=C9-F9	=T9-T10
10	=SDSS.B10	=SDSS.N10	=p.Y10	=p.Z10	=p.X10	=WDnorm.X10	=WDnorm.Y10	=WDnorm.W10	=WDnorm.D10	=p.AD10	=p.AE10	=p.AC10	=WDnorm.AC10	=WDnorm.AD10	=WDnorm.AB10	=SQRT(J10^2 + M10^2)	=SQRT(K10^2 + N10^2)	=SQRT(L10^2 + O10^2)	=fin.B10	=C10-F10	
11	=SDSS.B11	=SDSS.N11	=p.Y11	=p.Z11	=p.X11	=WDnorm.X11	=WDnorm.Y11	=WDnorm.W11	=WDnorm.D11	=p.AD11	=p.AE11	=p.AC11	=WDnorm.AC11	=WDnorm.AD11	=WDnorm.AB11	=SQRT(J11^2 + M11^2)	=SQRT(K11^2 + N11^2)	=SQRT(L11^2 + O11^2)	=fin.B11	=C11-F11	=T11-T12
12	=SDSS.B12	=SDSS.N12	=p.Y12	=p.Z12	=p.X12	=WDnorm.X12	=WDnorm.Y12	=WDnorm.W12	=WDnorm.D12	=p.AD12	=p.AE12	=p.AC12	=WDnorm.AC12	=WDnorm.AD12	=WDnorm.AB12	=SQRT(J12^2 + M12^2)	=SQRT(K12^2 + N12^2)	=SQRT(L12^2 + O12^2)	=fin.B12	=C12-F12	
13	=SDSS.B13	=SDSS.N13	=p.Y13	=p.Z13	=p.X13	=WDnorm.X13	=WDnorm.Y13	=WDnorm.W13	=WDnorm.D13	=p.AD13	=p.AE13	=p.AC13	=WDnorm.AC13	=WDnorm.AD13	=WDnorm.AB13	=SQRT(J13^2 + M13^2)	=SQRT(K13^2 + N13^2)	=SQRT(L13^2 + O13^2)	=fin.B13	=C13-F13	=T13-T14
14	=SDSS.B14	=SDSS.N14	=p.Y14	=p.Z14	=p.X14	=WDnorm.X14	=WDnorm.Y14	=WDnorm.W14	=WDnorm.D14	=p.AD14	=p.AE14	=p.AC14	=WDnorm.AC14	=WDnorm.AD14	=WDnorm.AB14	=SQRT(J14^2 + M14^2)	=SQRT(K14^2 + N14^2)	=SQRT(L14^2 + O14^2)	=fin.B14	=C14-F14	
15	=SDSS.B15	=SDSS.N15	=p.Y15	=p.Z15	=p.X15	=WDnorm.X15	=WDnorm.Y15	=WDnorm.W15	=WDnorm.D15	=p.AD15	=p.AE15	=p.AC15	=WDnorm.AC15	=WDnorm.AD15	=WDnorm.AB15	=SQRT(J15^2 + M15^2)	=SQRT(K15^2 + N15^2)	=SQRT(L15^2 + O15^2)	=fin.B15	=C15-F15	=T15-T16
16	=SDSS.B16	=SDSS.N16	=p.Y16	=p.Z16	=p.X16	=WDnorm.X16	=WDnorm.Y16	=WDnorm.W16	=WDnorm.D16	=p.AD16	=p.AE16	=p.AC16	=WDnorm.AC16	=WDnorm.AD16	=WDnorm.AB16	=SQRT(J16^2 + M16^2)	=SQRT(K16^2 + N16^2)	=SQRT(L16^2 + O16^2)	=fin.B16	=C16-F16	
17	=SDSS.B17	=SDSS.N17	=p.Y17	=p.Z17	=p.X17	=WDnorm.X17	=WDnorm.Y17	=WDnorm.W17	=WDnorm.D17	=p.AD17	=p.AE17	=p.AC17	=WDnorm.AC17	=WDnorm.AD17	=WDnorm.AB17	=SQRT(J17^2 + M17^2)	=SQRT(K17^2 + N17^2)	=SQRT(L17^2 + O17^2)	=fin.B17	=C17-F17	=T17-T18
18	=SDSS.B18	=SDSS.N18	=p.Y18	=p.Z18	=p.X18	=WDnorm.X18	=WDnorm.Y18	=WDnorm.W18	=WDnorm.D18	=p.AD18	=p.AE18	=p.AC18	=WDnorm.AC18	=WDnorm.AD18	=WDnorm.AB18	=SQRT(J18^2 + M18^2)	=SQRT(K18^2 + N18^2)	=SQRT(L18^2 + O18^2)	=fin.B18	=C18-F18	
19	=SDSS.B19	=SDSS.N19	=p.Y19	=p.Z19	=p.X19	=WDnorm.X19	=WDnorm.Y19	=WDnorm.W19	=WDnorm.D19	=p.AD19	=p.AE19	=p.AC19	=WDnorm.AC19	=WDnorm.AD19	=WDnorm.AB19	=SQRT(J19^2 + M19^2)	=SQRT(K19^2 + N19^2)	=SQRT(L19^2 + O19^2)	=fin.B19	NaN	=T19-T20
20	=SDSS.B20	=SDSS.N20	=p.Y20	=p.Z20	=p.X20	=WDnorm.X20	=WDnorm.Y20	=WDnorm.W20	=WDnorm.D20	=p.AD20	=p.AE20	=p.AC20	=WDnorm.AC20	=WDnorm.AD20	=WDnorm.AB20	=SQRT(J20^2 + M20^2)	=SQRT(K20^2 + N20^2)	=SQRT(L20^2 + O20^2)	=fin.B20	=C20-F20	
21	=SDSS.B21	=SDSS.N21	=p.Y21	=p.Z21	=p.X21	=WDnorm.X21	=WDnorm.Y21	=WDnorm.W21	=WDnorm.D21	=p.AD21	=p.AE21	=p.AC21	=WDnorm.AC21	=WDnorm.AD21	=WDnorm.AB21	=SQRT(J21^2 + M21^2)	=SQRT(K21^2 + N21^2)	=SQRT(L21^2 + O21^2)	=fin.B21	=C21-F21	=T21-T22
22	=SDSS.B22	=SDSS.N22	=p.Y22	=p.Z22	=p.X22	=WDnorm.X22	=WDnorm.Y22	=WDnorm.W22	=WDnorm.D22	=p.AD22	=p.AE22	=p.AC22	=WDnorm.AC22	=WDnorm.AD22	=WDnorm.AB22	=SQRT(J22^2 + M22^2)	=SQRT(K22^2 + N22^2)	=SQRT(L22^2 + O22^2)	=fin.B22	=C22-F22	
23	=SDSS.B23	=SDSS.N23	=p.Y23	=p.Z23	=p.X23	=WDnorm.X23	=WDnorm.Y23	=WDnorm.W23	=WDnorm.D23	=p.AD23	=p.AE23	=p.AC23	=WDnorm.AC23	=WDnorm.AD23	=WDnorm.AB23	=SQRT(J23^2 + M23^2)	=SQRT(K23^2 + N23^2)	=SQRT(L23^2 + O23^2)	=fin.B23	=C23-F23	=T23-T24
24	=SDSS.B24	=SDSS.N24	=p.Y24	=p.Z24	=p.X24	=WDnorm.X24	=WDnorm.Y24	=WDnorm.W24	=WDnorm.D24	=p.AD24	=p.AE24	=p.AC24	=WDnorm.AC24	=WDnorm.AD24	=WDnorm.AB24	=SQRT(J24^2 + M24^2)	=SQRT(K24^2 + N24^2)	=SQRT(L24^2 + O24^2)	=fin.B24	=C24-F24	
25	=SDSS.B25	=SDSS.N25	=p.Y25	=p.Z25	=p.X25	=WDnorm.X25	=WDnorm.Y25	=WDnorm.W25	=WDnorm.D25	=p.AD25	=p.AE25	=p.AC25	=WDnorm.AC25	=WDnorm.AD25	=WDnorm.AB25	=SQRT(J25^2 + M25^2)	=SQRT(K25^2 + N25^2)	=SQRT(L25^2 + O25^2)	=fin.B25	=C25-F25	=T25-T26
26	=SDSS.B26	=SDSS.N26	=p.Y26	=p.Z26	=p.X26	=WDnorm.X26	=WDnorm.Y26	=WDnorm.W26	=WDnorm.D26	=p.AD26	=p.AE26	=p.AC26	=WDnorm.AC26	=WDnorm.AD26	=WDnorm.AB26	=SQRT(J26^2 + M26^2)	=SQRT(K26^2 + N26^2)	=SQRT(L26^2 + O26^2)	=fin.B26	=C26-F26	
27	=SDSS.B27	=SDSS.N27	=p.Y27	=p.Z27	=p.X27	=WDnorm.X27	=WDnorm.Y27	=WDnorm.W27	=WDnorm.D27	=p.AD27	=p.AE27	=p.AC27	=WDnorm.AC27	=WDnorm.AD27	=WDnorm.AB27	=SQRT(J27^2 + M27^2)	=SQRT(K27^2 + N27^2)	=SQRT(L27^2 + O27^2)	=fin.B27	=C27-F27	=T27-T28
28	=SDSS.B28	=SDSS.N28	=p.Y28	=p.Z28	=p.X28	=WDnorm.X28	=WDnorm.Y28	=WDnorm.W28	=WDnorm.D28	=p.AD28	=p.AE28	=p.AC28	=WDnorm.AC28	=WDnorm.AD28	=WDnorm.AB28	=SQRT(J28^2 + M28^2)	=SQRT(K28^2 + N28^2)	=SQRT(L28^2 + O28^2)	=fin.B28	=C28-F28	
29																					
30																					
31																					
32																					
33	=SDSS.B33	=SDSS.N33	=p.Y33	=p.Z33	=p.X33	=WDnorm.X33	=WDnorm.Y33	=WDnorm.W33	=WDnorm.D33	=p.AD33	=p.AE33	=p.AC33	=WDnorm.AC33	=WDnorm.AD33	=WDnorm.AB33	=SQRT(J33^2 + M33^2)	=SQRT(K33^2 + N33^2)	=SQRT(L33^2 + O33^2)	=fin.B33	=C33-F33	=T33-T34
34	=SDSS.B34	=SDSS.N34	=p.Y34	=p.Z34	=p.X34	=WDnorm.X34	=WDnorm.Y34	=WDnorm.W34	=WDnorm.D34	=p.AD34	=p.AE34	=p.AC34	=WDnorm.AC34	=WDnorm.AD34	=WDnorm.AB34	=SQRT(J34^2 + M34^2)	=SQRT(K34^2 + N34^2)	=SQRT(L34^2 + O34^2)	=fin.B34	=C34-F34	
35	=SDSS.B35	=SDSS.N35	=p.Y35	=p.Z35	=p.X35	=WDnorm.X35	=WDnorm.Y35	=WDnorm.W35	=WDnorm.D35	=p.AD35	=p.AE35	=p.AC35	=WDnorm.AC35	=WDnorm.AD35	=WDnorm.AB35	=SQRT(J35^2 + M35^2)	=SQRT(K35^2 + N35^2)	=SQRT(L35^2 + O35^2)	=fin.B35	=C35-F35	=T35-T36
36	=SDSS.B36	=SDSS.N36	=p.Y36	=p.Z36	=p.X36	=WDnorm.X36	=WDnorm.Y36	=WDnorm.W36	=WDnorm.D36	=p.AD36	=p.AE36	=p.AC36	=WDnorm.AC36	=WDnorm.AD36	=WDnorm.AB36	=SQRT(J36^2 + M36^2)	=SQRT(K36^2 + N36^2)	=SQRT(L36^2 + O36^2)	=fin.B36	=C36-F36	

	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG
1								simulated probability of observed chance alignment				
2	distance modulus u (m – M) error	distance modulus g (m – M) error	distance modulus r (m – M) error					with cool2 correction				
3								input	input	output	output	
4						distance	distance error	separation (ar	distance (extreme max - extreme min)	probability of c	probability of chance alignme	
5	=SQRT(p.AC5^2 + WDnorm.AB5^2)	=SQRT(p.AD5^2 + WDnorm.AC5^2)	=SQRT(p.AE5^2 + WDnorm.AD5^2)		=A5	=WDcool2.AI5	=WDcool2.AJ5	=SDSS.U5	=IF(AA5>AA6;AA5+AB5;AA6+AB6)	8.48E-015	1.05E-003	
6	=SQRT(p.AC6^2 + WDnorm.AB6^2)	=SQRT(p.AD6^2 + WDnorm.AC6^2)	=SQRT(p.AE6^2 + WDnorm.AD6^2)		=A6	=WDnorm.AI6	=WDnorm.AJ6	=SDSS.U6	=IF(AA5<AA6;AA5-AB5;AA6-AB6)			
7	=SQRT(p.AC7^2 + WDnorm.AB7^2)	=SQRT(p.AD7^2 + WDnorm.AC7^2)	=SQRT(p.AE7^2 + WDnorm.AD7^2)		=A7	=WDnorm.AI7	=WDnorm.AJ7	=SDSS.U7	=IF(AA7>AA8;AA7+AB7;AA8+AB8)		3.29E-004	
8	0	0	0		=A8			=SDSS.U8	=IF(AA7<AA8;AA7-AB7;AA8-AB8)			
9	=SQRT(p.AC9^2 + WDnorm.AB9^2)	=SQRT(p.AD9^2 + WDnorm.AC9^2)	=SQRT(p.AE9^2 + WDnorm.AD9^2)		=A9	=WDcool2.AI9	=WDcool2.AJ9	=SDSS.U9	=IF(AA9>AA10;AA9+AB9;AA10+AB10)	4.73E-015	7.10E-003	
10	=SQRT(p.AC10^2 + WDnorm.AB10^2)	=SQRT(p.AD10^2 + WDnorm.AC10^2)	=SQRT(p.AE10^2 + WDnorm.AD10^2)		=A10	=WDcool2.AI10	=WDcool2.AJ10	=SDSS.U10	=IF(AA9<AA10;AA9-AB9;AA10-AB10)			
11	=SQRT(p.AC11^2 + WDnorm.AB11^2)	=SQRT(p.AD11^2 + WDnorm.AC11^2)	=SQRT(p.AE11^2 + WDnorm.AD11^2)		=A11	=WDnorm.AI11	=WDnorm.AJ11	=SDSS.U11	=IF(AA11>AA12;AA11+AB11;AA12+AB12)	3.03E-012	3.33E-003	
12	=SQRT(p.AC12^2 + WDnorm.AB12^2)	=SQRT(p.AD12^2 + WDnorm.AC12^2)	=SQRT(p.AE12^2 + WDnorm.AD12^2)		=A12	=WDnorm.AI12	=WDnorm.AJ12	=SDSS.U12	=IF(AA11<AA12;AA11-AB11;AA12-AB12)			
13	=SQRT(p.AC13^2 + WDnorm.AB13^2)	=SQRT(p.AD13^2 + WDnorm.AC13^2)	=SQRT(p.AE13^2 + WDnorm.AD13^2)		=A13	=WDnorm.AI13	=WDnorm.AJ13	=SDSS.U13	=IF(AA13>AA14;AA13+AB13;AA14+AB14)	2.99E-017	9.21E-004	
14	=SQRT(p.AC14^2 + WDnorm.AB14^2)	=SQRT(p.AD14^2 + WDnorm.AC14^2)	=SQRT(p.AE14^2 + WDnorm.AD14^2)		=A14	=WDnorm.AI14	=WDnorm.AJ14	=SDSS.U14	=IF(AA13<AA14;AA13-AB13;AA14-AB14)			
15	=SQRT(p.AC15^2 + WDnorm.AB15^2)	=SQRT(p.AD15^2 + WDnorm.AC15^2)	=SQRT(p.AE15^2 + WDnorm.AD15^2)		=A15	=WDnorm.AI15	=WDnorm.AJ15	=SDSS.U15	=IF(AA15>AA16;AA15+AB15;AA16+AB16)	8.34E-018	8.96E-006	
16	=SQRT(p.AC16^2 + WDnorm.AB16^2)	=SQRT(p.AD16^2 + WDnorm.AC16^2)	=SQRT(p.AE16^2 + WDnorm.AD16^2)		=A16	=WDnorm.AI16	=WDnorm.AJ16	=SDSS.U16	=IF(AA15<AA16;AA15-AB15;AA16-AB16)			
17	=SQRT(p.AC17^2 + WDnorm.AB17^2)	=SQRT(p.AD17^2 + WDnorm.AC17^2)	=SQRT(p.AE17^2 + WDnorm.AD17^2)		=A17	=WDnorm.AI17	=WDnorm.AJ17	=SDSS.U17	=IF(AA17>AA18;AA17+AB17;AA18+AB18)	6.04E-030	7.24E-004	
18	=SQRT(p.AC18^2 + WDnorm.AB18^2)	=SQRT(p.AD18^2 + WDnorm.AC18^2)	=SQRT(p.AE18^2 + WDnorm.AD18^2)		=A18	=WDcool2.AI18	=WDcool2.AJ18	=SDSS.U18	=IF(AA17<AA18;AA17-AB17;AA18-AB18)			
19	0	0	0		=A19			=SDSS.U19	=IF(AA19>AA20;AA19+AB19;AA20+AB20)		1.83E-004	
20	=SQRT(p.AC20^2 + WDnorm.AB20^2)	=SQRT(p.AD20^2 + WDnorm.AC20^2)	=SQRT(p.AE20^2 + WDnorm.AD20^2)		=A20	=WDcool2.AI20	=WDcool2.AJ20	=SDSS.U20	=IF(AA19<AA20;AA19-AB19;AA20-AB20)			
21	=SQRT(p.AC21^2 + WDnorm.AB21^2)	=SQRT(p.AD21^2 + WDnorm.AC21^2)	=SQRT(p.AE21^2 + WDnorm.AD21^2)		=A21	=WDcool2.AI21	=WDcool2.AJ21	=SDSS.U21	=IF(AA21>AA22;AA21+AB21;AA22+AB22)	1.83E-013	3.64E-004	
22	=SQRT(p.AC22^2 + WDnorm.AB22^2)	=SQRT(p.AD22^2 + WDnorm.AC22^2)	=SQRT(p.AE22^2 + WDnorm.AD22^2)		=A22	=WDnorm.AI22	=WDnorm.AJ22	=SDSS.U22	=IF(AA21<AA22;AA21-AB21;AA22-AB22)			
23	=SQRT(p.AC23^2 + WDnorm.AB23^2)	=SQRT(p.AD23^2 + WDnorm.AC23^2)	=SQRT(p.AE23^2 + WDnorm.AD23^2)		=A23	=WDnorm.AI23	=WDnorm.AJ23	=SDSS.U23	=IF(AA23>AA24;AA23+AB23;AA24+AB24)	2.85E-017	6.20E-004	
24	=SQRT(p.AC24^2 + WDnorm.AB24^2)	=SQRT(p.AD24^2 + WDnorm.AC24^2)	=SQRT(p.AE24^2 + WDnorm.AD24^2)		=A24	=WDnorm.AI24	=WDnorm.AJ24	=SDSS.U24	=IF(AA23<AA24;AA23-AB23;AA24-AB24)			
25	=SQRT(p.AC25^2 + WDnorm.AB25^2)	=SQRT(p.AD25^2 + WDnorm.AC25^2)	=SQRT(p.AE25^2 + WDnorm.AD25^2)		=A25	=WDnorm.AI25	=WDnorm.AJ25	=SDSS.U25	=IF(AA25>AA26;AA25+AB25;AA26+AB26)	4.27E-018	1.57E-006	
26	=SQRT(p.AC26^2 + WDnorm.AB26^2)	=SQRT(p.AD26^2 + WDnorm.AC26^2)	=SQRT(p.AE26^2 + WDnorm.AD26^2)		=A26	=WDnorm.AI26	=WDnorm.AJ26	=SDSS.U26	=IF(AA25<AA26;AA25-AB25;AA26-AB26)			
27	=SQRT(p.AC27^2 + WDnorm.AB27^2)	=SQRT(p.AD27^2 + WDnorm.AC27^2)	=SQRT(p.AE27^2 + WDnorm.AD27^2)		=A27	=WDnorm.AI27	=WDnorm.AJ27	=SDSS.U27	=IF(AA27>AA28;AA27+AB27;AA28+AB28)	2.16E-018	1.73E-004	
28	=SQRT(p.AC28^2 + WDnorm.AB28^2)	=SQRT(p.AD28^2 + WDnorm.AC28^2)	=SQRT(p.AE28^2 + WDnorm.AD28^2)		=A28	=WDcool2.AI28	=WDcool2.AJ28	=SDSS.U28	=IF(AA27<AA28;AA27-AB27;AA28-AB28)			
29												
30												
31												
32												
33	=SQRT(p.AC33^2 + WDnorm.AB33^2)	=SQRT(p.AD33^2 + WDnorm.AC33^2)	=SQRT(p.AE33^2 + WDnorm.AD33^2)		=A33	=WDnorm.AI33	=WDnorm.AJ33	=SDSS.U33	=IF(AA33>AA34;AA33+AB33;AA34+AB34)	4.27E-018	1.57E-006	
34	=SQRT(p.AC34^2 + WDnorm.AB34^2)	=SQRT(p.AD34^2 + WDnorm.AC34^2)	=SQRT(p.AE34^2 + WDnorm.AD34^2)		=A34	=WDnorm.AI34	=WDnorm.AJ34	=SDSS.U34	=IF(AA33<AA34;AA33-AB33;AA34-AB34)			
35	=SQRT(p.AC35^2 + WDnorm.AB35^2)	=SQRT(p.AD35^2 + WDnorm.AC35^2)	=SQRT(p.AE35^2 + WDnorm.AD35^2)		=A35	=WDnorm.AI35	=WDnorm.AJ35	=SDSS.U35	=IF(AA35>AA36;AA35+AB35;AA36+AB36)	2.16E-018	1.73E-004	
36	=SQRT(p.AC36^2 + WDnorm.AB36^2)	=SQRT(p.AD36^2 + WDnorm.AC36^2)	=SQRT(p.AE36^2 + WDnorm.AD36^2)		=A36	=WDcool2.AI36	=WDcool2.AJ36	=SDSS.U36	=IF(AA35<AA36;AA35-AB35;AA36-AB36)			

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA
1	**Norm WD par**	loggtempmag../rbb_wd_dd_param					Mode norm	must execute WDnorm, then WDcool, then WDcool2, then WDcool again (for DD-08A), then sysAgeDiffDist, IFMR, Fort.24																			
2	name	logg	logg error	temp	temp error	observed u_r	observed g_r	observed r_m	observed i_m	observed z_r	observed u_r	observed g_m	observed r_m	observed i_m	observed z_r	observed g_r	Mass	Mass (error)	Age/Ct (y)	Age/Ct error (y	logg	logg error	abs mag (u)	abs mag (g)	abs mag (r)	abs mag (i)	abs mag (z)	
3																											
4																											
5	=SDSS.B5	=s.AV5	=s.AW5	=s.AT5	=s.AU5	=p.X5	=p.Y5	=p.Z5	=p.AA5	=p.AB5	=p.AC5	=p.AD5	=p.AE5	=p.AF5	=p.AG5	18.89	0.67	0.04	6.00E+008	7.66E+007	8.11	0.07	12.51	12.15	12.25	12.41	12.63
6	=SDSS.B6	=s.AV6	=s.AW6	=s.AT6	=s.AU6	=p.X6	=p.Y6	=p.Z6	=p.AA6	=p.AB6	=p.AC6	=p.AD6	=p.AE6	=p.AF6	=p.AG6	17.71	0.61	0.04	7.82E+007	1.17E+007	7.97	0.07	10.68	10.65	11	11.31	11.62
7	=SDSS.B7	=s.AV7	=s.AW7	=s.AT7	=s.AU7	=p.X7	=p.Y7	=p.Z7	=p.AA7	=p.AB7	=p.AC7	=p.AD7	=p.AE7	=p.AF7	=p.AG7	16.91	0.7	0.04	1.46E+008	1.84E+007	8.13	0.07	11.21	11.1	11.42	11.71	12
8	=SDSS.B8	=s.AV8	=s.AW8	=s.AT8	=s.AU8	=p.X8	=p.Y8	=p.Z8	=p.AA8	=p.AB8	=p.AC8	=p.AD8	=p.AE8	=p.AF8	=p.AG8	17.76	0.26	0.01	1.25E+007	3.18E+006	0	0.07	9.38	9.24	9.61	9.91	10.21
9	=SDSS.B9	=s.AV9	=s.AW9	=s.AT9	=s.AU9	=p.X9	=p.Y9	=p.Z9	=p.AA9	=p.AB9	=p.AC9	=p.AD9	=p.AE9	=p.AF9	=p.AG9	18.81	0.89	0.05	1.32E+009	1.86E+008	8.45	0.07	13.23	12.93	12.98	13.11	13.3
10	=SDSS.B10	=s.AV10	=s.AW10	=s.AT10	=s.AU10	=p.X10	=p.Y10	=p.Z10	=p.AA10	=p.AB10	=p.AC10	=p.AD10	=p.AE10	=p.AF10	=p.AG10	18.16	0.76	0.05	6.32E+008	8.47E+007	8.25	0.07	12.54	12.21	12.34	12.51	12.74
11	=SDSS.B11	=s.AV11	=s.AW11	=s.AT11	=s.AU11	=p.X11	=p.Y11	=p.Z11	=p.AA11	=p.AB11	=p.AC11	=p.AD11	=p.AE11	=p.AF11	=p.AG11	18.57	0.71	0.04	2.91E+008	3.28E+007	8.16	0.07	11.86	11.58	11.81	12.06	12.33
12	=SDSS.B12	=s.AV12	=s.AW12	=s.AT12	=s.AU12	=p.X12	=p.Y12	=p.Z12	=p.AA12	=p.AB12	=p.AC12	=p.AD12	=p.AE12	=p.AF12	=p.AG12	16.36	0.91	0.04	5.42E+007	1.12E+007	8.45	0.07	10.34	10.62	11.06	11.41	11.76
13	=SDSS.B13	=s.AV13	=s.AW13	=s.AT13	=s.AU13	=p.X13	=p.Y13	=p.Z13	=p.AA13	=p.AB13	=p.AC13	=p.AD13	=p.AE13	=p.AF13	=p.AG13	15.6	0.85	0.04	1.69E+008	2.84E+007	8.37	0.07	11.31	11.3	11.64	11.94	12.25
14	=SDSS.B14	=s.AV14	=s.AW14	=s.AT14	=s.AU14	=p.X14	=p.Y14	=p.Z14	=p.AA14	=p.AB14	=p.AC14	=p.AD14	=p.AE14	=p.AF14	=p.AG14	16.01	0.65	0.04	2.65E+008	2.73E+007	8.07	0.07	11.75	11.46	11.69	11.94	12.21
15	=SDSS.B15	=s.AV15	=s.AW15	=s.AT15	=s.AU15	=p.X15	=p.Y15	=p.Z15	=p.AA15	=p.AB15	=p.AC15	=p.AD15	=p.AE15	=p.AF15	=p.AG15	16.76	0.67	0.04	2.89E+007	7.81E+006	8.06	0.07	10.1	10.27	10.69	11.03	11.36
16	=SDSS.B16	=s.AV16	=s.AW16	=s.AT16	=s.AU16	=p.X16	=p.Y16	=p.Z16	=p.AA16	=p.AB16	=p.AC16	=p.AD16	=p.AE16	=p.AF16	=p.AG16	17.26	0.65	0.04	1.04E+008	1.41E+007	8.06	0.07	10.91	10.85	11.19	11.5	11.8
17	=SDSS.B17	=s.AV17	=s.AW17	=s.AT17	=s.AU17	=p.X17	=p.Y17	=p.Z17	=p.AA17	=p.AB17	=p.AC17	=p.AD17	=p.AE17	=p.AF17	=p.AG17	15.64	0.65	0.04	6.90E+006	1.33E+006	8	0.07	9.11	9.47	9.96	10.33	10.68
18	=SDSS.B18	=s.AV18	=s.AW18	=s.AT18	=s.AU18	=p.X18	=p.Y18	=p.Z18	=p.AA18	=p.AB18	=p.AC18	=p.AD18	=p.AE18	=p.AF18	=p.AG18	18.2	0.8	0.05	9.06E+008	1.34E+008	8.31	0.07	12.88	12.55	12.63	12.78	12.98
19	=SDSS.B19	=s.AV19	=s.AW19	=s.AT19	=s.AU19	=p.X19	=p.Y19	=p.Z19	=p.AA19	=p.AB19	=p.AC19	=p.AD19	=p.AE19	=p.AF19	=p.AG19	18.34	0.22	0	1.88E+008	3.97E+007	0	0.07	11.48	10.95	11.01	11.15	11.33
20	=SDSS.B20	=s.AV20	=s.AW20	=s.AT20	=s.AU20	=p.X20	=p.Y20	=p.Z20	=p.AA20	=p.AB20	=p.AC20	=p.AD20	=p.AE20	=p.AF20	=p.AG20	18.4	0.96	0.04	1.35E+009	1.86E+008	8.57	0.07	13.25	12.97	13.05	13.19	13.39
21	=SDSS.B21	=s.AV21	=s.AW21	=s.AT21	=s.AU21	=p.X21	=p.Y21	=p.Z21	=p.AA21	=p.AB21	=p.AC21	=p.AD21	=p.AE21	=p.AF21	=p.AG21	18.07	0.71	0.05	5.08E+008	6.05E+007	8.17	0.07	12.33	12	12.14	12.33	12.57
22	=SDSS.B22	=s.AV22	=s.AW22	=s.AT22	=s.AU22	=p.X22	=p.Y22	=p.Z22	=p.AA22	=p.AB22	=p.AC22	=p.AD22	=p.AE22	=p.AF22	=p.AG22	18.28	1.17	0.03	8.99E+008	1.02E+008	8.94	0.07	12.8	12.72	12.97	13.23	13.52
23	=SDSS.B23	=s.AV23	=s.AW23	=s.AT23	=s.AU23	=p.X23	=p.Y23	=p.Z23	=p.AA23	=p.AB23	=p.AC23	=p.AD23	=p.AE23	=p.AF23	=p.AG23	17.8	0.88	0.05	5.54E+008	8.64E+007	8.43	0.07	12.4	12.14	12.33	12.55	12.81
24	=SDSS.B24	=s.AV24	=s.AW24	=s.AT24	=s.AU24	=p.X24	=p.Y24	=p.Z24	=p.AA24	=p.AB24	=p.AC24	=p.AD24	=p.AE24	=p.AF24	=p.AG24	17.49	0.76	0.05	4.28E+008	5.33E+007	8.24	0.07	12.19	11.9	12.08	12.3	12.56
25	=SDSS.B25	=s.AV25	=s.AW25	=s.AT25	=s.AU25	=p.X25	=p.Y25	=p.Z25	=p.AA25	=p.AB25	=p.AC25	=p.AD25	=p.AE25	=p.AF25	=p.AG25	16.27	0.74	0.04	1.00E+008	1.65E+007	8.19	0.07	10.89	10.91	11.27	11.58	11.89
26	=SDSS.B26	=s.AV26	=s.AW26	=s.AT26	=s.AU26	=p.X26	=p.Y26	=p.Z26	=p.AA26	=p.AB26	=p.AC26	=p.AD26	=p.AE26	=p.AF26	=p.AG26	17.65	0.89	0.04	4.71E+008	7.80E+007	8.45	0.07	12.27	12.05	12.26	12.5	12.77
27	=SDSS.B27	=s.AV27	=s.AW27	=s.AT27	=s.AU27	=p.X27	=p.Y27	=p.Z27	=p.AA27	=p.AB27	=p.AC27	=p.AD27	=p.AE27	=p.AF27	=p.AG27	16.41	0.63	0.04	1.69E+008	1.85E+007	8.02	0.07	11.3	11.11	11.4	11.67	11.96
28	=SDSS.B28	=s.AV28	=s.AW28	=s.AT28	=s.AU28	=p.X28	=p.Y28	=p.Z28	=p.AA28	=p.AB28	=p.AC28	=p.AD28	=p.AE28	=p.AF28	=p.AG28	17.11	0.79	0.05	6.27E+008	8.56E+007	8.29	0.07	12.53	12.21	12.35	12.53	12.77
29																											
30																											
31																											
32	**Third Party WDs for mass distribution**																										
33	=SDSS.B33	8.270	0.050	22740.0	230.0	=p.X33	=p.Y33	=p.Z33	=p.AA33	=p.AB33	=p.AC33	=p.AD33	=p.AE33	=p.AF33	=p.AG33	16.13	0.79	0.03	7.79E+007	1.03E+007	8.27	0.05	10.66	10.78	11.17	11.5	11.83
34	=SDSS.B34	8.780	0.050	22130.0	220.0	=p.X34	=p.Y34	=p.Z34	=p.AA34	=p.AB34	=p.AC34	=p.AD34	=p.AE34	=p.AF34	=p.AG34	17.13	1.09	0.03	2.48E+008	2.44E+007	8.78	0.05	11.62	11.76	12.13	12.44	12.77
35	=SDSS.B35	7.857	0.008	15636.0	9.0	=p.X35	=p.Y35	=p.Z35	=p.AA35	=p.AB35	=p.AC35	=p.AD35	=p.AE35	=p.AF35	=p.AG35	16.23	0.54	0	1.46E+008	1.49E+006	7.86	0.01	11.11	10.89	11.19	11.46	11.75
36	=SDSS.B36	7.989	0.012	13935.0	112.0	=p.X36	=p.Y36	=p.Z36	=p.AA36	=p.AB36	=p.AC36	=p.AD36	=p.AE36	=p.AF36	=p.AG36	16.5	0.6	0.01	2.44E+008	6.52E+006	7.99E+000	0.01	11.64	11.33	11.57	11.82	12.09

	AB	AC	AD	AE	AF	AG	AH	AI	AJ
1									
2	abs mag (u) e	abs mag (g) e	abs mag (r) e	abs mag (i) e	abs mag (z) e	rtemperature	temperature e	idistance mod	udistance modu
3									
4									
5	0.12	0.13	0.12	0.12	0.11	10648	244.9	6.74	0.14
6	0.12	0.12	0.11	0.11	0.11	19198.5	441.6	7.06	0.12
7	0.12	0.12	0.11	0.11	0.11	17285	397.6	5.81	0.12
8	0.17	0.12	0.11	0.1	0.1	18000	1000	8.52	0.13
9	0.14	0.15	0.14	0.14	0.13	10008.1	230.2	5.89	0.15
10	0.12	0.13	0.12	0.12	0.12	11222.3	258.1	5.95	0.13
11	0.12	0.12	0.11	0.11	0.11	13987	321.7	6.99	0.12
12	0.14	0.13	0.13	0.13	0.13	28066.6	645.5	5.74	0.14
13	0.13	0.13	0.12	0.12	0.12	19019.6	437.5	4.3	0.13
14	0.12	0.12	0.11	0.11	0.11	13950.4	320.9	4.55	0.12
15	0.13	0.12	0.12	0.12	0.12	24788.3	570.1	6.49	0.12
16	0.12	0.12	0.11	0.11	0.11	18502.2	425.6	6.41	0.15
17	0.13	0.12	0.12	0.12	0.12	33443.8	769.2	6.17	0.13
18	0.13	0.14	0.13	0.13	0.12	10392.9	239	5.65	0.14
19	0.2	0.23	0.19	0.16	0.14	9500	500	7.39	0.23
20	0.14	0.15	0.15	0.14	0.14	10492	241.3	5.43	0.16
21	0.12	0.13	0.12	0.11	0.11	11543	265.5	6.07	0.13
22	0.15	0.15	0.15	0.14	0.14	16037.2	368.9	5.56	0.15
23	0.11	0.12	0.12	0.12	0.11	13072.2	300.7	5.66	0.12
24	0.11	0.12	0.11	0.11	0.11	12757.9	293.4	5.59	0.12
25	0.13	0.12	0.12	0.12	0.11	20000.3	460	5.37	0.12
26	0.12	0.13	0.12	0.12	0.12	14036.7	322.8	5.61	0.27
27	0.12	0.12	0.11	0.11	0.11	15834.2	364.2	5.3	0.12
28	0.12	0.13	0.12	0.12	0.12	11452.5	263.4	4.9	0.15
29									
30									
31									
32									
33	0.08	0.09	0.08	0.08	0.08	22740	230	5.35	0.09
34	0.1	0.1	0.1	0.1	0.1	22130	220	5.38	0.12
35	0.01	0.01	0.01	0.01	0.01	15636	9	5.33	0.02
36	0.03	0.03	0.02	0.02	0.02	13935	112	5.17	0.03

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	**Cool WD par**	teffdistmag.dat (RBB cooler		./rbb_wd_dd_param			Mode cool		Strong Field DAH modeled externally		distance derived using alternate component					Fort.21										
2	name	Teff	Teff error	distance modulus	distance modulus e	observed u_m	observed g_m	observed r_m	observed i_m	observed z_m	observed u_m	observed g_m	observed r_m	observed i_m	observed z_m	observed g_m	Mass	Mass (error)	Age/Ct (y)	Age/Ct error (y	logg	logg error	abs mag (u)	abs mag (g)	abs mag (r)	abs mag (i)
3																										
4																										
5	=SDSS.B5	=s.AT5	=s.AU5	=dNorm.T6	=dNorm.V6	=p.X5	=p.Y5	=p.Z5	=p.AA5	=p.AB5	=p.AC5	=p.AD5	=p.AE5	=p.AF5	=p.AG5	18.89	0.54	0.06	4.45E+008	5.00E+007	7.9	0.1	12.23	11.83	11.94	12.1
6	=SDSS.B6																									
7	=SDSS.B7																									
8	=SDSS.B8	=s.AT8	=s.AU8	=dNorm.T7	=dNorm.V7	=p.X8	=p.Y8	=p.Z8	=p.AA8	=p.AB8	=p.AC8	=p.AD8	=p.AE8	=p.AF8	=p.AG8	17.76	1.03	0.05	3.91E+008	5.86E+007	8.67	0.09	11.99	11.95	12.26	12.55
9	=SDSS.B9																									
10	=SDSS.B10																									
11	=SDSS.B11																									
12	=SDSS.B12																									
13	=SDSS.B13																									
14	=SDSS.B14																									
15	=SDSS.B15																									
16	=SDSS.B16																									
17	=SDSS.B17																									
18	=SDSS.B18	=s.AT18	=s.AU18	=dNorm.T17	=dNorm.V17	=p.X18	=p.Y18	=p.Z18	=p.AA18	=p.AB18	=p.AC18	=p.AD18	=p.AE18	=p.AF18	=p.AG18	18.2	0.59	0.06	5.27E+008	6.54E+007	7.98	0.1	12.41	12.03	12.12	12.28
19	=SDSS.B19	=s.AT19	=s.AU19	=WDcool2.AI20	=WDcool2.AJ20	=p.X19	=p.Y19	=p.Z19	=p.AA19	=p.AB19	=p.AC19	=p.AD19	=p.AE19	=p.AF19	=p.AG19	18.34	0.62	0.11	7.30E+008	1.68E+008	8.03	0.18	12.81	12.45	12.49	12.6
20	=SDSS.B20																									
21	=SDSS.B21	=s.AT21	=s.AU21	=dNorm.T22	=dNorm.V22	=p.X21	=p.Y21	=p.Z21	=p.AA21	=p.AB21	=p.AC21	=p.AD21	=p.AE21	=p.AF21	=p.AG21	18.07	0.91	0.06	8.54E+008	1.33E+008	8.47	0.09	12.79	12.5	12.63	12.82
22	=SDSS.B22																									
23	=SDSS.B23																									
24	=SDSS.B24																									
25	=SDSS.B25																									
26	=SDSS.B26																									
27	=SDSS.B27																									
28	=SDSS.B28	=s.AT28	=s.AU28	=dNorm.T27	=dNorm.V27	=p.X28	=p.Y28	=p.Z28	=p.AA28	=p.AB28	=p.AC28	=p.AD28	=p.AE28	=p.AF28	=p.AG28	17.11	0.62	0.06	4.41E+008	5.50E+007	8.03	0.1	12.18	11.81	11.96	12.15
29																										

	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN
1														
2	abs mag (z)	abs mag (u) e	abs mag (g) e	abs mag (r) e	abs mag (i) e	abs mag (z) e	rtemperature	temperature e	distance modu	distance modu	spec mass (for com	spec mass offset	spec mass offset error	distance modulus differe
3														
4														
5	12.32	0.11	0.13	0.13	0.12	0.12	10648	244.9	7.06	0.12	=WDnorm.Q5	=Q5-AK5	=ABS($AL$29-AL5)	=AI5-WDnorm.AI5
6														
7														
8	12.85	0.13	0.13	0.13	0.13	0.13	18000	1000	5.81	0.13				
9														
10														
11														
12														
13														
14														
15														
16														
17														
18	12.48	0.11	0.12	0.12	0.12	0.12	10392.9	239	6.17	0.12	=WDnorm.Q18	=Q18-AK18	=ABS($AL$29-AL18)	=AI18-WDnorm.AI18
19	12.77	0.15	0.16	0.16	0.17	0.17	9500	500	5.89	0.16				
20														
21	13.05	0.14	0.15	0.15	0.15	0.15	11543	265.5	5.56	0.15	=WDnorm.Q21	=Q21-AK21	=ABS($AL$29-AL21)	=AI21-WDnorm.AI21
22														
23														
24														
25														
26														
27														
28	12.39	0.12	0.14	0.14	0.14	0.14	11452.5	263.4	5.3	0.12	=WDnorm.Q28	=Q28-AK28	=ABS($AL$29-AL28)	=AI28-WDnorm.AI28
29												=(AVERAGE(AL5;AL18;AL28))	=AVERAGE(AM5;AM18;AM28)	

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	**DC WD parm**	masstempmag.dat (PDD cool./rbb_wd_dd_param					Mode cool_2									Fort.23								
2	name	mass	mass error	temp	temp error	observed u_m	observed g_m	observed r_m	observed i_m	observed z_m	observed u_m	observed g_m	observed r_m	observed i_m	observed z_m	observed g_m	Mass	Mass (error)	Age/Ct (y)	Age/Ct error (y)	logg	logg error	abs mag (u)	abs mag (g)
3																								
4																								
5	=SDSS.B5	=WDnorm.Q5+$D$30	=SQRT(WDnorm.R5^2 + $D$31^2)	=s.AT5	=s.AU5	=p.X5	=p.Y5	=p.Z5	=p.AA5	=p.AB5	=p.AC5	=p.AD5	=p.AE5	=p.AF5	=p.AG5	18.89	0.5	0.05	4.28E+008	5.00E+007	7.83	0.1	12.15	11.75
6	=SDSS.B6																							
7	=SDSS.B7																							
8	=SDSS.B8																							
9	=SDSS.B9	=WDnorm.Q9+$D$30	=SQRT(WDnorm.R9^2 + $D$31^2)	=s.AT9	=s.AU9	=p.X9	=p.Y9	=p.Z9	=p.AA9	=p.AB9	=p.AC9	=p.AD9	=p.AE9	=p.AF9	=p.AG9	18.81	0.72	0.05	8.09E+008	1.34E+008	8.19	0.08	12.84	12.5
10	=SDSS.B10	=WDnorm.Q10+$D$30	=SQRT(WDnorm.R10^2 + $D$31^2)	=s.AT10	=s.AU10	=p.X10	=p.Y10	=p.Z10	=p.AA10	=p.AB10	=p.AC10	=p.AD10	=p.AE10	=p.AF10	=p.AG10	18.16	0.6	0.05	4.41E+008	4.52E+007	7.99	0.09	12.18	11.81
11	=SDSS.B11																							
12	=SDSS.B12																							
13	=SDSS.B13																							
14	=SDSS.B14																							
15	=SDSS.B15																							
16	=SDSS.B16																							
17	=SDSS.B17																							
18	=SDSS.B18	=WDnorm.Q18+$D$30	=SQRT(WDnorm.R18^2 + $D$31^2)	=s.AT18	=s.AU18	=p.X18	=p.Y18	=p.Z18	=p.AA18	=p.AB18	=p.AC18	=p.AD18	=p.AE18	=p.AF18	=p.AG18	18.2	0.63	0.05	5.78E+008	7.45E+007	8.05	0.09	12.51	12.14
19	=SDSS.B19																							
20	=SDSS.B20	=WDnorm.Q20+$D$30	=SQRT(WDnorm.R20^2 + $D$31^2)	=s.AT20	=s.AU20	=p.X20	=p.Y20	=p.Z20	=p.AA20	=p.AB20	=p.AC20	=p.AD20	=p.AE20	=p.AF20	=p.AG20	18.4	0.8	0.05	8.21E+008	1.41E+008	8.31	0.08	12.84	12.51
21	=SDSS.B21	=WDnorm.Q21+$D$30	=SQRT(WDnorm.R21^2 + $D$31^2)	=s.AT21	=s.AU21	=p.X21	=p.Y21	=p.Z21	=p.AA21	=p.AB21	=p.AC21	=p.AD21	=p.AE21	=p.AF21	=p.AG21	18.07	0.54	0.05	3.75E+008	4.08E+007	7.89	0.09	11.99	11.6
22	=SDSS.B22																							
23	=SDSS.B23																							
24	=SDSS.B24																							
25	=SDSS.B25																							
26	=SDSS.B26																							
27	=SDSS.B27																							
28	=SDSS.B28	=WDnorm.Q28+$D$30	=SQRT(WDnorm.R28^2 + $D$31^2)	=s.AT28	=s.AU28	=p.X28	=p.Y28	=p.Z28	=p.AA28	=p.AB28	=p.AC28	=p.AD28	=p.AE28	=p.AF28	=p.AG28	17.11	0.62	0.05	4.32E+008	4.38E+007	8.03	0.09	12.17	11.8
29																								
30		Low Temp Spectroscopic Mass overestimate		=WDcool.AL29																				
31		Low Temp Spectroscopic Mass overestimate error		=WDcool.AM29																				

	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK
1													
2	abs mag (r)	abs mag (i)	abs mag (z)	abs mag (u) er	abs mag (g) er	abs mag (r) er	abs mag (i) er	abs mag (z) er	temperature	temperature er	distance modu	distance modu	distance modulus differ
3													
4													
5	11.86	12.02	12.24	0.14	0.16	0.15	0.14	0.14	10648	244.9	7.14	0.16	
6													
7													
8													
9	12.56	12.69	12.88	0.15	0.16	0.15	0.15	0.14	10008.1	230.2	6.32	0.16	=AI9-WDnorm.AI9
10	11.94	12.12	12.36	0.13	0.14	0.13	0.13	0.13	11222.3	258.1	6.36	0.14	=AI10-WDnorm.AI10
11													
12													
13													
14													
15													
16													
17													
18	12.23	12.38	12.58	0.14	0.15	0.15	0.14	0.13	10392.9	239	6.06	0.16	
19													
20	12.6	12.75	12.96	0.15	0.16	0.15	0.14	0.14	10492	241.3	5.89	0.16	=AI20-WDnorm.AI20
21	11.75	11.95	12.19	0.13	0.14	0.13	0.13	0.13	11543	265.5	6.47	0.14	=AI21-WDnorm.AI21
22													
23													
24													
25													
26													
27													
28	11.95	12.14	12.38	0.13	0.14	0.13	0.13	0.13	11452.5	263.4	5.31	0.16	
29													
30													
31													

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	**Final Results (with combined tweaks)**					from "s" or derivatives						**from analytical parameter calculation / error propagation code (rbb_param ...f) NOT from compare**				**DDSystem**		**AgeDifference**	**Distribution**	**AcrossPublish**
2	name	Special Case	Special Case (for the purpose of mass/age SW calculations)	Magnetic? (not used in calcu		Logg	Logg (error)	Temperature	Temperature (error)			Mass	Mass (error)	Age/Ct (y)	Age/Ct error (y)	Radius	rerr	GR	Gre	abs mag (u)
3																				
4																				
5	=SDSS.B5	cool	=IF(((B5 = "cool")OR(B5 = "magnetic DA"));"nologg";"norm")	=s.AE5		=WDcool.U5	=WDcool.V5	=WDcool.AG5	=WDcool.AH5			=WDcool.Q5	=WDcool.R5	=WDcool.S5	=WDcool.T5	=#ref!.#ref!5	=#ref!.#ref!5	=#ref!.#ref!5	=#ref!.#ref!5	=WDcool.W5
6	=SDSS.B6	norm	=IF(((B6 = "cool")OR(B6 = "magnetic DA"));"nologg";"norm")	=s.AE6		=WDnorm.U6	=WDnorm.V6	=WDnorm.AG6	=WDnorm.AH6			=WDnorm.Q6	=WDnorm.R6	=WDnorm.S6	=WDnorm.T6	=#ref!.#ref!6	=#ref!.#ref!6	=#ref!.#ref!6	=#ref!.#ref!6	=WDnorm.W6
7	=SDSS.B7	norm	=IF(((B7 = "cool")OR(B7 = "magnetic DA"));"nologg";"norm")	=s.AE7		=WDnorm.U7	=WDnorm.V7	=WDnorm.AG7	=WDnorm.AH7			=WDnorm.Q7	=WDnorm.R7	=WDnorm.S7	=WDnorm.T7	=#ref!.#ref!7	=#ref!.#ref!7	=#ref!.#ref!7	=#ref!.#ref!7	=WDnorm.W7
8	=SDSS.B8	magnetic DA	=IF(((B8 = "cool")OR(B8 = "magnetic DA"));"nologg";"norm")	=s.AE8		=WDcool.U8	=WDcool.V8	=WDcool.AG8	=WDcool.AH8			=WDcool.Q8	=WDcool.R8	=WDcool.S8	=WDcool.T8	=#ref!.#ref!8	=#ref!.#ref!8	=#ref!.#ref!8	=#ref!.#ref!8	=WDcool.W8
9	=SDSS.B9	cool2	=IF(((B9 = "cool")OR(B9 = "magnetic DA"));"nologg";"norm")	=s.AE9		=WDcool2.U9	=WDcool2.V9	=WDcool2.AG9	=WDcool2.AH9			=WDcool2.Q9	=WDcool2.R9	=WDcool2.S9	=WDcool2.T9	=#ref!.#ref!9	=#ref!.#ref!9	=#ref!.#ref!9	=#ref!.#ref!9	=WDcool2.W9
10	=SDSS.B10	cool2	=IF(((B10 = "cool")OR(B10 = "magnetic DA"));"nologg";"norm")	=s.AE10		=WDcool2.U10	=WDcool2.V10	=WDcool2.AG10	=WDcool2.AH10			=WDcool2.Q10	=WDcool2.R10	=WDcool2.S10	=WDcool2.T10	=#ref!.#ref!10	=#ref!.#ref!10	=#ref!.#ref!10	=#ref!.#ref!10	=WDcool2.W10
11	=SDSS.B11	norm	=IF(((B11 = "cool")OR(B11 = "magnetic DA"));"nologg";"norm")	=s.AE11		=WDnorm.U11	=WDnorm.V11	=WDnorm.AG11	=WDnorm.AH11			=WDnorm.Q11	=WDnorm.R11	=WDnorm.S11	=WDnorm.T11	=#ref!.#ref!11	=#ref!.#ref!11	=#ref!.#ref!11	=#ref!.#ref!11	=WDnorm.W11
12	=SDSS.B12	norm	=IF(((B12 = "cool")OR(B12 = "magnetic DA"));"nologg";"norm")	=s.AE12		=WDnorm.U12	=WDnorm.V12	=WDnorm.AG12	=WDnorm.AH12			=WDnorm.Q12	=WDnorm.R12	=WDnorm.S12	=WDnorm.T12	=#ref!.#ref!12	=#ref!.#ref!12	=#ref!.#ref!12	=#ref!.#ref!12	=WDnorm.W12
13	=SDSS.B13	norm	=IF(((B13 = "cool")OR(B13 = "magnetic DA"));"nologg";"norm")	=s.AE13		=WDnorm.U13	=WDnorm.V13	=WDnorm.AG13	=WDnorm.AH13			=WDnorm.Q13	=WDnorm.R13	=WDnorm.S13	=WDnorm.T13	=#ref!.#ref!13	=#ref!.#ref!13	=#ref!.#ref!13	=#ref!.#ref!13	=WDnorm.W13
14	=SDSS.B14	norm	=IF(((B14 = "cool")OR(B14 = "magnetic DA"));"nologg";"norm")	=s.AE14		=WDnorm.U14	=WDnorm.V14	=WDnorm.AG14	=WDnorm.AH14			=WDnorm.Q14	=WDnorm.R14	=WDnorm.S14	=WDnorm.T14	=#ref!.#ref!14	=#ref!.#ref!14	=#ref!.#ref!14	=#ref!.#ref!14	=WDnorm.W14
15	=SDSS.B15	norm	=IF(((B15 = "cool")OR(B15 = "magnetic DA"));"nologg";"norm")	=s.AE15		=WDnorm.U15	=WDnorm.V15	=WDnorm.AG15	=WDnorm.AH15			=WDnorm.Q15	=WDnorm.R15	=WDnorm.S15	=WDnorm.T15	=#ref!.#ref!15	=#ref!.#ref!15	=#ref!.#ref!15	=#ref!.#ref!15	=WDnorm.W15
16	=SDSS.B16	norm	=IF(((B16 = "cool")OR(B16 = "magnetic DA"));"nologg";"norm")	=s.AE16		=WDnorm.U16	=WDnorm.V16	=WDnorm.AG16	=WDnorm.AH16			=WDnorm.Q16	=WDnorm.R16	=WDnorm.S16	=WDnorm.T16	=#ref!.#ref!16	=#ref!.#ref!16	=#ref!.#ref!16	=#ref!.#ref!16	=WDnorm.W16
17	=SDSS.B17	norm	=IF(((B17 = "cool")OR(B17 = "magnetic DA"));"nologg";"norm")	=s.AE17		=WDnorm.U17	=WDnorm.V17	=WDnorm.AG17	=WDnorm.AH17			=WDnorm.Q17	=WDnorm.R17	=WDnorm.S17	=WDnorm.T17	=#ref!.#ref!17	=#ref!.#ref!17	=#ref!.#ref!17	=#ref!.#ref!17	=WDnorm.W17
18	=SDSS.B18	cool	=IF(((B18 = "cool")OR(B18 = "magnetic DA"));"nologg";"norm")	=s.AE18		=WDcool.U18	=WDcool.V18	=WDcool.AG18	=WDcool.AH18			=WDcool.Q18	=WDcool.R18	=WDcool.S18	=WDcool.T18	=#ref!.#ref!18	=#ref!.#ref!18	=#ref!.#ref!18	=#ref!.#ref!18	=WDcool.W18
19	=SDSS.B19	magnetic DA	=IF(((B19 = "cool")OR(B19 = "magnetic DA"));"nologg";"norm")	=s.AE19		=WDcool.U19	=WDcool.V19	=WDcool.AG19	=WDcool.AH19			=WDcool.Q19	=WDcool.R19	=WDcool.S19	=WDcool.T19	=#ref!.#ref!19	=#ref!.#ref!19	=#ref!.#ref!19	=#ref!.#ref!19	=WDcool.W19
20	=SDSS.B20	cool2	=IF(((B20 = "cool")OR(B20 = "magnetic DA"));"nologg";"norm")	=s.AE20		=WDcool2.U20	=WDcool2.V20	=WDcool2.AG20	=WDcool2.AH20			=WDcool2.Q20	=WDcool2.R20	=WDcool2.S20	=WDcool2.T20	=#ref!.#ref!20	=#ref!.#ref!20	=#ref!.#ref!20	=#ref!.#ref!20	=WDcool2.W20
21	=SDSS.B21	cool2	=IF(((B21 = "cool")OR(B21 = "magnetic DA"));"nologg";"norm")	=s.AE21		=WDcool2.U21	=WDcool2.V21	=WDcool2.AG21	=WDcool2.AH21			=WDcool2.Q21	=WDcool2.R21	=WDcool2.S21	=WDcool2.T21	=#ref!.#ref!21	=#ref!.#ref!21	=#ref!.#ref!21	=#ref!.#ref!21	=WDcool2.W21
22	=SDSS.B22	norm	=IF(((B22 = "cool")OR(B22 = "magnetic DA"));"nologg";"norm")	=s.AE22		=WDnorm.U22	=WDnorm.V22	=WDnorm.AG22	=WDnorm.AH22			=WDnorm.Q22	=WDnorm.R22	=WDnorm.S22	=WDnorm.T22	=#ref!.#ref!22	=#ref!.#ref!22	=#ref!.#ref!22	=#ref!.#ref!22	=WDnorm.W22
23	=SDSS.B23	norm	=IF(((B23 = "cool")OR(B23 = "magnetic DA"));"nologg";"norm")	=s.AE23		=WDnorm.U23	=WDnorm.V23	=WDnorm.AG23	=WDnorm.AH23			=WDnorm.Q23	=WDnorm.R23	=WDnorm.S23	=WDnorm.T23	=#ref!.#ref!23	=#ref!.#ref!23	=#ref!.#ref!23	=#ref!.#ref!23	=WDnorm.W23
24	=SDSS.B24	norm	=IF(((B24 = "cool")OR(B24 = "magnetic DA"));"nologg";"norm")	=s.AE24		=WDnorm.U24	=WDnorm.V24	=WDnorm.AG24	=WDnorm.AH24			=WDnorm.Q24	=WDnorm.R24	=WDnorm.S24	=WDnorm.T24	=#ref!.#ref!24	=#ref!.#ref!24	=#ref!.#ref!24	=#ref!.#ref!24	=WDnorm.W24
25	=SDSS.B25	norm	=IF(((B25 = "cool")OR(B25 = "magnetic DA"));"nologg";"norm")	=s.AE25		=WDnorm.U25	=WDnorm.V25	=WDnorm.AG25	=WDnorm.AH25			=WDnorm.Q25	=WDnorm.R25	=WDnorm.S25	=WDnorm.T25	=#ref!.#ref!25	=#ref!.#ref!25	=#ref!.#ref!25	=#ref!.#ref!25	=WDnorm.W25
26	=SDSS.B26	norm	=IF(((B26 = "cool")OR(B26 = "magnetic DA"));"nologg";"norm")	=s.AE26		=WDnorm.U26	=WDnorm.V26	=WDnorm.AG26	=WDnorm.AH26			=WDnorm.Q26	=WDnorm.R26	=WDnorm.S26	=WDnorm.T26	=#ref!.#ref!26	=#ref!.#ref!26	=#ref!.#ref!26	=#ref!.#ref!26	=WDnorm.W26
27	=SDSS.B27	norm	=IF(((B27 = "cool")OR(B27 = "magnetic DA"));"nologg";"norm")	=s.AE27		=WDnorm.U27	=WDnorm.V27	=WDnorm.AG27	=WDnorm.AH27			=WDnorm.Q27	=WDnorm.R27	=WDnorm.S27	=WDnorm.T27	=#ref!.#ref!27	=#ref!.#ref!27	=#ref!.#ref!27	=#ref!.#ref!27	=WDnorm.W27
28	=SDSS.B28	cool	=IF(((B28 = "cool")OR(B28 = "magnetic DA"));"nologg";"norm")	=s.AE28		=WDcool.U28	=WDcool.V28	=WDcool.AG28	=WDcool.AH28			=WDcool.Q28	=WDcool.R28	=WDcool.S28	=WDcool.T28	=#ref!.#ref!28	=#ref!.#ref!28	=#ref!.#ref!28	=#ref!.#ref!28	=WDcool.W28
29																				
30																				
31																				
32	**Third Party WDs for mass distribution**																			
33	=SDSS.B33	norm	=IF(((B33 = "cool")OR(B33 = "magnetic DA"));"nologg";"norm")			8.27	0.05	22740	230			=WDnorm.Q33	=WDnorm.R33	7.79E+007	1.03E+007					=WDnorm.W33
34	=SDSS.B34	norm	=IF(((B34 = "cool")OR(B34 = "magnetic DA"));"nologg";"norm")			8.78	0.05	22130	220			=WDnorm.Q34	=WDnorm.R34	2.48E+008	2.44E+007					=WDnorm.W34
35	=SDSS.B35	norm	=IF(((B35 = "cool")OR(B35 = "magnetic DA"));"nologg";"norm")			7.86	0.01	15636	9			=WDnorm.Q35	=WDnorm.R35	1.46E+008	1.49E+006					=WDnorm.W35
36	=SDSS.B36	norm	=IF(((B36 = "cool")OR(B36 = "magnetic DA"));"nologg";"norm")			7.99	0.01	13935	112			=WDnorm.Q36	=WDnorm.R36	2.44E+008	6.52E+006					=WDnorm.W36
37																				
38																				
39																				
40																				
41																				
42																				
43																				
44																				
45																				

	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP
1	**edIFMRS ("plot")**				***NB absolute magnitudes are still used from these tables [as they cannot be derived from compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS]***													**from SDSS, ρ, and PM**				
2	abs mag (g)	abs mag (r)	abs mag (i)	abs mag (z)	abs mag (u) error	abs mag (g) error	abs mag (r) error	abs mag (i) error	abs mag (z) error	distance modulus	distance modulus	distance modulus diff		between components	min abs mag			psfMag_u	psfMag_g	psfMag_r	psfMag_i	psfMag_z
3																						
4																						
5	=WDcool.X5	=WDcool.Y5	=WDcool.Z5	=WDcool.AA5	=WDcool.AB5	=WDcool.AC5	=WDcool.AD5	=WDcool.AE5	=WDcool.AF5	=WDcool.AI5	=WDcool.AJ5	=AD6-AD5			=MIN(T5:X5)	=ROUND(AD5;2)	=ROUND(AE5;2)	=p.X5	=p.Y5	=p.Z5	=p.AA5	=p.AB5
6	=WDnorm.X6	=WDnorm.Y6	=WDnorm.Z6	=WDnorm.AA6	=WDnorm.AB6	=WDnorm.AC6	=WDnorm.AD6	=WDnorm.AE6	=WDnorm.AF6	=WDnorm.AI6	=WDnorm.AJ6				=MIN(T6:X6)	=ROUND(AD6;2)	=ROUND(AE6;2)	=p.X6	=p.Y6	=p.Z6	=p.AA6	=p.AB6
7	=WDnorm.X7	=WDnorm.Y7	=WDnorm.Z7	=WDnorm.AA7	=WDnorm.AB7	=WDnorm.AC7	=WDnorm.AD7	=WDnorm.AE7	=WDnorm.AF7	=WDnorm.AI7	=WDnorm.AJ7	=AD8-AD7			=MIN(T7:X7)	=ROUND(AD7;2)	=ROUND(AE7;2)	=p.X7	=p.Y7	=p.Z7	=p.AA7	=p.AB7
8	=WDcool.X8	=WDcool.Y8	=WDcool.Z8	=WDcool.AA8	=WDcool.AB8	=WDcool.AC8	=WDcool.AD8	=WDcool.AE8	=WDcool.AF8	=WDcool.AI8	=WDcool.AJ8				=MIN(T8:X8)	=ROUND(AD8;2)	=ROUND(AE8;2)	=p.X8	=p.Y8	=p.Z8	=p.AA8	=p.AB8
9	=WDcool2.X9	=WDcool2.Y9	=WDcool2.Z9	=WDcool2.AA9	=WDcool2.AB9	=WDcool2.AC9	=WDcool2.AD9	=WDcool2.AE9	=WDcool2.AF9	=WDcool2.AI9	=WDcool2.AJ9	=AD10-AD9			=MIN(T9:X9)	=ROUND(AD9;2)	=ROUND(AE9;2)	=p.X9	=p.Y9	=p.Z9	=p.AA9	=p.AB9
10	=WDcool2.X10	=WDcool2.Y10	=WDcool2.Z10	=WDcool2.AA10	=WDcool2.AB10	=WDcool2.AC10	=WDcool2.AD10	=WDcool2.AE10	=WDcool2.AF10	=WDcool2.AI10	=WDcool2.AJ10				###	=ROUND(AD10;2)	=ROUND(AE10;2)	=p.X10	=p.Y10	=p.Z10	=p.AA10	=p.AB10
11	=WDnorm.X11	=WDnorm.Y11	=WDnorm.Z11	=WDnorm.AA11	=WDnorm.AB11	=WDnorm.AC11	=WDnorm.AD11	=WDnorm.AE11	=WDnorm.AF11	=WDnorm.AI11	=WDnorm.AJ11	=AD12-AD11			###	=ROUND(AD11;2)	=ROUND(AE11;2)	=p.X11	=p.Y11	=p.Z11	=p.AA11	=p.AB11
12	=WDnorm.X12	=WDnorm.Y12	=WDnorm.Z12	=WDnorm.AA12	=WDnorm.AB12	=WDnorm.AC12	=WDnorm.AD12	=WDnorm.AE12	=WDnorm.AF12	=WDnorm.AI12	=WDnorm.AJ12				###	=ROUND(AD12;2)	=ROUND(AE12;2)	=p.X12	=p.Y12	=p.Z12	=p.AA12	=p.AB12
13	=WDnorm.X13	=WDnorm.Y13	=WDnorm.Z13	=WDnorm.AA13	=WDnorm.AB13	=WDnorm.AC13	=WDnorm.AD13	=WDnorm.AE13	=WDnorm.AF13	=WDnorm.AI13	=WDnorm.AJ13	=AD14-AD13			###	=ROUND(AD13;2)	=ROUND(AE13;2)	=p.X13	=p.Y13	=p.Z13	=p.AA13	=p.AB13
14	=WDnorm.X14	=WDnorm.Y14	=WDnorm.Z14	=WDnorm.AA14	=WDnorm.AB14	=WDnorm.AC14	=WDnorm.AD14	=WDnorm.AE14	=WDnorm.AF14	=WDnorm.AI14	=WDnorm.AJ14				###	=ROUND(AD14;2)	=ROUND(AE14;2)	=p.X14	=p.Y14	=p.Z14	=p.AA14	=p.AB14
15	=WDnorm.X15	=WDnorm.Y15	=WDnorm.Z15	=WDnorm.AA15	=WDnorm.AB15	=WDnorm.AC15	=WDnorm.AD15	=WDnorm.AE15	=WDnorm.AF15	=WDnorm.AI15	=WDnorm.AJ15	=AD16-AD15			###	=ROUND(AD15;2)	=ROUND(AE15;2)	=p.X15	=p.Y15	=p.Z15	=p.AA15	=p.AB15
16	=WDnorm.X16	=WDnorm.Y16	=WDnorm.Z16	=WDnorm.AA16	=WDnorm.AB16	=WDnorm.AC16	=WDnorm.AD16	=WDnorm.AE16	=WDnorm.AF16	=WDnorm.AI16	=WDnorm.AJ16				###	=ROUND(AD16;2)	=ROUND(AE16;2)	=p.X16	=p.Y16	=p.Z16	=p.AA16	=p.AB16
17	=WDnorm.X17	=WDnorm.Y17	=WDnorm.Z17	=WDnorm.AA17	=WDnorm.AB17	=WDnorm.AC17	=WDnorm.AD17	=WDnorm.AE17	=WDnorm.AF17	=WDnorm.AI17	=WDnorm.AJ17	=AD18-AD17			###	=ROUND(AD17;2)	=ROUND(AE17;2)	=p.X17	=p.Y17	=p.Z17	=p.AA17	=p.AB17
18	=WDcool.X18	=WDcool.Y18	=WDcool.Z18	=WDcool.AA18	=WDcool.AB18	=WDcool.AC18	=WDcool.AD18	=WDcool.AE18	=WDcool.AF18	=WDcool.AI18	=WDcool.AJ18				###	=ROUND(AD18;2)	=ROUND(AE18;2)	=p.X18	=p.Y18	=p.Z18	=p.AA18	=p.AB18
19	=WDcool.X19	=WDcool.Y19	=WDcool.Z19	=WDcool.AA19	=WDcool.AB19	=WDcool.AC19	=WDcool.AD19	=WDcool.AE19	=WDcool.AF19	=WDcool.AI19	=WDcool.AJ19	=AD20-AD19			###	=ROUND(AD19;2)	=ROUND(AE19;2)	=p.X19	=p.Y19	=p.Z19	=p.AA19	=p.AB19
20	=WDcool2.X20	=WDcool2.Y20	=WDcool2.Z20	=WDcool2.AA20	=WDcool2.AB20	=WDcool2.AC20	=WDcool2.AD20	=WDcool2.AE20	=WDcool2.AF20	=WDcool2.AI20	=WDcool2.AJ20				###	=ROUND(AD20;2)	=ROUND(AE20;2)	=p.X20	=p.Y20	=p.Z20	=p.AA20	=p.AB20
21	=WDcool2.X21	=WDcool2.Y21	=WDcool2.Z21	=WDcool2.AA21	=WDcool2.AB21	=WDcool2.AC21	=WDcool2.AD21	=WDcool2.AE21	=WDcool2.AF21	=WDcool2.AI21	=WDcool2.AJ21	=AD22-AD21			###	=ROUND(AD21;2)	=ROUND(AE21;2)	=p.X21	=p.Y21	=p.Z21	=p.AA21	=p.AB21
22	=WDnorm.X22	=WDnorm.Y22	=WDnorm.Z22	=WDnorm.AA22	=WDnorm.AB22	=WDnorm.AC22	=WDnorm.AD22	=WDnorm.AE22	=WDnorm.AF22	=WDnorm.AI22	=WDnorm.AJ22				###	=ROUND(AD22;2)	=ROUND(AE22;2)	=p.X22	=p.Y22	=p.Z22	=p.AA22	=p.AB22
23	=WDnorm.X23	=WDnorm.Y23	=WDnorm.Z23	=WDnorm.AA23	=WDnorm.AB23	=WDnorm.AC23	=WDnorm.AD23	=WDnorm.AE23	=WDnorm.AF23	=WDnorm.AI23	=WDnorm.AJ23	=AD24-AD23			###	=ROUND(AD23;2)	=ROUND(AE23;2)	=p.X23	=p.Y23	=p.Z23	=p.AA23	=p.AB23
24	=WDnorm.X24	=WDnorm.Y24	=WDnorm.Z24	=WDnorm.AA24	=WDnorm.AB24	=WDnorm.AC24	=WDnorm.AD24	=WDnorm.AE24	=WDnorm.AF24	=WDnorm.AI24	=WDnorm.AJ24				###	=ROUND(AD24;2)	=ROUND(AE24;2)	=p.X24	=p.Y24	=p.Z24	=p.AA24	=p.AB24
25	=WDnorm.X25	=WDnorm.Y25	=WDnorm.Z25	=WDnorm.AA25	=WDnorm.AB25	=WDnorm.AC25	=WDnorm.AD25	=WDnorm.AE25	=WDnorm.AF25	=WDnorm.AI25	=WDnorm.AJ25	=AD26-AD25			###	=ROUND(AD25;2)	=ROUND(AE25;2)	=p.X25	=p.Y25	=p.Z25	=p.AA25	=p.AB25
26	=WDnorm.X26	=WDnorm.Y26	=WDnorm.Z26	=WDnorm.AA26	=WDnorm.AB26	=WDnorm.AC26	=WDnorm.AD26	=WDnorm.AE26	=WDnorm.AF26	=WDnorm.AI26	=WDnorm.AJ26				###	=ROUND(AD26;2)	=ROUND(AE26;2)	=p.X26	=p.Y26	=p.Z26	=p.AA26	=p.AB26
27	=WDnorm.X27	=WDnorm.Y27	=WDnorm.Z27	=WDnorm.AA27	=WDnorm.AB27	=WDnorm.AC27	=WDnorm.AD27	=WDnorm.AE27	=WDnorm.AF27	=WDnorm.AI27	=WDnorm.AJ27	=AD28-AD27			###	=ROUND(AD27;2)	=ROUND(AE27;2)	=p.X27	=p.Y27	=p.Z27	=p.AA27	=p.AB27
28	=WDcool.X28	=WDcool.Y28	=WDcool.Z28	=WDcool.AA28	=WDcool.AB28	=WDcool.AC28	=WDcool.AD28	=WDcool.AE28	=WDcool.AF28	=WDcool.AI28	=WDcool.AJ28				###	=ROUND(AD28;2)	=ROUND(AE28;2)	=p.X28	=p.Y28	=p.Z28	=p.AA28	=p.AB28
29																						
30																						
31																						
32																						
33	=WDnorm.X33	=WDnorm.Y33	=WDnorm.Z33	=WDnorm.AA33	=WDnorm.AB33	=WDnorm.AC33	=WDnorm.AD33	=WDnorm.AE33	=WDnorm.AF33	=WDnorm.AI33	=WDnorm.AJ33	=AD34-AD33			###			=p.X33	=p.Y33	=p.Z33	=p.AA33	=p.AB33
34	=WDnorm.X34	=WDnorm.Y34	=WDnorm.Z34	=WDnorm.AA34	=WDnorm.AB34	=WDnorm.AC34	=WDnorm.AD34	=WDnorm.AE34	=WDnorm.AF34	=WDnorm.AI34	=WDnorm.AJ34				###			=p.X34	=p.Y34	=p.Z34	=p.AA34	=p.AB34
35	=WDnorm.X35	=WDnorm.Y35	=WDnorm.Z35	=WDnorm.AA35	=WDnorm.AB35	=WDnorm.AC35	=WDnorm.AD35	=WDnorm.AE35	=WDnorm.AF35	=WDnorm.AI35	=WDnorm.AJ35	=AD36-AD35			###			=p.X35	=p.Y35	=p.Z35	=p.AA35	=p.AB35
36	=WDnorm.X36	=WDnorm.Y36	=WDnorm.Z36	=WDnorm.AA36	=WDnorm.AB36	=WDnorm.AC36	=WDnorm.AD36	=WDnorm.AE36	=WDnorm.AF36	=WDnorm.AI36	=WDnorm.AJ36				###			=p.X36	=p.Y36	=p.Z36	=p.AA36	=p.AB36
37																						
38																						
39																						
40																						
41																						
42																						
43																						
44																						
45																						

	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ	BA	BB	BC	BD	BE	BF	BG	BH	BI
1																	**use analytical results?**		**0**
2	psfMagErr_u	psfMagErr_g	psfMagErr_r	psfMagErr_i	psfMagErr_z	min obs mag	separation (arc	PM RA	PM Dec (mas/	PM error RA (r	PM error Dec	probability shai	s data A brigh	PM (rad/year)	PM + error (rad/year)		***(1=analytical,0=monte carlo)***		
3																			
4																			
5	=p.AC5	=p.AD5	=p.AE5	=p.AF5	=p.AG5	=MIN(AL5:AP5)	=SDSS.U6	=PM.F5	=PM.G5	=PM.H5	=PM.I5	=DF5	=SDSS.K6	=SQRT(AX5^2+AY5^2)/1000/60/60/180*PI()	=(SQRT(AX5^2+AY5^2)+SQRT(AZ5^2+BA5^2))/1000/60/60/180*PI()				
6	=p.AC6	=p.AD6	=p.AE6	=p.AF6	=p.AG6	=MIN(AL6:AP6)	=SDSS.U5	=PM.F6	=PM.G6	=PM.H6	=PM.I6	=PM.V6	=SDSS.K5	=SQRT(AX6^2+AY6^2)/1000/60/60/180*PI()	=(SQRT(AX6^2+AY6^2)+SQRT(AZ6^2+BA6^2))/1000/60/60/180*PI()				
7	=p.AC7	=p.AD7	=p.AE7	=p.AF7	=p.AG7	=MIN(AL7:AP7)	=SDSS.U7	=PM.F7	=PM.G7	=PM.H7	=PM.I7	=DF7	=SDSS.K7	=SQRT(AX7^2+AY7^2)/1000/60/60/180*PI()	=(SQRT(AX7^2+AY7^2)+SQRT(AZ7^2+BA7^2))/1000/60/60/180*PI()				
8	=p.AC8	=p.AD8	=p.AE8	=p.AF8	=p.AG8	=MIN(AL8:AP8)	=SDSS.U8	=PM.F8	=PM.G8	=PM.H8	=PM.I8	=PM.V8	=SDSS.K8	=SQRT(AX8^2+AY8^2)/1000/60/60/180*PI()	=(SQRT(AX8^2+AY8^2)+SQRT(AZ8^2+BA8^2))/1000/60/60/180*PI()				
9	=p.AC9	=p.AD9	=p.AE9	=p.AF9	=p.AG9	=MIN(AL9:AP9)	=SDSS.U10	=PM.F9	=PM.G9	=PM.H9	=PM.I9	=DF9	=SDSS.K10	=SQRT(AX9^2+AY9^2)/1000/60/60/180*PI()	=(SQRT(AX9^2+AY9^2)+SQRT(AZ9^2+BA9^2))/1000/60/60/180*PI()				
10	=p.AC10	=p.AD10	=p.AE10	=p.AF10	=p.AG10	=MIN(AL10:AP10)	=SDSS.U9	=PM.F10	=PM.G10	=PM.H10	=PM.I10	=PM.V10	=SDSS.K9	=SQRT(AX10^2+AY10^2)/1000/60/60/180*PI()	=(SQRT(AX10^2+AY10^2)+SQRT(AZ10^2+BA10^2))/1000/60/60/180*PI()				
11	=p.AC11	=p.AD11	=p.AE11	=p.AF11	=p.AG11	=MIN(AL11:AP11)	=SDSS.U12	=PM.F11	=PM.G11	=PM.H11	=PM.I11	=DF11	=SDSS.K12	=SQRT(AX11^2+AY11^2)/1000/60/60/180*PI()	=(SQRT(AX11^2+AY11^2)+SQRT(AZ11^2+BA11^2))/1000/60/60/180*PI()				
12	=p.AC12	=p.AD12	=p.AE12	=p.AF12	=p.AG12	=MIN(AL12:AP12)	=SDSS.U11	=PM.F12	=PM.G12	=PM.H12	=PM.I12	=PM.V12	=SDSS.K11	=SQRT(AX12^2+AY12^2)/1000/60/60/180*PI()	=(SQRT(AX12^2+AY12^2)+SQRT(AZ12^2+BA12^2))/1000/60/60/180*PI()				
13	=p.AC13	=p.AD13	=p.AE13	=p.AF13	=p.AG13	=MIN(AL13:AP13)	=SDSS.U13	=PM.F13	=PM.G13	=PM.H13	=PM.I13	=DF13	=SDSS.K13	=SQRT(AX13^2+AY13^2)/1000/60/60/180*PI()	=(SQRT(AX13^2+AY13^2)+SQRT(AZ13^2+BA13^2))/1000/60/60/180*PI()				
14	=p.AC14	=p.AD14	=p.AE14	=p.AF14	=p.AG14	=MIN(AL14:AP14)	=SDSS.U14	=PM.F14	=PM.G14	=PM.H14	=PM.I14	=PM.V14	=SDSS.K14	=SQRT(AX14^2+AY14^2)/1000/60/60/180*PI()	=(SQRT(AX14^2+AY14^2)+SQRT(AZ14^2+BA14^2))/1000/60/60/180*PI()				
15	=p.AC15	=p.AD15	=p.AE15	=p.AF15	=p.AG15	=MIN(AL15:AP15)	=SDSS.U15	=PM.F15	=PM.G15	=PM.H15	=PM.I15	=DF15	=SDSS.K15	=SQRT(AX15^2+AY15^2)/1000/60/60/180*PI()	=(SQRT(AX15^2+AY15^2)+SQRT(AZ15^2+BA15^2))/1000/60/60/180*PI()				
16	=p.AC16	=p.AD16	=p.AE16	=p.AF16	=p.AG16	=MIN(AL16:AP16)	=SDSS.U16	=PM.F16	=PM.G16	=PM.H16	=PM.I16	=PM.V16	=SDSS.K16	=SQRT(AX16^2+AY16^2)/1000/60/60/180*PI()	=(SQRT(AX16^2+AY16^2)+SQRT(AZ16^2+BA16^2))/1000/60/60/180*PI()				
17	=p.AC17	=p.AD17	=p.AE17	=p.AF17	=p.AG17	=MIN(AL17:AP17)	=SDSS.U17	=PM.F17	=PM.G17	=PM.H17	=PM.I17	=DF17	=SDSS.K17	=SQRT(AX17^2+AY17^2)/1000/60/60/180*PI()	=(SQRT(AX17^2+AY17^2)+SQRT(AZ17^2+BA17^2))/1000/60/60/180*PI()				
18	=p.AC18	=p.AD18	=p.AE18	=p.AF18	=p.AG18	=MIN(AL18:AP18)	=SDSS.U18	=PM.F18	=PM.G18	=PM.H18	=PM.I18	=PM.V18	=SDSS.K18	=SQRT(AX18^2+AY18^2)/1000/60/60/180*PI()	=(SQRT(AX18^2+AY18^2)+SQRT(AZ18^2+BA18^2))/1000/60/60/180*PI()				
19	=p.AC19	=p.AD19	=p.AE19	=p.AF19	=p.AG19	=MIN(AL19:AP19)	=SDSS.U19	=PM.F19	=PM.G19	=PM.H19	=PM.I19	=DF19	=SDSS.K19	=SQRT(AX19^2+AY19^2)/1000/60/60/180*PI()	=(SQRT(AX19^2+AY19^2)+SQRT(AZ19^2+BA19^2))/1000/60/60/180*PI()				
20	=p.AC20	=p.AD20	=p.AE20	=p.AF20	=p.AG20	=MIN(AL20:AP20)	=SDSS.U20	=PM.F20	=PM.G20	=PM.H20	=PM.I20	=PM.V20	=SDSS.K20	=SQRT(AX20^2+AY20^2)/1000/60/60/180*PI()	=(SQRT(AX20^2+AY20^2)+SQRT(AZ20^2+BA20^2))/1000/60/60/180*PI()				
21	=p.AC21	=p.AD21	=p.AE21	=p.AF21	=p.AG21	=MIN(AL21:AP21)	=SDSS.U21	=PM.F21	=PM.G21	=PM.H21	=PM.I21	=DF21	=SDSS.K21	=SQRT(AX21^2+AY21^2)/1000/60/60/180*PI()	=(SQRT(AX21^2+AY21^2)+SQRT(AZ21^2+BA21^2))/1000/60/60/180*PI()				
22	=p.AC22	=p.AD22	=p.AE22	=p.AF22	=p.AG22	=MIN(AL22:AP22)	=SDSS.U22	=PM.F22	=PM.G22	=PM.H22	=PM.I22	=PM.V22	=SDSS.K22	=SQRT(AX22^2+AY22^2)/1000/60/60/180*PI()	=(SQRT(AX22^2+AY22^2)+SQRT(AZ22^2+BA22^2))/1000/60/60/180*PI()				
23	=p.AC23	=p.AD23	=p.AE23	=p.AF23	=p.AG23	=MIN(AL23:AP23)	=SDSS.U23	=PM.F23	=PM.G23	=PM.H23	=PM.I23	=DF23	=SDSS.K23	=SQRT(AX23^2+AY23^2)/1000/60/60/180*PI()	=(SQRT(AX23^2+AY23^2)+SQRT(AZ23^2+BA23^2))/1000/60/60/180*PI()				
24	=p.AC24	=p.AD24	=p.AE24	=p.AF24	=p.AG24	=MIN(AL24:AP24)	=SDSS.U24	=PM.F24	=PM.G24	=PM.H24	=PM.I24	=PM.V24	=SDSS.K24	=SQRT(AX24^2+AY24^2)/1000/60/60/180*PI()	=(SQRT(AX24^2+AY24^2)+SQRT(AZ24^2+BA24^2))/1000/60/60/180*PI()				
25	=p.AC25	=p.AD25	=p.AE25	=p.AF25	=p.AG25	=MIN(AL25:AP25)	=SDSS.U25	=PM.F25	=PM.G25	=PM.H25	=PM.I25	=DF25	=SDSS.K25	=SQRT(AX25^2+AY25^2)/1000/60/60/180*PI()	=(SQRT(AX25^2+AY25^2)+SQRT(AZ25^2+BA25^2))/1000/60/60/180*PI()				
26	=p.AC26	=p.AD26	=p.AE26	=p.AF26	=p.AG26	=MIN(AL26:AP26)	=SDSS.U26	=PM.F26	=PM.G26	=PM.H26	=PM.I26	=PM.V26	=SDSS.K26	=SQRT(AX26^2+AY26^2)/1000/60/60/180*PI()	=(SQRT(AX26^2+AY26^2)+SQRT(AZ26^2+BA26^2))/1000/60/60/180*PI()				
27	=p.AC27	=p.AD27	=p.AE27	=p.AF27	=p.AG27	=MIN(AL27:AP27)	=SDSS.U27	=PM.F27	=PM.G27	=PM.H27	=PM.I27	=DF27	=SDSS.K27	=SQRT(AX27^2+AY27^2)/1000/60/60/180*PI()	=(SQRT(AX27^2+AY27^2)+SQRT(AZ27^2+BA27^2))/1000/60/60/180*PI()				
28	=p.AC28	=p.AD28	=p.AE28	=p.AF28	=p.AG28	=MIN(AL28:AP28)	=SDSS.U28	=PM.F28	=PM.G28	=PM.H28	=PM.I28	=PM.V28	=SDSS.K28	=SQRT(AX28^2+AY28^2)/1000/60/60/180*PI()	=(SQRT(AX28^2+AY28^2)+SQRT(AZ28^2+BA28^2))/1000/60/60/180*PI()				
29																			
30																			
31																			
32																			
33	=p.AC33	=p.AD33	=p.AE33	=p.AF33	=p.AG33														
34	=p.AC34	=p.AD34	=p.AE34	=p.AF34	=p.AG34														
35	=p.AC35	=p.AD35	=p.AE35	=p.AF35	=p.AG35														
36	=p.AC36	=p.AD36	=p.AE36	=p.AF36	=p.AG36														
37																			
38																			
39																			
40																			
41																			
42																			
43																			
44																			
45																			

	BJ	BK	BL	BM	BN	BO	BP	BQ	BR	BS	BT
1			Take these from either sheet "fin" +"IFMR" OR "plots"								
2			mass	mass min	mass max	Cooling Age	Cooling Age Min	Cooling Age Max	initial mass	initial mass min	initial mass max
3											
4											
5			=IF($BI$1;IFMR.AB5;plots.BL5)	=IF($BI$1;IFMR.AC5;plots.BM5)	=IF($BI$1;IFMR.AD5;plots.BN5)	=IF($BI$1;IFMR.AE5;plots.BO5)	=IF($BI$1;IFMR.AF5;plots.BP5)	=IF($BI$1;IFMR.AG5;plots.BQ5)	=IF($BI$1;IFMR.V5;plots.BF5)	=IF($BI$1;IFMR.W5;plots.BG5)	=IF($BI$1;IFMR.X5;plots.BH5)
6			=IF($BI$1;IFMR.AB6;plots.BL6)	=IF($BI$1;IFMR.AC6;plots.BM6)	=IF($BI$1;IFMR.AD6;plots.BN6)	=IF($BI$1;IFMR.AE6;plots.BO6)	=IF($BI$1;IFMR.AF6;plots.BP6)	=IF($BI$1;IFMR.AG6;plots.BQ6)	=IF($BI$1;IFMR.V6;plots.BF6)	=IF($BI$1;IFMR.W6;plots.BG6)	=IF($BI$1;IFMR.X6;plots.BH6)
7			=IF($BI$1;IFMR.AB7;plots.BL7)	=IF($BI$1;IFMR.AC7;plots.BM7)	=IF($BI$1;IFMR.AD7;plots.BN7)	=IF($BI$1;IFMR.AE7;plots.BO7)	=IF($BI$1;IFMR.AF7;plots.BP7)	=IF($BI$1;IFMR.AG7;plots.BQ7)	=IF($BI$1;IFMR.V7;plots.BF7)	=IF($BI$1;IFMR.W7;plots.BG7)	=IF($BI$1;IFMR.X7;plots.BH7)
8			=IF($BI$1;IFMR.AB8;plots.BL8)	=IF($BI$1;IFMR.AC8;plots.BM8)	=IF($BI$1;IFMR.AD8;plots.BN8)	=IF($BI$1;IFMR.AE8;plots.BO8)	=IF($BI$1;IFMR.AF8;plots.BP8)	=IF($BI$1;IFMR.AG8;plots.BQ8)	=IF($BI$1;IFMR.V8;plots.BF8)	=IF($BI$1;IFMR.W8;plots.BG8)	=IF($BI$1;IFMR.X8;plots.BH8)
9			=IF($BI$1;IFMR.AB9;plots.BL9)	=IF($BI$1;IFMR.AC9;plots.BM9)	=IF($BI$1;IFMR.AD9;plots.BN9)	=IF($BI$1;IFMR.AE9;plots.BO9)	=IF($BI$1;IFMR.AF9;plots.BP9)	=IF($BI$1;IFMR.AG9;plots.BQ9)	=IF($BI$1;IFMR.V9;plots.BF9)	=IF($BI$1;IFMR.W9;plots.BG9)	=IF($BI$1;IFMR.X9;plots.BH9)
10			=IF($BI$1;IFMR.AB10;plots.BL10)	=IF($BI$1;IFMR.AC10;plots.BM10)	=IF($BI$1;IFMR.AD10;plots.BN10)	=IF($BI$1;IFMR.AE10;plots.BO10)	=IF($BI$1;IFMR.AF10;plots.BP10)	=IF($BI$1;IFMR.AG10;plots.BQ10)	=IF($BI$1;IFMR.V10;plots.BF10)	=IF($BI$1;IFMR.W10;plots.BG10)	=IF($BI$1;IFMR.X10;plots.BH10)
11			=IF($BI$1;IFMR.AB11;plots.BL11)	=IF($BI$1;IFMR.AC11;plots.BM11)	=IF($BI$1;IFMR.AD11;plots.BN11)	=IF($BI$1;IFMR.AE11;plots.BO11)	=IF($BI$1;IFMR.AF11;plots.BP11)	=IF($BI$1;IFMR.AG11;plots.BQ11)	=IF($BI$1;IFMR.V11;plots.BF11)	=IF($BI$1;IFMR.W11;plots.BG11)	=IF($BI$1;IFMR.X11;plots.BH11)
12			=IF($BI$1;IFMR.AB12;plots.BL12)	=IF($BI$1;IFMR.AC12;plots.BM12)	=IF($BI$1;IFMR.AD12;plots.BN12)	=IF($BI$1;IFMR.AE12;plots.BO12)	=IF($BI$1;IFMR.AF12;plots.BP12)	=IF($BI$1;IFMR.AG12;plots.BQ12)	=IF($BI$1;IFMR.V12;plots.BF12)	=IF($BI$1;IFMR.W12;plots.BG12)	=IF($BI$1;IFMR.X12;plots.BH12)
13			=IF($BI$1;IFMR.AB13;plots.BL13)	=IF($BI$1;IFMR.AC13;plots.BM13)	=IF($BI$1;IFMR.AD13;plots.BN13)	=IF($BI$1;IFMR.AE13;plots.BO13)	=IF($BI$1;IFMR.AF13;plots.BP13)	=IF($BI$1;IFMR.AG13;plots.BQ13)	=IF($BI$1;IFMR.V13;plots.BF13)	=IF($BI$1;IFMR.W13;plots.BG13)	=IF($BI$1;IFMR.X13;plots.BH13)
14			=IF($BI$1;IFMR.AB14;plots.BL14)	=IF($BI$1;IFMR.AC14;plots.BM14)	=IF($BI$1;IFMR.AD14;plots.BN14)	=IF($BI$1;IFMR.AE14;plots.BO14)	=IF($BI$1;IFMR.AF14;plots.BP14)	=IF($BI$1;IFMR.AG14;plots.BQ14)	=IF($BI$1;IFMR.V14;plots.BF14)	=IF($BI$1;IFMR.W14;plots.BG14)	=IF($BI$1;IFMR.X14;plots.BH14)
15			=IF($BI$1;IFMR.AB15;plots.BL15)	=IF($BI$1;IFMR.AC15;plots.BM15)	=IF($BI$1;IFMR.AD15;plots.BN15)	=IF($BI$1;IFMR.AE15;plots.BO15)	=IF($BI$1;IFMR.AF15;plots.BP15)	=IF($BI$1;IFMR.AG15;plots.BQ15)	=IF($BI$1;IFMR.V15;plots.BF15)	=IF($BI$1;IFMR.W15;plots.BG15)	=IF($BI$1;IFMR.X15;plots.BH15)
16			=IF($BI$1;IFMR.AB16;plots.BL16)	=IF($BI$1;IFMR.AC16;plots.BM16)	=IF($BI$1;IFMR.AD16;plots.BN16)	=IF($BI$1;IFMR.AE16;plots.BO16)	=IF($BI$1;IFMR.AF16;plots.BP16)	=IF($BI$1;IFMR.AG16;plots.BQ16)	=IF($BI$1;IFMR.V16;plots.BF16)	=IF($BI$1;IFMR.W16;plots.BG16)	=IF($BI$1;IFMR.X16;plots.BH16)
17			=IF($BI$1;IFMR.AB17;plots.BL17)	=IF($BI$1;IFMR.AC17;plots.BM17)	=IF($BI$1;IFMR.AD17;plots.BN17)	=IF($BI$1;IFMR.AE17;plots.BO17)	=IF($BI$1;IFMR.AF17;plots.BP17)	=IF($BI$1;IFMR.AG17;plots.BQ17)	=IF($BI$1;IFMR.V17;plots.BF17)	=IF($BI$1;IFMR.W17;plots.BG17)	=IF($BI$1;IFMR.X17;plots.BH17)
18			=IF($BI$1;IFMR.AB18;plots.BL18)	=IF($BI$1;IFMR.AC18;plots.BM18)	=IF($BI$1;IFMR.AD18;plots.BN18)	=IF($BI$1;IFMR.AE18;plots.BO18)	=IF($BI$1;IFMR.AF18;plots.BP18)	=IF($BI$1;IFMR.AG18;plots.BQ18)	=IF($BI$1;IFMR.V18;plots.BF18)	=IF($BI$1;IFMR.W18;plots.BG18)	=IF($BI$1;IFMR.X18;plots.BH18)
19			=IF($BI$1;IFMR.AB19;plots.BL19)	=IF($BI$1;IFMR.AC19;plots.BM19)	=IF($BI$1;IFMR.AD19;plots.BN19)	=IF($BI$1;IFMR.AE19;plots.BO19)	=IF($BI$1;IFMR.AF19;plots.BP19)	=IF($BI$1;IFMR.AG19;plots.BQ19)	=IF($BI$1;IFMR.V19;plots.BF19)	=IF($BI$1;IFMR.W19;plots.BG19)	=IF($BI$1;IFMR.X19;plots.BH19)
20			=IF($BI$1;IFMR.AB20;plots.BL20)	=IF($BI$1;IFMR.AC20;plots.BM20)	=IF($BI$1;IFMR.AD20;plots.BN20)	=IF($BI$1;IFMR.AE20;plots.BO20)	=IF($BI$1;IFMR.AF20;plots.BP20)	=IF($BI$1;IFMR.AG20;plots.BQ20)	=IF($BI$1;IFMR.V20;plots.BF20)	=IF($BI$1;IFMR.W20;plots.BG20)	=IF($BI$1;IFMR.X20;plots.BH20)
21			=IF($BI$1;IFMR.AB21;plots.BL21)	=IF($BI$1;IFMR.AC21;plots.BM21)	=IF($BI$1;IFMR.AD21;plots.BN21)	=IF($BI$1;IFMR.AE21;plots.BO21)	=IF($BI$1;IFMR.AF21;plots.BP21)	=IF($BI$1;IFMR.AG21;plots.BQ21)	=IF($BI$1;IFMR.V21;plots.BF21)	=IF($BI$1;IFMR.W21;plots.BG21)	=IF($BI$1;IFMR.X21;plots.BH21)
22			=IF($BI$1;IFMR.AB22;plots.BL22)	=IF($BI$1;IFMR.AC22;plots.BM22)	=IF($BI$1;IFMR.AD22;plots.BN22)	=IF($BI$1;IFMR.AE22;plots.BO22)	=IF($BI$1;IFMR.AF22;plots.BP22)	=IF($BI$1;IFMR.AG22;plots.BQ22)	=IF($BI$1;IFMR.V22;plots.BF22)	=IF($BI$1;IFMR.W22;plots.BG22)	=IF($BI$1;IFMR.X22;plots.BH22)
23			=IF($BI$1;IFMR.AB23;plots.BL23)	=IF($BI$1;IFMR.AC23;plots.BM23)	=IF($BI$1;IFMR.AD23;plots.BN23)	=IF($BI$1;IFMR.AE23;plots.BO23)	=IF($BI$1;IFMR.AF23;plots.BP23)	=IF($BI$1;IFMR.AG23;plots.BQ23)	=IF($BI$1;IFMR.V23;plots.BF23)	=IF($BI$1;IFMR.W23;plots.BG23)	=IF($BI$1;IFMR.X23;plots.BH23)
24			=IF($BI$1;IFMR.AB24;plots.BL24)	=IF($BI$1;IFMR.AC24;plots.BM24)	=IF($BI$1;IFMR.AD24;plots.BN24)	=IF($BI$1;IFMR.AE24;plots.BO24)	=IF($BI$1;IFMR.AF24;plots.BP24)	=IF($BI$1;IFMR.AG24;plots.BQ24)	=IF($BI$1;IFMR.V24;plots.BF24)	=IF($BI$1;IFMR.W24;plots.BG24)	=IF($BI$1;IFMR.X24;plots.BH24)
25			=IF($BI$1;IFMR.AB25;plots.BL25)	=IF($BI$1;IFMR.AC25;plots.BM25)	=IF($BI$1;IFMR.AD25;plots.BN25)	=IF($BI$1;IFMR.AE25;plots.BO25)	=IF($BI$1;IFMR.AF25;plots.BP25)	=IF($BI$1;IFMR.AG25;plots.BQ25)	=IF($BI$1;IFMR.V25;plots.BF25)	=IF($BI$1;IFMR.W25;plots.BG25)	=IF($BI$1;IFMR.X25;plots.BH25)
26			=IF($BI$1;IFMR.AB26;plots.BL26)	=IF($BI$1;IFMR.AC26;plots.BM26)	=IF($BI$1;IFMR.AD26;plots.BN26)	=IF($BI$1;IFMR.AE26;plots.BO26)	=IF($BI$1;IFMR.AF26;plots.BP26)	=IF($BI$1;IFMR.AG26;plots.BQ26)	=IF($BI$1;IFMR.V26;plots.BF26)	=IF($BI$1;IFMR.W26;plots.BG26)	=IF($BI$1;IFMR.X26;plots.BH26)
27			=IF($BI$1;IFMR.AB27;plots.BL27)	=IF($BI$1;IFMR.AC27;plots.BM27)	=IF($BI$1;IFMR.AD27;plots.BN27)	=IF($BI$1;IFMR.AE27;plots.BO27)	=IF($BI$1;IFMR.AF27;plots.BP27)	=IF($BI$1;IFMR.AG27;plots.BQ27)	=IF($BI$1;IFMR.V27;plots.BF27)	=IF($BI$1;IFMR.W27;plots.BG27)	=IF($BI$1;IFMR.X27;plots.BH27)
28			=IF($BI$1;IFMR.AB28;plots.BL28)	=IF($BI$1;IFMR.AC28;plots.BM28)	=IF($BI$1;IFMR.AD28;plots.BN28)	=IF($BI$1;IFMR.AE28;plots.BO28)	=IF($BI$1;IFMR.AF28;plots.BP28)	=IF($BI$1;IFMR.AG28;plots.BQ28)	=IF($BI$1;IFMR.V28;plots.BF28)	=IF($BI$1;IFMR.W28;plots.BG28)	=IF($BI$1;IFMR.X28;plots.BH28)
29											
30											
31											
32											
33			=IF($BI$1;IFMR.AB33;plots.BL33)	=IF($BI$1;IFMR.AC33;plots.BM33)	=IF($BI$1;IFMR.AD33;plots.BN33)	=IF($BI$1;IFMR.AE33;plots.BO33)	=IF($BI$1;IFMR.AF33;plots.BP33)	=IF($BI$1;IFMR.AG33;plots.BQ33)	=IF($BI$1;IFMR.V33;plots.BF33)	=IF($BI$1;IFMR.W33;plots.BG33)	=IF($BI$1;IFMR.X33;plots.BH33)
34			=IF($BI$1;IFMR.AB34;plots.BL34)	=IF($BI$1;IFMR.AC34;plots.BM34)	=IF($BI$1;IFMR.AD34;plots.BN34)	=IF($BI$1;IFMR.AE34;plots.BO34)	=IF($BI$1;IFMR.AF34;plots.BP34)	=IF($BI$1;IFMR.AG34;plots.BQ34)	=IF($BI$1;IFMR.V34;plots.BF34)	=IF($BI$1;IFMR.W34;plots.BG34)	=IF($BI$1;IFMR.X34;plots.BH34)
35			=IF($BI$1;IFMR.AB35;plots.BL35)	=IF($BI$1;IFMR.AC35;plots.BM35)	=IF($BI$1;IFMR.AD35;plots.BN35)	=IF($BI$1;IFMR.AE35;plots.BO35)	=IF($BI$1;IFMR.AF35;plots.BP35)	=IF($BI$1;IFMR.AG35;plots.BQ35)	=IF($BI$1;IFMR.V35;plots.BF35)	=IF($BI$1;IFMR.W35;plots.BG35)	=IF($BI$1;IFMR.X35;plots.BH35)
36			=IF($BI$1;IFMR.AB36;plots.BL36)	=IF($BI$1;IFMR.AC36;plots.BM36)	=IF($BI$1;IFMR.AD36;plots.BN36)	=IF($BI$1;IFMR.AE36;plots.BO36)	=IF($BI$1;IFMR.AF36;plots.BP36)	=IF($BI$1;IFMR.AG36;plots.BQ36)	=IF($BI$1;IFMR.V36;plots.BF36)	=IF($BI$1;IFMR.W36;plots.BG36)	=IF($BI$1;IFMR.X36;plots.BH36)
37											
38											
39											
40											
41			IFMR used;								
42			Final mass = m * InitialMass + c								
43			dobbie_new_2006	Theoretical Values	error						
44			IFMR m =	0.1330	0.0150						
45			IFMR c =	0.2890	0.0510						

fin

	BU	BV	BW	BX	BY	BZ	CA	CB	CC	CD	CE	CF
1												
2	MS Age	MS Age Min	MS Age Max	System Age	System Age Min	System Age Max	distance	distance min	distance max		Radius	Radius min
3												
4												
5	=IF($BI$1;IFMR.Y5;plots.BI5)	=IF($BI$1;IFMR.Z5;plots.BJ5)	=IF($BI$1;IFMR.AA5;plots.BK5)	=IF($BI$1;IFMR.AH5;plots.BR5)	=IF($BI$1;IFMR.AI5;plots.BS5)	=IF($BI$1;IFMR.AJ5;plots.BT5)	=AD5	=IF($BI$1;IFMR.AR5;plots.BY5)	=IF($BI$1;IFMR.AS5;plots.BZ5)		=plots.C5	=plots.E5
6	=IF($BI$1;IFMR.Y6;plots.BI6)	=IF($BI$1;IFMR.Z6;plots.BJ6)	=IF($BI$1;IFMR.AA6;plots.BK6)	=IF($BI$1;IFMR.AH6;plots.BR6)	=IF($BI$1;IFMR.AI6;plots.BS6)	=IF($BI$1;IFMR.AJ6;plots.BT6)	=AD6	=IF($BI$1;IFMR.AR6;plots.BY6)	=IF($BI$1;IFMR.AS6;plots.BZ6)		=plots.C6	=plots.E6
7	=IF($BI$1;IFMR.Y7;plots.BI7)	=IF($BI$1;IFMR.Z7;plots.BJ7)	=IF($BI$1;IFMR.AA7;plots.BK7)	=IF($BI$1;IFMR.AH7;plots.BR7)	=IF($BI$1;IFMR.AI7;plots.BS7)	=IF($BI$1;IFMR.AJ7;plots.BT7)	=AD7	=IF($BI$1;IFMR.AR7;plots.BY7)	=IF($BI$1;IFMR.AS7;plots.BZ7)		=plots.C7	=plots.E7
8	=IF($BI$1;IFMR.Y8;plots.BI8)	=IF($BI$1;IFMR.Z8;plots.BJ8)	=IF($BI$1;IFMR.AA8;plots.BK8)	=IF($BI$1;IFMR.AH8;plots.BR8)	=IF($BI$1;IFMR.AI8;plots.BS8)	=IF($BI$1;IFMR.AJ8;plots.BT8)	=AD8	=IF($BI$1;IFMR.AR8;plots.BY8)	=IF($BI$1;IFMR.AS8;plots.BZ8)		=plots.C8	=plots.E8
9	=IF($BI$1;IFMR.Y9;plots.BI9)	=IF($BI$1;IFMR.Z9;plots.BJ9)	=IF($BI$1;IFMR.AA9;plots.BK9)	=IF($BI$1;IFMR.AH9;plots.BR9)	=IF($BI$1;IFMR.AI9;plots.BS9)	=IF($BI$1;IFMR.AJ9;plots.BT9)	=AD9	=IF($BI$1;IFMR.AR9;plots.BY9)	=IF($BI$1;IFMR.AS9;plots.BZ9)		=plots.C9	=plots.E9
10	=IF($BI$1;IFMR.Y10;plots.BI10)	=IF($BI$1;IFMR.Z10;plots.BJ10)	=IF($BI$1;IFMR.AA10;plots.BK10)	=IF($BI$1;IFMR.AH10;plots.BR10)	=IF($BI$1;IFMR.AI10;plots.BS10)	=IF($BI$1;IFMR.AJ10;plots.BT10)	=AD10	=IF($BI$1;IFMR.AR10;plots.BY10)	=IF($BI$1;IFMR.AS10;plots.BZ10)		=plots.C10	=plots.E10
11	=IF($BI$1;IFMR.Y11;plots.BI11)	=IF($BI$1;IFMR.Z11;plots.BJ11)	=IF($BI$1;IFMR.AA11;plots.BK11)	=IF($BI$1;IFMR.AH11;plots.BR11)	=IF($BI$1;IFMR.AI11;plots.BS11)	=IF($BI$1;IFMR.AJ11;plots.BT11)	=AD11	=IF($BI$1;IFMR.AR11;plots.BY11)	=IF($BI$1;IFMR.AS11;plots.BZ11)		=plots.C11	=plots.E11
12	=IF($BI$1;IFMR.Y12;plots.BI12)	=IF($BI$1;IFMR.Z12;plots.BJ12)	=IF($BI$1;IFMR.AA12;plots.BK12)	=IF($BI$1;IFMR.AH12;plots.BR12)	=IF($BI$1;IFMR.AI12;plots.BS12)	=IF($BI$1;IFMR.AJ12;plots.BT12)	=AD12	=IF($BI$1;IFMR.AR12;plots.BY12)	=IF($BI$1;IFMR.AS12;plots.BZ12)		=plots.C12	=plots.E12
13	=IF($BI$1;IFMR.Y13;plots.BI13)	=IF($BI$1;IFMR.Z13;plots.BJ13)	=IF($BI$1;IFMR.AA13;plots.BK13)	=IF($BI$1;IFMR.AH13;plots.BR13)	=IF($BI$1;IFMR.AI13;plots.BS13)	=IF($BI$1;IFMR.AJ13;plots.BT13)	=AD13	=IF($BI$1;IFMR.AR13;plots.BY13)	=IF($BI$1;IFMR.AS13;plots.BZ13)		=plots.C13	=plots.E13
14	=IF($BI$1;IFMR.Y14;plots.BI14)	=IF($BI$1;IFMR.Z14;plots.BJ14)	=IF($BI$1;IFMR.AA14;plots.BK14)	=IF($BI$1;IFMR.AH14;plots.BR14)	=IF($BI$1;IFMR.AI14;plots.BS14)	=IF($BI$1;IFMR.AJ14;plots.BT14)	=AD14	=IF($BI$1;IFMR.AR14;plots.BY14)	=IF($BI$1;IFMR.AS14;plots.BZ14)		=plots.C14	=plots.E14
15	=IF($BI$1;IFMR.Y15;plots.BI15)	=IF($BI$1;IFMR.Z15;plots.BJ15)	=IF($BI$1;IFMR.AA15;plots.BK15)	=IF($BI$1;IFMR.AH15;plots.BR15)	=IF($BI$1;IFMR.AI15;plots.BS15)	=IF($BI$1;IFMR.AJ15;plots.BT15)	=AD15	=IF($BI$1;IFMR.AR15;plots.BY15)	=IF($BI$1;IFMR.AS15;plots.BZ15)		=plots.C15	=plots.E15
16	=IF($BI$1;IFMR.Y16;plots.BI16)	=IF($BI$1;IFMR.Z16;plots.BJ16)	=IF($BI$1;IFMR.AA16;plots.BK16)	=IF($BI$1;IFMR.AH16;plots.BR16)	=IF($BI$1;IFMR.AI16;plots.BS16)	=IF($BI$1;IFMR.AJ16;plots.BT16)	=AD16	=IF($BI$1;IFMR.AR16;plots.BY16)	=IF($BI$1;IFMR.AS16;plots.BZ16)		=plots.C16	=plots.E16
17	=IF($BI$1;IFMR.Y17;plots.BI17)	=IF($BI$1;IFMR.Z17;plots.BJ17)	=IF($BI$1;IFMR.AA17;plots.BK17)	=IF($BI$1;IFMR.AH17;plots.BR17)	=IF($BI$1;IFMR.AI17;plots.BS17)	=IF($BI$1;IFMR.AJ17;plots.BT17)	=AD17	=IF($BI$1;IFMR.AR17;plots.BY17)	=IF($BI$1;IFMR.AS17;plots.BZ17)		=plots.C17	=plots.E17
18	=IF($BI$1;IFMR.Y18;plots.BI18)	=IF($BI$1;IFMR.Z18;plots.BJ18)	=IF($BI$1;IFMR.AA18;plots.BK18)	=IF($BI$1;IFMR.AH18;plots.BR18)	=IF($BI$1;IFMR.AI18;plots.BS18)	=IF($BI$1;IFMR.AJ18;plots.BT18)	=AD18	=IF($BI$1;IFMR.AR18;plots.BY18)	=IF($BI$1;IFMR.AS18;plots.BZ18)		=plots.C18	=plots.E18
19	=IF($BI$1;IFMR.Y19;plots.BI19)	=IF($BI$1;IFMR.Z19;plots.BJ19)	=IF($BI$1;IFMR.AA19;plots.BK19)	=IF($BI$1;IFMR.AH19;plots.BR19)	=IF($BI$1;IFMR.AI19;plots.BS19)	=IF($BI$1;IFMR.AJ19;plots.BT19)	=AD19	=IF($BI$1;IFMR.AR19;plots.BY19)	=IF($BI$1;IFMR.AS19;plots.BZ19)		=plots.C19	=plots.E19
20	=IF($BI$1;IFMR.Y20;plots.BI20)	=IF($BI$1;IFMR.Z20;plots.BJ20)	=IF($BI$1;IFMR.AA20;plots.BK20)	=IF($BI$1;IFMR.AH20;plots.BR20)	=IF($BI$1;IFMR.AI20;plots.BS20)	=IF($BI$1;IFMR.AJ20;plots.BT20)	=AD20	=IF($BI$1;IFMR.AR20;plots.BY20)	=IF($BI$1;IFMR.AS20;plots.BZ20)		=plots.C20	=plots.E20
21	=IF($BI$1;IFMR.Y21;plots.BI21)	=IF($BI$1;IFMR.Z21;plots.BJ21)	=IF($BI$1;IFMR.AA21;plots.BK21)	=IF($BI$1;IFMR.AH21;plots.BR21)	=IF($BI$1;IFMR.AI21;plots.BS21)	=IF($BI$1;IFMR.AJ21;plots.BT21)	=AD21	=IF($BI$1;IFMR.AR21;plots.BY21)	=IF($BI$1;IFMR.AS21;plots.BZ21)		=plots.C21	=plots.E21
22	=IF($BI$1;IFMR.Y22;plots.BI22)	=IF($BI$1;IFMR.Z22;plots.BJ22)	=IF($BI$1;IFMR.AA22;plots.BK22)	=IF($BI$1;IFMR.AH22;plots.BR22)	=IF($BI$1;IFMR.AI22;plots.BS22)	=IF($BI$1;IFMR.AJ22;plots.BT22)	=AD22	=IF($BI$1;IFMR.AR22;plots.BY22)	=IF($BI$1;IFMR.AS22;plots.BZ22)		=plots.C22	=plots.E22
23	=IF($BI$1;IFMR.Y23;plots.BI23)	=IF($BI$1;IFMR.Z23;plots.BJ23)	=IF($BI$1;IFMR.AA23;plots.BK23)	=IF($BI$1;IFMR.AH23;plots.BR23)	=IF($BI$1;IFMR.AI23;plots.BS23)	=IF($BI$1;IFMR.AJ23;plots.BT23)	=AD23	=IF($BI$1;IFMR.AR23;plots.BY23)	=IF($BI$1;IFMR.AS23;plots.BZ23)		=plots.C23	=plots.E23
24	=IF($BI$1;IFMR.Y24;plots.BI24)	=IF($BI$1;IFMR.Z24;plots.BJ24)	=IF($BI$1;IFMR.AA24;plots.BK24)	=IF($BI$1;IFMR.AH24;plots.BR24)	=IF($BI$1;IFMR.AI24;plots.BS24)	=IF($BI$1;IFMR.AJ24;plots.BT24)	=AD24	=IF($BI$1;IFMR.AR24;plots.BY24)	=IF($BI$1;IFMR.AS24;plots.BZ24)		=plots.C24	=plots.E24
25	=IF($BI$1;IFMR.Y25;plots.BI25)	=IF($BI$1;IFMR.Z25;plots.BJ25)	=IF($BI$1;IFMR.AA25;plots.BK25)	=IF($BI$1;IFMR.AH25;plots.BR25)	=IF($BI$1;IFMR.AI25;plots.BS25)	=IF($BI$1;IFMR.AJ25;plots.BT25)	=AD25	=IF($BI$1;IFMR.AR25;plots.BY25)	=IF($BI$1;IFMR.AS25;plots.BZ25)		=plots.C25	=plots.E25
26	=IF($BI$1;IFMR.Y26;plots.BI26)	=IF($BI$1;IFMR.Z26;plots.BJ26)	=IF($BI$1;IFMR.AA26;plots.BK26)	=IF($BI$1;IFMR.AH26;plots.BR26)	=IF($BI$1;IFMR.AI26;plots.BS26)	=IF($BI$1;IFMR.AJ26;plots.BT26)	=AD26	=IF($BI$1;IFMR.AR26;plots.BY26)	=IF($BI$1;IFMR.AS26;plots.BZ26)		=plots.C26	=plots.E26
27	=IF($BI$1;IFMR.Y27;plots.BI27)	=IF($BI$1;IFMR.Z27;plots.BJ27)	=IF($BI$1;IFMR.AA27;plots.BK27)	=IF($BI$1;IFMR.AH27;plots.BR27)	=IF($BI$1;IFMR.AI27;plots.BS27)	=IF($BI$1;IFMR.AJ27;plots.BT27)	=AD27	=IF($BI$1;IFMR.AR27;plots.BY27)	=IF($BI$1;IFMR.AS27;plots.BZ27)		=plots.C27	=plots.E27
28	=IF($BI$1;IFMR.Y28;plots.BI28)	=IF($BI$1;IFMR.Z28;plots.BJ28)	=IF($BI$1;IFMR.AA28;plots.BK28)	=IF($BI$1;IFMR.AH28;plots.BR28)	=IF($BI$1;IFMR.AI28;plots.BS28)	=IF($BI$1;IFMR.AJ28;plots.BT28)	=AD28	=IF($BI$1;IFMR.AR28;plots.BY28)	=IF($BI$1;IFMR.AS28;plots.BZ28)		=plots.C28	=plots.E28
29												
30												
31												
32												
33	=IF($BI$1;IFMR.Y33;plots.BI33)	=IF($BI$1;IFMR.Z33;plots.BJ33)	=IF($BI$1;IFMR.AA33;plots.BK33)	=IF($BI$1;IFMR.AH33;plots.BR33)	=IF($BI$1;IFMR.AI33;plots.BS33)	=IF($BI$1;IFMR.AJ33;plots.BT33)	=AD33	=IF($BI$1;IFMR.AR33;plots.BY33)	=IF($BI$1;IFMR.AS33;plots.BZ33)		=plots.C33	=plots.E33
34	=IF($BI$1;IFMR.Y34;plots.BI34)	=IF($BI$1;IFMR.Z34;plots.BJ34)	=IF($BI$1;IFMR.AA34;plots.BK34)	=IF($BI$1;IFMR.AH34;plots.BR34)	=IF($BI$1;IFMR.AI34;plots.BS34)	=IF($BI$1;IFMR.AJ34;plots.BT34)	=AD34	=IF($BI$1;IFMR.AR34;plots.BY34)	=IF($BI$1;IFMR.AS34;plots.BZ34)		=plots.C34	=plots.E34
35	=IF($BI$1;IFMR.Y35;plots.BI35)	=IF($BI$1;IFMR.Z35;plots.BJ35)	=IF($BI$1;IFMR.AA35;plots.BK35)	=IF($BI$1;IFMR.AH35;plots.BR35)	=IF($BI$1;IFMR.AI35;plots.BS35)	=IF($BI$1;IFMR.AJ35;plots.BT35)	=AD35	=IF($BI$1;IFMR.AR35;plots.BY35)	=IF($BI$1;IFMR.AS35;plots.BZ35)		=plots.C35	=plots.E35
36	=IF($BI$1;IFMR.Y36;plots.BI36)	=IF($BI$1;IFMR.Z36;plots.BJ36)	=IF($BI$1;IFMR.AA36;plots.BK36)	=IF($BI$1;IFMR.AH36;plots.BR36)	=IF($BI$1;IFMR.AI36;plots.BS36)	=IF($BI$1;IFMR.AJ36;plots.BT36)	=AD36	=IF($BI$1;IFMR.AR36;plots.BY36)	=IF($BI$1;IFMR.AS36;plots.BZ36)		=plots.C36	=plots.E36
37												
38												
39							OLD: =IF($BK$1;IFMR.DG5;plots.BX5)					
40												
41												
42												
43												
44												
45												

	CG	CH	CI	CJ	CK	CL	CM	CN	CO	CP	CQ	CR	CS	CT	CU
1															
2	Radius max	GR	GR min	GR max	M(g)	M(g) min	M(g) max	STRICTsystemAgeCommon.txt	mass error	cooling age error	distance error		distance (m)	distance - error (m)	tangential velocity (m/year)
3															
4															
5	=plots.D5	=plots.I5	=plots.K5	=plots.J5	=plots.F5	=plots.H5	=plots.G5	=IF($BI$1;IFMR.AK5;plots.BU5)	=(ABS(BL5-BM5)+ABS(BL5-BN5))/2	=(ABS(BO5-BP5)+ABS(BO5-BQ5))/2	=AE5		=10^((CA5/5)+1)*(3.08568025*10^16)	=10^(((AD5-AE5)/5)+1)*(3.08568025*10^16)	=TAN(BD5)*CS5
6	=plots.D6	=plots.I6	=plots.K6	=plots.J6	=plots.F6	=plots.H6	=plots.G6	=IF($BI$1;IFMR.AK6;plots.BU6)	=(ABS(BL6-BM6)+ABS(BL6-BN6))/2	=(ABS(BO6-BP6)+ABS(BO6-BQ6))/2	=AE6		=10^((AD6/5)+1)*(3.08568025*10^16)	=10^(((AD6-AE6)/5)+1)*(3.08568025*10^16)	=TAN(BD6)*CS6
7	=plots.D7	=plots.I7	=plots.K7	=plots.J7	=plots.F7	=plots.H7	=plots.G7	=IF($BI$1;IFMR.AK7;plots.BU7)	=(ABS(BL7-BM7)+ABS(BL7-BN7))/2	=(ABS(BO7-BP7)+ABS(BO7-BQ7))/2	=AE7		=10^((AD7/5)+1)*(3.08568025*10^16)	=10^(((AD7-AE7)/5)+1)*(3.08568025*10^16)	=TAN(BD7)*CS7
8	=plots.D8	=plots.I8	=plots.K8	=plots.J8	=plots.F8	=plots.H8	=plots.G8	=IF($BI$1;IFMR.AK8;plots.BU8)	=(ABS(BL8-BM8)+ABS(BL8-BN8))/2	=(ABS(BO8-BP8)+ABS(BO8-BQ8))/2	=AE8		=10^((AD8/5)+1)*(3.08568025*10^16)	=10^(((AD8-AE8)/5)+1)*(3.08568025*10^16)	=TAN(BD8)*CS8
9	=plots.D9	=plots.I9	=plots.K9	=plots.J9	=plots.F9	=plots.H9	=plots.G9	=IF($BI$1;IFMR.AK9;plots.BU9)	=(ABS(BL9-BM9)+ABS(BL9-BN9))/2	=(ABS(BO9-BP9)+ABS(BO9-BQ9))/2	=AE9		=10^((AD9/5)+1)*(3.08568025*10^16)	=10^(((AD9-AE9)/5)+1)*(3.08568025*10^16)	=TAN(BD9)*CS9
10	=plots.D10	=plots.I10	=plots.K10	=plots.J10	=plots.F10	=plots.H10	=plots.G10	=IF($BI$1;IFMR.AK10;plots.BU10)	=(ABS(BL10-BM10)+ABS(BL10-BN10))/2	=(ABS(BO10-BP10)+ABS(BO10-BQ10))/2	=AE10		=10^((AD10/5)+1)*(3.08568025*10^16)	=10^(((AD10-AE10)/5)+1)*(3.08568025*10^16)	=TAN(BD10)*CS10
11	=plots.D11	=plots.I11	=plots.K11	=plots.J11	=plots.F11	=plots.H11	=plots.G11	=IF($BI$1;IFMR.AK11;plots.BU11)	=(ABS(BL11-BM11)+ABS(BL11-BN11))/2	=(ABS(BO11-BP11)+ABS(BO11-BQ11))/2	=AE11		=10^((AD11/5)+1)*(3.08568025*10^16)	=10^(((AD11-AE11)/5)+1)*(3.08568025*10^16)	=TAN(BD11)*CS11
12	=plots.D12	=plots.I12	=plots.K12	=plots.J12	=plots.F12	=plots.H12	=plots.G12	=IF($BI$1;IFMR.AK12;plots.BU12)	=(ABS(BL12-BM12)+ABS(BL12-BN12))/2	=(ABS(BO12-BP12)+ABS(BO12-BQ12))/2	=AE12		=10^((AD12/5)+1)*(3.08568025*10^16)	=10^(((AD12-AE12)/5)+1)*(3.08568025*10^16)	=TAN(BD12)*CS12
13	=plots.D13	=plots.I13	=plots.K13	=plots.J13	=plots.F13	=plots.H13	=plots.G13	=IF($BI$1;IFMR.AK13;plots.BU13)	=(ABS(BL13-BM13)+ABS(BL13-BN13))/2	=(ABS(BO13-BP13)+ABS(BO13-BQ13))/2	=AE13		=10^((AD13/5)+1)*(3.08568025*10^16)	=10^(((AD13-AE13)/5)+1)*(3.08568025*10^16)	=TAN(BD13)*CS13
14	=plots.D14	=plots.I14	=plots.K14	=plots.J14	=plots.F14	=plots.H14	=plots.G14	=IF($BI$1;IFMR.AK14;plots.BU14)	=(ABS(BL14-BM14)+ABS(BL14-BN14))/2	=(ABS(BO14-BP14)+ABS(BO14-BQ14))/2	=AE14		=10^((AD14/5)+1)*(3.08568025*10^16)	=10^(((AD14-AE14)/5)+1)*(3.08568025*10^16)	=TAN(BD14)*CS14
15	=plots.D15	=plots.I15	=plots.K15	=plots.J15	=plots.F15	=plots.H15	=plots.G15	=IF($BI$1;IFMR.AK15;plots.BU15)	=(ABS(BL15-BM15)+ABS(BL15-BN15))/2	=(ABS(BO15-BP15)+ABS(BO15-BQ15))/2	=AE15		=10^((AD15/5)+1)*(3.08568025*10^16)	=10^(((AD15-AE15)/5)+1)*(3.08568025*10^16)	=TAN(BD15)*CS15
16	=plots.D16	=plots.I16	=plots.K16	=plots.J16	=plots.F16	=plots.H16	=plots.G16	=IF($BI$1;IFMR.AK16;plots.BU16)	=(ABS(BL16-BM16)+ABS(BL16-BN16))/2	=(ABS(BO16-BP16)+ABS(BO16-BQ16))/2	=AE16		=10^((AD16/5)+1)*(3.08568025*10^16)	=10^(((AD16-AE16)/5)+1)*(3.08568025*10^16)	=TAN(BD16)*CS16
17	=plots.D17	=plots.I17	=plots.K17	=plots.J17	=plots.F17	=plots.H17	=plots.G17	=IF($BI$1;IFMR.AK17;plots.BU17)	=(ABS(BL17-BM17)+ABS(BL17-BN17))/2	=(ABS(BO17-BP17)+ABS(BO17-BQ17))/2	=AE17		=10^((AD17/5)+1)*(3.08568025*10^16)	=10^(((AD17-AE17)/5)+1)*(3.08568025*10^16)	=TAN(BD17)*CS17
18	=plots.D18	=plots.I18	=plots.K18	=plots.J18	=plots.F18	=plots.H18	=plots.G18	=IF($BI$1;IFMR.AK18;plots.BU18)	=(ABS(BL18-BM18)+ABS(BL18-BN18))/2	=(ABS(BO18-BP18)+ABS(BO18-BQ18))/2	=AE18		=10^((AD18/5)+1)*(3.08568025*10^16)	=10^(((AD18-AE18)/5)+1)*(3.08568025*10^16)	=TAN(BD18)*CS18
19	=plots.D19	=plots.I19	=plots.K19	=plots.J19	=plots.F19	=plots.H19	=plots.G19	=IF($BI$1;IFMR.AK19;plots.BU19)	=(ABS(BL19-BM19)+ABS(BL19-BN19))/2	=(ABS(BO19-BP19)+ABS(BO19-BQ19))/2	=AE19		=10^((AD19/5)+1)*(3.08568025*10^16)	=10^(((AD19-AE19)/5)+1)*(3.08568025*10^16)	=TAN(BD19)*CS19
20	=plots.D20	=plots.I20	=plots.K20	=plots.J20	=plots.F20	=plots.H20	=plots.G20	=IF($BI$1;IFMR.AK20;plots.BU20)	=(ABS(BL20-BM20)+ABS(BL20-BN20))/2	=(ABS(BO20-BP20)+ABS(BO20-BQ20))/2	=AE20		=10^((AD20/5)+1)*(3.08568025*10^16)	=10^(((AD20-AE20)/5)+1)*(3.08568025*10^16)	=TAN(BD20)*CS20
21	=plots.D21	=plots.I21	=plots.K21	=plots.J21	=plots.F21	=plots.H21	=plots.G21	=IF($BI$1;IFMR.AK21;plots.BU21)	=(ABS(BL21-BM21)+ABS(BL21-BN21))/2	=(ABS(BO21-BP21)+ABS(BO21-BQ21))/2	=AE21		=10^((AD21/5)+1)*(3.08568025*10^16)	=10^(((AD21-AE21)/5)+1)*(3.08568025*10^16)	=TAN(BD21)*CS21
22	=plots.D22	=plots.I22	=plots.K22	=plots.J22	=plots.F22	=plots.H22	=plots.G22	=IF($BI$1;IFMR.AK22;plots.BU22)	=(ABS(BL22-BM22)+ABS(BL22-BN22))/2	=(ABS(BO22-BP22)+ABS(BO22-BQ22))/2	=AE22		=10^((AD22/5)+1)*(3.08568025*10^16)	=10^(((AD22-AE22)/5)+1)*(3.08568025*10^16)	=TAN(BD22)*CS22
23	=plots.D23	=plots.I23	=plots.K23	=plots.J23	=plots.F23	=plots.H23	=plots.G23	=IF($BI$1;IFMR.AK23;plots.BU23)	=(ABS(BL23-BM23)+ABS(BL23-BN23))/2	=(ABS(BO23-BP23)+ABS(BO23-BQ23))/2	=AE23		=10^((AD23/5)+1)*(3.08568025*10^16)	=10^(((AD23-AE23)/5)+1)*(3.08568025*10^16)	=TAN(BD23)*CS23
24	=plots.D24	=plots.I24	=plots.K24	=plots.J24	=plots.F24	=plots.H24	=plots.G24	=IF($BI$1;IFMR.AK24;plots.BU24)	=(ABS(BL24-BM24)+ABS(BL24-BN24))/2	=(ABS(BO24-BP24)+ABS(BO24-BQ24))/2	=AE24		=10^((AD24/5)+1)*(3.08568025*10^16)	=10^(((AD24-AE24)/5)+1)*(3.08568025*10^16)	=TAN(BD24)*CS24
25	=plots.D25	=plots.I25	=plots.K25	=plots.J25	=plots.F25	=plots.H25	=plots.G25	=IF($BI$1;IFMR.AK25;plots.BU25)	=(ABS(BL25-BM25)+ABS(BL25-BN25))/2	=(ABS(BO25-BP25)+ABS(BO25-BQ25))/2	=AE25		=10^((AD25/5)+1)*(3.08568025*10^16)	=10^(((AD25-AE25)/5)+1)*(3.08568025*10^16)	=TAN(BD25)*CS25
26	=plots.D26	=plots.I26	=plots.K26	=plots.J26	=plots.F26	=plots.H26	=plots.G26	=IF($BI$1;IFMR.AK26;plots.BU26)	=(ABS(BL26-BM26)+ABS(BL26-BN26))/2	=(ABS(BO26-BP26)+ABS(BO26-BQ26))/2	=AE26		=10^((AD26/5)+1)*(3.08568025*10^16)	=10^(((AD26-AE26)/5)+1)*(3.08568025*10^16)	=TAN(BD26)*CS26
27	=plots.D27	=plots.I27	=plots.K27	=plots.J27	=plots.F27	=plots.H27	=plots.G27	=IF($BI$1;IFMR.AK27;plots.BU27)	=(ABS(BL27-BM27)+ABS(BL27-BN27))/2	=(ABS(BO27-BP27)+ABS(BO27-BQ27))/2	=AE27		=10^((AD27/5)+1)*(3.08568025*10^16)	=10^(((AD27-AE27)/5)+1)*(3.08568025*10^16)	=TAN(BD27)*CS27
28	=plots.D28	=plots.I28	=plots.K28	=plots.J28	=plots.F28	=plots.H28	=plots.G28	=IF($BI$1;IFMR.AK28;plots.BU28)	=(ABS(BL28-BM28)+ABS(BL28-BN28))/2	=(ABS(BO28-BP28)+ABS(BO28-BQ28))/2	=AE28		=10^((AD28/5)+1)*(3.08568025*10^16)	=10^(((AD28-AE28)/5)+1)*(3.08568025*10^16)	=TAN(BD28)*CS28
29															
30															
31															
32															
33	=plots.D33	=plots.I33	=plots.K33	=plots.J33	=plots.F33	=plots.H33	=plots.G33	=IF($BI$1;IFMR.AK33;plots.BU33)	=(ABS(BL33-BM33)+ABS(BL33-BN33))/2	=(ABS(BO33-BP33)+ABS(BO33-BQ33))/2	=AE33		=10^((AD33/5)+1)*(3.08568025*10^16)	=10^(((AD33-AE33)/5)+1)*(3.08568025*10^16)	=TAN(BD33)*CS33
34	=plots.D34	=plots.I34	=plots.K34	=plots.J34	=plots.F34	=plots.H34	=plots.G34	=IF($BI$1;IFMR.AK34;plots.BU34)	=(ABS(BL34-BM34)+ABS(BL34-BN34))/2	=(ABS(BO34-BP34)+ABS(BO34-BQ34))/2	=AE34		=10^((AD34/5)+1)*(3.08568025*10^16)	=10^(((AD34-AE34)/5)+1)*(3.08568025*10^16)	=TAN(BD34)*CS34
35	=plots.D35	=plots.I35	=plots.K35	=plots.J35	=plots.F35	=plots.H35	=plots.G35	=IF($BI$1;IFMR.AK35;plots.BU35)	=(ABS(BL35-BM35)+ABS(BL35-BN35))/2	=(ABS(BO35-BP35)+ABS(BO35-BQ35))/2	=AE35		=10^((AD35/5)+1)*(3.08568025*10^16)	=10^(((AD35-AE35)/5)+1)*(3.08568025*10^16)	=TAN(BD35)*CS35
36	=plots.D36	=plots.I36	=plots.K36	=plots.J36	=plots.F36	=plots.H36	=plots.G36	=IF($BI$1;IFMR.AK36;plots.BU36)	=(ABS(BL36-BM36)+ABS(BL36-BN36))/2	=(ABS(BO36-BP36)+ABS(BO36-BQ36))/2	=AE36		=10^((AD36/5)+1)*(3.08568025*10^16)	=10^(((AD36-AE36)/5)+1)*(3.08568025*10^16)	=TAN(BD36)*CS36
37															
38															
39										OLD: (ABS(CC5-CD5)+ABS(CC5-CE5))/2					
40															
41															
42															
43															
44															
45															

	CV	CW	CX
1			
2	tangential velocity error (m/year)	tangential velocity (km/s)	tangential velocity error (km/s)
3			
4			
5	=(TAN(BE5)*CT5)-(TAN(BD5)*CS5)	=CU5/1000/365/24/60/60	=CV5/1000/365/24/60/60
6	=(TAN(BE6)*CT6)-(TAN(BD6)*CS6)	=CU6/1000/365/24/60/60	=CV6/1000/365/24/60/60
7	=(TAN(BE7)*CT7)-(TAN(BD7)*CS7)	=CU7/1000/365/24/60/60	=CV7/1000/365/24/60/60
8	=(TAN(BE8)*CT8)-(TAN(BD8)*CS8)	=CU8/1000/365/24/60/60	=CV8/1000/365/24/60/60
9	=(TAN(BE9)*CT9)-(TAN(BD9)*CS9)	=CU9/1000/365/24/60/60	=CV9/1000/365/24/60/60
10	=(TAN(BE10)*CT10)-(TAN(BD10)*CS10)	=CU10/1000/365/24/60/60	=CV10/1000/365/24/60/60
11	=(TAN(BE11)*CT11)-(TAN(BD11)*CS11)	=CU11/1000/365/24/60/60	=CV11/1000/365/24/60/60
12	=(TAN(BE12)*CT12)-(TAN(BD12)*CS12)	=CU12/1000/365/24/60/60	=CV12/1000/365/24/60/60
13	=(TAN(BE13)*CT13)-(TAN(BD13)*CS13)	=CU13/1000/365/24/60/60	=CV13/1000/365/24/60/60
14	=(TAN(BE14)*CT14)-(TAN(BD14)*CS14)	=CU14/1000/365/24/60/60	=CV14/1000/365/24/60/60
15	=(TAN(BE15)*CT15)-(TAN(BD15)*CS15)	=CU15/1000/365/24/60/60	=CV15/1000/365/24/60/60
16	=(TAN(BE16)*CT16)-(TAN(BD16)*CS16)	=CU16/1000/365/24/60/60	=CV16/1000/365/24/60/60
17	=(TAN(BE17)*CT17)-(TAN(BD17)*CS17)	=CU17/1000/365/24/60/60	=CV17/1000/365/24/60/60
18	=(TAN(BE18)*CT18)-(TAN(BD18)*CS18)	=CU18/1000/365/24/60/60	=CV18/1000/365/24/60/60
19	=(TAN(BE19)*CT19)-(TAN(BD19)*CS19)	=CU19/1000/365/24/60/60	=CV19/1000/365/24/60/60
20	=(TAN(BE20)*CT20)-(TAN(BD20)*CS20)	=CU20/1000/365/24/60/60	=CV20/1000/365/24/60/60
21	=(TAN(BE21)*CT21)-(TAN(BD21)*CS21)	=CU21/1000/365/24/60/60	=CV21/1000/365/24/60/60
22	=(TAN(BE22)*CT22)-(TAN(BD22)*CS22)	=CU22/1000/365/24/60/60	=CV22/1000/365/24/60/60
23	=(TAN(BE23)*CT23)-(TAN(BD23)*CS23)	=CU23/1000/365/24/60/60	=CV23/1000/365/24/60/60
24	=(TAN(BE24)*CT24)-(TAN(BD24)*CS24)	=CU24/1000/365/24/60/60	=CV24/1000/365/24/60/60
25	=(TAN(BE25)*CT25)-(TAN(BD25)*CS25)	=CU25/1000/365/24/60/60	=CV25/1000/365/24/60/60
26	=(TAN(BE26)*CT26)-(TAN(BD26)*CS26)	=CU26/1000/365/24/60/60	=CV26/1000/365/24/60/60
27	=(TAN(BE27)*CT27)-(TAN(BD27)*CS27)	=CU27/1000/365/24/60/60	=CV27/1000/365/24/60/60
28	=(TAN(BE28)*CT28)-(TAN(BD28)*CS28)	=CU28/1000/365/24/60/60	=CV28/1000/365/24/60/60
29			
30			
31			
32			
33	=(TAN(BE33)*CT33)-(TAN(BD33)*CS33)	=CU33/1000/365/24/60/60	=CV33/1000/365/24/60/60
34	=(TAN(BE34)*CT34)-(TAN(BD34)*CS34)	=CU34/1000/365/24/60/60	=CV34/1000/365/24/60/60
35	=(TAN(BE35)*CT35)-(TAN(BD35)*CS35)	=CU35/1000/365/24/60/60	=CV35/1000/365/24/60/60
36	=(TAN(BE36)*CT36)-(TAN(BD36)*CS36)	=CU36/1000/365/24/60/60	=CV36/1000/365/24/60/60
37			
38			
39			
40			
41			
42			
43			
44			
45			

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	WDinputParar	compareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRS.pro																
2																		
3																		
4	=WDnorm.A2	=WDnorm.B2	=WDnorm.C2	=WDnorm.D2	=WDnorm.E2	=WDnorm.F2	=WDnorm.G2	=WDnorm.H2	=WDnorm.I2	=WDnorm.J2	=WDnorm.K2	=WDnorm.L2	=WDnorm.M2	=WDnorm.N2	=WDnorm.O2	Special Case	Special Case (for the purpos	
5	=WDnorm.A5	=WDnorm.B5	=WDnorm.C5	=WDnorm.D5	=WDnorm.E5	=p.X5	=p.Y5	=p.Z5	=p.AA5	=p.AB5	=p.AC5	=p.AD5	=p.AE5	=p.AF5	=p.AG5	=fin.C5	=fin.B5	
6	=WDnorm.A6	=WDnorm.B6	=WDnorm.C6	=WDnorm.D6	=WDnorm.E6	=p.X6	=p.Y6	=p.Z6	=p.AA6	=p.AB6	=p.AC6	=p.AD6	=p.AE6	=p.AF6	=p.AG6	=fin.C6	=fin.B6	
7	=WDnorm.A7	=WDnorm.B7	=WDnorm.C7	=WDnorm.D7	=WDnorm.E7	=p.X7	=p.Y7	=p.Z7	=p.AA7	=p.AB7	=p.AC7	=p.AD7	=p.AE7	=p.AF7	=p.AG7	=fin.C7	=fin.B7	
8	=WDnorm.A8	=$D$38	=WDnorm.C8	=WDnorm.D8	=WDnorm.E8	=p.X8	=p.Y8	=p.Z8	=p.AA8	=p.AB8	=p.AC8	=p.AD8	=p.AE8	=p.AF8	=p.AG8	=fin.C8	=fin.B8	
9	=WDcool2.A9	=WDcool2.U9	=WDcool2.V9	=WDcool2.D9	=WDcool2.E9	=WDcool2.F9	=WDcool2.G9	=WDcool2.H9	=WDcool2.I9	=WDcool2.J9	=WDcool2.K9	=WDcool2.L9	=WDcool2.M9	=WDcool2.N9	=WDcool2.O9	=fin.C9	=fin.B9	
10	=WDcool2.A1(	=WDcool2.U10	=WDcool2.V10	=WDcool2.D10	=WDcool2.E10	=WDcool2.F10	=WDcool2.G10	=WDcool2.H10	###	=WDcool2.J10	=WDcool2.K10	=WDcool2.L10	=WDcool2.M10	=WDcool2.N10	=WDcool2.O10	=fin.C10	=fin.B10	
11	=WDnorm.A11	=WDnorm.B11	=WDnorm.C11	=WDnorm.D11	=WDnorm.E11	=p.X11	=p.Y11	=p.Z11	=p.AA11	=p.AB11	=p.AC11	=p.AD11	=p.AE11	=p.AF11	=p.AG11	=fin.C11	=fin.B11	
12	=WDnorm.A12	=WDnorm.B12	=WDnorm.C12	=WDnorm.D12	=WDnorm.E12	=p.X12	=p.Y12	=p.Z12	=p.AA12	=p.AB12	=p.AC12	=p.AD12	=p.AE12	=p.AF12	=p.AG12	=fin.C12	=fin.B12	
13	=WDnorm.A13	=WDnorm.B13	=WDnorm.C13	=WDnorm.D13	=WDnorm.E13	=p.X13	=p.Y13	=p.Z13	=p.AA13	=p.AB13	=p.AC13	=p.AD13	=p.AE13	=p.AF13	=p.AG13	=fin.C13	=fin.B13	
14	=WDnorm.A14	=WDnorm.B14	=WDnorm.C14	=WDnorm.D14	=WDnorm.E14	=p.X14	=p.Y14	=p.Z14	=p.AA14	=p.AB14	=p.AC14	=p.AD14	=p.AE14	=p.AF14	=p.AG14	=fin.C14	=fin.B14	
15	=WDnorm.A15	=WDnorm.B15	=WDnorm.C15	=WDnorm.D15	=WDnorm.E15	=p.X15	=p.Y15	=p.Z15	=p.AA15	=p.AB15	=p.AC15	=p.AD15	=p.AE15	=p.AF15	=p.AG15	=fin.C15	=fin.B15	
16	=WDnorm.A16	=WDnorm.B16	=WDnorm.C16	=WDnorm.D16	=WDnorm.E16	=p.X16	=p.Y16	=p.Z16	=p.AA16	=p.AB16	=p.AC16	=p.AD16	=p.AE16	=p.AF16	=p.AG16	=fin.C16	=fin.B16	
17	=WDnorm.A17	=WDnorm.B17	=WDnorm.C17	=WDnorm.D17	=WDnorm.E17	=p.X17	=p.Y17	=p.Z17	=p.AA17	=p.AB17	=p.AC17	=p.AD17	=p.AE17	=p.AF17	=p.AG17	=fin.C17	=fin.B17	
18	=WDnorm.A18	=WDnorm.B18	=WDnorm.C18	=WDnorm.D18	=WDnorm.E18	=p.X18	=p.Y18	=p.Z18	=p.AA18	=p.AB18	=p.AC18	=p.AD18	=p.AE18	=p.AF18	=p.AG18	=fin.C18	=fin.B18	
19	=WDnorm.A19	=$D$38	=WDnorm.C19	=WDnorm.D19	=WDnorm.E19	=p.X19	=p.Y19	=p.Z19	=p.AA19	=p.AB19	=p.AC19	=p.AD19	=p.AE19	=p.AF19	=p.AG19	=fin.C19	=fin.B19	
20	=WDcool2.A2(	=WDcool2.U20	=WDcool2.V20	=WDcool2.D20	=WDcool2.E20	=WDcool2.F20	=WDcool2.G20	=WDcool2.H20	###	=WDcool2.J20	=WDcool2.K20	=WDcool2.L20	=WDcool2.M20	=WDcool2.N20	=WDcool2.O20	=fin.C20	=fin.B20	
21	=WDcool2.A2:	=WDcool2.U21	=WDcool2.V21	=WDcool2.D21	=WDcool2.E21	=WDcool2.F21	=WDcool2.G21	=WDcool2.H21	###	=WDcool2.J21	=WDcool2.K21	=WDcool2.L21	=WDcool2.M21	=WDcool2.N21	=WDcool2.O21	=fin.C21	=fin.B21	
22	=WDnorm.A22	=WDnorm.B22	=WDnorm.C22	=WDnorm.D22	=WDnorm.E22	=p.X22	=p.Y22	=p.Z22	=p.AA22	=p.AB22	=p.AC22	=p.AD22	=p.AE22	=p.AF22	=p.AG22	=fin.C22	=fin.B22	
23	=WDnorm.A23	=WDnorm.B23	=WDnorm.C23	=WDnorm.D23	=WDnorm.E23	=p.X23	=p.Y23	=p.Z23	=p.AA23	=p.AB23	=p.AC23	=p.AD23	=p.AE23	=p.AF23	=p.AG23	=fin.C23	=fin.B23	
24	=WDnorm.A24	=WDnorm.B24	=WDnorm.C24	=WDnorm.D24	=WDnorm.E24	=p.X24	=p.Y24	=p.Z24	=p.AA24	=p.AB24	=p.AC24	=p.AD24	=p.AE24	=p.AF24	=p.AG24	=fin.C24	=fin.B24	
25	=WDnorm.A25	=WDnorm.B25	=WDnorm.C25	=WDnorm.D25	=WDnorm.E25	=p.X25	=p.Y25	=p.Z25	=p.AA25	=p.AB25	=p.AC25	=p.AD25	=p.AE25	=p.AF25	=p.AG25	=fin.C25	=fin.B25	
26	=WDnorm.A26	=WDnorm.B26	=WDnorm.C26	=WDnorm.D26	=WDnorm.E26	=p.X26	=p.Y26	=p.Z26	=p.AA26	=p.AB26	=p.AC26	=p.AD26	=p.AE26	=p.AF26	=p.AG26	=fin.C26	=fin.B26	
27	=WDnorm.A27	=WDnorm.B27	=WDnorm.C27	=WDnorm.D27	=WDnorm.E27	=p.X27	=p.Y27	=p.Z27	=p.AA27	=p.AB27	=p.AC27	=p.AD27	=p.AE27	=p.AF27	=p.AG27	=fin.C27	=fin.B27	
28	=WDnorm.A28	=WDnorm.B28	=WDnorm.C28	=WDnorm.D28	=WDnorm.E28	=p.X28	=p.Y28	=p.Z28	=p.AA28	=p.AB28	=p.AC28	=p.AD28	=p.AE28	=p.AF28	=p.AG28	=fin.C28	=fin.B28	
29																		
30																		
31																		
32																		
33	=WDnorm.A33	=WDnorm.B33	=WDnorm.C33	=WDnorm.D33	=WDnorm.E33	=p.X33	=p.Y33	=p.Z33	=p.AA33	=p.AB33	=p.AC33	=p.AD33	=p.AE33	=p.AF33	=p.AG33	=fin.C33	=fin.B33	
34	=WDnorm.A34	=WDnorm.B34	=WDnorm.C34	=WDnorm.D34	=WDnorm.E34	=p.X34	=p.Y34	=p.Z34	=p.AA34	=p.AB34	=p.AC34	=p.AD34	=p.AE34	=p.AF34	=p.AG34	=fin.C34	=fin.B34	
35	=WDnorm.A35	=WDnorm.B35	=WDnorm.C35	=WDnorm.D35	=WDnorm.E35	=p.X35	=p.Y35	=p.Z35	=p.AA35	=p.AB35	=p.AC35	=p.AD35	=p.AE35	=p.AF35	=p.AG35	=fin.C35	=fin.B35	
36	=WDnorm.A36	=WDnorm.B36	=WDnorm.C36	=WDnorm.D36	=WDnorm.E36	=p.X36	=p.Y36	=p.Z36	=p.AA36	=p.AB36	=p.AC36	=p.AD36	=p.AE36	=p.AF36	=p.AG36	=fin.C36	=fin.B36	
37																		
38		starting point value only		8.5														

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	**observed versus modelled photometry comparison**																									
2	Normalised						Normalised						Normalised							Normalised						
3		psfMag_u	psfMag_g	psfMag_r	psfMag_i	psfmag_z		abs mag (u)	abs mag (g)	abs mag (r)	abs mag (i)	abs mag (z)		obs/mod mag	obs/mod mag	obs/mod mag	obs/mod mag	obs/mod mag	min obs/mod mag	diff	obs/mod mag	obs/mod mag	obs/mod mag	obs/mod mag	obs/mod mag	Special Case
4	wavelength (A)	3551	4686	6165	7481	8931	wavelength (A	=B4	=C4	=D4	=E4	=F4	name	=B4	=C4	=D4	=E4	=F4		name	=B4	=C4	=D4	=E4	=F4	
5	=pOrig.A5&"obs"	=fin.AL5	=fin.AM5	=fin.AN5	=fin.AO5	=fin.AP5	=pOrig.A5 & "r	=fin.T5	=fin.U5	=fin.V5	=fin.W5	=fin.X5	=pOrig.A5	=H5-B5	=I5-C5	=J5-D5	=K5-E5	=L5-F5	=MIN(N5:R5)	=pOrig.A5 & "-	=N5-$S5	=O5-$S5	=P5-$S5	=Q5-$S5	=R5-$S5	=fin.B5	
6	=pOrig.A6&"obs"	=fin.AL6	=fin.AM6	=fin.AN6	=fin.AO6	=fin.AP6	=pOrig.A6 & "r	=fin.T6	=fin.U6	=fin.V6	=fin.W6	=fin.X6	=pOrig.A6	=H6-B6	=I6-C6	=J6-D6	=K6-E6	=L6-F6	=MIN(N6:R6)	=pOrig.A6 & "-	=N6-$S6	=O6-$S6	=P6-$S6	=Q6-$S6	=R6-$S6	=fin.B6	
7	=pOrig.A7&"obs"	=fin.AL7	=fin.AM7	=fin.AN7	=fin.AO7	=fin.AP7	=pOrig.A7 & "r	=fin.T7	=fin.U7	=fin.V7	=fin.W7	=fin.X7	=pOrig.A7	=H7-B7	=I7-C7	=J7-D7	=K7-E7	=L7-F7	=MIN(N7:R7)	=pOrig.A7 & "-	=N7-$S7	=O7-$S7	=P7-$S7	=Q7-$S7	=R7-$S7	=fin.B7	
8	=pOrig.A8&"obs"	=fin.AL8	=fin.AM8	=fin.AN8	=fin.AO8	=fin.AP8	=pOrig.A8 & "r	=fin.T8	=fin.U8	=fin.V8	=fin.W8	=fin.X8	=pOrig.A8	=H8-B8	=I8-C8	=J8-D8	=K8-E8	=L8-F8	=MIN(N8:R8)	=pOrig.A8 & "-	=N8-$S8	=O8-$S8	=P8-$S8	=Q8-$S8	=R8-$S8	=fin.B8	
9	=pOrig.A9&"obs"	=fin.AL9	=fin.AM9	=fin.AN9	=fin.AO9	=fin.AP9	=pOrig.A9 & "r	=fin.T9	=fin.U9	=fin.V9	=fin.W9	=fin.X9	=pOrig.A9	=H9-B9	=I9-C9	=J9-D9	=K9-E9	=L9-F9	=MIN(N9:R9)	=pOrig.A9 & "-	=N9-$S9	=O9-$S9	=P9-$S9	=Q9-$S9	=R9-$S9	=fin.B9	
10	=pOrig.A10&"obs"	=fin.AL10	=fin.AM10	=fin.AN10	=fin.AO10	=fin.AP10	=pOrig.A10 & "	=fin.T10	=fin.U10	=fin.V10	=fin.W10	=fin.X10	=pOrig.A10	=H10-B10	=I10-C10	=J10-D10	=K10-E10	=L10-F10	=MIN(N10:R10)	=pOrig.A10 & "	=N10-$S10	=O10-$S10	=P10-$S10	=Q10-$S10	=R10-$S10	=fin.B10	
11	=pOrig.A11&"obs"	=fin.AL11	=fin.AM11	=fin.AN11	=fin.AO11	=fin.AP11	=pOrig.A11 & "	=fin.T11	=fin.U11	=fin.V11	=fin.W11	=fin.X11	=pOrig.A11	=H11-B11	=I11-C11	=J11-D11	=K11-E11	=L11-F11	=MIN(N11:R11)	=pOrig.A11 & "	=N11-$S11	=O11-$S11	=P11-$S11	=Q11-$S11	=R11-$S11	=fin.B11	
12	=pOrig.A12&"obs"	=fin.AL12	=fin.AM12	=fin.AN12	=fin.AO12	=fin.AP12	=pOrig.A12 & "	=fin.T12	=fin.U12	=fin.V12	=fin.W12	=fin.X12	=pOrig.A12	=H12-B12	=I12-C12	=J12-D12	=K12-E12	=L12-F12	=MIN(N12:R12)	=pOrig.A12 & "	=N12-$S12	=O12-$S12	=P12-$S12	=Q12-$S12	=R12-$S12	=fin.B12	
13	=pOrig.A13&"obs"	=fin.AL13	=fin.AM13	=fin.AN13	=fin.AO13	=fin.AP13	=pOrig.A13 & "	=fin.T13	=fin.U13	=fin.V13	=fin.W13	=fin.X13	=pOrig.A13	=H13-B13	=I13-C13	=J13-D13	=K13-E13	=L13-F13	=MIN(N13:R13)	=pOrig.A13 & "	=N13-$S13	=O13-$S13	=P13-$S13	=Q13-$S13	=R13-$S13	=fin.B13	
14	=pOrig.A14&"obs"	=fin.AL14	=fin.AM14	=fin.AN14	=fin.AO14	=fin.AP14	=pOrig.A14 & "	=fin.T14	=fin.U14	=fin.V14	=fin.W14	=fin.X14	=pOrig.A14	=H14-B14	=I14-C14	=J14-D14	=K14-E14	=L14-F14	=MIN(N14:R14)	=pOrig.A14 & "	=N14-$S14	=O14-$S14	=P14-$S14	=Q14-$S14	=R14-$S14	=fin.B14	
15	=pOrig.A15&"obs"	=fin.AL15	=fin.AM15	=fin.AN15	=fin.AO15	=fin.AP15	=pOrig.A15 & "	=fin.T15	=fin.U15	=fin.V15	=fin.W15	=fin.X15	=pOrig.A15	=H15-B15	=I15-C15	=J15-D15	=K15-E15	=L15-F15	=MIN(N15:R15)	=pOrig.A15 & "	=N15-$S15	=O15-$S15	=P15-$S15	=Q15-$S15	=R15-$S15	=fin.B15	
16	=pOrig.A16&"obs"	=fin.AL16	=fin.AM16	=fin.AN16	=fin.AO16	=fin.AP16	=pOrig.A16 & "	=fin.T16	=fin.U16	=fin.V16	=fin.W16	=fin.X16	=pOrig.A16	=H16-B16	=I16-C16	=J16-D16	=K16-E16	=L16-F16	=MIN(N16:R16)	=pOrig.A16 & "	=N16-$S16	=O16-$S16	=P16-$S16	=Q16-$S16	=R16-$S16	=fin.B16	
17	=pOrig.A17&"obs"	=fin.AL17	=fin.AM17	=fin.AN17	=fin.AO17	=fin.AP17	=pOrig.A17 & "	=fin.T17	=fin.U17	=fin.V17	=fin.W17	=fin.X17	=pOrig.A17	=H17-B17	=I17-C17	=J17-D17	=K17-E17	=L17-F17	=MIN(N17:R17)	=pOrig.A17 & "	=N17-$S17	=O17-$S17	=P17-$S17	=Q17-$S17	=R17-$S17	=fin.B17	
18	=pOrig.A18&"obs"	=fin.AL18	=fin.AM18	=fin.AN18	=fin.AO18	=fin.AP18	=pOrig.A18 & "	=fin.T18	=fin.U18	=fin.V18	=fin.W18	=fin.X18	=pOrig.A18	=H18-B18	=I18-C18	=J18-D18	=K18-E18	=L18-F18	=MIN(N18:R18)	=pOrig.A18 & "	=N18-$S18	=O18-$S18	=P18-$S18	=Q18-$S18	=R18-$S18	=fin.B18	
19	=pOrig.A19&"obs"	=fin.AL19	=fin.AM19	=fin.AN19	=fin.AO19	=fin.AP19	=pOrig.A19 & "	=fin.T19	=fin.U19	=fin.V19	=fin.W19	=fin.X19	=pOrig.A19	=H19-B19	=I19-C19	=J19-D19	=K19-E19	=L19-F19	=MIN(N19:R19)	=pOrig.A19 & "	=N19-$S19	=O19-$S19	=P19-$S19	=Q19-$S19	=R19-$S19	=fin.B19	
20	=pOrig.A20&"obs"	=fin.AL20	=fin.AM20	=fin.AN20	=fin.AO20	=fin.AP20	=pOrig.A20 & "	=fin.T20	=fin.U20	=fin.V20	=fin.W20	=fin.X20	=pOrig.A20	=H20-B20	=I20-C20	=J20-D20	=K20-E20	=L20-F20	=MIN(N20:R20)	=pOrig.A20 & "	=N20-$S20	=O20-$S20	=P20-$S20	=Q20-$S20	=R20-$S20	=fin.B20	
21	=pOrig.A21&"obs"	=fin.AL21	=fin.AM21	=fin.AN21	=fin.AO21	=fin.AP21	=pOrig.A21 & "	=fin.T21	=fin.U21	=fin.V21	=fin.W21	=fin.X21	=pOrig.A21	=H21-B21	=I21-C21	=J21-D21	=K21-E21	=L21-F21	=MIN(N21:R21)	=pOrig.A21 & "	=N21-$S21	=O21-$S21	=P21-$S21	=Q21-$S21	=R21-$S21	=fin.B21	
22	=pOrig.A22&"obs"	=fin.AL22	=fin.AM22	=fin.AN22	=fin.AO22	=fin.AP22	=pOrig.A22 & "	=fin.T22	=fin.U22	=fin.V22	=fin.W22	=fin.X22	=pOrig.A22	=H22-B22	=I22-C22	=J22-D22	=K22-E22	=L22-F22	=MIN(N22:R22)	=pOrig.A22 & "	=N22-$S22	=O22-$S22	=P22-$S22	=Q22-$S22	=R22-$S22	=fin.B22	
23	=pOrig.A23&"obs"	=fin.AL23	=fin.AM23	=fin.AN23	=fin.AO23	=fin.AP23	=pOrig.A23 & "	=fin.T23	=fin.U23	=fin.V23	=fin.W23	=fin.X23	=pOrig.A23	=H23-B23	=I23-C23	=J23-D23	=K23-E23	=L23-F23	=MIN(N23:R23)	=pOrig.A23 & "	=N23-$S23	=O23-$S23	=P23-$S23	=Q23-$S23	=R23-$S23	=fin.B23	
24	=pOrig.A24&"obs"	=fin.AL24	=fin.AM24	=fin.AN24	=fin.AO24	=fin.AP24	=pOrig.A24 & "	=fin.T24	=fin.U24	=fin.V24	=fin.W24	=fin.X24	=pOrig.A24	=H24-B24	=I24-C24	=J24-D24	=K24-E24	=L24-F24	=MIN(N24:R24)	=pOrig.A24 & "	=N24-$S24	=O24-$S24	=P24-$S24	=Q24-$S24	=R24-$S24	=fin.B24	
25	=pOrig.A25&"obs"	=fin.AL25	=fin.AM25	=fin.AN25	=fin.AO25	=fin.AP25	=pOrig.A25 & "	=fin.T25	=fin.U25	=fin.V25	=fin.W25	=fin.X25	=pOrig.A25	=H25-B25	=I25-C25	=J25-D25	=K25-E25	=L25-F25	=MIN(N25:R25)	=pOrig.A25 & "	=N25-$S25	=O25-$S25	=P25-$S25	=Q25-$S25	=R25-$S25	=fin.B25	
26	=pOrig.A26&"obs"	=fin.AL26	=fin.AM26	=fin.AN26	=fin.AO26	=fin.AP26	=pOrig.A26 & "	=fin.T26	=fin.U26	=fin.V26	=fin.W26	=fin.X26	=pOrig.A26	=H26-B26	=I26-C26	=J26-D26	=K26-E26	=L26-F26	=MIN(N26:R26)	=pOrig.A26 & "	=N26-$S26	=O26-$S26	=P26-$S26	=Q26-$S26	=R26-$S26	=fin.B26	
27	=pOrig.A27&"obs"	=fin.AL27	=fin.AM27	=fin.AN27	=fin.AO27	=fin.AP27	=pOrig.A27 & "	=fin.T27	=fin.U27	=fin.V27	=fin.W27	=fin.X27	=pOrig.A27	=H27-B27	=I27-C27	=J27-D27	=K27-E27	=L27-F27	=MIN(N27:R27)	=pOrig.A27 & "	=N27-$S27	=O27-$S27	=P27-$S27	=Q27-$S27	=R27-$S27	=fin.B27	
28	=pOrig.A28&"obs"	=fin.AL28	=fin.AM28	=fin.AN28	=fin.AO28	=fin.AP28	=pOrig.A28 & "	=fin.T28	=fin.U28	=fin.V28	=fin.W28	=fin.X28	=pOrig.A28	=H28-B28	=I28-C28	=J28-D28	=K28-E28	=L28-F28	=MIN(N28:R28)	=pOrig.A28 & "	=N28-$S28	=O28-$S28	=P28-$S28	=Q28-$S28	=R28-$S28	=fin.B28	
29	=pOrig.A5 & "mod"	=fin.T5	=fin.U5	=fin.V5	=fin.W5	=fin.X5																					
30	=pOrig.A6 & "mod"	=fin.T6	=fin.U6	=fin.V6	=fin.W6	=fin.X6																					
31	=pOrig.A7 & "mod"	=fin.T7	=fin.U7	=fin.V7	=fin.W7	=fin.X7																					
32	=pOrig.A8 & "mod"	=fin.T8	=fin.U8	=fin.V8	=fin.W8	=fin.X8																					
33	=pOrig.A9 & "mod"	=fin.T9	=fin.U9	=fin.V9	=fin.W9	=fin.X9																					
34	=pOrig.A10 & "mod"	=fin.T10	=fin.U10	=fin.V10	=fin.W10	=fin.X10																					
35	=pOrig.A11 & "mod"	=fin.T11	=fin.U11	=fin.V11	=fin.W11	=fin.X11																					
36	=pOrig.A12 & "mod"	=fin.T12	=fin.U12	=fin.V12	=fin.W12	=fin.X12																					
37	=pOrig.A13 & "mod"	=fin.T13	=fin.U13	=fin.V13	=fin.W13	=fin.X13																					
38	=pOrig.A14 & "mod"	=fin.T14	=fin.U14	=fin.V14	=fin.W14	=fin.X14																					
39	=pOrig.A15 & "mod"	=fin.T15	=fin.U15	=fin.V15	=fin.W15	=fin.X15																					
40	=pOrig.A16 & "mod"	=fin.T16	=fin.U16	=fin.V16	=fin.W16	=fin.X16																					
41	=pOrig.A17 & "mod"	=fin.T17	=fin.U17	=fin.V17	=fin.W17	=fin.X17																					
42	=pOrig.A18 & "mod"	=fin.T18	=fin.U18	=fin.V18	=fin.W18	=fin.X18																					
43	=pOrig.A19 & "mod"	=fin.T19	=fin.U19	=fin.V19	=fin.W19	=fin.X19																					
44	=pOrig.A20 & "mod"	=fin.T20	=fin.U20	=fin.V20	=fin.W20	=fin.X20																					
45	=pOrig.A21 & "mod"	=fin.T21	=fin.U21	=fin.V21	=fin.W21	=fin.X21																					
46	=pOrig.A22 & "mod"	=fin.T22	=fin.U22	=fin.V22	=fin.W22	=fin.X22																					
47	=pOrig.A23 & "mod"	=fin.T23	=fin.U23	=fin.V23	=fin.W23	=fin.X23																					
48	=pOrig.A24 & "mod"	=fin.T24	=fin.U24	=fin.V24	=fin.W24	=fin.X24																					
49	=pOrig.A25 & "mod"	=fin.T25	=fin.U25	=fin.V25	=fin.W25	=fin.X25																					
50	=pOrig.A26 & "mod"	=fin.T26	=fin.U26	=fin.V26	=fin.W26	=fin.X26																					
51	=pOrig.A27 & "mod"	=fin.T27	=fin.U27	=fin.V27	=fin.W27	=fin.X27																					
52	=pOrig.A28 & "mod"	=fin.T28	=fin.U28	=fin.V28	=fin.W28	=fin.X28																					

	AA	AB	AC	AD	AE	AF
1						
2					Normalised	
3	Separation (aris	data A bright	edge of chip?	bad result exp	ected?	obs/mod mag diff (averag
4					name	obs/mod mag diff (averag
5	=fin.AW5	=fin.BC5	0	=IF(((AA5<3)	=pOrig.A5	=AVERAGE(U5:Y5)
6	=fin.AW6	=fin.BC6	0	=IF(((AA6<3)	=pOrig.A6	=AVERAGE(U6:Y6)
7	=fin.AW7	=fin.BC7	1	=IF(((AA7<3)	=pOrig.A7	=AVERAGE(U7:Y7)
8	=fin.AW8	=fin.BC8	1	=IF(((AA8<3)	=pOrig.A8	=AVERAGE(U8:Y8)
9	=fin.AW9	=fin.BC9	0	=IF(((AA9<3)	=pOrig.A9	=AVERAGE(U9:Y9)
10	=fin.AW10	=fin.BC10	0	=IF(((AA10<3)	=pOrig.A10	=AVERAGE(U10:Y10)
11	=fin.AW11	=fin.BC11	1	=IF(((AA11<3)	=pOrig.A11	=AVERAGE(U11:Y11)
12	=fin.AW12	=fin.BC12	1	=IF(((AA12<3)	=pOrig.A12	=AVERAGE(U12:Y12)
13	=fin.AW13	=fin.BC13	0	=IF(((AA13<3)	=pOrig.A13	=AVERAGE(U13:Y13)
14	=fin.AW14	=fin.BC14	0	=IF(((AA14<3)	=pOrig.A14	=AVERAGE(U14:Y14)
15	=fin.AW15	=fin.BC15	1	=IF(((AA15<3)	=pOrig.A15	=AVERAGE(U15:Y15)
16	=fin.AW16	=fin.BC16	1	=IF(((AA16<3)	=pOrig.A16	=AVERAGE(U16:Y16)
17	=fin.AW17	=fin.BC17	1	=IF(((AA17<3)	=pOrig.A17	=AVERAGE(U17:Y17)
18	=fin.AW18	=fin.BC18	1	=IF(((AA18<3)	=pOrig.A18	=AVERAGE(U18:Y18)
19	=fin.AW19	=fin.BC19	1	=IF(((AA19<3)	=pOrig.A19	=AVERAGE(U19:Y19)
20	=fin.AW20	=fin.BC20	1	=IF(((AA20<3)	=pOrig.A20	=AVERAGE(U20:Y20)
21	=fin.AW21	=fin.BC21	1	=IF(((AA21<3)	=pOrig.A21	=AVERAGE(U21:Y21)
22	=fin.AW22	=fin.BC22	1	=IF(((AA22<3)	=pOrig.A22	=AVERAGE(U22:Y22)
23	=fin.AW23	=fin.BC23	0	=IF(((AA23<3)	=pOrig.A23	=AVERAGE(U23:Y23)
24	=fin.AW24	=fin.BC24	0	=IF(((AA24<3)	=pOrig.A24	=AVERAGE(U24:Y24)
25	=fin.AW25	=fin.BC25	0	=IF(((AA25<3)	=pOrig.A25	=AVERAGE(U25:Y25)
26	=fin.AW26	=fin.BC26	0	=IF(((AA26<3)	=pOrig.A26	=AVERAGE(U26:Y26)
27	=fin.AW27	=fin.BC27	0	=IF(((AA27<3)	=pOrig.A27	=AVERAGE(U27:Y27)
28	=fin.AW28	=fin.BC28	0	=IF(((AA28<3)	=pOrig.A28	=AVERAGE(U28:Y28)
29						
30						
31						
32						
33						
34						
35						
36						
37						
38						
39						
40						
41						
42						
43						
44						
45						
46						
47						
48						
49						
50						
51						
52						

dFin

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	**Distance**	DDphotometricBands.txt (after switch)			.run plotDistanceModulus.pro																		
2	Paper name	DD_NUMBER	psfMag_g	psfMag_r	psfMag_u	abs mag (g [v])	abs mag (r)	abs mag (u)	Temperature	psfMagErr_g	psfMagErr_r	psfMagErr_u	delta M (g)	delta M (r)	delta M (u)	g (m – M) error	r (m – M) error	u (m – M) error	Special Case	distance modu	distance modulus g difference		
3																							
4																							
5	=SDSS.B5	=SDSS.N6	=p.Y5	=p.Z5	=p.X5	=fin.U5	=fin.V5	=fin.T5	=fin.H5	=p.AD5	=p.AE5	=p.AC5	=fin.Z5	=fin.AA5	=fin.Y5	=SQRT(J5^2 + M5^2)	=SQRT(K5^2 + N5^2)	=SQRT(L5^2 + O5^2)	=fin.B5	=C5-F5	=T5-T6		
6	=SDSS.B6	=SDSS.N5	=p.Y6	=p.Z6	=p.X6	=fin.U6	=fin.V6	=fin.T6	=fin.H6	=p.AD6	=p.AE6	=p.AC6	=fin.Z6	=fin.AA6	=fin.Y6	=SQRT(J6^2 + M6^2)	=SQRT(K6^2 + N6^2)	=SQRT(L6^2 + O6^2)	=fin.B6	=C6-F6			
7	=SDSS.B7	=SDSS.N7	=p.Y7	=p.Z7	=p.X7	=fin.U7	=fin.V7	=fin.T7	=fin.H7	=p.AD7	=p.AE7	=p.AC7	=fin.Z7	=fin.AA7	=fin.Y7	=SQRT(J7^2 + M7^2)	=SQRT(K7^2 + N7^2)	=SQRT(L7^2 + O7^2)	=fin.B7	=C7-F7	=T7-T8		
8	=SDSS.B8	=SDSS.N8	=p.Y8	=p.Z8	=p.X8	=fin.U8	=fin.V8	=fin.T8	=fin.H8	=p.AD8	=p.AE8	=p.AC8	=fin.Z8	=fin.AA8	=fin.Y8	=SQRT(J8^2 + M8^2)	=SQRT(K8^2 + N8^2)	=SQRT(L8^2 + O8^2)	=fin.B8	NaN			
9	=SDSS.B9	=SDSS.N10	=p.Y9	=p.Z9	=p.X9	=fin.U9	=fin.V9	=fin.T9	=fin.H9	=p.AD9	=p.AE9	=p.AC9	=fin.Z9	=fin.AA9	=fin.Y9	=SQRT(J9^2 + M9^2)	=SQRT(K9^2 + N9^2)	=SQRT(L9^2 + O9^2)	=fin.B9	=C9-F9	=T9-T10		
10	=SDSS.B10	=SDSS.N9	=p.Y10	=p.Z10	=p.X10	=fin.U10	=fin.V10	=fin.T10	=fin.H10	=p.AD10	=p.AE10	=p.AC10	=fin.Z10	=fin.AA10	=fin.Y10	=SQRT(J10^2 + M10^2)	=SQRT(K10^2 + N10^2)	=SQRT(L10^2 + O10^2)	=fin.B10	=C10-F10			
11	=SDSS.B11	=SDSS.N12	=p.Y11	=p.Z11	=p.X11	=fin.U11	=fin.V11	=fin.T11	=fin.H11	=p.AD11	=p.AE11	=p.AC11	=fin.Z11	=fin.AA11	=fin.Y11	=SQRT(J11^2 + M11^2)	=SQRT(K11^2 + N11^2)	=SQRT(L11^2 + O11^2)	=fin.B11	=C11-F11	=T11-T12		
12	=SDSS.B12	=SDSS.N11	=p.Y12	=p.Z12	=p.X12	=fin.U12	=fin.V12	=fin.T12	=fin.H12	=p.AD12	=p.AE12	=p.AC12	=fin.Z12	=fin.AA12	=fin.Y12	=SQRT(J12^2 + M12^2)	=SQRT(K12^2 + N12^2)	=SQRT(L12^2 + O12^2)	=fin.B12	=C12-F12			
13	=SDSS.B13	=SDSS.N13	=p.Y13	=p.Z13	=p.X13	=fin.U13	=fin.V13	=fin.T13	=fin.H13	=p.AD13	=p.AE13	=p.AC13	=fin.Z13	=fin.AA13	=fin.Y13	=SQRT(J13^2 + M13^2)	=SQRT(K13^2 + N13^2)	=SQRT(L13^2 + O13^2)	=fin.B13	=C13-F13	=T13-T14		
14	=SDSS.B14	=SDSS.N14	=p.Y14	=p.Z14	=p.X14	=fin.U14	=fin.V14	=fin.T14	=fin.H14	=p.AD14	=p.AE14	=p.AC14	=fin.Z14	=fin.AA14	=fin.Y14	=SQRT(J14^2 + M14^2)	=SQRT(K14^2 + N14^2)	=SQRT(L14^2 + O14^2)	=fin.B14	=C14-F14			
15	=SDSS.B15	=SDSS.N15	=p.Y15	=p.Z15	=p.X15	=fin.U15	=fin.V15	=fin.T15	=fin.H15	=p.AD15	=p.AE15	=p.AC15	=fin.Z15	=fin.AA15	=fin.Y15	=SQRT(J15^2 + M15^2)	=SQRT(K15^2 + N15^2)	=SQRT(L15^2 + O15^2)	=fin.B15	=C15-F15	=T15-T16		
16	=SDSS.B16	=SDSS.N16	=p.Y16	=p.Z16	=p.X16	=fin.U16	=fin.V16	=fin.T16	=fin.H16	=p.AD16	=p.AE16	=p.AC16	=fin.Z16	=fin.AA16	=fin.Y16	=SQRT(J16^2 + M16^2)	=SQRT(K16^2 + N16^2)	=SQRT(L16^2 + O16^2)	=fin.B16	=C16-F16			
17	=SDSS.B17	=SDSS.N17	=p.Y17	=p.Z17	=p.X17	=fin.U17	=fin.V17	=fin.T17	=fin.H17	=p.AD17	=p.AE17	=p.AC17	=fin.Z17	=fin.AA17	=fin.Y17	=SQRT(J17^2 + M17^2)	=SQRT(K17^2 + N17^2)	=SQRT(L17^2 + O17^2)	=fin.B17	=C17-F17	=T17-T18		
18	=SDSS.B18	=SDSS.N18	=p.Y18	=p.Z18	=p.X18	=fin.U18	=fin.V18	=fin.T18	=fin.H18	=p.AD18	=p.AE18	=p.AC18	=fin.Z18	=fin.AA18	=fin.Y18	=SQRT(J18^2 + M18^2)	=SQRT(K18^2 + N18^2)	=SQRT(L18^2 + O18^2)	=fin.B18	=C18-F18			
19	=SDSS.B19	=SDSS.N19	=p.Y19	=p.Z19	=p.X19	=fin.U19	=fin.V19	=fin.T19	=fin.H19	=p.AD19	=p.AE19	=p.AC19	=fin.Z19	=fin.AA19	=fin.Y19	=SQRT(J19^2 + M19^2)	=SQRT(K19^2 + N19^2)	=SQRT(L19^2 + O19^2)	=fin.B19	NaN	=T19-T20		
20	=SDSS.B20	=SDSS.N20	=p.Y20	=p.Z20	=p.X20	=fin.U20	=fin.V20	=fin.T20	=fin.H20	=p.AD20	=p.AE20	=p.AC20	=fin.Z20	=fin.AA20	=fin.Y20	=SQRT(J20^2 + M20^2)	=SQRT(K20^2 + N20^2)	=SQRT(L20^2 + O20^2)	=fin.B20	=C20-F20			
21	=SDSS.B21	=SDSS.N21	=p.Y21	=p.Z21	=p.X21	=fin.U21	=fin.V21	=fin.T21	=fin.H21	=p.AD21	=p.AE21	=p.AC21	=fin.Z21	=fin.AA21	=fin.Y21	=SQRT(J21^2 + M21^2)	=SQRT(K21^2 + N21^2)	=SQRT(L21^2 + O21^2)	=fin.B21	=C21-F21	=T21-T22		
22	=SDSS.B22	=SDSS.N22	=p.Y22	=p.Z22	=p.X22	=fin.U22	=fin.V22	=fin.T22	=fin.H22	=p.AD22	=p.AE22	=p.AC22	=fin.Z22	=fin.AA22	=fin.Y22	=SQRT(J22^2 + M22^2)	=SQRT(K22^2 + N22^2)	=SQRT(L22^2 + O22^2)	=fin.B22	=C22-F22			
23	=SDSS.B23	=SDSS.N23	=p.Y23	=p.Z23	=p.X23	=fin.U23	=fin.V23	=fin.T23	=fin.H23	=p.AD23	=p.AE23	=p.AC23	=fin.Z23	=fin.AA23	=fin.Y23	=SQRT(J23^2 + M23^2)	=SQRT(K23^2 + N23^2)	=SQRT(L23^2 + O23^2)	=fin.B23	=C23-F23	=T23-T24		
24	=SDSS.B24	=SDSS.N24	=p.Y24	=p.Z24	=p.X24	=fin.U24	=fin.V24	=fin.T24	=fin.H24	=p.AD24	=p.AE24	=p.AC24	=fin.Z24	=fin.AA24	=fin.Y24	=SQRT(J24^2 + M24^2)	=SQRT(K24^2 + N24^2)	=SQRT(L24^2 + O24^2)	=fin.B24	=C24-F24			
25	=SDSS.B25	=SDSS.N25	=p.Y25	=p.Z25	=p.X25	=fin.U25	=fin.V25	=fin.T25	=fin.H25	=p.AD25	=p.AE25	=p.AC25	=fin.Z25	=fin.AA25	=fin.Y25	=SQRT(J25^2 + M25^2)	=SQRT(K25^2 + N25^2)	=SQRT(L25^2 + O25^2)	=fin.B25	=C25-F25	=T25-T26		
26	=SDSS.B26	=SDSS.N26	=p.Y26	=p.Z26	=p.X26	=fin.U26	=fin.V26	=fin.T26	=fin.H26	=p.AD26	=p.AE26	=p.AC26	=fin.Z26	=fin.AA26	=fin.Y26	=SQRT(J26^2 + M26^2)	=SQRT(K26^2 + N26^2)	=SQRT(L26^2 + O26^2)	=fin.B26	=C26-F26			
27	=SDSS.B27	=SDSS.N27	=p.Y27	=p.Z27	=p.X27	=fin.U27	=fin.V27	=fin.T27	=fin.H27	=p.AD27	=p.AE27	=p.AC27	=fin.Z27	=fin.AA27	=fin.Y27	=SQRT(J27^2 + M27^2)	=SQRT(K27^2 + N27^2)	=SQRT(L27^2 + O27^2)	=fin.B27	=C27-F27	=T27-T28		
28	=SDSS.B28	=SDSS.N28	=p.Y28	=p.Z28	=p.X28	=fin.U28	=fin.V28	=fin.T28	=fin.H28	=p.AD28	=p.AE28	=p.AC28	=fin.Z28	=fin.AA28	=fin.Y28	=SQRT(J28^2 + M28^2)	=SQRT(K28^2 + N28^2)	=SQRT(L28^2 + O28^2)	=fin.B28	=C28-F28			
29																							
30																							
31																							
32																							
33	=SDSS.B33	=SDSS.N33	=p.Y33	=p.Z33	=p.X33	=fin.U33	=fin.V33	=fin.T33	=fin.H33	=p.AD33	=p.AE33	=p.AC33	=fin.Z33	=fin.AA33	=fin.Y33	=SQRT(J33^2 + M33^2)	=SQRT(K33^2 + N33^2)	=SQRT(L33^2 + O33^2)	=fin.B33				
34	=SDSS.B34	=SDSS.N34	=p.Y34	=p.Z34	=p.X34	=fin.U34	=fin.V34	=fin.T34	=fin.H34	=p.AD34	=p.AE34	=p.AC34	=fin.Z34	=fin.AA34	=fin.Y34	=SQRT(J34^2 + M34^2)	=SQRT(K34^2 + N34^2)	=SQRT(L34^2 + O34^2)	=fin.B34				
35	=SDSS.B35	=SDSS.N35	=p.Y35	=p.Z35	=p.X35	=fin.U35	=fin.V35	=fin.T35	=fin.H35	=p.AD35	=p.AE35	=p.AC35	=fin.Z35	=fin.AA35	=fin.Y35	=SQRT(J35^2 + M35^2)	=SQRT(K35^2 + N35^2)	=SQRT(L35^2 + O35^2)	=fin.B35				
36	=SDSS.B36	=SDSS.N36	=p.Y36	=p.Z36	=p.X36	=fin.U36	=fin.V36	=fin.T36	=fin.H36	=p.AD36	=p.AE36	=p.AC36	=fin.Z36	=fin.AA36	=fin.Y36	=SQRT(J36^2 + M36^2)	=SQRT(K36^2 + N36^2)	=SQRT(L36^2 + O36^2)	=fin.B36				

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	**Mass Distribution**		**Mass Distribution**						White dwarf mass distribution in the SDSS (DR4) – Kepler 200				The Formation Rate and Mass and Luminosity Functions of DA White Dwarfs from the Palomar Green Survey (Liebert, James; Bergeron, P.; Holberg								
2									Figure 1. Histogram for the 1859		Figure 8. Histogram for the 1		1/Vmax weighted distributior		1/Vmax weighted distribution of the 2		1/Vmax weighted distribution of the 29		PG DA stars with Teff > 13000K (using 0.		
3																					
4	mass (msol)	mass (max)	frequency DD		Mass (min)	**Mass Range (Mass (mid)**	**=C4**		mass WD	frequency WD (Ke	mass WD	frequency WD	mass WD	frequency WD	mass WD	frequency WD 1/	mass WD	frequency WD 1/Vmax –	normalised using summed freq		
5	=fin.L5	0.3	{=FREQUENCY(A5:A32;B5:B16)}		=B5-0.1 = " " & D5 & "-"	=D5+0.05	=C5		0.2125	0	=I5	0	0.225	0.0000	=(M5+M6)/2	=N5+N6	=O5	=P5/$P$30*$G$18			
6	=fin.L6	0.4	{=FREQUENCY(A5:A32;B5:B16)}		=B6-0.1 = " " & D6 & "-"	=D6+0.05	=C6		0.2375	1	=I6	0	0.275	0.0000							
7	=fin.L7	0.5	{=FREQUENCY(A5:A32;B5:B16)}		=B7-0.1 = " " & D7 & "-"	=D7+0.05	=C7		0.2625	4	=I7	1.50E-007	0.325	0.0050	=(M7+M8)/2	=N7+N8	=O7	=P7/$P$30*$G$18			
8	=fin.L8	0.6	{=FREQUENCY(A5:A32;B5:B16)}		=B8-0.1 = " " & D8 & "-"	=D8+0.05	=C8		0.2875	4	=I8	2.00E-007	0.375	0.0090							
9	=fin.L9	0.7	{=FREQUENCY(A5:A32;B5:B16)}		=B9-0.1 = " " & D9 & "-"	=D9+0.05	=C9		0.3125	13	=I9	5.00E-007	0.425	0.0275	=(M9+M10)/2	=N9+N10	=O9	=P9/$P$30*$G$18			
10	=fin.L10	0.8	{=FREQUENCY(A5:A32;B5:B16)}		=B10-0.1 = " " & D10 & "	=D10+0.05	=C10		0.3375	17	=I10	8.00E-007	0.475	0.0330							
11		0.9	{=FREQUENCY(A5:A32;B5:B16)}		=B11-0.1 = " " & D11 & "	=D11+0.05	=C11		0.3625	36	=I11	1.10E-006	0.525	0.3200	=(M11+M12)/2	=N11+N12	=O11	=P11/$P$30*$G$18			
12		1	{=FREQUENCY(A5:A32;B5:B16)}		=B12-0.1 = " " & D12 & "	=D12+0.05	=C12		0.3875	26	=I12	9.00E-007	0.575	0.3600							
13	=fin.L13	1.1	{=FREQUENCY(A5:A32;B5:B16)}		=B13-0.1 = " " & D13 & "	=D13+0.05	=C13		0.4125	35	=I13	1.00E-006	0.625	0.2640	=(M13+M14)/2	=N13+N14	=O13	=P13/$P$30*$G$18			
14	=fin.L14	1.2	{=FREQUENCY(A5:A32;B5:B16)}		=B14-0.1 = " " & D14 & "	=D14+0.05	=C14		0.4375	20	=I14	9.00E-007	0.675	0.0680							
15	=fin.L15	1.3	{=FREQUENCY(A5:A32;B5:B16)}		=B15-0.1 = " " & D15 & "	=D15+0.05	=C15		0.4625	37	=I15	1.80E-006	0.725	0.0830	=(M15+M16)/2	=N15+N16	=O15	=P15/$P$30*$G$18			
16	=fin.L16	1.4	{=FREQUENCY(A5:A32;B5:B16)}		=B16-0.1 = " " & D16 & "	=D16+0.05	=C16		0.4875	63	=I16	3.00E-006	0.775	0.0600							
17	=fin.L17		{=FREQUENCY(A5:A32;B5:B16)}			max freq	=MAX(G5:G16)		0.5125	110	=I17	8.00E-006	0.825	0.0530	=(M17+M18)/2	=N17+N18	=O17	=P17/$P$30*$G$18			
18	=fin.L18					summed freq	=SUM(G5:G16)		0.5375	198	=I18	1.20E-005	0.875	0.0610							
19	=fin.L19								0.5625	280	=I19	1.80E-005	0.925	0.1130	=(M19+M20)/2	=N19+N20	=O19	=P19/$P$30*$G$18			
20	=fin.L20								0.5875	295	=I20	1.75E-005	0.975	0.0220							
21									0.6125	212	=I21	1.30E-005	1.025	0.0080	=(M21+M22)/2	=N21+N22	=O21	=P21/$P$30*$G$18			
22									0.6375	145	=I22	8.70E-006	1.075	0.0175							
23	=fin.L23								0.6625	78	=I23	5.40E-006	1.125	0.0000	=(M23+M24)/2	=N23+N24	=O23	=P23/$P$30*$G$18			
24	=fin.L24								0.6875	56	=I24	4.70E-006	1.175	0.0190							
25	=fin.L25								0.7125	32	=I25	2.75E-006	1.225	0.0010	=(M25+M26)/2	=N25+N26	=O25	=P25/$P$30*$G$18			
26	=fin.L26								0.7375	24	=I26	2.30E-006	1.275	0.0000							
27	=fin.L27								0.7625	24	=I27	2.20E-006	1.325	0.0500	=(M27+M28)/2	=N27+N28	=O27	=P27/$P$30*$G$18			
28	=fin.L28								0.7875	15	=I28	1.10E-006	1.375	0.0000							
29	=fin.L33								0.8125	17	=I29	1.70E-006			max freq	=MAX(P5:P28)					
30	=fin.L34								0.8375	10	=I30	1.60E-006			summed freq	=SUM(P5:P27)					
31	=fin.L35								0.8625	16	=I31	1.50E-006									
32	=fin.L36								0.8875	16	=I32	1.80E-006									
33									0.9125	3	=I33	1.50E-007									
34									0.9375	8	=I34	5.00E-007									
35									0.9625	9	=I35	1.40E-006									
36									0.9875	5	=I36	3.00E-007									
37									1.0125	6	=I37	6.50E-007									
38									1.0375	3	=I38	1.00E-007									
39									1.0625	5	=I39	1.15E-006									
40									1.0875	2	=I40	6.00E-007									
41									1.1125	4	=I41	2.50E-006									
42									1.1375	1	=I42	3.00E-007									
43									1.1625	7	=I43	3.10E-006									
44									1.1875	2	=I44	8.00E-007									
45									1.2125	3	=I45	1.60E-006									
46									1.2375	1	=I46	0									
47									1.2625	2	=I47	1.60E-006									
48									1.2875	1	=I48	4.00E-007									
49									1.3125	0	=I49	0									
50									1.3375	1	=I50	1.60E-006									
51									1.3625	0	=I51	0									
52									1.3875	0	=I52	0									
53										=MAX(J5:J52)											

IFMR refinement

		Normal						Low Mass Extreme - With IFMR Errors						High Mass Extreme - With IFMR Errors						
		mass.txt (1) input		mass.txt (1) output				mass.txt (1) input		mass.txt (1) output				mass.txt (1) input		mass.txt (1) output				
4	name	Final Mass (Wi	nitial Mass (from theoretical I	Main Sequenc	Main Sequence l	Cooling Time (y)	System Age	Final Mass (WD)	Initial Mass (from theoretical IFMR)	Main Sequenc	Main Sequenc	Cooling Time (y)	System Age	Final Mass (WD)	Initial Mass (from theoretical IFMR)	Main Sequenc	Main Sequenc	Cooling Time (y)	System Age	
5	=fin.A5	=fin.L5	= (B5 - $B$47) / $B$46	1.43E+009	=D5	=fin.N5	=E5+F5	=fin.L5-fin.M5	= (H5 - ($B$47+$C$47)) / ($B$46+$C$46)	1.24E+010	=J5	=fin.N5-fin.O5	=K5+L5	=fin.L5+fin.M5	= (N5 - ($B$47-$C$47)) / ($B$46-$C$46)	4.50E+008	=P5	=fin.N5+fin.O5	=Q5+R5	
6	=fin.A6	=fin.L6	= (B6 - $B$47) / $B$46	9.17E+008	=D6	=fin.N6	=E6+F6	=fin.L6-fin.M6	= (H6 - ($B$47+$C$47)) / ($B$46+$C$46)	2.77E+009	=J6	=fin.N6-fin.O6	=K6+L6	=fin.L6+fin.M6	= (N6 - ($B$47-$C$47)) / ($B$46-$C$46)	3.22E+008	=P6	=fin.N6+fin.O6	=Q6+R6	
7	=fin.A7	=fin.L7	= (B7 - $B$47) / $B$46	4.50E+008	=D7	=fin.N7	=E7+F7	=fin.L7-fin.M7	= (H7 - ($B$47+$C$47)) / ($B$46+$C$46)	1.26E+009	=J7	=fin.N7-fin.O7	=K7+L7	=fin.L7+fin.M7	= (N7 - ($B$47-$C$47)) / ($B$46-$C$46)	1.84E+008	=P7	=fin.N7+fin.O7	=Q7+R7	
8	=fin.A8	=fin.L8	= (B8 - $B$47) / $B$46	3.13E+008	=D8	=fin.N8	=E8+F8	=fin.L8-fin.M8	= (H8 - ($B$47+$C$47)) / ($B$46+$C$46)	1.76E+008	=J8	=fin.N8-fin.O8	=K8+L8	=fin.L8+fin.M8	= (N8 - ($B$47-$C$47)) / ($B$46-$C$46)	2.88E+009	=P8	=fin.N8+fin.O8	=Q8+R8	
9	=fin.A9	=fin.L9	= (B9 - $B$47) / $B$46	3.82E+008	=D9	=fin.N9	=E9+F9	=fin.L9-fin.M9	= (H9 - ($B$47+$C$47)) / ($B$46+$C$46)	1.12E+009	=J9	=fin.N9-fin.O9	=K9+L9	=fin.L9+fin.M9	= (N9 - ($B$47-$C$47)) / ($B$46-$C$46)	1.54E+008	=P9	=fin.N9+fin.O9	=Q9+R9	
10	=fin.A10	=fin.L10	= (B10 - $B$47) / $B$46	1.00E+009	=D10	=fin.N10	=E10+F10	=fin.L10-fin.M10	= (H10 - ($B$47+$C$47)) / ($B$46+$C$46)	3.95E+009	=J10	=fin.N10-fin.O10	=K10+L10	=fin.L10+fin.M10	= (N10 - ($B$47-$C$47)) / ($B$46-$C$46)	3.12E+008	=P10	=fin.N10+fin.O10	=Q10+R10	
11	=fin.A11	=fin.L11	= (B11 - $B$47) / $B$46	4.23E+008	=D11	=fin.N11	=E11+F11	=fin.L11-fin.M11	= (H11 - ($B$47+$C$47)) / ($B$46+$C$46)	1.18E+009	=J11	=fin.N11-fin.O11	=K11+L11	=fin.L11+fin.M11	= (N11 - ($B$47-$C$47)) / ($B$46-$C$46)	1.75E+008	=P11	=fin.N11+fin.O11	=Q11+R11	
12	=fin.A12	=fin.L12	= (B12 - $B$47) / $B$46	1.45E+008	=D12	=fin.N12	=E12+F12	=fin.L12-fin.M12	= (H12 - ($B$47+$C$47)) / ($B$46+$C$46)	2.96E+008	=J12	=fin.N12-fin.O12	=K12+L12	=fin.L12+fin.M12	= (N12 - ($B$47-$C$47)) / ($B$46-$C$46)	6.79E+008	=P12	=fin.N12+fin.O12	=Q12+R12	
13	=fin.A13	=fin.L13	= (B13 - $B$47) / $B$46	1.88E+008	=D13	=fin.N13	=E13+F13	=fin.L13-fin.M13	= (H13 - ($B$47+$C$47)) / ($B$46+$C$46)	4.20E+008	=J13	=fin.N13-fin.O13	=K13+L13	=fin.L13+fin.M13	= (N13 - ($B$47-$C$47)) / ($B$46-$C$46)	2.95E+008	=P13	=fin.N13+fin.O13	=Q13+R13	
14	=fin.A14	=fin.L14	= (B14 - $B$47) / $B$46	6.23E+008	=D14	=fin.N14	=E14+F14	=fin.L14-fin.M14	= (H14 - ($B$47+$C$47)) / ($B$46+$C$46)	1.68E+009	=J14	=fin.N14-fin.O14	=K14+L14	=fin.L14+fin.M14	= (N14 - ($B$47-$C$47)) / ($B$46-$C$46)	2.34E+008	=P14	=fin.N14+fin.O14	=Q14+R14	
15	=fin.A15	=fin.L15	= (B15 - $B$47) / $B$46	5.42E+008	=D15	=fin.N15	=E15+F15	=fin.L15-fin.M15	= (H15 - ($B$47+$C$47)) / ($B$46+$C$46)	1.46E+009	=J15	=fin.N15-fin.O15	=K15+L15	=fin.L15+fin.M15	= (N15 - ($B$47-$C$47)) / ($B$46-$C$46)	2.14E+008	=P15	=fin.N15+fin.O15	=Q15+R15	
16	=fin.A16	=fin.L16	= (B16 - $B$47) / $B$46	6.18E+008	=D16	=fin.N16	=E16+F16	=fin.L16-fin.M16	= (H16 - ($B$47+$C$47)) / ($B$46+$C$46)	1.64E+009	=J16	=fin.N16-fin.O16	=K16+L16	=fin.L16+fin.M16	= (N16 - ($B$47-$C$47)) / ($B$46-$C$46)	2.34E+008	=P16	=fin.N16+fin.O16	=Q16+R16	
17	=fin.A17	=fin.L17	= (B17 - $B$47) / $B$46	6.18E+008	=D17	=fin.N17	=E17+F17	=fin.L17-fin.M17	= (H17 - ($B$47+$C$47)) / ($B$46+$C$46)	1.59E+009	=J17	=fin.N17-fin.O17	=K17+L17	=fin.L17+fin.M17	= (N17 - ($B$47-$C$47)) / ($B$46-$C$46)	2.40E+008	=P17	=fin.N17+fin.O17	=Q17+R17	
18	=fin.A18	=fin.L18	= (B18 - $B$47) / $B$46	1.07E+009	=D18	=fin.N18	=E18+F18	=fin.L18-fin.M18	= (H18 - ($B$47+$C$47)) / ($B$46+$C$46)	4.76E+009	=J18	=fin.N18-fin.O18	=K18+L18	=fin.L18+fin.M18	= (N18 - ($B$47-$C$47)) / ($B$46-$C$46)	3.19E+008	=P18	=fin.N18+fin.O18	=Q18+R18	
19	=fin.A19	=fin.L19	= (B19 - $B$47) / $B$46	8.34E+008	=D19	=fin.N19	=E19+F19	=fin.L19-fin.M19	= (H19 - ($B$47+$C$47)) / ($B$46+$C$46)	7.17E+009	=J19	=fin.N19-fin.O19	=K19+L19	=fin.L19+fin.M19	= (N19 - ($B$47-$C$47)) / ($B$46-$C$46)	2.01E+008	=P19	=fin.N19+fin.O19	=Q19+R19	
20	=fin.A20	=fin.L20	= (B20 - $B$47) / $B$46	2.46E+008	=D20	=fin.N20	=E20+F20	=fin.L20-fin.M20	= (H20 - ($B$47+$C$47)) / ($B$46+$C$46)	6.32E+008	=J20	=fin.N20-fin.O20	=K20+L20	=fin.L20+fin.M20	= (N20 - ($B$47-$C$47)) / ($B$46-$C$46)	1.68E+008	=P20	=fin.N20+fin.O20	=Q20+R20	
21	=fin.A21	=fin.L21	= (B21 - $B$47) / $B$46	1.38E+009	=D21	=fin.N21	=E21+F21	=fin.L21-fin.M21	= (H21 - ($B$47+$C$47)) / ($B$46+$C$46)	1.27E+010	=J21	=fin.N21-fin.O21	=K21+L21	=fin.L21+fin.M21	= (N21 - ($B$47-$C$47)) / ($B$46-$C$46)	4.76E+008	=P21	=fin.N21+fin.O21	=Q21+R21	
22	=fin.A22	=fin.L22	= (B22 - $B$47) / $B$46	1.54E+009	=D22	=fin.N22	=E22+F22	=fin.L22-fin.M22	= (H22 - ($B$47+$C$47)) / ($B$46+$C$46)	2.23E+008	=J22	=fin.N22-fin.O22	=K22+L22	=fin.L22+fin.M22	= (N22 - ($B$47-$C$47)) / ($B$46-$C$46)	7.31E+009	=P22	=fin.N22+fin.O22	=Q22+R22	
23	=fin.A23	=fin.L23	= (B23 - $B$47) / $B$46	1.65E+008	=D23	=fin.N23	=E23+F23	=fin.L23-fin.M23	= (H23 - ($B$47+$C$47)) / ($B$46+$C$46)	3.56E+008	=J23	=fin.N23-fin.O23	=K23+L23	=fin.L23+fin.M23	= (N23 - ($B$47-$C$47)) / ($B$46-$C$46)	4.54E+008	=P23	=fin.N23+fin.O23	=Q23+R23	
24	=fin.A24	=fin.L24	= (B24 - $B$47) / $B$46	3.00E+008	=D24	=fin.N24	=E24+F24	=fin.L24-fin.M24	= (H24 - ($B$47+$C$47)) / ($B$46+$C$46)	7.80E+008	=J24	=fin.N24-fin.O24	=K24+L24	=fin.L24+fin.M24	= (N24 - ($B$47-$C$47)) / ($B$46-$C$46)	1.41E+008	=P24	=fin.N24+fin.O24	=Q24+R24	
25	=fin.A25	=fin.L25	= (B25 - $B$47) / $B$46	3.43E+008	=D25	=fin.N25	=E25+F25	=fin.L25-fin.M25	= (H25 - ($B$47+$C$47)) / ($B$46+$C$46)	9.00E+008	=J25	=fin.N25-fin.O25	=K25+L25	=fin.L25+fin.M25	= (N25 - ($B$47-$C$47)) / ($B$46-$C$46)	1.49E+008	=P25	=fin.N25+fin.O25	=Q25+R25	
26	=fin.A26	=fin.L26	= (B26 - $B$47) / $B$46	1.56E+008	=D26	=fin.N26	=E26+F26	=fin.L26-fin.M26	= (H26 - ($B$47+$C$47)) / ($B$46+$C$46)	3.30E+008	=J26	=fin.N26-fin.O26	=K26+L26	=fin.L26+fin.M26	= (N26 - ($B$47-$C$47)) / ($B$46-$C$46)	5.30E+008	=P26	=fin.N26+fin.O26	=Q26+R26	
27	=fin.A27	=fin.L27	= (B27 - $B$47) / $B$46	7.74E+008	=D27	=fin.N27	=E27+F27	=fin.L27-fin.M27	= (H27 - ($B$47+$C$47)) / ($B$46+$C$46)	2.20E+009	=J27	=fin.N27-fin.O27	=K27+L27	=fin.L27+fin.M27	= (N27 - ($B$47-$C$47)) / ($B$46-$C$46)	2.79E+008	=P27	=fin.N27+fin.O27	=Q27+R27	
28	=fin.A28	=fin.L28	= (B28 - $B$47) / $B$46	7.95E+008	=D28	=fin.N28	=E28+F28	=fin.L28-fin.M28	= (H28 - ($B$47+$C$47)) / ($B$46+$C$46)	2.96E+009	=J28	=fin.N28-fin.O28	=K28+L28	=fin.L28+fin.M28	= (N28 - ($B$47-$C$47)) / ($B$46-$C$46)	2.51E+008	=P28	=fin.N28+fin.O28	=Q28+R28	
29				0						0						0				
30				0						0						0				
31				0						0						0				
32																				
33	=fin.A33	=fin.L33	= (B33 - $B$47) / $B$46	2.50E+008	=D33	=fin.N33	=E33+F33	=fin.L33-fin.M33	= (H33 - ($B$47+$C$47)) / ($B$46+$C$46)	5.57E+008	=J33	=fin.N33-fin.O33	=K33+L33	=fin.L33+fin.M33	= (N33 - ($B$47-$C$47)) / ($B$46-$C$46)	1.44E+008	=P33	=fin.N33+fin.O33	=Q33+R33	
34	=fin.A34	=fin.L34	= (B34 - $B$47) / $B$46	6.81E+008	=D34	=fin.N34	=E34+F34	=fin.L34-fin.M34	= (H34 - ($B$47+$C$47)) / ($B$46+$C$46)	1.42E+008	=J34	=fin.N34-fin.O34	=K34+L34	=fin.L34+fin.M34	= (N34 - ($B$47-$C$47)) / ($B$46-$C$46)	3.95E+009	=P34	=fin.N34+fin.O34	=Q34+R34	
35	=fin.A35	=fin.L35	= (B35 - $B$47) / $B$46	1.62E+009	=D35	=fin.N35	=E35+F35	=fin.L35-fin.M35	= (H35 - ($B$47+$C$47)) / ($B$46+$C$46)	4.76E+009	=J35	=fin.N35-fin.O35	=K35+L35	=fin.L35+fin.M35	= (N35 - ($B$47-$C$47)) / ($B$46-$C$46)	7.52E+008	=P35	=fin.N35+fin.O35	=Q35+R35	
36	=fin.A36	=fin.L36	= (B36 - $B$47) / $B$46	9.33E+008	=D36	=fin.N36	=E36+F36	=fin.L36-fin.M36	= (H36 - ($B$47+$C$47)) / ($B$46+$C$46)	1.93E+009	=J36	=fin.N36-fin.O36	=K36+L36	=fin.L36+fin.M36	= (N36 - ($B$47-$C$47)) / ($B$46-$C$46)	4.13E+008	=P36	=fin.N36+fin.O36	=Q36+R36	
37												Use This								
38																				
39																				
40																				
41																				
42																				
43	IFMR used;																			
44	Final mass = m * InitialMass + c																			
45	dobbie_new_2Theoretical Vaerror																			
46	IFMR m =	=fin.BM44	=fin.BN44																	
47	IFMR c =	=fin.BM45	=fin.BN45																	

Assume Girardi models
Solar-metallicity (Z = 0.019) and Overshoot
assume initial mass calculated @ AGBt

	T	U	V	W	X	Y	Z	AA	AB	AC	AD
1			findMainSequenceCoolingTimeOrMassInGirardiModel.pro								
2			initialMassInput.txt			initialAgeInput.txt			finalMassInput.txt		
3	alignment with IFMR? (with errors)		Low/High Mass Extreme - With IFMR Errors								
4		Initial Mass dif	Est. MS Mass	Est. MS Mass	Est. MS Mass	Est. MS Age	Est. MS Age min	Est. MS Age m	WD Mass	WD Mass min	WD Mass max
5	=IF(($G6>$G5); IF(((MIN(M6;S6) < MAX(M5;S5)) AND (MAX(M5;S5) > MIN(M6;S6))); 1; 0); IF(((MIN(M5;S5) < MAX(M6;S6)) AND (MAX(M6;S6) > MIN(M5;S5))); 1; 0))	=C5-C6	=C5	=I5	=O5	=D5/1000000	=P5/1000000	=J5/1000000	=B5	=fin.L5-fin.M5	=fin.L5+fin.M5
6			=C6	=I6	=O6	=D6/1000000	=P6/1000000	=J6/1000000	=B6	=fin.L6-fin.M6	=fin.L6+fin.M6
7	=IF(($G8>$G7); IF(((MIN(M8;S8) < MAX(M7;S7)) AND (MAX(M7;S7) > MIN(M8;S8))); 1; 0); IF(((MIN(M7;S7) < MAX(M8;S8)) AND (MAX(M8;S8) > MIN(M7;S7))); 1; 0))	=C7-C8	=C7	=I7	=O7	=D7/1000000	=P7/1000000	=J7/1000000	=B7	=fin.L7-fin.M7	=fin.L7+fin.M7
8			=C8	=I8	=O8	=D8/1000000	=P8/1000000	=J8/1000000	=B8	=fin.L8-fin.M8	=fin.L8+fin.M8
9	=IF(($G10>$G9); IF(((MIN(M10;S10) < MAX(M9;S9)) AND (MAX(M9;S9) > MIN(M10;S10))); 1; 0); IF(((MIN(M9;S9) < MAX(M10;S10)) AND (MAX(M10;S10) > MIN(M9;S9))); 1; 0))	=C9-C10	=C9	=I9	=O9	=D9/1000000	=P9/1000000	=J9/1000000	=B9	=fin.L9-fin.M9	=fin.L9+fin.M9
10			=C10	=I10	=O10	=D10/1000000	=P10/1000000	=J10/1000000	=B10	=fin.L10-fin.M10	=fin.L10+fin.M10
11	=IF(($G12>$G11); IF(((MIN(M12;S12) < MAX(M11;S11)) AND (MAX(M11;S11) > MIN(M12;S12))); 1; 0); IF(((MIN(M11;S11) < MAX(M12;S12)) AND (MAX(M12;S12) > MIN(M11;S11))); 1; 0))	=C11-C12	=C11	=I11	=O11	=D11/1000000	=P11/1000000	=J11/1000000	=B11	=fin.L11-fin.M11	=fin.L11+fin.M11
12			=C12	=I12	=O12	=D12/1000000	=P12/1000000	=J12/1000000	=B12	=fin.L12-fin.M12	=fin.L12+fin.M12
13	=IF(($G14>$G13); IF(((MIN(M14;S14) < MAX(M13;S13)) AND (MAX(M13;S13) > MIN(M14;S14))); 1; 0); IF(((MIN(M13;S13) < MAX(M14;S14)) AND (MAX(M14;S14) > MIN(M13;S13))); 1; 0))	=C13-C14	=C13	=I13	=O13	=D13/1000000	=P13/1000000	=J13/1000000	=B13	=fin.L13-fin.M13	=fin.L13+fin.M13
14			=C14	=I14	=O14	=D14/1000000	=P14/1000000	=J14/1000000	=B14	=fin.L14-fin.M14	=fin.L14+fin.M14
15	=IF(($G16>$G15); IF(((MIN(M16;S16) < MAX(M15;S15)) AND (MAX(M15;S15) > MIN(M16;S16))); 1; 0); IF(((MIN(M15;S15) < MAX(M16;S16)) AND (MAX(M16;S16) > MIN(M15;S15))); 1; 0))	=C15-C16	=C15	=I15	=O15	=D15/1000000	=P15/1000000	=J15/1000000	=B15	=fin.L15-fin.M15	=fin.L15+fin.M15
16			=C16	=I16	=O16	=D16/1000000	=P16/1000000	=J16/1000000	=B16	=fin.L16-fin.M16	=fin.L16+fin.M16
17	=IF(($G18>$G17); IF(((MIN(M18;S18) < MAX(M17;S17)) AND (MAX(M17;S17) > MIN(M18;S18))); 1; 0); IF(((MIN(M17;S17) < MAX(M18;S18)) AND (MAX(M18;S18) > MIN(M17;S17))); 1; 0))	=C17-C18	=C17	=I17	=O17	=D17/1000000	=P17/1000000	=J17/1000000	=B17	=fin.L17-fin.M17	=fin.L17+fin.M17
18			=C18	=I118	=O18	=D18/1000000	=P18/1000000	=J18/1000000	=B18	=fin.L18-fin.M18	=fin.L18+fin.M18
19	=IF(($G20>$G19); IF(((MIN(M20;S20) < MAX(M19;S19)) AND (MAX(M19;S19) > MIN(M20;S20))); 1; 0); IF(((MIN(M19;S19) < MAX(M20;S20)) AND (MAX(M20;S20) > MIN(M19;S19))); 1; 0))	=C19-C20	=C19	=I19	=O19	=D19/1000000	=P19/1000000	=J19/1000000	=B19	=fin.L19-fin.M19	=fin.L19+fin.M19
20			=C20	=I20	=O20	=D20/1000000	=P20/1000000	=J20/1000000	=B20	=fin.L20-fin.M20	=fin.L20+fin.M20
21	=IF(($G22>$G21); IF(((MIN(M22;S22) < MAX(M21;S21)) AND (MAX(M21;S21) > MIN(M22;S22))); 1; 0); IF(((MIN(M21;S21) < MAX(M22;S22)) AND (MAX(M22;S22) > MIN(M21;S21))); 1; 0))	=C21-C22	=C21	=I21	=O21	=D21/1000000	=P21/1000000	=J21/1000000	=B21	=fin.L21-fin.M21	=fin.L21+fin.M21
22			=C22	=I22	=O22	=D22/1000000	=P22/1000000	=J22/1000000	=B22	=fin.L22-fin.M22	=fin.L22+fin.M22
23	=IF(($G24>$G23); IF(((MIN(M24;S24) < MAX(M23;S23)) AND (MAX(M23;S23) > MIN(M24;S24))); 1; 0); IF(((MIN(M23;S23) < MAX(M24;S24)) AND (MAX(M24;S24) > MIN(M23;S23))); 1; 0))	=C23-C24	=C23	=I23	=O23	=D23/1000000	=P23/1000000	=J23/1000000	=B23	=fin.L23-fin.M23	=fin.L23+fin.M23
24			=C24	=I24	=O24	=D24/1000000	=P24/1000000	=J24/1000000	=B24	=fin.L24-fin.M24	=fin.L24+fin.M24
25	=IF(($G26>$G25); IF(((MIN(M26;S26) < MAX(M25;S25)) AND (MAX(M25;S25) > MIN(M26;S26))); 1; 0); IF(((MIN(M25;S25) < MAX(M26;S26)) AND (MAX(M26;S26) > MIN(M25;S25))); 1; 0))	=C25-C26	=C25	=I25	=O25	=D25/1000000	=P25/1000000	=J25/1000000	=B25	=fin.L25-fin.M25	=fin.L25+fin.M25
26			=C26	=I26	=O26	=D26/1000000	=P26/1000000	=J26/1000000	=B26	=fin.L26-fin.M26	=fin.L26+fin.M26
27	=IF(($G28>$G27); IF(((MIN(M28;S28) < MAX(M27;S27)) AND (MAX(M27;S27) > MIN(M28;S28))); 1; 0); IF(((MIN(M27;S27) < MAX(M28;S28)) AND (MAX(M28;S28) > MIN(M27;S27))); 1; 0))	=C27-C28	=C27	=I27	=O27	=D27/1000000	=P27/1000000	=J27/1000000	=B27	=fin.L27-fin.M27	=fin.L27+fin.M27
28			=C28	=I28	=O28	=D28/1000000	=P28/1000000	=J28/1000000	=B28	=fin.L28-fin.M28	=fin.L28+fin.M28
29											
30											
31											
32											
33	=IF(($G34>$G33); IF(((MIN(M34;S34) < MAX(M33;S33)) AND (MAX(M33;S33) > MIN(M34;S34))); 1; 0); IF(((MIN(M33;S33) < MAX(M34;S34)) AND (MAX(M34;S34) > MIN(M33;S33))); 1; 0))		=C33	=I33	=O33	=D33/1000000	=P33/1000000	=J33/1000000	=B33	=fin.L33-fin.M33	=fin.L33+fin.M33
34			=C34	=I34	=O34	=D34/1000000	=P34/1000000	=J34/1000000	=B34	=fin.L34-fin.M34	=fin.L34+fin.M34
35	=IF(($G36>$G35); IF(((MIN(M36;S36) < MAX(M35;S35)) AND (MAX(M35;S35) > MIN(M36;S36))); 1; 0); IF(((MIN(M35;S35) < MAX(M36;S36)) AND (MAX(M36;S36) > MIN(M35;S35))); 1; 0))		=C35	=I35	=O35	=D35/1000000	=P35/1000000	=J35/1000000	=B35	=fin.L35-fin.M35	=fin.L35+fin.M35
36			=C36	=I36	=O36	=D36/1000000	=P36/1000000	=J36/1000000	=B36	=fin.L36-fin.M36	=fin.L36+fin.M36
37											
38											
39											
40											
41											
42											
43											
44											
45											
46											
47											

	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS
1															
2	finalAgeInput.txt			systemAgeConstraints.txt			STRICTsystensystemAgeCommon.txt [NoSysWithMagneticsCompOrDD-09]								
3															
4	Cooling Age Age	Cooling Age min	Cooling Age max	System Age	Low Mass Extreme	High Mass Extreme S	System Age (My)			line	DD		distance	distance min	distance max
5	=fin.N5/1000000	=(fin.N5-fin.O5)/1000000	=(fin.N5+fin.O5)/1000000	=G5/1000000	=S5/1000000	=M5/1000000	=T5	1			0	=A5	=fin.AD5	=fin.AD5-fin.AE5	=fin.AD5+fin.AE5
6	=fin.N6/1000000	=(fin.N6-fin.O6)/1000000	=(fin.N6+fin.O6)/1000000	=G6/1000000	=S6/1000000	=M6/1000000	=AK5	1			1	=A6	=fin.AD6	=fin.AD6-fin.AE6	=fin.AD6+fin.AE6
7	=fin.N7/1000000	=(fin.N7-fin.O7)/1000000	=(fin.N7+fin.O7)/1000000	=G7/1000000	=S7/1000000	=M7/1000000	=T7	0			2	=A7	=fin.AD7	=fin.AD7-fin.AE7	=fin.AD7+fin.AE7
8	=fin.N8/1000000	=(fin.N8-fin.O8)/1000000	=(fin.N8+fin.O8)/1000000	=G8/1000000	=S8/1000000	=M8/1000000	=AK7	0			3	=A8	=fin.AD8	=fin.AD8-fin.AE8	=fin.AD8+fin.AE8
9	=fin.N9/1000000	=(fin.N9-fin.O9)/1000000	=(fin.N9+fin.O9)/1000000	=G9/1000000	=S9/1000000	=M9/1000000	=T9	1			4	=A9	=fin.AD9	=fin.AD9-fin.AE9	=fin.AD9+fin.AE9
10	=fin.N10/1000000	=(fin.N10-fin.O10)/1000000	=(fin.N10+fin.O10)/1000000	=G10/1000000	=S10/1000000	=M10/1000000	=AK9	1			5	=A10	=fin.AD10	=fin.AD10-fin.AE10	=fin.AD10+fin.AE10
11	=fin.N11/1000000	=(fin.N11-fin.O11)/1000000	=(fin.N11+fin.O11)/1000000	=G11/1000000	=S11/1000000	=M11/1000000	=T11	0			6	=A11	=fin.AD11	=fin.AD11-fin.AE11	=fin.AD11+fin.AE11
12	=fin.N12/1000000	=(fin.N12-fin.O12)/1000000	=(fin.N12+fin.O12)/1000000	=G12/1000000	=S12/1000000	=M12/1000000	=AK11	0			7	=A12	=fin.AD12	=fin.AD12-fin.AE12	=fin.AD12+fin.AE12
13	=fin.N13/1000000	=(fin.N13-fin.O13)/1000000	=(fin.N13+fin.O13)/1000000	=G13/1000000	=S13/1000000	=M13/1000000	1	1			8	=A13	=fin.AD13	=fin.AD13-fin.AE13	=fin.AD13+fin.AE13
14	=fin.N14/1000000	=(fin.N14-fin.O14)/1000000	=(fin.N14+fin.O14)/1000000	=G14/1000000	=S14/1000000	=M14/1000000	=AK13	1			9	=A14	=fin.AD14	=fin.AD14-fin.AE14	=fin.AD14+fin.AE14
15	=fin.N15/1000000	=(fin.N15-fin.O15)/1000000	=(fin.N15+fin.O15)/1000000	=G15/1000000	=S15/1000000	=M15/1000000	=T15	1			10	=A15	=fin.AD15	=fin.AD15-fin.AE15	=fin.AD15+fin.AE15
16	=fin.N16/1000000	=(fin.N16-fin.O16)/1000000	=(fin.N16+fin.O16)/1000000	=G16/1000000	=S16/1000000	=M16/1000000	=AK15	1			11	=A16	=fin.AD16	=fin.AD16-fin.AE16	=fin.AD16+fin.AE16
17	=fin.N17/1000000	=(fin.N17-fin.O17)/1000000	=(fin.N17+fin.O17)/1000000	=G17/1000000	=S17/1000000	=M17/1000000	=T17	1			12	=A17	=fin.AD17	=fin.AD17-fin.AE17	=fin.AD17+fin.AE17
18	=fin.N18/1000000	=(fin.N18-fin.O18)/1000000	=(fin.N18+fin.O18)/1000000	=G18/1000000	=S18/1000000	=M18/1000000	=AK17	1			13	=A18	=fin.AD18	=fin.AD18-fin.AE18	=fin.AD18+fin.AE18
19	=fin.N19/1000000	=(fin.N19-fin.O19)/1000000	=(fin.N19+fin.O19)/1000000	=G19/1000000	=S19/1000000	=M19/1000000	=T19	0			14	=A19	=fin.AD19	=fin.AD19-fin.AE19	=fin.AD19+fin.AE19
20	=fin.N20/1000000	=(fin.N20-fin.O20)/1000000	=(fin.N20+fin.O20)/1000000	=G20/1000000	=S20/1000000	=M20/1000000	=AK19	0			15	=A20	=fin.AD20	=fin.AD20-fin.AE20	=fin.AD20+fin.AE20
21	=fin.N21/1000000	=(fin.N21-fin.O21)/1000000	=(fin.N21+fin.O21)/1000000	=G21/1000000	=S21/1000000	=M21/1000000	=T21	0			16	=A21	=fin.AD21	=fin.AD21-fin.AE21	=fin.AD21+fin.AE21
22	=fin.N22/1000000	=(fin.N22-fin.O22)/1000000	=(fin.N22+fin.O22)/1000000	=G22/1000000	=S22/1000000	=M22/1000000	=AK21	0			17	=A22	=fin.AD22	=fin.AD22-fin.AE22	=fin.AD22+fin.AE22
23	=fin.N23/1000000	=(fin.N23-fin.O23)/1000000	=(fin.N23+fin.O23)/1000000	=G23/1000000	=S23/1000000	=M23/1000000	=T23	1			18	=A23	=fin.AD23	=fin.AD23-fin.AE23	=fin.AD23+fin.AE23
24	=fin.N24/1000000	=(fin.N24-fin.O24)/1000000	=(fin.N24+fin.O24)/1000000	=G24/1000000	=S24/1000000	=M24/1000000	=AK23	1			19	=A24	=fin.AD24	=fin.AD24-fin.AE24	=fin.AD24+fin.AE24
25	=fin.N25/1000000	=(fin.N25-fin.O25)/1000000	=(fin.N25+fin.O25)/1000000	=G25/1000000	=S25/1000000	=M25/1000000	=T25	1			20	=A25	=fin.AD25	=fin.AD25-fin.AE25	=fin.AD25+fin.AE25
26	=fin.N26/1000000	=(fin.N26-fin.O26)/1000000	=(fin.N26+fin.O26)/1000000	=G26/1000000	=S26/1000000	=M26/1000000	=AK25	1			21	=A26	=fin.AD26	=fin.AD26-fin.AE26	=fin.AD26+fin.AE26
27	=fin.N27/1000000	=(fin.N27-fin.O27)/1000000	=(fin.N27+fin.O27)/1000000	=G27/1000000	=S27/1000000	=M27/1000000	=T27	1			22	=A27	=fin.AD27	=fin.AD27-fin.AE27	=fin.AD27+fin.AE27
28	=fin.N28/1000000	=(fin.N28-fin.O28)/1000000	=(fin.N28+fin.O28)/1000000	=G28/1000000	=S28/1000000	=M28/1000000	=AK27	1			23	=A28	=fin.AD28	=fin.AD28-fin.AE28	=fin.AD28+fin.AE28
29															
30															
31															
32															
33	=fin.N33/1000000	=(fin.N33-fin.O33)/1000000	=(fin.N33+fin.O33)/1000000	=G33/1000000	=S33/1000000	=M33/1000000	=T33	1					=fin.AD33	=fin.AD33-fin.AE33	=fin.AD33+fin.AE33
34	=fin.N34/1000000	=(fin.N34-fin.O34)/1000000	=(fin.N34+fin.O34)/1000000	=G34/1000000	=S34/1000000	=M34/1000000	=AK33	1					=fin.AD34	=fin.AD34-fin.AE34	=fin.AD34+fin.AE34
35	=fin.N35/1000000	=(fin.N35-fin.O35)/1000000	=(fin.N35+fin.O35)/1000000	=G35/1000000	=S35/1000000	=M35/1000000	=T35	1					=fin.AD35	=fin.AD35-fin.AE35	=fin.AD35+fin.AE35
36	=fin.N36/1000000	=(fin.N36-fin.O36)/1000000	=(fin.N36+fin.O36)/1000000	=G36/1000000	=S36/1000000	=M36/1000000	=AK35	1					=fin.AD36	=fin.AD36-fin.AE36	=fin.AD36+fin.AE36
37															
38															
39															
40															
41															
42															
43															
44															
45															
46															
47															

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA
1	**RBBcompareDDSystemAgeDifferenceDistributionAcrossPublishedIFMRSOutput**																										
2																											
3																					**Dobbie IFMR**						
4	WDindex	DD index	Median distan	Median distan	Median distan	Median M	Median M erro	Median M erro	Radius (solar)	Radius error m	Radius error m	GR (km/s)	GR error max	GR error min (Mf		Mf error max	Mf error min	Cooling time (t	Cooling time e	Cooling time d	Mi (Myr)	Mi error max	Mi error min	Agei	Agei error max	Agei error min	binary lifetime
5	0	DD-01A													0.510	0.089	-0.047	418.500	79.300	-38.000	1.664	1.036	-0.505	2149.000	4089.000	-1584.000	2568.000
6	1	DD-01B													0.605	0.040	-0.038	76.350	15.860	-13.120	2.376	0.601	-0.517	912.500	738.500	-424.700	988.800
7	2	DD-02A													0.695	0.043	-0.042	156.700	24.500	-21.800	3.050	0.655	-0.573	454.100	364.200	-191.600	610.800
8	3	DD-02B													1.027	0.044	-0.050	353.900	49.900	-45.200	5.545	0.882	-0.768	93.900	41.500	-27.210	447.800
9	4	DD-03A													0.717	0.051	-0.050	794.900	115.700	-101.200	3.221	0.711	-0.620	387.600	330.300	-164.100	1182.000
10	5	DD-03B													0.598	0.054	-0.051	439.800	65.000	-52.800	2.323	0.660	-0.581	970.800	919.200	-486.600	1411.000
11	6	DD-04A													0.707	0.044	-0.044	309.800	42.000	-36.400	3.142	0.665	-0.577	416.600	329.100	-172.800	726.400
12	7	DD-04B													0.911	0.043	-0.044	54.170	13.570	-12.370	4.675	0.792	-0.672	142.900	70.200	-45.720	197.100
13	8	DD-05A													0.849	0.045	-0.045	180.000	28.500	-24.600	4.207	0.757	-0.648	187.000	106.300	-63.800	367.000
14	9	DD-05B													0.652	0.043	-0.042	272.500	36.200	-31.400	2.726	0.626	-0.546	630.000	540.000	-283.600	902.500
15	10	DD-06A													0.668	0.041	-0.040	27.340	8.240	-5.490	2.851	0.628	-0.550	553.500	443.700	-241.400	580.900
16	11	DD-06B													0.655	0.042	-0.041	108.100	19.900	-17.230	2.752	0.629	-0.542	613.100	508.900	-275.000	721.200
17	12	DD-07A													0.653	0.039	-0.037	6.155	2.757	-0.167	2.736	0.610	-0.534	623.300	510.700	-275.400	629.500
18	13	DD-07B													0.597	0.084	-0.055	539.500	104.200	-54.200	2.314	0.946	-0.578	980.700	925.300	-606.200	1520.000
19	14	DD-08A													0.614	0.136	-0.089	714.600	183.400	-90.500	2.444	1.173	-0.777	847.600	1213.000	-572.700	1562.000
20	15	DD-08B													0.796	0.052	-0.051	849.400	113.700	-105.700	3.813	0.756	-0.645	242.700	164.000	-91.400	1092.000
21	16	DD-09A													0.542	0.051	-0.047	360.200	48.700	-42.100	1.905	0.619	-0.548	1589.000	2359.000	-820.000	1950.000
22	17	DD-09B													1.173	0.031	-0.033	893.000	111.000	-135.800	6.648	0.981	-0.822	61.910	21.610	-15.760	954.900
23	18	DD-10A													0.879	0.044	-0.045	573.600	81.600	-70.000	4.433	0.789	-0.667	163.300	87.700	-54.600	736.900
24	19	DD-10B													0.755	0.046	-0.044	450.200	61.600	-52.100	3.505	0.702	-0.603	305.700	220.300	-118.700	755.900
25	20	DD-11A													0.737	0.044	-0.043	107.300	19.500	-17.890	3.364	0.686	-0.577	342.700	249.000	-136.100	450.000
26	21	DD-11B													0.893	0.045	-0.045	489.000	70.000	-59.900	4.541	0.785	-0.668	153.700	79.000	-50.200	642.700
27	22	DD-12A													0.626	0.042	-0.040	171.600	25.800	-22.200	2.535	0.604	-0.535	770.300	699.700	-352.400	941.900
28	23	DD-12B													0.631	0.122	-0.045	448.900	144.400	-41.400	2.571	1.084	-0.536	741.100	675.900	-468.700	1190.000
29																											
30																											
31																											
32																											
33															0.791	0.031	-0.031	79.460	10.960	-10.220	3.773	0.678	-0.575	249.900	145.800	-88.200	329.300
34															1.098	0.026	-0.027	245.100	24.900	-22.400	6.082	0.908	-0.755	75.580	27.920	-20.020	320.700
35															0.537	0.005	-0.005	136.400	2.400	-2.200	1.865	0.469	-0.418	1642.000	1658.000	-685.000	1778.000
36															0.604	0.006	-0.006	242.800	6.900	-6.600	2.371	0.509	-0.442	917.400	641.600	-379.900	1160.000
37																											
38																											

	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ	BA	BB
1																											
2																											
3						**Kalirai IFMR**												**Williams IFMR**									
4	binary lifetime	binary lifetime	Delta binary lif	Delta binary lif	Delta binary lif	Mi (Myr)	Mi error max	Mi error min	Agei	Agei error max	Agei error min	binary lifetime	binary lifetime	binary lifetime	Delta binary lif	Delta binary lif	Delta binary lif	Mi (Myr)	Mi error max	Mi error min	Agei	Agei error max	Agei error min	binary lifetime	binary lifetime	binary lifetime	Delta binary lif
5	4108.000	-1410.000	-1579.000	1008.000	-3420.000	1.067	0.927	-0.483	9756.000	5604.000	-9569.000	10170.000	5610.000	-9136.000	-8547.000	9736.000	-4903.000	1.328	0.720	-0.380	4313.000	7997.000	-3119.000	4731.000	8019.000	-3033.000	-3282.000
6	732.200	-417.200	0.000	0.000	0.000	1.936	0.462	-0.434	1551.000	1391.000	-661.400	1627.000	1382.000	-649.900	0.000	0.000	0.000	2.062	0.339	-0.327	1373.000	539.000	-485.300	1450.000	529.000	-475.500	0.000
7	357.400	-180.600	163.000	338.100	-179.800	2.759	0.504	-0.472	609.100	404.900	-235.300	765.800	391.200	-218.700	327.600	396.400	-238.200	2.757	0.365	-0.356	609.900	277.500	-185.400	766.700	261.300	-166.100	309.300
8	65.900	-46.000	0.000	0.000	0.000	5.803	0.628	-0.619	84.290	26.410	-17.680	438.200	54.400	-41.200	0.000	0.000	0.000	5.330	0.398	-0.435	103.400	24.000	-16.490	457.300	51.800	-40.000	0.000
9	296.000	-127.000	-228.000	386.300	-675.200	2.967	0.568	-0.543	492.400	373.900	-193.600	1287.000	311.000	-116.000	-785.800	773.600	-2260.000	2.934	0.433	-0.417	509.200	275.700	-167.200	1304.000	201.000	-82.000	-595.400
10	887.000	-442.400	0.000	0.000	0.000	1.871	0.564	-0.532	1633.000	2472.000	-785.300	2073.000	2435.000	-732.000	0.000	0.000	0.000	2.007	0.446	-0.418	1460.000	1010.000	-620.500	1900.000	966.000	-566.000	0.000
11	314.600	-153.500	529.300	256.700	-120.900	2.870	0.516	-0.490	542.900	366.200	-206.300	852.700	342.300	-175.800	660.500	318.000	-169.300	2.852	0.377	-0.367	553.300	258.400	-168.200	863.100	232.900	-134.900	645.500
12	67.100	-41.200	0.000	0.000	0.000	4.741	0.561	-0.545	138.000	50.300	-33.300	192.100	44.600	-25.900	0.000	0.000	0.000	4.432	0.382	-0.384	163.500	43.400	-30.700	217.600	35.100	-20.900	0.000
13	101.400	-54.800	-535.500	227.800	-439.600	4.170	0.562	-0.533	191.300	84.900	-52.700	371.300	71.700	-37.200	-827.300	343.500	-607.700	3.950	0.391	-0.382	220.800	70.400	-48.500	400.900	54.000	-29.100	-738.200
14	526.500	-265.900	0.000	0.000	0.000	2.363	0.490	-0.461	926.200	665.800	-373.600	1199.000	648.000	-349.800	0.000	0.000	0.000	2.423	0.364	-0.348	866.600	486.400	-275.100	1139.000	462.000	-246.000	0.000
15	442.100	-237.600	-140.300	234.000	-269.600	2.516	0.480	-0.453	785.500	586.500	-307.000	812.900	583.100	-300.900	-188.800	469.000	-537.900	2.552	0.351	-0.332	756.100	350.900	-230.800	783.500	347.500	-224.500	-166.500
16	501.800	-265.400	0.000	0.000	0.000	2.395	0.477	-0.454	893.500	652.500	-351.500	1002.000	641.000	-338.000	0.000	0.000	0.000	2.450	0.353	-0.339	841.900	447.100	-260.500	950.000	434.000	-245.200	0.000
17	631.500	-268.200	-890.800	295.700	-503.200	2.376	0.465	-0.431	913.100	627.900	-353.800	919.200	705.800	-345.600	-1268.000	661.400	-2032.000	2.434	0.333	-0.315	856.700	419.300	-252.800	862.900	511.100	-249.000	-1149.000
18	902.000	-454.000	0.000	0.000	0.000	1.861	0.876	-0.552	1648.000	2668.000	-1050.000	2188.000	2633.000	-935.000	0.000	0.000	0.000	1.998	0.688	-0.444	1472.000	1170.000	-815.900	2012.000	1128.000	-717.000	0.000
19	1159.000	-405.000	-469.900	427.200	-1102.000	2.018	1.314	-0.853	1443.000	3032.000	-1153.000	2158.000	2986.000	-899.000	-1043.000	953.000	-2934.000	2.132	1.075	-0.699	1253.000	1888.000	-883.300	1967.000	1819.000	-637.000	-821.200
20	179.000	-92.000	0.000	0.000	0.000	3.689	0.598	-0.558	265.500	155.400	-87.400	1115.000	146.000	-62.000	0.000	0.000	0.000	3.544	0.439	-0.424	296.700	128.300	-80.700	1146.000	109.000	-48.000	0.000
21	2328.000	-792.000	994.600	2317.000	-772.000	1.361	0.534	-0.494	3996.000	8824.000	-2531.000	4356.000	8794.000	-2485.000	3410.000	8780.000	-2490.000	1.576	0.413	-0.384	2534.000	3689.000	-1056.000	2894.000	3649.000	-1015.000	1935.000
22	110.100	-128.500	0.000	0.000	0.000	7.148	0.608	-0.577	52.970	10.530	-8.410	945.900	107.100	-129.900	0.000	0.000	0.000	6.466	0.335	-0.345	65.800	8.680	-6.870	958.800	106.200	-129.700	0.000
23	105.800	-61.200	-18.990	78.570	-131.500	4.446	0.574	-0.537	162.100	65.000	-42.400	735.700	78.700	-43.000	-72.010	85.030	-153.800	4.183	0.396	-0.388	189.700	56.300	-39.200	763.300	65.000	-35.900	-72.660
24	205.700	-99.100	0.000	0.000	0.000	3.314	0.534	-0.504	357.500	219.800	-120.600	807.700	190.100	-83.400	0.000	0.000	0.000	3.227	0.392	-0.370	385.800	164.700	-105.900	836.000	130.100	-61.700	0.000
25	243.800	-126.800	-192.700	176.400	-102.700	3.142	0.516	-0.488	416.600	262.800	-144.600	523.900	251.300	-131.100	-115.800	219.100	-129.600	3.081	0.377	-0.367	441.000	196.900	-123.600	548.200	182.600	-108.400	-118.000
26	93.000	-54.100	0.000	0.000	0.000	4.577	0.577	-0.543	150.600	58.200	-38.300	639.600	68.500	-38.100	0.000	0.000	0.000	4.294	0.391	-0.386	177.300	49.900	-35.200	666.300	55.800	-30.500	0.000
27	691.100	-339.300	-248.100	243.200	-185.600	2.129	0.474	-0.446	1257.000	821.000	-540.300	1429.000	807.000	-524.200	-200.000	520.600	-417.800	2.226	0.353	-0.338	1099.000	511.000	-363.900	1270.000	496.000	-343.800	-223.700
28	655.000	-325.300	0.000	0.000	0.000	2.174	1.227	-0.472	1180.000	832.000	-848.700	1629.000	807.000	-654.800	0.000	0.000	0.000	2.263	0.992	-0.372	1045.000	561.000	-668.700	1494.000	531.000	-480.000	0.000
29																											
30																											
31																											
32																											
33	143.200	-84.600	8.638	118.400	-69.480	3.640	0.451	-0.417	275.500	111.400	-74.300	355.000	106.200	-67.600	43.910	97.090	-64.170	3.502	0.296	-0.284	306.500	82.400	-61.100	386.000	74.700	-52.400	59.270
34	34.100	-24.600	0.000	0.000	0.000	6.458	0.554	-0.506	65.990	13.460	-10.800	311.100	23.600	-18.600	0.000	0.000	0.000	5.883	0.303	-0.296	81.630	10.600	-8.940	326.700	20.500	-16.500	0.000
35	1659.000	-685.000	618.000	1026.000	-312.400	1.312	0.256	-0.245	4482.000	5218.000	-1910.000	4618.000	5218.000	-1909.000	2817.000	4600.000	-1521.000	1.535	0.136	-0.127	2749.000	853.000	-624.000	2886.000	852.000	-624.000	1261.000
36	642.000	-379.200	0.000	0.000	0.000	1.930	0.275	-0.261	1558.000	572.000	-428.000	1801.000	572.000	-427.000	0.000	0.000	0.000	2.057	0.145	-0.135	1381.000	187.000	-247.000	1624.000	186.000	-247.000	0.000
37																											
38																											

	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN	BO	BP	BQ	BR	BS	BT	BU	BV	BW	BX	BY
1				findMainSequenceCoolingTimeOrMassInGirardiModel.pro																			
2				initialMassInput.txt			initialAgeInput.txt			finalMassInput.txt			finalAgeInput.txt			systemAgeConstraints.txt			STRICTsystensystemAgeCo	systemAgeCommon.txt [no DD-04 and DD-09]			
3				Low/High Mass Extreme - With IFMR Errors																			
4	Delta binary lif	Delta binary li	etime (Myr) en	Est. MS Mass	Est. MS Mass	Est. MS Mass	Est. MS Age	Est. MS Age n	Est. MS Age m	WD Mass	WD Mass min	WD Mass max	Cooling Age A	Cooling Age m	Cooling Age m	System Age	Low Mass Ext	High Mass Ext	alignment with	alignment with	alignment with	distance	distance min
5	2551.000	-7478.000		=U5	=BF5+W5	=BF5+V5	=X5	=BI5+Z5	=BI5+Y5	=O5	=BL5+Q5	=BL5+P5	=R5	=BO5+T5	=BO5+S5	=AA5	=BR5+AC5	=BR5+AB5	=CG5	1	1	=IF(NOT(C5 = 0); C5; C6)	=IF(NOT(E5 = 0); E5; E6)
6	0.000	0.000		=U6	=BF6+W6	=BF6+V6	=X6	=BI6+Z6	=BI6+Y6	=O6	=BL6+Q6	=BL6+P6	=R6	=BO6+T6	=BO6+S6	=AA6	=BR6+AC6	=BR6+AB6	=BU5	1	=BW5	=IF(NOT(C6 = 0); C6; C5)	=IF(NOT(E6 = 0); E6; E5)
7	273.600	-190.800		=U7	=BF7+W7	=BF7+V7	=X7	=BI7+Z7	=BI7+Y7	=O7	=BL7+Q7	=BL7+P7	=R7	=BO7+T7	=BO7+S7	=AA7	=BR7+AC7	=BR7+AB7	=CG7	0	1	=IF(NOT(C7 = 0); C7; C8)	=IF(NOT(E7 = 0); E7; E8)
8	0.000	0.000		=U8	=BF8+W8	=BF8+V8	=X8	=BI8+Z8	=BI8+Y8	=O8	=BL8+Q8	=BL8+P8	=R8	=BO8+T8	=BO8+S8	=AA8	=BR8+AC8	=BR8+AB8	=BU7	0	=BW7	=IF(NOT(C8 = 0); C8; C7)	=IF(NOT(E8 = 0); E8; E7)
9	607.400	-893.600		=U9	=BF9+W9	=BF9+V9	=X9	=BI9+Z9	=BI9+Y9	=O9	=BL9+Q9	=BL9+P9	=R9	=BO9+T9	=BO9+S9	=AA9	=BR9+AC9	=BR9+AB9	=CG9	1	1	=IF(NOT(C9 = 0); C9; C10)	=IF(NOT(E9 = 0); E9; E10)
10	0.000	0.000		=U10	=BF10+W10	=BF10+V10	=X10	=BI10+Z10	=BI10+Y10	=O10	=BL10+Q10	=BL10+P10	=R10	=BO10+T10	=BO10+S10	=AA10	=BR10+AC10	=BR10+AB10	=BU9	1	=BW9	=IF(NOT(C10 = 0); C10; C9)	=IF(NOT(E10 = 0); E10; E9)
11	221.000	-135.900		=U11	=BF11+W11	=BF11+V11	=X11	=BI11+Z11	=BI11+Y11	=O11	=BL11+Q11	=BL11+P11	=R11	=BO11+T11	=BO11+S11	=AA11	=BR11+AC11	=BR11+AB11	=CG11	0	0	=IF(NOT(C11 = 0); C11; C12)	=IF(NOT(E11 = 0); E11; E12)
12	0.000	0.000		=U12	=BF12+W12	=BF12+V12	=X12	=BI12+Z12	=BI12+Y12	=O12	=BL12+Q12	=BL12+P12	=R12	=BO12+T12	=BO12+S12	=AA12	=BR12+AC12	=BR12+AB12	=BU11	0	=BW11	=IF(NOT(C12 = 0); C12; C11)	=IF(NOT(E12 = 0); E12; E11)
13	252.500	-439.800		=U13	=BF13+W13	=BF13+V13	=X13	=BI13+Z13	=BI13+Y13	=O13	=BL13+Q13	=BL13+P13	=R13	=BO13+T13	=BO13+S13	=AA13	=BR13+AC13	=BR13+AB13	=CG13	1	1	=IF(NOT(C13 = 0); C13; C14)	=IF(NOT(E13 = 0); E13; E14)
14	0.000	0.000		=U14	=BF14+W14	=BF14+V14	=X14	=BI14+Z14	=BI14+Y14	=O14	=BL14+Q14	=BL14+P14	=R14	=BO14+T14	=BO14+S14	=AA14	=BR14+AC14	=BR14+AB14	=BU13	1	=BW13	=IF(NOT(C14 = 0); C14; C13)	=IF(NOT(E14 = 0); E14; E13)
15	368.100	-413.800		=U15	=BF15+W15	=BF15+V15	=X15	=BI15+Z15	=BI15+Y15	=O15	=BL15+Q15	=BL15+P15	=R15	=BO15+T15	=BO15+S15	=AA15	=BR15+AC15	=BR15+AB15	=CG15	1	1	=IF(NOT(C15 = 0); C15; C16)	=IF(NOT(E15 = 0); E15; E16)
16	0.000	0.000		=U16	=BF16+W16	=BF16+V16	=X16	=BI16+Z16	=BI16+Y16	=O16	=BL16+Q16	=BL16+P16	=R16	=BO16+T16	=BO16+S16	=AA16	=BR16+AC16	=BR16+AB16	=BU15	1	=BW15	=IF(NOT(C16 = 0); C16; C15)	=IF(NOT(E16 = 0); E16; E15)
17	543.700	-727.000		=U17	=BF17+W17	=BF17+V17	=X17	=BI17+Z17	=BI17+Y17	=O17	=BL17+Q17	=BL17+P17	=R17	=BO17+T17	=BO17+S17	=AA17	=BR17+AC17	=BR17+AB17	=CG17	1	1	=IF(NOT(C17 = 0); C17; C18)	=IF(NOT(E17 = 0); E17; E18)
18	0.000	0.000		=U18	=BF18+W18	=BF18+V18	=X18	=BI18+Z18	=BI18+Y18	=O18	=BL18+Q18	=BL18+P18	=R18	=BO18+T18	=BO18+S18	=AA18	=BR18+AC18	=BR18+AB18	=BU17	1	=BW17	=IF(NOT(C18 = 0); C18; C17)	=IF(NOT(E18 = 0); E18; E17)
19	692.400	-1797.000		=U19	=BF19+W19	=BF19+V19	=X19	=BI19+Z19	=BI19+Y19	=O19	=BL19+Q19	=BL19+P19	=R19	=BO19+T19	=BO19+S19	=AA19	=BR19+AC19	=BR19+AB19	=CG19	0	1	=IF(NOT(C19 = 0); C19; C20)	=IF(NOT(E19 = 0); E19; E20)
20	0.000	0.000		=U20	=BF20+W20	=BF20+V20	=X20	=BI20+Z20	=BI20+Y20	=O20	=BL20+Q20	=BL20+P20	=R20	=BO20+T20	=BO20+S20	=AA20	=BR20+AC20	=BR20+AB20	=BU19	0	=BW19	=IF(NOT(C20 = 0); C20; C19)	=IF(NOT(E20 = 0); E20; E19)
21	3662.000	-1022.000		=U21	=BF21+W21	=BF21+V21	=X21	=BI21+Z21	=BI21+Y21	=O21	=BL21+Q21	=BL21+P21	=R21	=BO21+T21	=BO21+S21	=AA21	=BR21+AC21	=BR21+AB21	=CG21	0	0	=IF(NOT(C21 = 0); C21; C22)	=IF(NOT(E21 = 0); E21; E22)
22	0.000	0.000		=U22	=BF22+W22	=BF22+V22	=X22	=BI22+Z22	=BI22+Y22	=O22	=BL22+Q22	=BL22+P22	=R22	=BO22+T22	=BO22+S22	=AA22	=BR22+AC22	=BR22+AB22	=BU21	0	=BW21	=IF(NOT(C22 = 0); C22; C21)	=IF(NOT(E22 = 0); E22; E21)
23	77.230	-115.000		=U23	=BF23+W23	=BF23+V23	=X23	=BI23+Z23	=BI23+Y23	=O23	=BL23+Q23	=BL23+P23	=R23	=BO23+T23	=BO23+S23	=AA23	=BR23+AC23	=BR23+AB23	=CG23	1	1	=IF(NOT(C23 = 0); C23; C24)	=IF(NOT(E23 = 0); E23; E24)
24	0.000	0.000		=U24	=BF24+W24	=BF24+V24	=X24	=BI24+Z24	=BI24+Y24	=O24	=BL24+Q24	=BL24+P24	=R24	=BO24+T24	=BO24+S24	=AA24	=BR24+AC24	=BR24+AB24	=BU23	1	=BW23	=IF(NOT(C24 = 0); C24; C23)	=IF(NOT(E24 = 0); E24; E23)
25	166.500	-116.200		=U25	=BF25+W25	=BF25+V25	=X25	=BI25+Z25	=BI25+Y25	=O25	=BL25+Q25	=BL25+P25	=R25	=BO25+T25	=BO25+S25	=AA25	=BR25+AC25	=BR25+AB25	=CG25	1	1	=IF(NOT(C25 = 0); C25; C26)	=IF(NOT(E25 = 0); E25; E26)
26	0.000	0.000		=U26	=BF26+W26	=BF26+V26	=X26	=BI26+Z26	=BI26+Y26	=O26	=BL26+Q26	=BL26+P26	=R26	=BO26+T26	=BO26+S26	=AA26	=BR26+AC26	=BR26+AB26	=BU25	1	=BW25	=IF(NOT(C26 = 0); C26; C25)	=IF(NOT(E26 = 0); E26; E25)
27	372.400	-291.700		=U27	=BF27+W27	=BF27+V27	=X27	=BI27+Z27	=BI27+Y27	=O27	=BL27+Q27	=BL27+P27	=R27	=BO27+T27	=BO27+S27	=AA27	=BR27+AC27	=BR27+AB27	=CG27	1	1	=IF(NOT(C27 = 0); C27; C28)	=IF(NOT(E27 = 0); E27; E28)
28	0.000	0.000		=U28	=BF28+W28	=BF28+V28	=X28	=BI28+Z28	=BI28+Y28	=O28	=BL28+Q28	=BL28+P28	=R28	=BO28+T28	=BO28+S28	=AA28	=BR28+AC28	=BR28+AB28	=BU27	1	=BW27	=IF(NOT(C28 = 0); C28; C27)	=IF(NOT(E28 = 0); E28; E27)
29																							
30																							
31																							
32																							
33	71.230	-52.910		=U33	=BF33+W33	=BF33+V33	=X33	=BI33+Z33	=BI33+Y33	=O33	=BL33+Q33	=BL33+P33	=R33	=BO33+T33	=BO33+S33	=AA33	=BR33+AC33	=BR33+AB33	=CG33	1	1	=C33	=E33
34	0.000	0.000		=U34	=BF34+W34	=BF34+V34	=X34	=BI34+Z34	=BI34+Y34	=O34	=BL34+Q34	=BL34+P34	=R34	=BO34+T34	=BO34+S34	=AA34	=BR34+AC34	=BR34+AB34	=BU33	1	=BW33	=C34	=E34
35	689.000	-416.500		=U35	=BF35+W35	=BF35+V35	=X35	=BI35+Z35	=BI35+Y35	=O35	=BL35+Q35	=BL35+P35	=R35	=BO35+T35	=BO35+S35	=AA35	=BR35+AC35	=BR35+AB35	=CG35	1	1	=C35	=E35
36	0.000	0.000		=U36	=BF36+W36	=BF36+V36	=X36	=BI36+Z36	=BI36+Y36	=O36	=BL36+Q36	=BL36+P36	=R36	=BO36+T36	=BO36+S36	=AA36	=BR36+AC36	=BR36+AB36	=BU35	1	=BW35	=C36	=E36
37																						=C37	=E37
38																							

	BZ	CA	CB	CC	CD	CE	CF	CG	CH	CI	CJ	CK	CL	CM	CN	CO	CP	CQ
1			SDSSDDsystemAgeConfirmation [plotToFileSystemAges.pro]															
2			DDsystemAges0.txt [Dobbie]						DDsystemAges1.txt [Kalirai]						DDsystemAges2.txt [Williams]			
3																		
4	distance max		DD_NUMBER	System Age (N	Low Mass Extr	High Mass Ext	Special Case	alignment with IFMR? (with errors)	DD_NUMBER	System Age (N	Low Mass Extr	High Mass Ext	Special Case	alignment with IFMR? (with errors)	DD_NUMBER	System Age (N	Low Mass Extr	High Mass Ext
5	=IF(NOT(D5 = 0); D5; D6)		=SDSS.N5	=AA5	=CC5+AC5	=CC5+AB5	=fin.B5	=IF(AD5<0; IF(AD5+AE5>0; 1; 0); IF(AD5+AF5<0; 1; 0))	=SDSS.N5	=AM5	=CI5+AO5	=CI5+AN5	=fin.B5	=IF(AP5<0; IF(AP5+AQ5>0; 1; 0); IF(AP5+AR5<0; 1; 0))	=SDSS.N5	=AY5	=CO5+BA5	=CO5+AZ5
6	=IF(NOT(D6 = 0); D6; D5)		=SDSS.N6	=AA6	=CC6+AC6	=CC6+AB6	=fin.B6		=SDSS.N6	=AM6	=CI6+AO6	=CI6+AN6	=fin.B6		=SDSS.N6	=AY6	=CO6+BA6	=CO6+AZ6
7	=IF(NOT(D7 = 0); D7; D8)		=SDSS.N7	=AA7	=CC7+AC7	=CC7+AB7	=fin.B7	=IF(AD7<0; IF(AD7+AE7>0; 1; 0); IF(AD7+AF7<0; 1; 0))	=SDSS.N7	=AM7	=CI7+AO7	=CI7+AN7	=fin.B7	=IF(AP7<0; IF(AP7+AQ7>0; 1; 0); IF(AP7+AR7<0; 1; 0))	=SDSS.N7	=AY7	=CO7+BA7	=CO7+AZ7
8	=IF(NOT(D8 = 0); D8; D7)		=SDSS.N8	=AA8	=CC8+AC8	=CC8+AB8	=fin.B8		=SDSS.N8	=AM8	=CI8+AO8	=CI8+AN8	=fin.B8		=SDSS.N8	=AY8	=CO8+BA8	=CO8+AZ8
9	=IF(NOT(D9 = 0); D9; D10)		=SDSS.N9	=AA9	=CC9+AC9	=CC9+AB9	=fin.B9	=IF(AD9<0; IF(AD9+AE9>0; 1; 0); IF(AD9+AF9<0; 1; 0))	=SDSS.N9	=AM9	=CI9+AO9	=CI9+AN9	=fin.B9	=IF(AP9<0; IF(AP9+AQ9>0; 1; 0); IF(AP9+AR9<0; 1; 0))	=SDSS.N9	=AY9	=CO9+BA9	=CO9+AZ9
10	=IF(NOT(D10 = 0); D10; D9)		=SDSS.N10	=AA10	=CC10+AC10	=CC10+AB10	=fin.B10		=SDSS.N10	=AM10	=CI10+AO10	=CI10+AN10	=fin.B10		=SDSS.N10	=AY10	=CO10+BA10	=CO10+AZ10
11	=IF(NOT(D11 = 0); D11; D12)		=SDSS.N11	=AA11	=CC11+AC11	=CC11+AB11	=fin.B11	=IF(AD11<0; IF(AD11+AE11>0; 1; 0); IF(AD11+AF11<0; 1; 0))	=SDSS.N11	=AM11	=CI11+AO11	=CI11+AN11	=fin.B11	=IF(AP11<0; IF(AP11+AQ11>0; 1; 0); IF(AP11+AR11<0; 1; 0))	=SDSS.N11	=AY11	=CO11+BA11	=CO11+AZ11
12	=IF(NOT(D12 = 0); D12; D11)		=SDSS.N12	=AA12	=CC12+AC12	=CC12+AB12	=fin.B12		=SDSS.N12	=AM12	=CI12+AO12	=CI12+AN12	=fin.B12		=SDSS.N12	=AY12	=CO12+BA12	=CO12+AZ12
13	=IF(NOT(D13 = 0); D13; D14)		=SDSS.N13	=AA13	=CC13+AC13	=CC13+AB13	=fin.B13	=IF(AD13<0; IF(AD13+AE13>0; 1; 0); IF(AD13+AF13<0; 1; 0))	=SDSS.N13	=AM13	=CI13+AO13	=CI13+AN13	=fin.B13	=IF(AP13<0; IF(AP13+AQ13>0; 1; 0); IF(AP13+AR13<0; 1; 0))	=SDSS.N13	=AY13	=CO13+BA13	=CO13+AZ13
14	=IF(NOT(D14 = 0); D14; D13)		=SDSS.N14	=AA14	=CC14+AC14	=CC14+AB14	=fin.B14		=SDSS.N14	=AM14	=CI14+AO14	=CI14+AN14	=fin.B14		=SDSS.N14	=AY14	=CO14+BA14	=CO14+AZ14
15	=IF(NOT(D15 = 0); D15; D16)		=SDSS.N15	=AA15	=CC15+AC15	=CC15+AB15	=fin.B15	=IF(AD15<0; IF(AD15+AE15>0; 1; 0); IF(AD15+AF15<0; 1; 0))	=SDSS.N15	=AM15	=CI15+AO15	=CI15+AN15	=fin.B15	=IF(AP15<0; IF(AP15+AQ15>0; 1; 0); IF(AP15+AR15<0; 1; 0))	=SDSS.N15	=AY15	=CO15+BA15	=CO15+AZ15
16	=IF(NOT(D16 = 0); D16; D15)		=SDSS.N16	=AA16	=CC16+AC16	=CC16+AB16	=fin.B16		=SDSS.N16	=AM16	=CI16+AO16	=CI16+AN16	=fin.B16		=SDSS.N16	=AY16	=CO16+BA16	=CO16+AZ16
17	=IF(NOT(D17 = 0); D17; D18)		=SDSS.N17	=AA17	=CC17+AC17	=CC17+AB17	=fin.B17	=IF(AD17<0; IF(AD17+AE17>0; 1; 0); IF(AD17+AF17<0; 1; 0))	=SDSS.N17	=AM17	=CI17+AO17	=CI17+AN17	=fin.B17	=IF(AP17<0; IF(AP17+AQ17>0; 1; 0); IF(AP17+AR17<0; 1; 0))	=SDSS.N17	=AY17	=CO17+BA17	=CO17+AZ17
18	=IF(NOT(D18 = 0); D18; D17)		=SDSS.N18	=AA18	=CC18+AC18	=CC18+AB18	=fin.B18		=SDSS.N18	=AM18	=CI18+AO18	=CI18+AN18	=fin.B18		=SDSS.N18	=AY18	=CO18+BA18	=CO18+AZ18
19	=IF(NOT(D19 = 0); D19; D20)		=SDSS.N19	=AA19	=CC19+AC19	=CC19+AB19	=fin.B19	=IF(AD19<0; IF(AD19+AE19>0; 1; 0); IF(AD19+AF19<0; 1; 0))	=SDSS.N19	=AM19	=CI19+AO19	=CI19+AN19	=fin.B19	=IF(AP19<0; IF(AP19+AQ19>0; 1; 0); IF(AP19+AR19<0; 1; 0))	=SDSS.N19	=AY19	=CO19+BA19	=CO19+AZ19
20	=IF(NOT(D20 = 0); D20; D19)		=SDSS.N20	=AA20	=CC20+AC20	=CC20+AB20	=fin.B20		=SDSS.N20	=AM20	=CI20+AO20	=CI20+AN20	=fin.B20		=SDSS.N20	=AY20	=CO20+BA20	=CO20+AZ20
21	=IF(NOT(D21 = 0); D21; D22)		=SDSS.N21	=AA21	=CC21+AC21	=CC21+AB21	=fin.B21	=IF(AD21<0; IF(AD21+AE21>0; 1; 0); IF(AD21+AF21<0; 1; 0))	=SDSS.N21	=AM21	=CI21+AO21	=CI21+AN21	=fin.B21	=IF(AP21<0; IF(AP21+AQ21>0; 1; 0); IF(AP21+AR21<0; 1; 0))	=SDSS.N21	=AY21	=CO21+BA21	=CO21+AZ21
22	=IF(NOT(D22 = 0); D22; D21)		=SDSS.N22	=AA22	=CC22+AC22	=CC22+AB22	=fin.B22		=SDSS.N22	=AM22	=CI22+AO22	=CI22+AN22	=fin.B22		=SDSS.N22	=AY22	=CO22+BA22	=CO22+AZ22
23	=IF(NOT(D23 = 0); D23; D24)		=SDSS.N23	=AA23	=CC23+AC23	=CC23+AB23	=fin.B23	=IF(AD23<0; IF(AD23+AE23>0; 1; 0); IF(AD23+AF23<0; 1; 0))	=SDSS.N23	=AM23	=CI23+AO23	=CI23+AN23	=fin.B23	=IF(AP23<0; IF(AP23+AQ23>0; 1; 0); IF(AP23+AR23<0; 1; 0))	=SDSS.N23	=AY23	=CO23+BA23	=CO23+AZ23
24	=IF(NOT(D24 = 0); D24; D23)		=SDSS.N24	=AA24	=CC24+AC24	=CC24+AB24	=fin.B24		=SDSS.N24	=AM24	=CI24+AO24	=CI24+AN24	=fin.B24		=SDSS.N24	=AY24	=CO24+BA24	=CO24+AZ24
25	=IF(NOT(D25 = 0); D25; D26)		=SDSS.N25	=AA25	=CC25+AC25	=CC25+AB25	=fin.B25	=IF(AD25<0; IF(AD25+AE25>0; 1; 0); IF(AD25+AF25<0; 1; 0))	=SDSS.N25	=AM25	=CI25+AO25	=CI25+AN25	=fin.B25	=IF(AP25<0; IF(AP25+AQ25>0; 1; 0); IF(AP25+AR25<0; 1; 0))	=SDSS.N25	=AY25	=CO25+BA25	=CO25+AZ25
26	=IF(NOT(D26 = 0); D26; D25)		=SDSS.N26	=AA26	=CC26+AC26	=CC26+AB26	=fin.B26		=SDSS.N26	=AM26	=CI26+AO26	=CI26+AN26	=fin.B26		=SDSS.N26	=AY26	=CO26+BA26	=CO26+AZ26
27	=IF(NOT(D27 = 0); D27; D28)		=SDSS.N27	=AA27	=CC27+AC27	=CC27+AB27	=fin.B27	=IF(AD27<0; IF(AD27+AE27>0; 1; 0); IF(AD27+AF27<0; 1; 0))	=SDSS.N27	=AM27	=CI27+AO27	=CI27+AN27	=fin.B27	=IF(AP27<0; IF(AP27+AQ27>0; 1; 0); IF(AP27+AR27<0; 1; 0))	=SDSS.N27	=AY27	=CO27+BA27	=CO27+AZ27
28	=IF(NOT(D28 = 0); D28; D27)		=SDSS.N28	=AA28	=CC28+AC28	=CC28+AB28	=fin.B28		=SDSS.N28	=AM28	=CI28+AO28	=CI28+AN28	=fin.B28		=SDSS.N28	=AY28	=CO28+BA28	=CO28+AZ28
29																		
30																		
31																		
32																		
33	=D33		=SDSS.N33	=AA33	=CC33+AC33	=CC33+AB33	=fin.B33	=IF(AD33<0; IF(AD33+AE33>0; 1; 0); IF(AD33+AF33<0; 1; 0))	=SDSS.N33	=AM33	=CI33+AO33	=CI33+AN33	=fin.B33	=IF(AP33<0; IF(AP33+AQ33>0; 1; 0); IF(AP33+AR33<0; 1; 0))	=SDSS.N33	=AY33	=CO33+BA33	=CO33+AZ33
34	=D34		=SDSS.N34	=AA34	=CC34+AC34	=CC34+AB34	=fin.B34		=SDSS.N34	=AM34	=CI34+AO34	=CI34+AN34	=fin.B34		=SDSS.N34	=AY34	=CO34+BA34	=CO34+AZ34
35	=D35		=SDSS.N35	=AA35	=CC35+AC35	=CC35+AB35	=fin.B35	=IF(AD35<0; IF(AD35+AE35>0; 1; 0); IF(AD35+AF35<0; 1; 0))	=SDSS.N35	=AM35	=CI35+AO35	=CI35+AN35	=fin.B35	=IF(AP35<0; IF(AP35+AQ35>0; 1; 0); IF(AP35+AR35<0; 1; 0))	=SDSS.N35	=AY35	=CO35+BA35	=CO35+AZ35
36	=D36		=SDSS.N36	=AA36	=CC36+AC36	=CC36+AB36	=fin.B36		=SDSS.N36	=AM36	=CI36+AO36	=CI36+AN36	=fin.B36		=SDSS.N36	=AY36	=CO36+BA36	=CO36+AZ36
37	=D37																	
38								=SUM(CG5:CG36)						=SUM(CM5:CM36)				

	CR	CS
1		
2		
3		
4	Special Case alignment with IFMR? (with errors)	
5	=fin.B5	=IF(BB5<0; IF(BB5+BC5>0; 1; 0); IF(BB5+BD5<0; 1; 0))
6	=fin.B6	
7	=fin.B7	=IF(BB7<0; IF(BB7+BC7>0; 1; 0); IF(BB7+BD7<0; 1; 0))
8	=fin.B8	
9	=fin.B9	=IF(BB9<0; IF(BB9+BC9>0; 1; 0); IF(BB9+BD9<0; 1; 0))
10	=fin.B10	
11	=fin.B11	=IF(BB11<0; IF(BB11+BC11>0; 1; 0); IF(BB11+BD11<0; 1; 0))
12	=fin.B12	
13	=fin.B13	=IF(BB13<0; IF(BB13+BC13>0; 1; 0); IF(BB13+BD13<0; 1; 0))
14	=fin.B14	
15	=fin.B15	=IF(BB15<0; IF(BB15+BC15>0; 1; 0); IF(BB15+BD15<0; 1; 0))
16	=fin.B16	
17	=fin.B17	=IF(BB17<0; IF(BB17+BC17>0; 1; 0); IF(BB17+BD17<0; 1; 0))
18	=fin.B18	
19	=fin.B19	=IF(BB19<0; IF(BB19+BC19>0; 1; 0); IF(BB19+BD19<0; 1; 0))
20	=fin.B20	
21	=fin.B21	=IF(BB21<0; IF(BB21+BC21>0; 1; 0); IF(BB21+BD21<0; 1; 0))
22	=fin.B22	
23	=fin.B23	=IF(BB23<0; IF(BB23+BC23>0; 1; 0); IF(BB23+BD23<0; 1; 0))
24	=fin.B24	
25	=fin.B25	=IF(BB25<0; IF(BB25+BC25>0; 1; 0); IF(BB25+BD25<0; 1; 0))
26	=fin.B26	
27	=fin.B27	=IF(BB27<0; IF(BB27+BC27>0; 1; 0); IF(BB27+BD27<0; 1; 0))
28	=fin.B28	
29		
30		
31		
32		
33	=fin.B33	=IF(BB33<0; IF(BB33+BC33>0; 1; 0); IF(BB33+BD33<0; 1; 0))
34	=fin.B34	
35	=fin.B35	=IF(BB35<0; IF(BB35+BC35>0; 1; 0); IF(BB35+BD35<0; 1; 0))
36	=fin.B36	
37		
38		=SUM(CS5:CS36)

A.7.2 Target Selection Mass Bias Confirmation

estimateTargetSelectionMassBias.ods

Estimate of SDSS DD Target Selection Mass Bias

Richard Baxter

07/05/10

References;

IFMR Theoretical Mass values; Casewell+Dobbie 'High Resolution optical spectroscopy of Praesepe White Dwarf' 2009 Paper;
Theoretical Radius values; WhiteDwarf_mass-radius.jpg - only used for verification of calculations, and not calculations themselves
Abs Mag values from Bergeron Table_Mass_0.6 and Table_Mass_0.9 at given cooling ages
Temperature values for a given cooling age derived from 'theoretical WD cooling models 2b.ods' (Bergeron Table_Mass_0.6 and Table_Mass_0.9) - only used for verification of calculations, and not calculations themselves
Binary Separation Limit; 'ARE BINARY SEPARATIONS RELATED TO THEIR SYSTEM MASS?' - Michael F. Sterzik1 and Richard H. Durisen - 2004 - only used for verification of calculations, and not calculations themselves
M. F. Sterzik and R. H. Durisen: Dynamical interactions and brown dwarfs

initial mass	est. final mass	est. orbital separation (multiplication	example Radi	example Abs	example Absolute Magnitude at a given cooling age (0.9=15K and 0.6=10K respectively)
0.5	= 0.1197 * A16 + 0.3569	=(B16 / A16)/(((2*B16)/A16) - 1)			
1	= 0.1197 * A17 + 0.3569	=(B17 / A17)/(((2*B17)/A17) - 1)			
1.5	= 0.1197 * A18 + 0.3569	=(B18 / A18)/(((2*B18)/A18) - 1)			
2	= 0.1197 * A19 + 0.3569	=(B19 / A19)/(((2*B19)/A19) - 1)	8700	11.158	12.205
2.5	= 0.1197 * A20 + 0.3569	=(B20 / A20)/(((2*B20)/A20) - 1)			
3	= 0.1197 * A21 + 0.3569	=(B21 / A21)/(((2*B21)/A21) - 1)			
3.5	= 0.1197 * A22 + 0.3569	=(B22 / A22)/(((2*B22)/A22) - 1)			
4	= 0.1197 * A23 + 0.3569	=(B23 / A23)/(((2*B23)/A23) - 1)			
4.5	= 0.1197 * A24 + 0.3569	=(B24 / A24)/(((2*B24)/A24) - 1)	6300	11.337	11.933
5	= 0.1197 * A25 + 0.3569	=(B25 / A25)/(((2*B25)/A25) - 1)			
5.5	= 0.1197 * A26 + 0.3569	=(B26 / A26)/(((2*B26)/A26) - 1)			
6	= 0.1197 * A27 + 0.3569	=(B27 / A27)/(((2*B27)/A27) - 1)			
6.5	= 0.1197 * A28 + 0.3569	=(B28 / A28)/(((2*B28)/A28) - 1)			
7	= 0.1197 * A29 + 0.3569	=(B29 / A29)/(((2*B29)/A29) - 1)			
7.5	= 0.1197 * A30 + 0.3569	=(B30 / A30)/(((2*B30)/A30) - 1)			
8	= 0.1197 * A31 + 0.3569	=(B31 / A31)/(((2*B31)/A31) - 1)			

IFMR used;
Final mass = 0.1197 * InitialMass + 0.3569

0.6Msol to 0.9Msol WD system comparison:

Fractional Separation Difference (single WD system)	=C19/C24	
Fractional Separation Difference (binary WD system [DD])	=(C19*C19)/(C24*C24)	
Fractional Radius Difference (for a given temperature)	=D24/D19	
Fractional Temperature Difference (for a given cooling age) ~1.5		
Fractional Absolute Mag Difference (for a given temperature)	=100 ^ ((E24-E19)/5)	=100 ^ ((F24-F19)/5)
Fractional Absolute Mag Difference (for a given cooling age) ~1.0		

dependency #1. higher mass binary system gives larger separation after AGB phase, giving higher separations on average for higher mass DD systems (making them easier to detect);
So for example a 0.9 Msol WD would have undergone a (0.74 / 0.33) = 2.23x separation increase over a 0.6Msol WD over the course of it's life time
So the probability of observing a visible but gravitationally bound DD system with a 0.9Msol WD [and an 0.x Msol companion] versus a DD system with a 0.6Msol WD [and an 0.x Msol companion] would be ~2.23^3 = x11.08 higher [(4/3 * PI * 2.23^3) / (4/3 * PI * 1^3)] (NB volumetric)
[NB a dual 0.9 Msol DD system would have undergone a (0.74 * 0.74) / (0.33 * 0.33) = 5x separation increase over a dual 0.6Msol DD system over the course of it's life time, and probability of observing would be ~5^3 = 125x higher]
dependency #2. assuming both high and low mass WDs have different brightness at a given distance; high mass WDs have smaller radii than lower mass WDs (WDs have a strong mass-radii dependence) [which if considered in isolation would lower this probability]
dependency #3. assuming both high and low mass WDs cool at different rates; low mass WDs cool much faster to 10000K than high mass WDs [which if considered in isolation would lower this probability]
NB the temp diff is max ~5000K between 0.6Msol and 0.9Msol for any given cooling age, so for for example a 20000K 0.9Msol WD has abs mag (g) of 11.337, and a a 15000K 0.6Msol WD has abs mag (g) of 11.158 [Fractional Absolute Mag Diff = 1.18]
NB the temp diff is max ~5000K between 0.6Msol and 0.9Msol for any given cooling age, so for for example a 15000K 0.9Msol WD has abs mag (g) of 11.933, and a a 10000K 0.6Msol WD has abs mag (g) of 12.205 [Fractional Absolute Mag Diff = 0.78]
so the fractional flux difference (due to radii and temperature differences) is approximately equal to ~1.0 for WDs 0.6Msol and 0.9Msol at a given cooling age, such that combined effect of both dependency #2. and dependency #3. can be considered negligible compared to dependency #1. (~x1.0
dependency #4. The average binary separation difference between the progenitors of high and low mass WD, with higher mass binary systems having larger separations to begin with

Note This theory also assumes a limited absolute binary separation and our SDSS sample is able to identify DD systems to the limit of this allowable separation. Ie, it assumes our ability to identify binaries pairs within the SDSS is not dependent upon database query capabilities or sample contamination due to overlap with field WDs (preventing the identification of true

=C24*C24

Max (0.9Msol WD) fractional change in separation (from main sequence to WD stage) =

Assume 30 arc seconds max separation; 1 parsecs away, 30AU separation
Assume 30 arc seconds max separation; 100 parsecs away, 3000AU separation
Assume 30 arc seconds max separation; 33 parsecs away, 1000AU separation

Max stellar binary separation (AU) [~95%] [for ~B/A star] 130
Max DD binary separation (AU) =H64/H57

So by applying a 30arc second separation restriction we are sampling 90% of all DD systems with visible (bright enough) individual components in the SDSS > 33 parsecs away from earth [we are not unable to detect very high separation high mass DD systems]

[we are able to detect all high separation low mass DD systems - as expected in theory]
[we are able to detect all high separation high mass DD systems - as expected in theory]

Min (0.6Msol WD) fractional change in separation (from main sequence to WD stage) = =C19*C19

Assume 3 arc seconds min separation; 1 parsecs away, 3AU separation
Assume 3 arc seconds min separation; 6 parsecs away, 18AU separation
Assume 3 arc seconds min separation; 33 parsecs away, 100AU (~91.67) separation
Assume 3 arc seconds min separation; 1000 parsecs away, 3000AU separation

Min stellar binary Separation (AU) [~95%] [for ~A/F star] 10
Min DD binary Separation (AU) [~95%] =H79/H72
Mean stellar binary Separation (AU) 50
Mean DD binary Separation (AU) =H81/H72

So by being limited to approx a min 3 arc second separation (resolution) restriction we are only able to individually distinguish 95% of all DD systems with visible (bright enough) components in the SDSS @ 6 parsecs away from earth
So by being limited to approx a min 3 arc second separation (resolution) restriction we are only able to individually distinguish ~50% of all DD systems with visible (bright enough) components in the SDSS @ 33 parsecs away from earth
So by being limited to approx a min 3 arc second separation (resolution) restriction we are only able to individually distinguish ~50% of all DD systems with visible (bright enough) components in the SDSS @ 33 parsecs away from earth

[we are unable to detect low separation low mass DD systems - as expected in theory]
[we are unable to detect low separation high mass DD systems - as expected in theory]

A.7.3 Target Selection Basic WD Population Synthesis

RBB Basic WD Population Synthesis Model B.ods

A.7.4 Private Projects

RBB IFMR private project 1

Aim: Derive empirical WD Cooling Model using open cluster data
empirical WD cooling models using open cluster WD masses.ods
theoretical WD cooling models.ods
Results suggest this is not possible due to current errors in data.

RBB IFMR private project 2

Aim: use the SDSS to calculate a mass distribution of main sequence stars, model, then compare with this with SDSS WD survey. Compare theoretical WD mass distribution (based upon observed main sequence mass distribution and currently defined IFMR) and observed WD mass distribution.
main sequence luminosity function.ods
Results suggest this is not possible due to current error in data.

RBB IFMR private project 3

Aim: use the Initial Mass Function (IMF) to calculate a mass distribution of main sequence stars, model, then compare with this with SDSS WD survey
mass function derivation.ods
Results suggest this is not possible due to current error in data.

RBB IFMR private project 4

Aim: compare theoretical double degenerate mass distribution (based upon observed WD mass distribution) and observed double degenerate mass distribution
EstimateIFMRusingSDSSDDMassDistributionBias.ods

A.7.5 WHT Observing Proposals

A.7.6 WHT 2008A Proposal

Particle Physics and Astronomy Research Council
Polaris House, North Star Avenue, Swindon, SN2 1SZ
Telephone 01793 442000 Fax 01793 442002

PATT2
Version 02/2003

APPLICATION FOR TELESCOPE TIME (OPTICAL AND INFRARED)

1 TELESCOPE *(AAT, UKST, WHT, INT or UKIRT)*	WHT	Reference:	Date Stamp:
2 SEMESTER	2008A	3 SCIENTIFIC CATEGORY	3
4 COORDINATED PATT PROPOSALS	*AAT:*☐ *UKST:*☐ *WHT:*☐ *INT:*☐ *UKIRT:*☐ *JCMT:*☐ *GEMINI:*☐ *LT:*☐ *MERLIN:*☐		

5 PRINCIPAL APPLICANT

Surname:	Dobbie	Title:	Dr	Initials:	P.D.
Post held:	Research Astronomer				
Address:	Anglo-Australian Observatories, Epping, NSW 1710				
Telephone:		Fax:			
E-mail:	pdd@aao.gov.au	Is the applicant a possible observer?	Yes		

6 COLLABORATORS

Name:	Institute:	Observer?
M.R. Burleigh	University of Leicester	Yes

7 SHORT TITLE OF PROPOSAL *(maximum 12 words)*

Constraining the masses of cool white dwarfs with SDSS candidate wide double degenerates

8 SUMMARY OF PROPOSED OBSERVATIONS

We propose to obtain ISIS spectroscopy of the components of a number of spatially resolved candidate WD binary systems where one component has $T_{eff}>12500$K and the other $T_{eff}\leq 12500$K. We will use these data to 1) confirm that these objects are WDs, 2) place stringent constraints on the effective temperatures and (spectroscopic) surface gravities of both members of each system and 3) measure the wavelength shift of the H-α line core to determine the system line of sight velocity. Subsequently, using gravitational redshift we will determine the mass of the cooler WD in each system. We will compare this to the spectroscopic mass estimate to investigate the physics of cool WD model atmospheres. This physics is crucial to interpreting the WD luminosity function which contains information on the local star formation history and the age of the Galactic disk and to constraining the form of initial mass-final mass relation at near solar masses.

9 FOCAL STATION, INSTRUMENT AND DETECTOR

Focal station:	Instrument:	Detector(s):	Gratings/Filters:
Cass	ISIS	EEV12+REDPLUS	R300B+R1200R

10 OBSERVING TIME REQUESTED THIS SEMESTER

	Dark:	Grey:	Bright:	specify nights *or* weeks:
Time requested this semester		3		n
Minimum useful allocation this semester		2		

UKIRT applicants requiring dark time must justify this in section 18

11 COMPLETE THIS SECTION ONLY IF THIS IS A LONG TERM PROPOSAL

	Dark:	Grey:	Bright:	specify nights *or* weeks:
Total time requested				

Justification for long term status must be given in section 17

Text fonts must *not* be smaller than the style-file defaults

12 SCHEDULING INFORMATION

Preferred dates:	Feb and Jul
Impossible dates: *Give justification for impossible dates*	The RA distribution of our targets is best suited to the above months
If observations are to be simultaneous with other telescopes or satellites, give details:	
Any other scheduling constraints: *Include likely clashes with other time applications, constraints on lunar position or quarter, instrument preparation requirements, etc*	

13 SERVICE OBSERVING

yes: [ ] no: [x] maybe: [ ]

14 SUPPORT ASTRONOMER REQUESTED AT TELESCOPE

every night: [ ] no: [ ] first night only: [x]

15 LIST OF PRINCIPAL TARGETS

Object(s):	RA(h,m):	Dec(degs):	Mag(type):	Colour:	Exp. Time:
SDSS0332-00	03 32 37	-00 49 18	r=18.3	u-r=0.3	2hr
SDSS0750+30	07 50 53	+30 25 44	r=17.9	u-r=-0.4	1.5hr
SDSS0851+47	08 51 53	+47 12 50	r=17.8	u-r=0.3	1.5hr
SDSS1157+13	11 57 38	+13 44 14	r=18.5	(u-r=0.3)	3hr
SDSS1509+52	15 09 47	+52 10 02	r=18.1	u-r=-0.1	1.5hr
SDSS1705+33	17 05 56	+33 04 38	r=18.9	u-r=0.35	4hr
SDSS2224-08	22 24 36	-08 28 08	r=17.3	u-r=0.25	0.75hr
SDSS2257+14	22 57 32	+14 04 39	r=18.7	u-r=0.35	3.5hr
SDSS0054+13	00 54 12	+13 53 02	r=18.9	u-r=0.45	4hr

16 LIST ALL SIMILAR/SUPPORTING APPLICATIONS TO ANY PATT OR OTHER TIME ASSIGNMENT COMMITTEE

You must include a brief description of any other applications whose targets or science goals are similar to those requested here

Telescope/satellite:	Title/Description of programme:

White dwarf masses: Spectroscopy of samples of young (12500K<$T_{\rm eff}$$\lesssim$80000K) H-rich (DA) white dwarfs (WDs) has revealed that their mass distribution has a comparatively sharp peak centered on M$\sim$0.6$M_{\odot}$ (e.g. 129 hot WDs, Bergeron et al. 1992; the ROSAT WDs, Marsh et al. 1997). This contains $\sim$75% of these objects by number (Liebert et al. 2005) once the dependence of the sampling volume on mass is accounted for (massive WDs have small radii). However, spectroscopy of older, cooler ($T_{\rm eff}$$\leq$12500K) samples reveal a shift in this peak to larger masses. For example, Kepler et al. (2007) examine 964 DAs with 8000K$\leq$$T_{\rm eff}$$\leq$12000K and find the peak mass to be M=0.818$M_{\odot}$. It is still not completely settled as to whether this is a real trend in WD masses or a problem with spectroscopic mass determinations. Either way, better characterisation and understanding of the increase in spectroscopic masses at $T_{\rm eff}$<12500K is **important for** any astrophysical study which relies on good knowledge of WD masses and cooling times (very sensitive to the former) e.g. **deciphering information on the history of local star formation and the age of the Galactic disk from the WD luminosity function and constraining the form of the initial mass-final mass relation for stars with near solar mass, which is integral to modelling Galactic chemical evolution.**

Willson (2000) argue that cooler WDs, which presumably are generally the descendents of older, more metal poor stars, should have larger masses due to the lower efficiency of mass loss on the AGB. However, Kepler et al. (2007) highlight that recent spectroscopic studies of relatively hot WDs in globular clusters indicate that these have near normal masses (M$\sim$0.53$M_{\odot}$, Moehler et al. 2004). Bergeron et al. (1991) argue instead that the deeping of the convective zone in the atmospheres of cooler WDs may lead to material from the underlying He layer being mixed into the predominantly H photosphere. The presence of He increases the gas pressure in the photosphere, mimicking a higher surface gravity and hence larger mass. However, astroseismological measurements of the surface H-layer mass ($M_{\rm H}$) in pulsating WDs have all concluded that $M_{\rm H}$ >$10^{-7}$$M_{\odot}$ (e.g. Bradley 2006). Thus convection should not protrude into the underlying He layer until $T_{\rm eff}$$\leq$6500K. A seemingly more credible explanation is that there is some shortcoming in the treatment of the pressure dissolution of the excited energy levels of HI. The degree to which these are dissolved in model calculations affects the predicted strength of the high order Balmer lines, which are the primary diagnostic of surface gravity. At $T_{\rm eff}$>12500K the perturbation of these energy levels is dominated by interactions with charged (e^-, H^+) particles. The good agreement between spectroscopic and gravitational redshift (GR) mass determinations for objects with $T_{\rm eff}$$\gtrsim$12500K (e.g. Bergeron et al. 1995) indicates that the model physics is robust in this $T_{\rm eff}$ regime. However, at lower $T_{\rm eff}$, where the quality of the small number of existing GR mass determinations is poor, perturbations by neutral particles become increasingly dominant. These are treated in terms of a simplistic hard sphere model where an HI energy level is assumed to be dissolved if it's characteristic interaction radius, which is a fraction, f, of its hydrogenic radius, is greater than the mean interatomic distance in the gas. Unfortunately, f is largely unconstrained by theory.

Proposed observations: We have recently performed a survey for wide (ie. spatially resolved) double degenerate binaries ($T_{\rm eff}$>8000K), using photometry from the vast SDSS DR6, proper motions and (very) limited spectroscopy. We have identified 25 candidates, recovering two previously known examples of such systems (PG0922+162, HS2240+1234). We wish to obtain follow-up data for the components of a subsample of these with the WHT and ISIS to 1) confirm that those objects lacking spectroscopy (ie. 14/18 WDs) are indeed WDs, 2) where none yet exist place stringent constraints on (spectroscopic) $T_{\rm eff}$ and surface gravites using high S/N low resolution blue arm data and 3) using moderate S/N high resolution red arm data measure the shift of the H-α line core of all objects to $\sim$4kms^{-1}. For the targets we are concentrating on here the SDSS photometry indicates that one component has $T_{\rm eff}$>12500K, while the other has $T_{\rm eff}$<12500K. We will determine the mass of the hotter WD in each pair by comparing the $T_{\rm eff}$ and surface gravity estimate to evolutionary models and then deconvolve the GR and line of sight velocity components of the H-α line shift. Note that the necessary high resolution data currently exists for only one WD in this subsample (SDSS0332-00A). With the system line of sight velocity in hand we will determine the GR component of the H-α line core shift in the spectrum of the cooler member of each binary. From this we will estimate the "true" mass of the second degenerate which can be used to constrain the physics of cool WD model atmospheres.

Give details of the technical feasibility of the proposal (S/N,etc) AND any non-standard technical requirements

While the common proper motion of the components of our systems argue that they are physically associated, the combination of velocity constraints from the high resolution red arm data and the SDSS photometry can put this assertion beyond reasonable doubt (or otherwise). Our confirmed double degenerate systems will also be useful for placing constraints on the form of the initial mass-final mass relation, in particular at lower progenitor masses ($M_{init} \lesssim 2M_{\odot}$) than cannot be accessed through the study of open cluster WDs (e.g. Dobbie et al. 2006). Thus these systems can potentially provide clues to the fate of stars with masses similar to the Sun.

To undertake the proposed work we require high S/N, low resolution blue and moderate S/N red optical spectroscopy of the 18 WDs of our candidate wide double degenerate systems (which all have separations of <30"). We propose to use the ISIS spectrograph with EEV12 and REDPLUS CCDs, the "new" dichroic, the R300B and R1200R gratings and a 1.0" wide slit suitably orientated to cover both components of each system simultaneously and provide coverage over the range 3600-5500Å at $\lambda/\delta\lambda \sim 1000$ and 6240-6875Å at $\lambda/\delta\lambda \sim 7500$. The blue arm observations will give us access to the Balmer series members ranging from H-β– H-8, coverage of which are necessary to determine robustly effective temperature and **spectroscopic** surface gravity in the temperature range (∼8000-30000K) occupied by our candidate white dwarfs (e.g. Bergeron et al. 1992). Our extensive previous experience of fitting synthetic profiles to the observed hydrogen Balmer lines of DAs indicates that a $S/N \gtrsim 50$ at the above resolution allows rather tight constraints to be placed on the effective temperature, the surface gravity (±few 100K and a few 0.01 dex respectively) and with reference to state-of-the-art evolutionary models the **spectroscopic** mass ($\Delta M \lesssim 0.05M_{\odot}$) and the cooling time ($\Delta\tau$<50Myrs) of a WD (e.g. see Dobbie et al. 2006). Moreover, using the masses and the radii derived from the evolutionary models, the GR of the hot WDs (T_{eff}>12500K) can be constrained to $\sim 2 kms^{-1}$. The red arm observations will give us access to the H-α line, from which we will determine the line shift due to the combination of line of sight velocity and GR. The H-α line profile formed in the atmospheres of DA WDs features a relatively sharp core (see Figure 2). Based on our previous experience with high resolution data covering this part of the spectral energy distribution of WDs (with UVES data at $\lambda/\delta\lambda \sim 20000$ and a S/N∼17 per resolution element it is possible to constrain the line shift to $\sim 2 kms^{-1}$ by fitting a synthetic profile to the observed line), we estimate that with $S/N \gtrsim 25$ per resolution element at $\lambda/\delta\lambda \sim 7500$ we can constrain the line shift to $\sim 4 kms^{-1}$ and pin down the system line of sight velocity to $\sim 4.5 kms^{-1}$. Thus we can determine the "true" mass of the cooler component in each binary to ∼10%. We note that the shift in the peak of the WD mass distribution is $\sim 0.2M_{\odot}$ at T_{eff}=10000K, which may indicate that the currently adopted value of f is underestimated by ∼60%. This work offers the chance to constrain f to ∼15%.

The magnitudes given in Section 15 correspond to the fainter WD of each pair. We have used the SIGNAL exposure time calculator to determine that a 5400s observation will provide $S/N \approx 25$ per resolution element on a r'=17.8 (SDSS AB magnitude!) object at H-α in 1.0" seeing and grey moon conditions (1.0" slit, sec z=1.2). This integration time provides ample S/N (>50) in the blue arm for our purposes. We find a S/N∼25 per resolution element can be achieved at H-α on our faintest 2 candidates ($r' \approx 18.9$) in grey conditions in an exposure time of 14400s. Note that useful S/N cannot be achieved on these fainter candidates during bright time in reasonable integration times. We intend splitting each observation up into a series of shorter exposures to allow the removal of CR hits from the data. Allowing for overheads such as target and guide star acquisition (300s), the readout time of the CCD (∼30s), a number of observations of the arc lamp and a standard star (600s every other target; e.g. sdOC star PG0823+546) we determine that we will need 24 hours in total.

Our targets span a wide range of RAs and are not all observable on a given night. However, 5/9 objects are visible in February (10 hours of darkness) and 5/9 (including the three faintest objects) are visible in July (7.5 hours of darkness). Therefore, we request one grey night in February and two grey nights in July to complete this program.

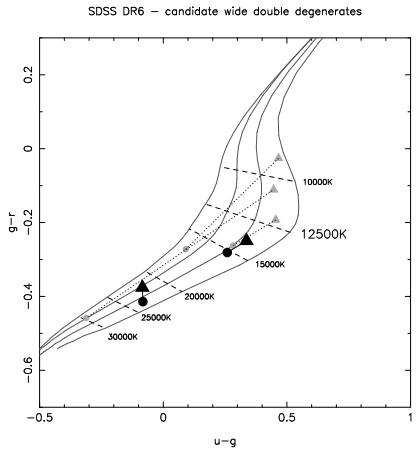


Figure 1. A selection of our candidate wide double degenerate binaries in ugr colour-colour space, with the DA models of Pierre Bergeron overplotted (top to bottom, 1.2$M_\odot$, 0.9$M_\odot$,0.6$M_\odot$ and 0.3$M_\odot$). Two previously known wide double degenerate systems are shown in bold.

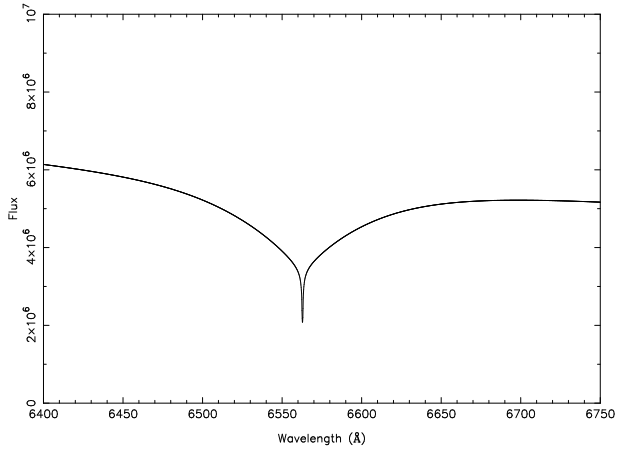


Figure 2. Synthetic white dwarf spectrum (T_{eff}=15000K, log g = 8.0) showing the H-α line with the pronounced non-LTE core.

References

Bergeron, P. et al. 1991, ApJ, 367, 252
Bergeron, P., et al. 1992, ApJ, 394, 228
Bergeron, P., et al. 1995, ApJ, 444, 810
Bradley, P.A., 2006, MmSAI, 77, 437
Dobbie, P. et al. 2006, MNRAS, 369, 383
Kepler, S., et al., 2007, MNRAS, 375, 1315
Liebert, J. et al. 2005, ApJ, 156, 47
Marsh, M. et al. 1997, MNRAS, 286,369
Moehler, S., et al., 2004, A&A, 420, 515
Willson, L-A., 2000, ARA&A, 38, 573

If instrumentation or setup differs from main programme, give full details

20 RELATED PATT APPLICATIONS OVER THE LAST FOUR SEMESTERS *(including unsuccesful applications)*

PATT reference:	Award:	Clear nights:	Comments:
W/2006A/1	3G	3	2 published papers

21 PUBLICATIONS BASED ON PATT TIME PUBLISHED DURING THE LAST FOUR SEMESTERS *(maximum 6)*

Dobbie et al., 2006, MNRAS, 369, 383

22 EXPERIENCE OF INTENDED OBSERVERS WHO HAVE NOT PREVIOUSLY USED THIS TELESCOPE

all are experienced

23 COMPLETE IF THE OBSERVATIONS ARE PRIMARILY FOR A STUDENT RESEARCH TRAINING PROGRAMME

Name of student:	
Project title:	

24 COMPLETE IF THE OBSERVATIONS ARE ASSOCIATED WITH A CURRENT PPARC/SERC RESEARCH GRANT

Name of principal investigator:	
Grant title:	
Grant number:	

25 NON-STANDARD TRAVEL AND SUBSISTENCE REQUIREMENTS *(UK observers only)*

Justify requests for travel and subsistence for more than one person:

Details of any other expenditure (eg freight, remote observing):

A.7.7 Gemini Observing Proposals

Gemini North 2009B Proposal

GEMINI OBSERVATORY

observing time request summary

Semester: 2009B

Observing Mode: queue

Instruments:
GMOS North

Gemini Reference:

Time Awarded:

Thesis:
no

Title: **Probing the stellar initial mass-final mass relation with wide double-degenerate binaries - the northern systems**

Principal Investigator: **Paul Dobbie**
PI institution: Anglo-Australian Observatory, P.O. Box 296 (167 Vimiera Road), Epping, NSW 2121, Australia
PI status: PhD/Doctorate
PI phone/fax/e-mail: +61 293724846 / +61 293724880 / pdd@aao.gov.au
Co-Investigators: Quentin Parker: Anglo-Australian Observatory, qap@aao.gov.au
Richard Baxter: Macquarie University, richardbrucebaxter@gmail.com

---

Partner Submission Details *(multiple entries for joint proposals)*

				NTAC			
Partner	**Partner Lead Scientist**	**Time Requested**	**Minimum Time Requested**	**Reference Number**	**Reco-mmended Time**	**Minimum Time Reco-mmended**	**Rank**
Australia	Dobbie	17.8 hours	8.5 hours		0.0	0.0	
	Total Time	*17.8 hours*					

Abstract *(118 words)*

We propose to obtain high S/N low-resolution GMOS spectroscopy of several wide double-degenerate binary systems which will allow us to test and extend our understanding of the fundamentally important stellar initial mass-final mass relation (IFMR). Several parts of this relation remain poorly delineated e.g. at initial masses M_init<2.5-3.0Msolar and M_init>5-6Msolar. Our new data will be used 1) to place tight limits on the effective temperatures and surface gravities of the binary components 2) in conjunction with WD evolutionary models to determine their masses and cooling times. Subsequently, we will exploit existing knowledge of the IFMR to estimate the progenitor masses, leading to the provision of extremely valuable additional data points on the IFMR in these initial mass regimes.

Science Justification *(996 words)*

We propose GMOS spectroscopy of several wide double-degenerate binaries that will allow us to test and extend understanding of the fundamentally important stellar initial mass-final mass relation (IFMR). This is a theoretically predicted positive correlation between the masses of single low/intermediate mass stars (M<10Msolar) and the remnants left after they have expired. A comprehensive knowledge of its form is important to several areas of astrophysics: it is a key ingredient of galaxy chemical evolution models as it provides an estimate of the amount of gas, enriched with C, N and other metals, these stars return to the ISM; it is crucial to deciphering information locked up in the white dwarf (WD) luminosity functions of stellar populations (Oswalt et al. 1996); the relations upper end can be used to probe directly the maximum mass of a star that will evolve through the planetary nebula phase and thus indirectly the minimum mass of a star that will expire via a Type II SNe (Williams et al. 2009).

The finer details of the IFMR cannot be determined from first principles as the late stages in the lifecycle of a star, while brief, remain extremely challenging to model (e.g. Marigo & Girardi 2007). Observations play a key role in advancing our understanding of the relations form. Most existing empirical data has been obtained via study of open cluster WDs (Weidemann 2000). Here progenitor lifetimes can be determined from the difference between the age of the cluster and the cooling times of WD members. Subsequently, initial masses (M_init) can be estimated by refering to stellar evolutionary models. Greater access to mosaic imagers and 8/10m telescopes with blue sensitive spectrographs has led to substantial recent progress in mapping the IFMR (Kalirai et al. 2007). Our latest work (Dobbie et al. 2009), where we compiled data for ~50 open cluster WDs, indicates that the bulk of stars follow closely a monotonic relation. It has also shown that the IFMR is somewhat steeper in the range 3Msolar<M_init<4Msolar than elsewhere (Figure 1). This is consistent with theoretical expectations and the sharp drop seen in the number density of objects on the high mass side of the main peak in the field WD mass distribution (Ferrario et al. 2005).

Despite this headway, the IFMR remains sparsely sampled in several important initial mass regimes. For example, there are only two data points at M_init>~5-6Msolar. Thus, the form of the upper IFMR remains substantially uncertain (Figure 1). Furthermore, there are very few objects at M_init<2.5-3Msolar yet it is crucial to have a good understanding of the fate of the numerous stars with masses more closely resembling that of our Sun. In the open cluster approach, mapping the higher initial mass regime of the IFMR requires targeting young (~100-300Myrs) populations sufficiently rich to have harboured a substantial number of stars with M_init>~5-6Msolar and where the progeny of these stars are still relatively young, hot and luminous. Investigation of the lower initial mass regime requires targeting old (>1-2Gyrs) open clusters, populations sufficiently mature that stars with M_init<2.5-3Msolar have evolved beyond the main sequence. Such clusters are comparatively rare so it is neccessary to probe large volumes to identify good examples. Consequently, while high S/N optical spectroscopy is critical to determining WD masses and cooling times, the great distances involved make it extremely challenging to obtain the necessary data even with todays large telescopes.

We focus here on a novel independent approach to probing the IFMR, involving the exploitation of wide (spatially resolved) double-degenerate field binaries in which the WDs have substantially different masses (delta_M>~0.1Msolar). The components of these co-eval systems have esentially evolved as single stars since they have remained sufficiently well separated that neither object has ever overflowed it's Roche Lobe and initiated mass transfer onto the other. If one of the WDs in such a system has a mass (M_final) which places it on the part of the IFMR which is reasonably well delineated by existing data (M_final~0.65-1.0Msolar; Figure 1) then it is possible to estimate the initial mass (and lifetime) of this component and thus the total age of the binary. The progenitor mass of the other component then follows as for a WD cluster member and can be used to probe a different initial mass regime.

We have used SDSS DR7 photometry and SuperCOSMOS proper motions to identify ~70 likely wide double-degenerate binaries containing components with effective temperatures, Teff>~10000K

(where atmospheres can be reliably modelled). The magnitudes and colours of all systems have been scrutinised to identify those where there appears to be a significant difference in the masses of the components (ie. delta_M>~0.1Msolar). For example, the components of one system both have u-g=-0.08, indicating similar effective temperature. However, one WD is 1.0magnitudes fainter than the other suggesting a smaller radius (~65%) which translates to a larger mass (~0.3Msolar). As the finite age of the Galaxy dictates that single star evolution cannot yet have produced WDs with M_fin<0.5-0.55solar, it is highly probable that the mass of at least one component of each system in this subset lies in the range M_fin=0.65-1.0Msolar. Here we propose to obtain high S/N, low resolution spectroscopy spanning 3800-5200angstroms for the components of the 5 systems visible this semester from Gemini-North. With this data we will measure the effective temperature and the surface gravity for each WD by comparing the observed profiles of H-beta-H-8 lines to the predictions of state-of-the-art model atmospheres. Our extensive experience of the line fitting process indicates that S/N>~50 per 3-4angstroms binned pixel across this wavelength range will keep the formal uncertainties to ~200K and ~0.02dex in effective temperature and surface gravity respectively (the limited existing SDSS spectroscopy of some objects is not of sufficient quality). Subsequently we will estimate the masses and the cooling times of the components by refering to the WD evolutionary tracks of Fontaine et al. (2001). The tight limits on effective temperature and surface gravity will allow the mass and cooling time of each WD to be determined to ~0.02Msolar and ~8% respectively.

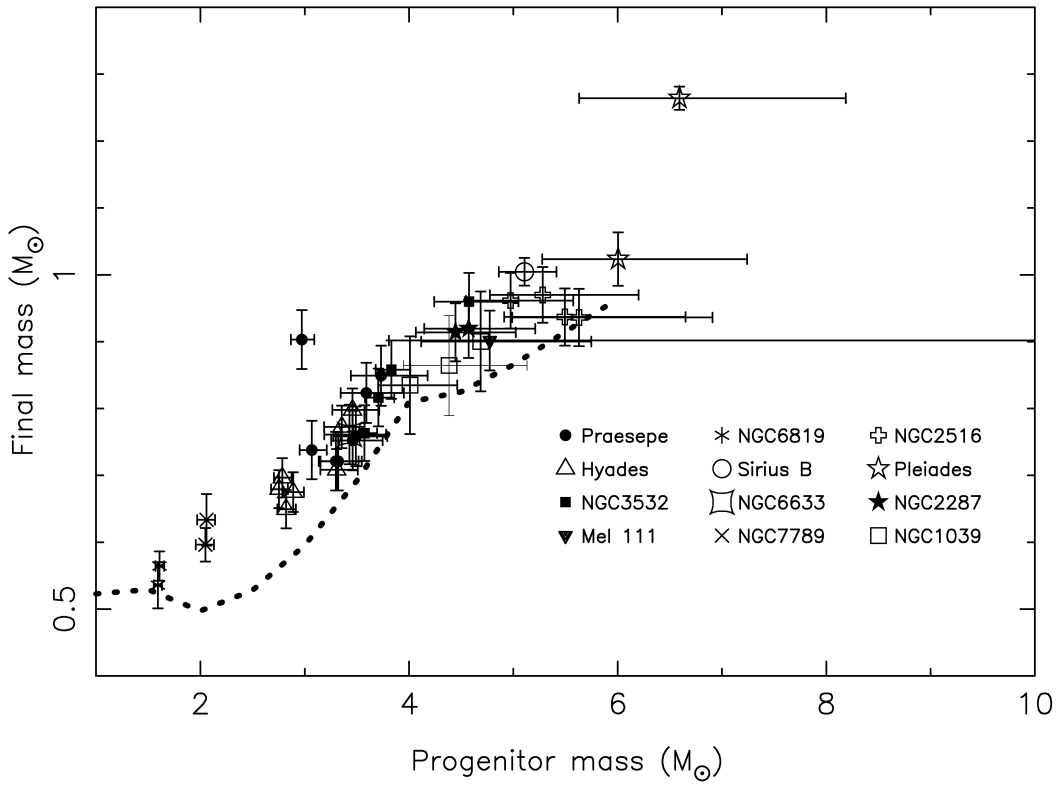


Figure 1: The IFMR for a compilation of WD members of open clusters (excepting the heavily studied Sirius B). The bulk of stars delineate a relatively tight trend which reflects the form of the core mass at the time of the first thermal pulse via initial mass relation (dotted line; Girardi et al. 2000), particularly at M_init>~3Msolar.

Technical Justification *(380 words)*

As discussed in the Scientific Justification we require S/N>~50 per 3-4 angstrom binned pixel across the range 3800-5200angstroms. We intend to observe our targets using the following settings: B600_G5307 grating, 4x4 detector binning and a 1" or 1.5" slit, depending on the separation of the binary system. We have used the GMOS-N exposure time calculator (V4.0) to derive the following exposure times for the components of our wide double-degenerate binary systems, required to obtain S/N~50 at H-8 to H-9: 60mins for DD-03A (airmass <1.5, 1" slit, IQ=70%, CC=70%, WV=Any and SB=Any), 60mins for DD-03B (airmass <1.5, 1" slit, IQ=70%, CC=70%, WV=Any and SB=80%), 60mins for DD-04A (airmass <1.5, 1" slit, IQ=70%, CC=90%, WV=Any and SB=80%), 60mins for DD-04B (airmass <1.5, 1" slit, IQ=70%, CC=70%, WV=Any and SB=80%), 120mins for DD-05A (airmass <1.5, 1" slit, IQ=70%, CC=70%, WV=Any and SB=80%), 120mins for DD-05B(airmass <1.5, 1" slit, IQ=70%, CC=70%, WV=Any and SB=80%), 125mins for DD-06A (airmass <1.5, 1.5" slit, IQ=85%, CC=70%, WV=Any and SB=50%), 125mins for DD-06B(airmass <1.5, 1.5" slit, IQ=85%, CC=70%, WV=Any and SB=50%), 38mins for DD-08A (airmass <1.5, 1" slit, IQ=70%, CC=90%, WV=Any and SB=80%) and 60mins for DD-08B (airmass <1.5, 1" slit, IQ=70%, CC=90%, WV=Any and SB=80%). These S/N estimates allow for a factor 2 binning in the dispersion direction at the reduction stage. To ensure successful cosmic ray rejection from our data the observation of each star will be split into three sub-integrations. While arc observations from the standard calibration plan are adequate for wavelength calibration, we request the additional observation of a DC (spectrally featureless) white dwarf (WD1918+38; 3x10mins) so that we can reliably remove instrumental signature from our data (this is essential for detailed line profile modeling). Allowing for overheads (~18 mins of telescope and instrument set-up per target and 24 seconds per CCD readout), we estimate that our total time request is 17.8 hours.

References

Dobbie, P.D., et al. 2009, MNRAS, accepted Ferrario, L., et al., 2005, MNRAS, 361, 1131 Fontaine, G., et al. 2001, PASP, 113, 409 Girardi, L., et al. 2000, A&AS, 141, 371 Kalirai, J., et al., 2007, ApJ, 671, 748 Marigo, P., & Girardi, L., 2007, A&A, 469, 239 Oswalt, T., et al., 1996, Nature, 382, 692 Weidemann, V., 2000, A&A , 363, 647 Williams et al. 2009, ApJ, 693, 355

Band 3 Information

Requested time in case of band 3 allocation: 17.8 hours

Minimum required time for a usable band 3 allocation: 8.4 hours

Use the following conditions for band 3 only:

Name	Image Quality	Sky Background	Water Vapor	Cloud Cover
Band 3 Observing Conditions	70 %	80 %	Any	70 %

Band 3 Consideration Comments *(95 words)*
All our northern targets can be observed in mediocre sky conditions and thus our program can be considered for Band 3. The components of some of our target systems are separated by only ~4-5 arcseconds so it will be neccessary to observe these in IQ=70% conditions to minimise the contamination of the spectrum of an object with the light of its companion. In CC=70% our faintest targets, DD-06A and 06B, require a dark sky but are sufficiently well separated that we can open up the slit to 1.5" and observe them in poorer seeing (IQ=85%).

Observation Details

Observation	RA	Dec	Brightness	Total Time *(including overheads)*
DD-03A	08:49:52.87	47:12:49.47	V=17.2	79.0 minutes
47636499(oiwfs)	8:49:29.964	47:13:43.77	12.94 UCmag,12.104 Jmag,11.763 Kmag	separation 3.99
Observing conditions: DD-03A		resources: GMOS North		
DD-03B	08:49:52.48	47:12:47.77	V=18.2	79.0 minutes
47636499(oiwfs)	8:49:29.964	47:13:43.77	12.94 UCmag,12.104 Jmag,11.763 Kmag	separation 3.94
Observing conditions: DD-03B		resources: GMOS North		
DD-04A	09:25:13.48	16:01:44.15	V=16.5	79.0 minutes
37361750(oiwfs)	9:25:03.002	15:58:19.7	14.02 UCmag,13.064 Jmag,12.710 Kmag	separation 4.24
Observing conditions: DD-04A		resources: GMOS North		
DD-04B	09:25:13.18	16:01:45.41	V=17.6	79.0 minutes
37361750(oiwfs)	9:25:03.002	15:58:19.7	14.02 UCmag,13.064 Jmag,12.710 Kmag	separation 4.21
Observing conditions: DD-04B		resources: GMOS North		
DD-05A	09:26:46.88	13:21:34.52	V=18.8	139.0 minutes
36470185(oiwfs)	9:26:45.256	13:22:18.9	9.63 UCmag,8.548 Jmag,8.243 Kmag	separation 0.84
Observing conditions: DD-05A		resources: GMOS North		
DD-05B	09:26:47.0	13:21:38.49	V=18.9	139.0 minutes
36470185(oiwfs)	9:26:45.256	13:22:18.9	9.63 UCmag,8.548 Jmag,8.243 Kmag	separation 0.8
Observing conditions: DD-05B		resources: GMOS North		
DD-06A	10:02:44.88	36:06:29.68	V=19.3	144.0 minutes
44485534(oiwfs)	10:02:24.839	36:04:25.8	10.33 UCmag,9.403 Jmag,9.014 Kmag	separation 4.54

Observing conditions: DD-06A		resources: GMOS North		
DD-06B	10:02:45.86	36:06:53.39	V=19.5	144.0 minutes
44485535(oiwfs)	10:02:25.621	36:05:29.61	12.83 UCmag,11.594 Jmag,11.117 Kmag	separation 4.32
Observing conditions: DD-06B		resources: GMOS North		
DD-08A	22:23:1.62	22:01:31.36	V=16.0	57.0 minutes
39594658(oiwfs)	22:23:16.834	21:59:00.96	9.04 UCmag,6.765 Jmag,5.821 Kmag	separation 4.33
Observing conditions: DD-08A		resources: GMOS North		
DD-08B	22:23:1.72	22:01:24.95	V=16.5	79.0 minutes
39594658(oiwfs)	22:23:16.834	21:59:00.96	9.04 UCmag,6.765 Jmag,5.821 Kmag	separation 4.25
Observing conditions: DD-08B		resources: GMOS North		
WD1918+38	19:18:58.0	38:43:35.0	B=15.1	49.0 minutes
50101419(oiwfs)	7:30:57.192	48:08:36.83		separation 7,732.59
Observing conditions: DC STANDARD		resources: GMOS North		

Observing Conditions

Name	Image Quality	Sky Background	Water Vapor	Cloud Cover
Band 3 Observing Conditions	70 %	80 %	Any	70 %
DD-03A	70 %	Any	Any	70 %
DC STANDARD	Any	Any	Any	70 %
DD-03B	70 %	80 %	Any	70 %
DD-04A	70 %	80 %	Any	90 %
DD-04B	70 %	80 %	Any	70 %
DD-05A	70 %	80 %	Any	70 %
DD-05B	70 %	80 %	Any	70 %
DD-06A	85 %	50 %	Any	70 %
DD-06B	85 %	50 %	Any	70 %
DD-08A	70 %	80 %	Any	90 %
DD-08B	70 %	80 %	Any	90 %
Global Default	70 %	50 %	Any	50 %

Resources

- Gemini North
  - GMOS North
    - Focal Plane Unit
      - Longslit 1.00 arcsec
    - Disperser
      - B600_G5307
  - GMOS North
    - Focal Plane Unit
      - Longslit 1.00 arcsec
    - Disperser
      - B600_G5307
  - GMOS North
    - Focal Plane Unit
      - Longslit 1.50 arcsec
    - Disperser

B600_G5307

Scheduling Information

Scheduling constraints and non-usable dates

- (impossible):
- (optimal):
- (synchronous):

Additional Information

Keyword Category: galactic
Keywords: Binaries
Evolution
White dwarfs

Allocations:

Reference	Time	% Useful	Status of previous data
GN-2007B-Q-88	2.0 hours	100%	Published: MNRAS, 2009 accepted, "Two distant brown dwarfs in the UKIRT Infrared Deep Sky Survey Deep Extragalactic Survey Data Release 2"
GN-2008B-Q-90	1.6 hours	100%	Published: MNRAS, 2009 accepted, "Two distant brown dwarfs in the UKIRT Infrared Deep Sky Survey Deep Extragalactic Survey Data Release 2"

Publications:

- Casewell, S., Dobbie, P., et al. 2009 MNRAS, accepted, "High resolution optical spectroscopy of Praesepe white dwarfs"
- Dobbie P., et al., 2009 MNRAS, accepted, "A new detailed examination of white dwarfs in NGC3532 and NGC2287"
- Dobbie, P., et al. 2009 MNRAS, accepted, "A massive white dwarf in the Coma Berenices open cluster"

Gemini South 2010B Proposal

GEMINI OBSERVATORY

observing time request summary

Semester: 2010A

Observing Mode: queue

Instruments:
GMOS South

Gemini Reference:

Time Awarded:

Thesis:
no

Title: **Probing the stellar initial mass-final mass relation with wide double-degenerate binaries - the southern A semester systems**

Principal Investigator: **Paul Dobbie**
PI institution: Anglo-Australian Observatory, P.O. Box 296 (167 Vimiera Road), Epping, NSW 2121, Australia
PI status: PhD/Doctorate
PI phone/fax/e-mail: +61 293724846 / +61 293724880 / pdd@aao.gov.au
Co-Investigators: Quentin Parker: Anglo-Australian Observatory, qap@aao.gov.au
Richard Baxter: Macquarie University, richardbrucebaxter@gmail.com

---

Partner Submission Details *(multiple entries for joint proposals)*

				NTAC			
Partner	**Partner Lead Scientist**	**Time Requested**	**Minimum Time Requested**	**Reference Number**	**Reco-mmended Time**	**Minimum Time Reco-mmended**	**Rank**
Australia	Dobbie	6.5 hours	3.2 hours		0.0	0.0	
	Total Time	*6.5 hours*					

Abstract *(118 words)*

We propose to obtain high S/N low-resolution GMOS spectroscopy of two wide double-degenerate binary systems which will allow us to test and extend our understanding of the fundamentally important stellar initial mass-final mass relation (IFMR). Several parts of this relation remain poorly delineated e.g. at initial masses M_init<2.5-3.0Msolar and M_init>5-6Msolar. Our new data will be used 1) to place tight limits on the effective temperatures and surface gravities of the binary components 2) in conjunction with WD evolutionary models to determine their masses and cooling times. Subsequently, we will exploit existing knowledge of the IFMR to estimate the progenitor masses, leading to the provision of extremely valuable additional data points on the IFMR in these initial mass regimes.

Science Justification *(923 words)*

We propose GMOS spectroscopy of several wide double-degenerate binaries that will allow us to test and extend understanding of the fundamentally important stellar initial mass-final mass relation (IFMR). This is a theoretically predicted positive correlation between the masses of single low/intermediate mass stars (M<10Msolar) and the remnants left after they have expired. A comprehensive knowledge of its form is important to several areas of astrophysics: it is a key ingredient of galaxy chemical evolution models as it provides an estimate of the amount of gas, enriched with C, N and other metals, these stars return to the ISM; it is crucial to deciphering information locked up in the white dwarf (WD) luminosity functions of stellar populations (Oswalt et al. 1996); the relations upper end can be used to probe directly the maximum mass of a star that will evolve through the planetary nebula phase and thus indirectly the minimum mass of a star that will expire via a Type II SNe (Williams et al. 2009).

The finer details of the IFMR cannot be determined from first principles as the late stages in the lifecycle of a star, while brief, remain extremely challenging to model (e.g. Marigo & Girardi 2007). Observations play a key role in advancing our understanding of the relations form. Most existing empirical data has been obtained via study of open cluster WDs (Weidemann 2000). Here progenitor lifetimes can be determined from the difference between the age of the cluster and the cooling times of WD members. Subsequently, initial masses (M_init) can be estimated by refering to stellar evolutionary models. Greater access to mosaic imagers and 8/10m telescopes with blue sensitive spectrographs has led to substantial recent progress in mapping the IFMR (Kalirai et al. 2007). Our latest work (Dobbie et al. 2009), where we compiled data for ~50 open cluster WDs, indicates that the bulk of stars follow closely a monotonic relation. It has also shown that the IFMR is somewhat steeper in the range 3Msolar<M_init<4Msolar than elsewhere (Figure 1). This is consistent with theoretical expectations and the sharp drop seen in the number density of objects on the high mass side of the main peak in the field WD mass distribution (Ferrario et al. 2005).

Despite this headway, the IFMR remains sparsely sampled in several important initial mass regimes. For example, there are only two data points at M_init>~5-6Msolar. Thus, the form of the upper IFMR remains substantially uncertain (Figure 1). Furthermore, there are very few objects at M_init<2.5-3Msolar yet it is crucial to have a good understanding of the fate of the numerous stars with masses more closely resembling that of our Sun. In the open cluster approach, mapping the higher initial mass regime of the IFMR requires targeting young (~100-300Myrs) populations sufficiently rich to have harboured a substantial number of stars with M_init>~5-6Msolar and where the progeny of these stars are still relatively young, hot and luminous. Investigation of the lower initial mass regime requires targeting old (>1-2Gyrs) open clusters, populations sufficiently mature that stars with M_init<2.5-3Msolar have evolved beyond the main sequence. Such clusters are comparatively rare so it is neccessary to probe large volumes to identify good examples. Consequently, while high S/N optical spectroscopy is critical to determining WD masses and cooling times, the great distances involved make it extremely challenging to obtain the necessary data even with todays large telescopes.

We focus here on a novel independent approach to probing the IFMR, involving the exploitation of wide (spatially resolved) double-degenerate field binaries in which the WDs have substantially different masses (delta_M>~0.1Msolar). The components of these co-eval systems have esentially evolved as single stars since they have remained sufficiently well separated that neither object has ever overflowed it's Roche Lobe and initiated mass transfer onto the other. If one of the WDs in such a system has a mass (M_final) which places it on the part of the IFMR which is reasonably well delineated by existing data (M_final~0.65-1.0Msolar; Figure 1) then it is possible to estimate the initial mass (and lifetime) of this component and thus the total age of the binary. The progenitor mass of the other component then follows as for a WD cluster member and can be used to probe a different initial mass regime.

We have used SDSS DR7 photometry and SuperCOSMOS proper motions to identify ~70 likely wide double-degenerate binaries containing components with effective temperatures, Teff>~10000K

(where atmospheres can be reliably modelled). The magnitudes and colours of all systems have been scrutinised to identify those where there appears to be a significant difference in the masses of the components (ie. delta_M>~0.1Msolar). For example, the components of one system both have u-g=-0.08, indicating similar effective temperature. However, one WD is 1.0magnitudes fainter than the other suggesting a smaller radius (~65%) which translates to a larger mass (~0.3Msolar). As the finite age of the Galaxy dictates that single star evolution cannot yet have produced WDs with M_fin<0.5-0.55solar, it is highly probable that the mass of at least one component of each system in this subset lies in the range M_fin=0.65-1.0Msolar. Here we propose to obtain high S/N, low resolution spectroscopy spanning 3800-5200angstroms for the components of the 2 systems visible this semester from Gemini-South.

Note that the two proposed targets are currently in the Gemini South queue but given their band 3 status there is considerable uncertaintly as to whether they will be observed in the current semester. If they were to be observed in 09B we would notify the Australian Gemini scientist so that this 10A programe could be modified accordingly.

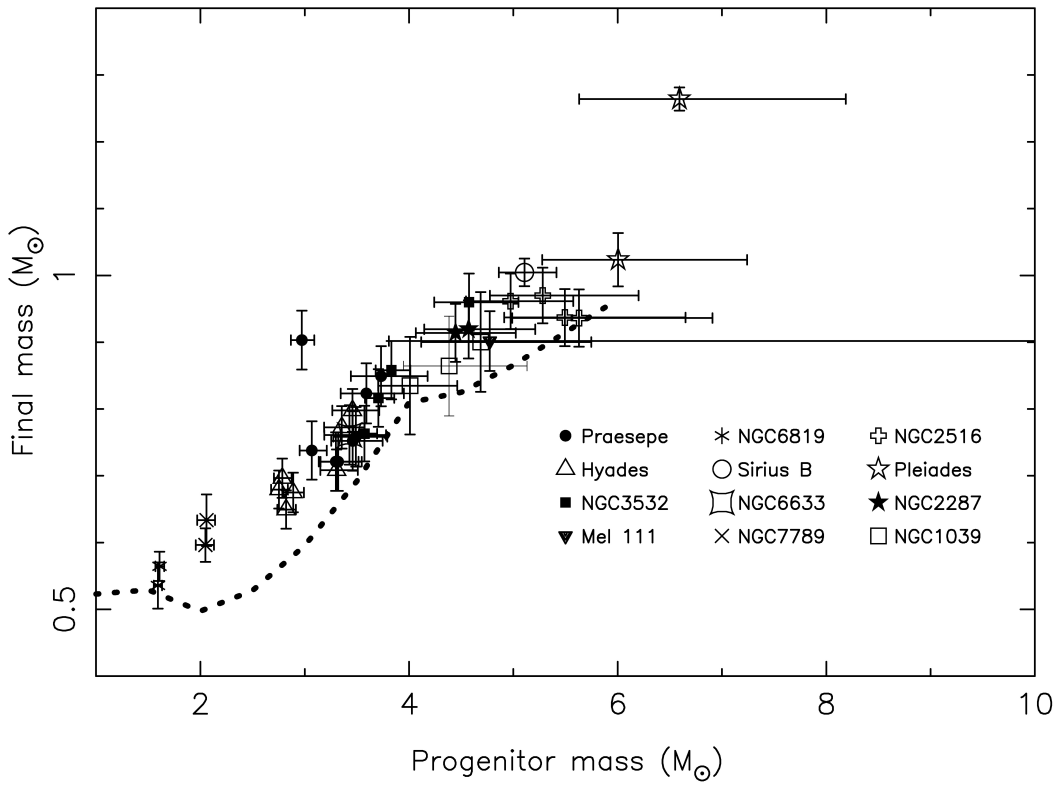


Figure 1: The IFMR for a compilation of WD members of open clusters (excepting the heavily studied Sirius B). The bulk of stars delineate a relatively tight trend which reflects the form of the core mass at the time of the first thermal pulse via initial mass relation (dotted line; Girardi et al. 2000), particularly at M_init>~3Msolar.

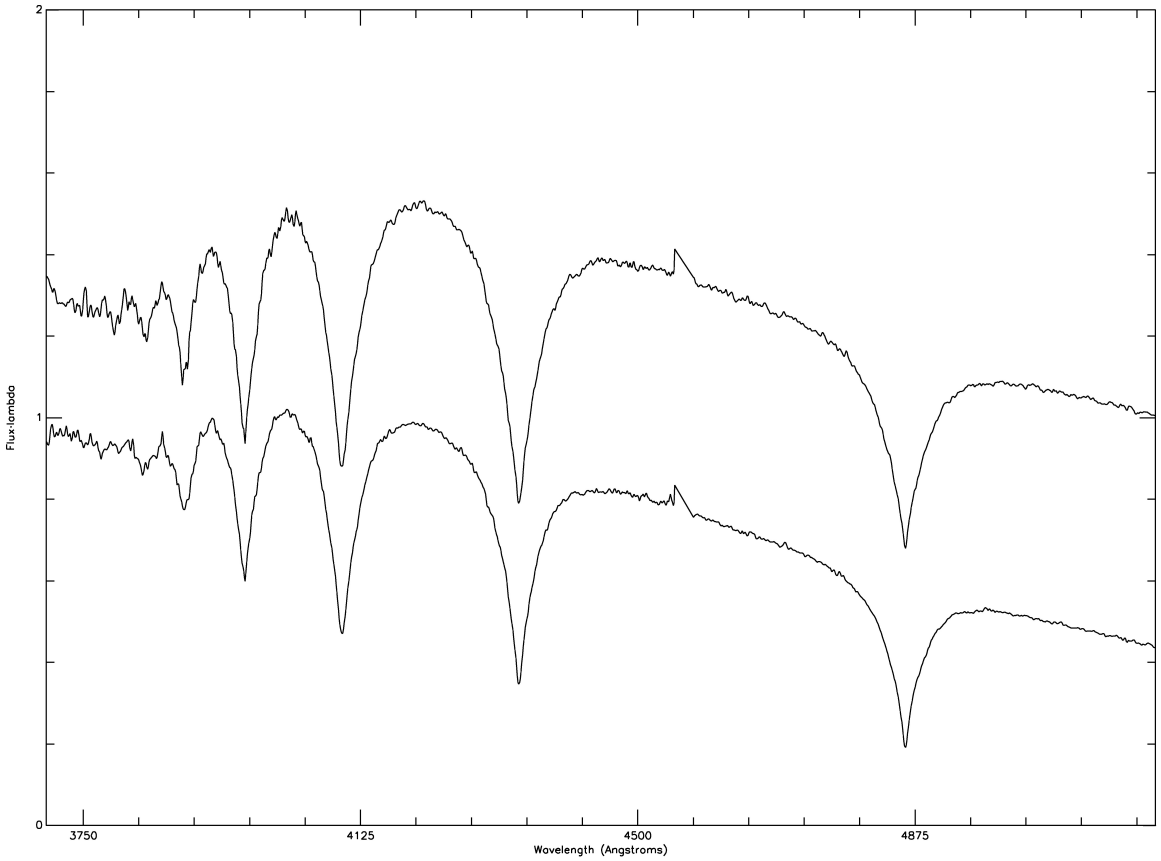


Figure 2: Gemini-N GMOS spectroscopy of DD-20A+B observed in the current semester. Preliminary analysis indicates Teff=19000K, 25500K and log g=8.1 and 8.0 for A and B respectively. The system lies at a distance of ~200pc. As expected, the cooler component is the more massive.

Technical Justification *(487 words)*

We will measure the effective temperature and the surface gravity for each WD by comparing the observed profiles of H-beta to H-8/9 lines to the predictions of state-of-the-art model atmospheres. Our extensive experience of the line fitting process indicates that S/N>~50 per 3-4angstroms binned pixel across the wavelength range 3800-5200angstroms will keep the formal uncertainties to ~200K and ~0.02dex in effective temperature and surface gravity respectively (the limited existing SDSS spectroscopy of some objects is not of sufficient quality, particularly at lambda<~4000angstroms). Subsequently we will estimate the masses and the cooling times of the components by refering to the WD evolutionary tracks of Fontaine et al. (2001). The tight limits on effective temperature and surface gravity will allow the mass and cooling time of each WD to be determined to ~0.02Msolar and ~8% respectively. When coupled with existing knowledge about the form of the IFMR this will permit the system age to be measured to ~25%. As the lifetime of the progenitor star of each WD will be determined to a similar level of accuracy, the progenitor masses can be constrained to ~10-15% (since M_progenitor approx proportional to lifetime_progenitor^-0.4).

To obtain the neccessary data (S/N>~50 per 3-4 angstrom binned pixel ALL THE WAY from 3800-5200angstroms ie. including ~3800-3850angstroms) we intend to observe our targets using the following settings: B600_G5323 grating, 4x4 detector binning and a 2.0" slit, aligned along the PA of the system (differential refraction should not be a problem with our wide slit and all systems have separation <30"). Using the GMOS-S exposure time calculator (V4.0) we derive an exposure time of 180mins to reach a S/N~50 at H-8 to H-9 for DD-01A+B: V=19,20 (airmass <1.5, 2" slit, IQ=70%, CC=70%, WV=Any and SB=50%). For DD-07A+B (V=16.8, 17.5) we derive an exposure time of 120mins (airmass <1.5, 1" slit, IQ=85%, CC=90%, WV=Any and SB=80%). These S/N estimates allow for a factor 2 binning in the dispersion direction at the reduction stage. To ensure successful cosmic ray rejection from our data the observation of each star will be split into three sub-integrations. While arc observations from the standard calibration plan are adequate for wavelength calibration, we request the additional observation of a DC (spectrally featureless) white dwarf (WD0000-345; 3x10mins) so that we can reliably remove instrumental signature from our data (this is essential for detailed line profile modeling). Allowing for overheads (~18 mins of telescope and instrument set-up per target and 24 seconds per CCD readout), we estimate that our total time request is 9.9 hours.

References

Dobbie, P.D., et al. 2009, MNRAS, accepted Ferrario, L., et al., 2005, MNRAS, 361, 1131 Fontaine, G., et al. 2001, PASP, 113, 409 Girardi, L., et al. 2000, A&AS, 141, 371 Kalirai, J., et al., 2007, ApJ, 671, 748 Marigo, P., & Girardi, L., 2007, A&A, 469, 239 Oswalt, T., et al., 1996, Nature, 382, 692 Weidemann, V., 2000, A&A , 363, 647 Williams et al. 2009, ApJ, 693, 355

Band 3 Information

Requested time in case of band 3 allocation: 3.5 hours

Minimum required time for a usable band 3 allocation: 3.5 hours

Use the following conditions for band 3 only:

Name	Image Quality	Sky Background	Water Vapor	Cloud Cover
Band 3 Observing Conditions	85 %	80 %	Any	90 %

Band 3 Consideration Comments *(42 words)*
One of our targets can be observed in mediocre sky conditions and thus our program could be considered for Band 3. Our faintest target, DD-01A+B could not be observed in poorer conditions. It requires better sky conditions to obtain the neccessary S/N.

Observation Details

Observation	RA	Dec	Brightness	Total Time *(including overheads)*
DD-01A+B	00:29:25.29	00:15:59.8	V=20.0	200.0 minutes
31795371(oiwfs)	0:29:16.112	0:18:10.82	14.85 UCmag,13.397 Jmag,12.774 Kmag	separation 3.17
Observing conditions: DD-01A+B		resources: GMOS South		
DD-07A+B	22:22:36.57	-8:28:6.05	V=17.5	140.0 minutes
29023488(oiwfs)	22:22:19.397	-8:29:20.67	12.25 UCmag,11.239 Jmag,10.869 Kmag	separation 4.42
Observing conditions: DD-07A+B		resources: GMOS South		
WD0000-345	00:02:40.08	-34:13:39.6	B=15.4	49.0 minutes
17177811(oiwfs)	0:02:17.536	-34:14:14.76	10.53 UCmag,9.013 Jmag,8.345 Kmag	separation 4.7
Observing conditions: DC standard		resources: GMOS South		

Observing Conditions

Name	Image Quality	Sky Background	Water Vapor	Cloud Cover
Band 3 Observing Conditions	85 %	80 %	Any	90 %
Global Default	70 %	50 %	Any	50 %
DD-01A+B	70 %	50 %	Any	70 %
DD-07A+B	70 %	80 %	Any	90 %
DC standard	85 %	Any	Any	90 %

Resources

- Gemini South
  - GMOS South
    - Focal Plane Unit
      - Longslit 1.50 arcsec
    - Disperser
      - B600_G5323

GMOS South
- Focal Plane Unit
  - Longslit 1.00 arcsec
- Disperser
  - B600_G5323

GMOS South
- Focal Plane Unit
  - Longslit 2.00 arcsec
- Disperser
  - B600_G5323

GMOS South
- Focal Plane Unit
  - Longslit 2.00 arcsec
- Disperser
  - B600_G5323

GMOS South
- Focal Plane Unit
  - Longslit 2.00 arcsec
- Disperser
  - B600_G5323

Scheduling Information

Scheduling constraints and non-usable dates

- (impossible):
- (optimal):
- (synchronous):

Additional Information

Keyword Category: galactic
Keywords: Binaries
Evolution
White dwarfs

Allocations:

Reference	Time	% Useful	Status of previous data
GN-2007B-Q-88	2.0 hours	100%	Published: MNRAS, 2009 accepted, "Two distant brown dwarfs in the UKIRT Infrared Deep Sky Survey Deep Extragalactic Survey Data Release 2"
GN-2008B-Q-90	1.6 hours	100%	Published: MNRAS, 2009 accepted, "Two distant brown dwarfs in the UKIRT Infrared Deep Sky Survey Deep Extragalactic Survey Data Release 2"
GN-2009B-Q-80	4.8 hours	100%	Observations completed. Data reduced and a preliminary analysis performed (e.g. Figure 2)
GS-2009B-Q-63	9.4 hours	0%	No data obtained as yet.

Publications:

- Casewell, S., Dobbie, P., et al. 2009 MNRAS, accepted, "High resolution optical spectroscopy of Praesepe white dwarfs"
- Dobbie P., et al., 2009 MNRAS, 395, 2248, "A new detailed examination of white dwarfs in NGC3532 and NGC2287"
- Dobbie, P., et al. 2009 MNRAS, 395, 1591, "A massive white dwarf in the Coma Berenices open cluster"

A.7.8 VLT Observing Proposals

VLT 2010A Proposal

EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral
Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

OBSERVING PROGRAMMES OFFICE • Karl-Schwarzschild-Straße 2 • D-85748 Garching bei München • e-mail: opo@eso.org • Tel.: +49-89-32 00 64 73

APPLICATION FOR OBSERVING TIME PERIOD: **84A**

Important Notice:

By submitting this proposal, the PI takes full responsibility for the content of the proposal, in particular with regard to the names of CoIs and the agreement to act according to the ESO policy and regulations, should observing time be granted

1. Title Category: **D–7**

Probing the pivotal initial mass between the PNe and Type II SNe evolutionary channels

2. Abstract / Total Time Requested

Total Amount of Time:

We aim to greatly improve understanding of the form of the upper end of the fundamentally important stellar initial mass-final mass relation (IFMR). Despite recent progress the relation remains sparsely sampled by observations at M_{init} ≥5-6$M_{\odot}$. We propose to use the VLT+FORS to obtain high S/N low-resolution spectra of 10 WD candidate members of the open clusters NGC2287 and NGC3532 which have been identified as the likely progeny of stars in this initial mass regime. These new data will be used 1) to place **tight** limits on the T_{eff}s and log g's of WDs so that their membership status can be thoroughly examined and 2) alongside WD evolutionary models, to accurately determine their masses and cooling times so we can apply prior knowledge of the cluster ages to estimate lifetimes and masses of progenitors. Our improved mapping of the IFMR at M_{init} ≥5-6$M_{\odot}$ will lead to the augmention of limits on the maximum mass of WD progenitors.

3.

Run	Period	Instrument	Time	Month	Moon	Seeing	Sky Trans.	Obs.Mode
A	84	FORS2	2n	feb	g	≤ 1.4″	THN	v

4.

Number of nights/hours	Telescope(s)	Amount of time
a) already awarded to this project:	VLT	24hr total in 079.D-0490/080.D-0654
b) still required to complete this project:	0	0

5. Special remarks:

6. Principal Investigator: PDOBBIE

CoI(s): Q. Parker (1018), M. Burleigh (1244)

7. Is this proposal linked to a PhD thesis preparation? State role of PhD student in this project

8. Description of the proposed programme

A) Scientific Rationale:

We propose VLT and FORS spectroscopy of 10 faint WD candidate members of the open clusters NGC2287 and NGC3535 that will provide crucial new data points at the high mass end of the fundamentally important stellar initial mass-final mass relation (IFMR). This is a theoretically predicted positive correlation between the masses of single, non-magnetic, low and intermediate mass stars ($M \lesssim 10 M_{\odot}$) and the remnants left behind after they have expired. A comprehensive knowledge of its form is important to several areas of astrophysics: it is a key ingredient of galaxy chemical evolution models as it provides an estimate of the amount of gas, enriched with C, N and other metals, these stars return to the ISM; it is crucial to deciphering information locked up in the white dwarf (WD) luminosity functions of stellar populations (Oswalt et al. 1996); the relations upper end can be used to directly probe the maximum mass of a star that will evolve through the planetary nebula phase and thus indirectly the minimum mass of a star that will expire via a Type II SNe (Williams et al. 2009).

The finer details of the IFMR cannot be determined from first principles since the late stages in the lifecycle of a star, while brief, are extremely challenging to model (e.g. Marigo & Girardi 2007). Therefore, observations play a key role in advancing our knowledge about the relations form. Arguably the best observational based constraints are obtained via the study of open cluster WDs (Weidemann 2000). Here progenitor lifetimes can be determined from the difference between the age of the cluster and the cooling times of WD members. Subsequently, initial masses can be estimated by refering to stellar evolutionary models. Greater access to mosaic imagers and 8/10m telescopes with blue sensitive spectrographs has led to major recent progress in mapping the IFMR (e.g. Kalirai et al. 2007). In our latest work (Dobbie et al. 2009), where we compiled data for ∼50 WDs from open clusters with near solar metalicity, we found that the bulk of these stars follow a monotonic relation relatively closely. We also showed that the IFMR is somewhat steeper in the range $3 M_{\odot} \lesssim M_{\rm init} \lesssim 4 M_{\odot}$ than elsewhere (Figure 1). This is consistent with theoretical expectations and the sharp drop seen in the number density of objects on the high mass side of the main peak in the field WD mass distribution (Ferrario et al. 2005).

Despite recent headway, the IFMR remains very poorly sampled by observations at $M_{\rm init} \gtrsim$5-6$M_{\odot}$. Indeed, there are only two data points (LB1497, GD50) and a limit (NGC2099-WD24) here, and the association of GD50 with the Pleiades is questionable as it lies ∼90pc beyond the cluster's tidal bounds. Thus, the form of the upper IFMR remains substantially uncertain (Figure 1). To move towards obtaining crucial additional data points in this initial mass regime, we have used CCD photometry obtained as part of the ESO Imaging Survey to search the near solar metalicity open clusters NGC2287 and NGC3532 for their oldest WD members. These populations have several key properties which make them especially suitable for probing the uppermost reaches of the IFMR. NGC2287 and NGC3532 are relatively nearby, m-M=$9.26^{+0.37}_{-0.32}$ (Sharma et al. 2006) and m-M=$8.04^{+0.37}_{-0.32}$ (Robichon et al. 1999) respectively and despite both residing at low Galactic latitude, extinction along these lines of sight is low, E(B-V)≈0.01 (Sharma et al. 2006) and E(B-V)≈0.04 (Fernandez & Salgado 1980) respectively. Consequently, intrinsically faint high mass WD members appear comparatively bright and can be studied in detail with spectrographs on 8m class telescopes in reasonable integration times. Moreover, the ages of both clusters are relatively well constrained (NGC2287, $\tau \approx 240 \pm 40$Myrs; NGC3532, $\tau \approx 300 \pm 25$Myrs; from Dobbie et al. 2009), curtailing uncertainty in the progenitor mass determinations. Finally, NGC2287 and NGC3532 are sufficiently mature to have formed WD populations, but still young enough that the oldest and probably most massive of degenerates remain at $T_{\rm eff} \gtrsim 12500$K where there is excellent agreement between spectroscopic and gravitational redshift mass determinations (e.g. Bergeron et al. 1995; Casewell et al. 2009).

Photographic plate surveys and spectroscopic follow-up investigations have, to date, led to the identification of two and four WD members of NGC2287 and NGC3532 respectively (e.g. Reimers & Koester 1993, Dobbie et al. 2009). However, none of these degenerates appears to have formed from a progenitor with a mass $M_{\rm init} \geq 5 M_{\odot}$. As both NGC2287 and NGC3532 are comparatively well populated it is probable that there are older fainter WD members that have merely remained undetected until now because they lie beyond the limits of the early surveys. For example, NGC3532 has an estimated total mass of $M_{\rm cluster} \gtrsim 2000 M_{\odot}$ (Fernandez & Salgado 1980). This is about twice the mass of the 150Myr old cluster NGC2516 (Jeffries et al. 2001), the central regions of which harbour four massive WDs from progenitor stars with $M_{\rm init} \approx 5$ - $5.5 M_{\odot}$ (Figure 1). On the reasonable assumption that the initial mass functions of these two populations were comparable in form, it could be expected that 4±2 degenerates from $M_{\rm init} \geq$5.5-6$M_{\odot}$ reside within NGC3532. Based on the number of known WDs in our two target clusters, approximately half this number could be expected to reside in NGC2287.

V,V-I colour-magnitude diagrams for NGC2287 and NGC3532, based on the newer CCD data, are shown in Figure 2. The expected location of the WD cooling sequence in each cluster is highlighted (Holberg & Bergeron 2006, Fontaine et al. 2001). We identify a total of 15 objects, five in NGC2287 and ten in NGC3532 lying in the vicinity of the theoretical tracks, with magnitudes fainter (and which are thus probably older) than the known WD cluster members. Follow-up low-resolution multiobject spectroscopy obtained with the AAT and AAOmega confirms that 10 objects, 4 in NGC2287 and 6 in NGC3532, show the broad hydrogen Balmer lines characteristic of DA WDs. Our extensive experience of modelling these lines indicates that to obtain the accurate measurements on both the effective temperature and the surface gravity needed to reliably assess cluster membership status and estimate initial and final masses, we require a S/N≥50 per ∼5Å binned pixel

8. Description of the proposed programme (continued)

from H-β to H-8/-9. AAOmega spectroscopy cannot be of sufficient quality to achieve this goal due to the poor response of this system shortward of $\sim$4000Å where the key gravity diagnostic H-ϵ, H-8 and H-9 lines are located (see Figure 3).

B) Immediate Objective:

With the proposed FORS spectra of the WD candidate members of NGC2287 and NGC3532 we will:

(1) accurately constrain their effective temperatures and surface gravities by comparing the observed profiles of the H-8 to H-β lines to the predictions of state-of-the-art model atmospheres. Our extensive previous experience of the line fitting process indicates that S/N$\geq$50 per pixel at a resolution $\approx$10Å allows formal uncertainties to be kept to $\pm$few 100K and $\pm$few 0.01 dex in effective temperature and surface gravity respectively. We will determine the absolute visual magnitude of each star by refering to the model WD photometry of Holberg & Bergeron (2006) and use this in conjunction with the CCD photometry to estimate distance. This will allow us to confirm (or otherwise) the cluster membership status of each WD (see Dobbie et al. 2009).

(2) estimate the masses and cooling times of the objects by refering to the WD evolutionary tracks of Fontaine et al. (2001). Our recent high resolution spectroscopic study of the Praesepe WDs has confirmed that the combination of TLUSTY model atmospheres and these evolutionary models lead to robust estimates of WD masses (see Casewell et al. 2009). The anticipated tight limits on the effective temperatures and surface gravities will allow us to constrain the mass and cooling time of each WD to ΔM<0.05M$_{\odot}$ and $\sim$10% respectively.

(3) determine the masses of the progenitor stars using solar composition stellar evolutionary models, having first estimated their lifetimes by subtracting the WD cooling times from the host cluster age. The stringent limits on the WD cooling times will ensure that the internal errors on the progenitor mass estimates are $\leq 0.25 M_{\odot}$. We will mitigate systematic uncertainty in our progenitor mass estimates by adopting for the age of each cluster the mean of several recently published determinations obtained using a number of independent stellar evolutionary model grids that all assume a moderate level of convective core overshooting (see Dobbie et al. 2009). Additionally, to translate progenitor lifetimes to masses we will use the stellar models of Girardi et al. (2000), which also assume moderate levels of core overshooting. We note that the core contraction gap observed in the sequences of the rich LMC open clusters NGC2173, SL556 and NGC2155 is best reproduced by evolutionary calculations which include moderate core overshooting (Woo et al. 2003). Moreover, data from eclipsing binary systems spanning the broad mass range $M_{\rm init}\sim$2-30M$_{\odot}$ are found to be generally consistent with stellar models which adopt moderate levels of overshooting (Claret 2007).

We anticipate that our new data points will lead to substantial progress in mapping the form of the IFMR at $M_{\rm init}>$5-6M$_{\odot}$ and augment the definition of the pivitol initial mass between the PNe and Type II SNe evolutionary channels.

C) Telescope Justification: For our primary science goals we require effective temperature and surface gravity estimates to a few 100K and a few 0.01 dex respectively. This can only be accomplished by obtaining good S/N optical spectroscopy with a resolution $\leq$10Å spanning the wavelength range 3750-5200Å. Given the magnitudes of our targets data of this quality could only be obtained in reasonable integrations times on a 8/10m class telescope with a blue sensitive spectrograph. The combination of the VLT and FORS (with the E2V CCD) offers unparalleled sensitivity at λ<4000Å. Additionally, all our targets have southern declinations so are most efficiently viewed from the southern hemisphere.

D) Observing Mode Justification (visitor or service): We need to use the E2V CCD with FORS to attain sufficient S/N at λ<4000Å. This set-up is only now available in Visitor mode.

E) Strategy for Data Reduction and Analysis: All investigators have extensive experience of reducing optical spectroscopic data. Dobbie will reduce the data. Spectral analysis of each dataset to determine effective temperature and surface gravity will be performed by Dobbie/Burleigh. Parker will contribute through his extensive knowledge of the late stages of stellar evolution.

8. Attachments (Figures)

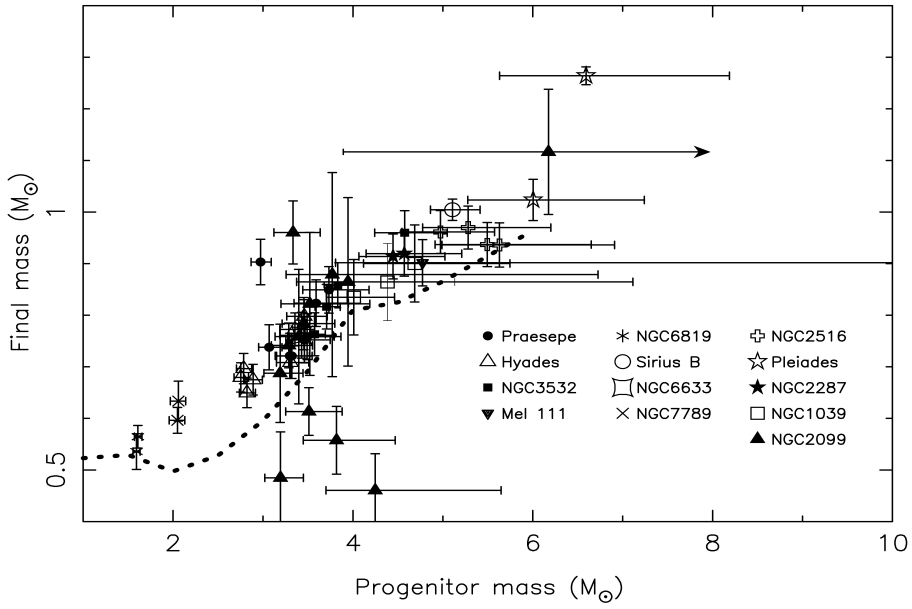


Fig. 1: The IFMR from ∼50 WD members of near-solar metalicity open star clusters (and Sirius B). The bulk of stars delineate a relatively tight trend which appears to be somewhat steeper between $3M_\odot \lesssim M_{\rm init} \lesssim 4M_\odot$ than elsewhere. The core mass as 1st TP v $M_{\rm init}$ relation from Girardi et al. (2000) is overplotted (dotted line).

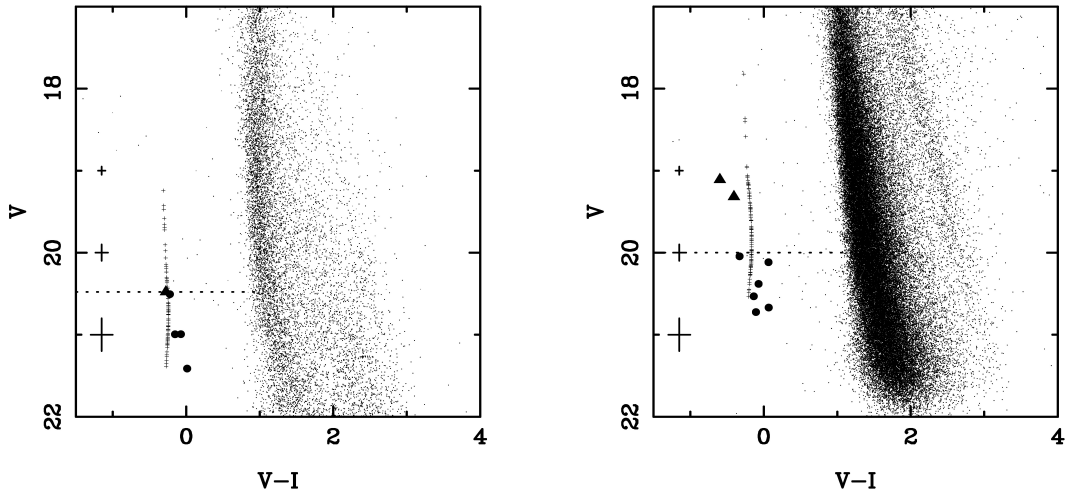


Fig. 2: V,V-I CMDs for NGC2287 (left) and NGC3532 (right) with the predicted location of the WD cooling sequence, overplotted. The faint magnitude limits of the currently known WD population in each cluster are shown (dotted horizontal lines). WD candidate members which we now wish to observe with VLT+FORS and known WD members of the two clusters are also marked (filled circles and triangles respectively).

8. Attachments (Figures)

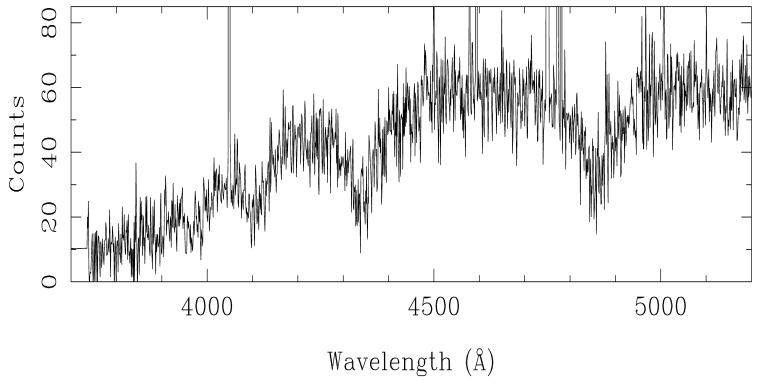


Fig. 3: An AAT and AAOmega spectrum of WD candidate WDC1107a in NGC3532. Despite 4hrs exposure the S/N is an order of magnitude below the level required to obtain accurately $T_{\rm eff}$ and log g, particularly at $\lambda \lesssim 4000$Å.

References

Bergeron, P., Wesemael, F., & Beauchamp, A. 1995, PASP, 107, 1047;
Casewell, S., et al. 2009, MNRAS, accepted;
Claret, A. 2007, A&A, 475, 1019;
Dobbie, P.D., et al. 2009, MNRAS, accepted;
Fernandez, J.A. & Salgado, C.W. 1980, A&AS, 39, 11;
Ferrario, L., et al. 2005, MNRAS, 361, 1131;
Fontaine, G., Brassard, P., Bergeron, P. 2001, PASP, 113, 409;
Girardi, L., et al. 2000, A&AS 141, 371;
Holberg, J.B., Bergeron, P. 2006, AJ, 132, 1221;
Kalirai, J., et al. 2007, ApJ, 671, 748;
Marigo, P. & Girardi, L. 2007, A&A, 469, 239;
Oswalt, T., et al. 1996, Nature, 382, 692;
Reimers, D., & Koester, D. 1993, A&A, 275, 479;
Robichon, N, et al. 1999, A&A, 345, 471;
Sharma, S., et al. 2006, AJ, 132, 1669;
Weidemann, V., 2000, A&A, 188, 74;
Williams, K.A., Bolte, M., & Koester, D. 2009, ApJ, 693, 355

9. Justification of requested observing time and lunar phase

Lunar Phase Justification: We need good S/N from λ=3750-5200Å on our faint targets. This can be achieved in modest integration times (0.5-3hrs) during grey sky conditions (7 day old moon). Indeed, all ten of our targets can be observed in only two grey nights. Data of the required quality can in principle be obtained during bright sky conditions (14 day moon), but with integration times that are $\sim$3$\times$ as long. We would be happy to accept a larger allocation of bright nights if these conditions are substantially less oversubscribed than grey time.

Time Justification: (including seeing overhead) The following numbers are based on the FORS exposure time calculator (Ver. 3.2.7). We will observe our targets using standard settings, namely with the GRIS-600B+22 grism and a 1.0" slit. With this set-up we derive exposure times of 1500s, 2700s, 5400s and 11200s to reach a S/N$\approx$50 at H-8 to H-9 for V=20.0,20.5, 21.0 and 21.5 AO stars respectively (airmass 1.4, seeing 0.8", 7 day old moon). Our S/N estimate allows for a factor 3 binning in the dispersion direction at the reduction stage, since for our purposes we require data with a resolution of only $\sim$10Å. To ensure successful cosmic ray rejection from our data the observation of each star will consist of a number of sub-integrations, namely, 3x500, 3x900,3x1800s and 4x2800s. Since our 10 targets are split approximately 2, 4, 3 and 1 per magnitude catagory listed above, we request a total time of 15.2 hours (which equates to the 2 nights in February). This estimate allows 16 minutes of overheads per OB (e.g. telescope preset, acquisition, centering on slit, CCD readout) and one observation each night of our calibration star (see below).

Calibration Request: Special Calibration - We plan to observe a DC white dwarf calibrator each night (900 secs on WD1055-072) so that we can remove to a high degree the signature of the FORS spectral response from the data.

10. Report on the use of ESO facilities during the last 2 years

Programme 79.D-0490 (PI: Dobbie), VLT/FORS1, 10 hours: published: "A new detailed examination of the white dwarf members of NGC3532 and NGC2287", Dobbie, P.D., Napiwotzki, R., Barstow, M.A., Burleigh, M.R., Jameson, R.F., MNRAS, in press, arXiv: 0902.4259

Programme 80.D-0654 (PI: Dobbie), VLT/FORS1, 14 hours: partly published: "A new detailed examination of the white dwarf members of NGC3532 and NGC2287", Dobbie, P.D., Napiwotzki, R., Barstow, M.A., Burleigh, M.R., Jameson, R.F., MNRAS, in press, arXiv: 0902.4259. Paper on remaining data about to be submitted. All results presented at 16th European Workshop on White Dwarfs (2008) and at MASH PNe workshop (02/09).

11. Applicant's publications related to the subject of this application during the last 2 years

Casewell S., Dobbie P., et al., 2009 MNRAS, accepted: *High resolution optical spectroscopy of Praesepe white dwarfs*

Dobbie P., et al., 2009 MNRAS, accepted: *A new detailed examination of white dwarfs in NGC3532 and NGC2287*

Dobbie P., et al., 2009 MNRAS, accepted: *A massive white dwarf member of the Coma Berenices Open Cluster*

12. List of targets proposed in this programme

Run	Target/Field	α(J2000)	δ(J2000)	ToT	Mag.	Diam.	Additional info	Reference star
A	WDC0644	06 44 40.28	-20 51 45.21	4.3	20.51		White dwarf	
A	WDC0645a	06 45 08.65	-20 51 30.16	4.3	21.41		White dwarf	
A	WDC0645b	06 45 22.06	-20 25 09.75	4.3	20.98		White dwarf	
A	WDC0646	06 46 01.08	-20 32 03.56	4.3	20.99		White dwarf	
A	WDC1103	11 03 27.82	-58 35 06.80	4.3	20.76		White dwarf	
A	WDC1105	11 05 23.87	-58 57 22.97	4.3	20.53		White dwarf	
A	WDC1106a	11 06 18.34	-59 05 17.27	4.3	20.04		White dwarf	
A	WDC1106b	11 06 51.64	-58 43 48.96	4.3	20.12		White dwarf	
A	WDC1107a	11 07 21.97	-58 42 12.60	4.3	20.38		White dwarf	
A	WDC1107b	11 07 29.49	-58 36 40.98	4.3	20.67		White dwarf	

12b. ESO Archive - Are the data requested by this proposal in the ESO Archive (http://archive.eso.org)? If yes, explain why the need for new data.

No

13. Scheduling requirements

14. Instrument configuration

Period	Instrument	Run ID	Parameter	Value or list
84	FORS2	A	LSS	GRIS-600B+22
84	FORS2	A	Detector	E2V

A.8 original analytical method for determining the properties of the components of the wide double-degenerate systems

.

Although, the properties of the components of the wide DD systems are now calculated using a numerical method as described in Chapter 2, the original method for determining the properties of the components of the wide DD systems utilised a purely analytical approach. This method has been added here for reference.

The masses, the cooling times and the absolute magnitudes for the white dwarfs presented here were in general obtained by comparing my spectroscopic effective temperature (T_{eff}) and surface gravity ($\log g$) estimates to grids of synthetic photometry (Holberg and Bergeron 2005) and evolutionary models (Fontaine et al. 2001). Points in these grids were interpolated using a fortran software routine, rbb_wd_dd_param.f (see Appendix Section A.5.2), to apply bi-cubic spline fits. As these binaries consisted of white dwarfs with quite different characteristics, I had to apply a number of different methods to determine their component properties.

A.8.1 Uncertainty Calculations

Uncertainties were calculated by the software by propagating the errors on the inputs (T_{eff} and $\log g$, distances and masses in some cases) using the standard error formula, given in equation (A.1).

$$\sigma^2_{output} = \sigma^2_{input1}(\frac{\mathrm{d}_{output}}{\mathrm{d}_{input1}})^2 + \sigma^2_{input2}(\frac{\mathrm{d}_{output}}{\mathrm{d}_{input2}})^2 \quad \text{(A.1)}$$

The same software used in the derivation of the output parameters was used in the derivation of the errors in the output parameters (i.e. the derivatives were calculated numerically).

For the case of a Hot DA, for example within a Hot + Hot DA scenario, this involved propagating the error on the spectroscopic surface gravity and temperature through for the calculation of the absolute magnitude, mass, and cooling age errors. In this case the first input parameter was T_{eff} and the second input parameter was $\log g$. The distance error was calculated using both the apparent magnitude (g band) error and absolute magnitude error.

For the case of an isolated Cool DA, for example within a Cool DA + Cool DA (or Cool DA without Hot DA partner) scenario or a magnetic DA + cool DA scenario, this involved propagating the errors on the spectroscopic surface gravities and temperatures through for the calculation of the mass error (where the first input parameter was T_{eff} and the second input parameter was $\log g$), and it then involved propagating the error on the temperature and corrected mass and through for the calculation of the absolute magnitude, surface gravities, and cooling age errors (where the first input parameter was T_{eff} and the second input parameter was M). The distance error was calculated using both the apparent magnitude (g band) error and absolute magnitude error.

The error of a particular output parameter was calculated as follows;

First it involved calculation of the maximum error in the output parameter when the first input parameter (IP_{First}) was set to its two respective extremes, divided by the maximum error in the first input parameter;

$$\frac{\mathrm{d}OP}{\mathrm{d}IP_{First}} = \frac{OP_{WhenFirstIPsetatMinErrRange} - OP_{WhenFirstIPSetAtMaxErrRange}}{IP_{FirstsetatMaxErrRange} - IP_{FirstsetatMinErrRange}} \tag{A.2}$$

Next it involved the calculation of the maximum error in the output parameter when the second input parameter (IP_{Second}) was set to its two respective extremes, divided by the maximum error in the second input parameter;

$$\frac{\mathrm{d}OP}{\mathrm{d}IP_{Second}} = \frac{OP_{WhenSecondIPsetatMinErrRange} - OP_{WhenSecondIPSetAtMaxErrRange}}{IP_{SecondsetatMaxErrRange} - IP_{SecondsetatMinErrRange}} \tag{A.3}$$

And finally, the final error in the output parameter (OP) was calculated as follows;

$$\mathrm{d}OP = \sqrt{((\frac{\mathrm{d}OP}{\mathrm{d}IP_{Second}}^2 IP^2_{SecondErr}) + (\frac{\mathrm{d}OP}{\mathrm{d}IP_{First}}^2 IP_{FirstErr})^2)} \tag{A.4}$$

The distance error was calculated as follows, where OP_{AbsMag} is the absolute magnitude output parameter, and IP_{AppMag} is the apparent magnitude input parameter;

$$\mathrm{d}d_{OP} = \sqrt{\mathrm{d}OP^2_{AbsMag} + \mathrm{d}IP^2_{AppMag}} \tag{A.5}$$

For the case of a non-isolated Cool DA (i.e. with a Hot DA partner) or magnetic DA (with a non-magnetic partner), for example within a Cool DA + Hot DA, Magnetic DA and Hot DA, or Magnetic DA + Cool DA scenario, where one of the input parameters was a distance value and an absolute magnitude was inferred based upon this distance value and the apparent magnitudes of the system, then the errors of the apparent magnitudes of the system were also propagated through to the errors in the output parameters. This therefore involved propagating the errors in the distance, apparent magnitudes and temperature through for the calculation of the mass, absolute magnitude, surface gravities, and cooling age errors. In this case the first input parameter was T_{eff}, the second input parameter was distance, and the third input parameter was the apparent magnitude (in g band).

This method therefore additionally involved the calculation of the maximum error in the output parameter when the third input parameter was set to its two respective extremes, divided by the maximum error in the third input parameter;

$$\frac{\mathrm{d}OP}{\mathrm{d}IP_{Third}} = \frac{OP_{WhenThirdIPsetatMinErrRange} - OP_{WhenThirdIPSetAtMaxErrRange}}{IP_{ThirdsetatMaxErrRange} - IP_{ThirdsetatMinErrRange}} \tag{A.6}$$

Where the final error in the output parameter was calculated as follows;

$$\mathrm{d}OP = \sqrt{((\frac{\mathrm{d}OP}{\mathrm{d}IP_{First}}^2 IP^2_{FirstErr}) + ... + (\frac{\mathrm{d}OP}{\mathrm{d}IP_{Third}}^2 IP^2_{ThirdErr}))} \tag{A.7}$$

Main Sequence Progenitor Calculations

For each WD component, an estimate of the DD system age may be derived by first calculating the initial (main sequence progenitor) mass of the WD component using a known IFMR form, for example Dobbie et al. (2006), calculating the age of the progenitor using the main sequence evolutionary grids of Girardi et al. (2000), and then adding the cooling age of the WD to its main sequence progenitor age. This method has been used extensively throughout this project for various purposes.

All system age comparisons using this method were derived using the Dobbie et al. (2006) linear IFMR form in this commentary. These system age calculations have been performed using code described in Appendix Section A.6.1, and my DD Data spreadsheet referenced in Appendix Section A.7.1.

A.9 example IDL routine

```
.

; Title: Count Close Pairs
; File: CCPtestDetailed.pro
; Author: Richard Baxter
; Organisation: Macquarie University / AAO
; Version: 25 March 09a
; Summary:
;
;   Postconditions;
;
;   This program generates a list of blue object pairs within a given angular separation. In the
    output list each close pair is listed twice, A with B and B with A. The list has been ordered
    in terms of pair distance, DISTANCE_BETWEEN_PAIR. The output files are of the following format;
;
;   SDSSblueObjectsDR7detailedClosePairsSortedOutputPart1.txt [RIGHT_ASCENSION A, DECLINATION A,
    GALATIC_LATITUDE A, GALATIC_LONGITUDE A, U A, G A, R A, POK A, DISTANCE_BETWEEN_PAIR,
    RIGHT_ASCENSION B, DECLINATION B, GALATIC_LATITUDE B, GALATIC_LONGITUDE B, U B, G B, R B, POK B
    ]
;   SDSSblueObjectsDR7detailedClosePairsSortedOutputPart2.txt [ID A, ID B]
;
;   Preconditions
;
;   The input files are of the following format;
;
;   repchr.pro [NASA]
;   gettok.pro [NASA]
;   readcol64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   remchar.pro [NASA]
;   skip_lun.pro [required for GDL]
;   strsplit.pro [GDL library]
;   strnumber64bitvars.pro [RBB - modified NASA pro for 64bit unsigned longs {ID}]
;   SDSSblueObjectsDR7detailed.txt [ID, RIGHT_ASCENSION, DECLINATION, GALATIC_LATITUDE,
    GALATIC_LONGITUDE, ID, U, G, R, POK] for 56587 blue objects, tab delimited


;distances are given in arc seconds here

firstPairMaxDistance = 60
lastPairMaxDistance = 60
diffPairMaxDistance = 1

;ORIGINAL;
fieldSquareDegrees = 1000L  ; or 1000   ;or 11663
fieldNumberOfStars = 4852L  ; or 4852   ;or 56587
numberOfSamples = 100L

;TEST HARNESS - load from file;
fieldSquareDegrees = 11663L         ; or 1000   ;or 11663
fieldNumberOfStars = 56587L         ; or 4852   ;or 56587
numberOfSamples = 1L

fieldxDegrees = SQRT(fieldSquareDegrees)    ; or 100
fieldyDegrees = SQRT(fieldSquareDegrees)    ; or 100

IDLrandomNumberMin = 0L
IDLrandomNumberMax = 1L
IDLrandomNumberVariation = (IDLrandomNumberMax - IDLrandomNumberMin)

for maxDistanceIndex=firstPairMaxDistance, lastPairMaxDistance, diffPairMaxDistance do begin
```

```
maxDistanceForConsideredPairInArcSeconds = double(maxDistanceIndex)
maxDistanceForConsideredPairInArcDegrees = maxDistanceForConsideredPairInArcSeconds/60D/60D
totalNumberOfPairsFoundAcrossAllSamples = 0L

for sampleIndex=1L, numberOfSamples, 1L do begin

        numberOfLines = 56587
        numberOfColumns = 5
        NameOfFile = 'SDSSblueObjectsDR7detailed.txt'

            ;long precision is required for ID
        readcol64bitvars, NameOfFile, ID, RIGHT_ASCENSION, DECLINATION, GALATIC_LATITUDE,
            GALATIC_LONGITUDE, U, G, R, POK, FORMAT='G,D,D,D,D,D,D,D,D', SKIPLINE=1

        starID = transpose(ID)
        starYCoordinates = transpose(DECLINATION) ;GALATIC_LATITUDE
        starXCoordinates = transpose(RIGHT_ASCENSION) ;GALATIC_LONGITUDE

    starXCoordinatesSortIndicies = sort(starXCoordinates)
    starXCoordinatesSorted = starXCoordinates[starXCoordinatesSortIndicies]
    starYCoordinatesSortIndicies = sort(starYCoordinates)
    starYCoordinatesSorted = starYCoordinates[starYCoordinatesSortIndicies]

    maxNumberOfPairsExpected = 10000 ;300
    starIDArrayForClosePairs1 = lon64arr(maxNumberOfPairsExpected)
    starIDArrayForClosePairs2 = lon64arr(maxNumberOfPairsExpected)
    starDistanceArrayForClosePairs = dblarr(maxNumberOfPairsExpected)
    starDataArrayForClosePairs1 = dblarr(8, maxNumberOfPairsExpected)
    starDataArrayForClosePairs2 = dblarr(8, maxNumberOfPairsExpected)
    data = [transpose(RIGHT_ASCENSION), transpose(DECLINATION), transpose(GALATIC_LATITUDE),
         transpose(GALATIC_LONGITUDE), transpose(U), transpose(G), transpose(R), transpose(POK)]

    ;PRINT, "data = "
    ;PRINT, data

    ;stringmessage = "starXCoordinatesSorted ="
    ;PRINT, starXCoordinatesSorted

    totalNumberOfPairsFound = 0L

    for starIndex=1L, fieldNumberOfStars, 1L do begin

        numberOfPairsFound = 0L

        starIndexArray = starIndex-1

        centralPositionInStarXCoordinatesSorted = where(starXCoordinatesSortIndicies eq
             starIndexArray)
        centralValueInStarXCoordinatesSorted = starXCoordinatesSorted[
             centralPositionInStarXCoordinatesSorted]
            ;centralPositionInStarXCoordinates = starIndexArray
            ;or centralValueInStarXCoordinates = starXCoordinates[
                 centralPositionInStarXCoordinates]
        centralPositionInStarYCoordinatesSorted = where(starYCoordinatesSortIndicies eq
             starIndexArray)
        centralValueInStarYCoordinatesSorted = starYCoordinatesSorted[
             centralPositionInStarYCoordinatesSorted]
            ;centralPositionInStarYCoordinates = starIndexArray
            ;or centralValueInStarYCoordinates = starYCoordinates[
                 centralPositionInStarYCoordinates]

            ;hunt in positive x direction
        consideredPairIndex = 1L
        currentMaxDistanceBetweenPairX = 0D

        while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
             begin

            if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) gt (
                 fieldNumberOfStars-1) then begin
                currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
            endif else begin

                currentPositionInStarXCoordinatesSorted =
                     centralPositionInStarXCoordinatesSorted + consideredPairIndex
                currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
                     currentPositionInStarXCoordinatesSorted]
                currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
                     currentPositionInStarXCoordinatesSorted]
                currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
                currentValueInStarYCoordinates = starYCoordinates[
                     currentPositionInStarYCoordinates]
                currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

                currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
                     centralValueInStarXCoordinatesSorted)
                currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
                     centralValueInStarYCoordinatesSorted)
                ;averageYDeclination = currentValueInStarYCoordinatesSorted +
                     centralValueInStarYCoordinatesSorted) / 2
                averageYDeclination = centralValueInStarYCoordinatesSorted
```

```
            currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
                currentMaxDistanceBetweenPairX
            currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

            currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
                 currentMaxDistanceBetweenPairYCorrected^2)

            if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
                then begin

                starIDArrayForClosePairs1[totalNumberOfPairsFound+numberOfPairsFound] =
                    starID[starIndexArray]
                starIDArrayForClosePairs2[totalNumberOfPairsFound+numberOfPairsFound] =
                    starID[currentPositionInStarXCoordinates]
                starDistanceArrayForClosePairs[totalNumberOfPairsFound+numberOfPairsFound] =
                     currentMaxDistanceBetweenPair
                starDataArrayForClosePairs1[*, totalNumberOfPairsFound+numberOfPairsFound] =
                     data[*, starIndexArray]
                starDataArrayForClosePairs2[*, totalNumberOfPairsFound+numberOfPairsFound] =
                     data[*, currentPositionInStarXCoordinates]

                numberOfPairsFound = numberOfPairsFound + 1

            endif

            consideredPairIndex = consideredPairIndex + 1

        endelse

    endwhile

        ;hunt in negative x direction
    consideredPairIndex = -1
    currentMaxDistanceBetweenPairX = 0
    while (currentMaxDistanceBetweenPairX lt maxDistanceForConsideredPairInArcDegrees) do
         begin

        if (centralPositionInStarXCoordinatesSorted+consideredPairIndex) lt 0 then begin
            currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees+9999
        endif else begin

            currentPositionInStarXCoordinatesSorted =
                centralPositionInStarXCoordinatesSorted + consideredPairIndex
            currentValueInStarXCoordinatesSorted = starXCoordinatesSorted[
                currentPositionInStarXCoordinatesSorted]
            currentPositionInStarXCoordinates = starXCoordinatesSortIndicies[
                currentPositionInStarXCoordinatesSorted]
            currentPositionInStarYCoordinates = currentPositionInStarXCoordinates
            currentValueInStarYCoordinates = starYCoordinates[
                currentPositionInStarYCoordinates]
            currentValueInStarYCoordinatesSorted = currentValueInStarYCoordinates

            currentMaxDistanceBetweenPairX = ABS(currentValueInStarXCoordinatesSorted -
                centralValueInStarXCoordinatesSorted)
            currentMaxDistanceBetweenPairY = ABS(currentValueInStarYCoordinatesSorted -
                centralValueInStarYCoordinatesSorted)
            ;averageYDeclination = (currentValueInStarYCoordinatesSorted +
                centralValueInStarYCoordinatesSorted) / 2
            averageYDeclination = centralValueInStarYCoordinatesSorted
            currentMaxDistanceBetweenPairXCorrected = COS(averageYDeclination/180.0*!PI) *
                currentMaxDistanceBetweenPairX
            currentMaxDistanceBetweenPairYCorrected = currentMaxDistanceBetweenPairY

            currentMaxDistanceBetweenPair = SQRT(currentMaxDistanceBetweenPairXCorrected^2 +
                 currentMaxDistanceBetweenPairYCorrected^2)

            if (currentMaxDistanceBetweenPair lt maxDistanceForConsideredPairInArcDegrees)
                then begin

                starIDArrayForClosePairs1[totalNumberOfPairsFound+numberOfPairsFound] =
                    starID[starIndexArray]
                starIDArrayForClosePairs2[totalNumberOfPairsFound+numberOfPairsFound] =
                    starID[currentPositionInStarXCoordinates]
                starDistanceArrayForClosePairs[totalNumberOfPairsFound+numberOfPairsFound] =
                     currentMaxDistanceBetweenPair
                starDataArrayForClosePairs1[*, totalNumberOfPairsFound+numberOfPairsFound] =
                     data[*, starIndexArray]
                starDataArrayForClosePairs2[*, totalNumberOfPairsFound+numberOfPairsFound] =
                     data[*, currentPositionInStarXCoordinates]

                numberOfPairsFound = numberOfPairsFound + 1
                ;currentMaxDistanceBetweenPairX = maxDistanceForConsideredPairInArcDegrees
                    +9999

            endif

            consideredPairIndex = consideredPairIndex - 1

        endelse

    endwhile
```

```
            totalNumberOfPairsFound = totalNumberOfPairsFound+numberOfPairsFound
        endfor

        stringmessage = "totalNumberOfPairsFound = "
        PRINT, stringmessage
        PRINT, (totalNumberOfPairsFound/2)

        totalNumberOfPairsFoundAcrossAllSamples = totalNumberOfPairsFoundAcrossAllSamples +
            totalNumberOfPairsFound
    endfor

    ;PRINT, starIDArrayForClosePairs
    starDistanceArrayForClosePairs = starDistanceArrayForClosePairs[0: (
        totalNumberOfPairsFoundAcrossAllSamples-1)] ;clip starDistanceArrayForClosePairs
    starDistanceArrayForClosePairsSortIndicies = sort(starDistanceArrayForClosePairs)
    starIDArrayForClosePairsSorted1 = starIDArrayForClosePairs1[
        starDistanceArrayForClosePairsSortIndicies]
    starIDArrayForClosePairsSorted2 = starIDArrayForClosePairs2[
        starDistanceArrayForClosePairsSortIndicies]
    starDistanceArrayForClosePairsSorted = starDistanceArrayForClosePairs[
        starDistanceArrayForClosePairsSortIndicies]
    starDataArrayForClosePairsSorted1 = starDataArrayForClosePairs1[*,
        starDistanceArrayForClosePairsSortIndicies]
    starDataArrayForClosePairsSorted2 = starDataArrayForClosePairs2[*,
        starDistanceArrayForClosePairsSortIndicies]


    starIDArrayForClosePairsSorted1 = transpose(starIDArrayForClosePairsSorted1)
    starIDArrayForClosePairsSorted2 = transpose(starIDArrayForClosePairsSorted2)
    starDistanceArrayForClosePairsSorted = transpose(starDistanceArrayForClosePairsSorted)


    finalData = [starDataArrayForClosePairsSorted1, starDistanceArrayForClosePairsSorted,
        starDataArrayForClosePairsSorted2]
    finalDataID = [starIDArrayForClosePairsSorted1, starIDArrayForClosePairsSorted2]

    PRINT, "finalData = "
    PRINT, finalData
    PRINT, "finalDataID = "
    PRINT, finalDataID

    outputFileName1='SDSSblueObjectsDR7detailedClosePairsSortedOutputPart1.txt'
    OPENW,1,outputFileName1
    PRINTF,1,finalData,FORMAT='(F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1X,F,1
        X,F,1X,F,1X,F)'
    CLOSE,1
    outputFileName2='SDSSblueObjectsDR7detailedClosePairsSortedOutputPart2.txt'
    OPENW,1,outputFileName2
    PRINTF,1,finalDataID,FORMAT='(I18,1X,I18)'
    CLOSE,1

    stringmessage = "maxDistanceForConsideredPairInArcSeconds = "
    PRINT, stringmessage
    PRINT, maxDistanceForConsideredPairInArcSeconds

    stringmessage = "fieldNumberOfStars = "
    PRINT, stringmessage
    PRINT, fieldNumberOfStars

    stringmessage = "fieldSquareDegrees = "
    PRINT, stringmessage
    PRINT, fieldSquareDegrees

    averageNumberOfPairsFound = (DOUBLE(totalNumberOfPairsFoundAcrossAllSamples)/DOUBLE(
        numberOfSamples))/DOUBLE(2)
    stringmessage = "averageNumberOfPairsFound = "
    PRINT, stringmessage
    PRINT, averageNumberOfPairsFound

endfor

end
```

List of Symbols

The following list is neither exhaustive nor exclusive, but may be helpful.

$''$ ..... arcsecond

mas ..... milliarcsecond

$M_\odot$ ..... solar mass

τ_c ..... cooling time

T_{eff} ..... effective temperature

$\log g$ ..... log of surface gravity

References

K. N. Abazajian, J. K. Adelman-McCarthy, M. A. Ageros, S. S. Allam, C. Allende Prieto, D. An, K. S. J. Anderson, S. F. Anderson, J. Annis, N. A. Bahcall, C. A. L. Bailer-Jones, J. C. Barentine, B. A. Bassett, A. C. Becker, T. C. Beers, E. F. Bell, V. Belokurov, A. A. Berlind, E. F. Berman, M. Bernardi, S. J. Bickerton, D. Bizyaev, J. P. Blakeslee, M. R. Blanton, J. J. Bochanski, W. N. Boroski, H. J. Brewington, J. Brinchmann, J. Brinkmann, R. J. Brunner, T. Budavri, L. N. Carey, S. Carliles, M. A. Carr, F. J. Castander, D. Cinabro, A. J. Connolly, I. Csabai, C. E. Cunha, P. C. Czarapata, J. R. A. Davenport, E. de Haas, B. Dilday, M. Doi, D. J. Eisenstein, M. L. Evans, N. W. Evans, X. Fan, S. D. Friedman, J. A. Frieman, M. Fukugita, B. T. Gnsicke, E. Gates, B. Gillespie, G. Gilmore, B. Gonzalez, C. F. Gonzalez, E. K. Grebel, J. E. Gunn, Z. Gyry, P. B. Hall, P. Harding, F. H. Harris, M. Harvanek, S. L. Hawley, J. J. E. Hayes, T. M. Heckman, J. S. Hendry, G. S. Hennessy, R. B. Hindsley, J. Hoblitt, C. J. Hogan, D. W. Hogg, J. A. Holtzman, J. B. Hyde, S.-i. Ichikawa, T. Ichikawa, M. Im, . Ivezi, S. Jester, L. Jiang, J. A. Johnson, A. M. Jorgensen, M. Juri, S. M. Kent, R. Kessler, S. J. Kleinman, G. R. Knapp, K. Konishi, R. G. Kron, J. Krzesinski, N. Kuropatkin, H. Lampeitl, S. Lebedeva, M. G. Lee, Y. S. Lee, R. French Leger, S. Lpine, N. Li, M. Lima, H. Lin, D. C. Long, C. P. Loomis, J. Loveday, R. H. Lupton, E. Magnier, O. Malanushenko, V. Malanushenko, R. Mandelbaum, B. Margon, J. P. Marriner, D. Martnez-Delgado, T. Matsubara, P. M. McGehee, T. A. McKay, A. Meiksin, H. L. Morrison, F. Mullally, J. A. Munn, T. Murphy, T. Nash, A. Nebot, E. H. Neilsen, H. J. Newberg, P. R. Newman, R. C. Nichol, T. Nicinski, M. Nieto-Santisteban, A. Nitta, S. Okamura, D. J. Oravetz, J. P. Ostriker, R. Owen, N. Padmanabhan, K. Pan, C. Park, G. Pauls, J. Peoples, W. J. Percival, J. R. Pier, A. C. Pope, D. Pourbaix, P. A. Price, N. Purger, T. Quinn, M. J. Raddick, P. Re Fiorentin, G. T. Richards, M. W. Richmond, A. G. Riess, H. Rix, C. M. Rockosi, M. Sako, D. J. Schlegel, D. P. Schneider, R. Scholz, M. R. Schreiber, A. D. Schwope, U. Seljak, B. Sesar, E. Sheldon, K. Shimasaku, V. C. Sibley, A. E. Simmons, T. Sivarani, J. Allyn Smith, M. C. Smith, V. Smoli, S. A. Snedden, A. Stebbins, M. Steinmetz, C. Stoughton, M. A. Strauss, M. SubbaRao, Y. Suto, A. S. Szalay, I. Szapudi, P. Szkody, M. Tanaka, M. Tegmark, L. F. A. Teodoro, A. R. Thakar,

C. A. Tremonti, D. L. Tucker, A. Uomoto, D. E. Vanden Berk, J. Vandenberg, S. Vidrih, M. S. Vogeley, W. Voges, N. P. Vogt, Y. Wadadekar, S. Watters, D. H. Weinberg, A. A. West, S. D. M. White, B. C. Wilhite, A. C. Wonders, B. Yanny, D. R. Yocum, D. G. York, I. Zehavi, S. Zibetti, and D. B. Zucker. The seventh data release of the sloan digital sky survey. *The Astrophysical Journal Supplement Series*, 182:543–558, June 2009. 23, 77

H. A. Abt and S. G. Levy. Multiplicity among solar-type stars. *The Astrophysical Journal Supplement Series*, 30:273–306, Mar. 1976. 79

N. F. Allard, G. Hbrard, J. Dupuis, P. Chayer, J. W. Kruk, J. Kielkopf, and I. Hubeny. Far ultraviolet spectroscopic explorer observations of g226-29: First detection of the h2 quasi-molecular satellite at 1150 . *The Astrophysical Journal*, 601:L183–L186, Feb. 2004. 59

L. H. Auer and D. Mihalas. Non-Lte model atmospheres. III. a Complete-Linearization method. *The Astrophysical Journal*, 158:641, Nov. 1969. 57

M. R. Bate, I. A. Bonnell, and N. M. Price. Modelling accretion in protobinary systems. *Monthly Notices of the Royal Astronomical Society*, 277:362–376, Nov. 1995. 14

P. Bergeron, R. A. Saffer, and J. Liebert. A spectroscopic determination of the mass distribution of DA white dwarfs. *Astrophysical Journal*, 394:228–247, July 1992. 13, 60, 80

P. Bergeron, D. Saumon, and F. Wesemael. New model atmospheres for very cool white dwarfs with mixed H/He and pure he compositions. *Astrophysical Journal*, 443:764–779, Apr. 1995a. 76

P. Bergeron, F. Wesemael, and A. Beauchamp. Photometric calibration of hydrogen- and Helium-Rich white dwarf models. *Publications of the Astronomical Society of the Pacific*, 107:1047, Nov. 1995b. xiv, 26, 61

P. Bergeron, M. T. Ruiz, and S. K. Leggett. The chemical evolution of cool white dwarfs and the age of the local galactic disk. *The Astrophysical Journal Supplement Series*, 108:339, Jan. 1997. 49

P. Bergeron, S. K. Leggett, and M. T. Ruiz. Photometric and spectroscopic analysis of cool white dwarfs with trigonometric parallax measurements. *Astrophysical Journal Supplement Series*, 133:413–449, Apr. 2001. 49, 61

T. Bloecker. Evolutionary tracks and observational constraints. *Acta Astronomica*, 43: 305–313, Oct. 1993. 7

G. H. Bowen. Dynamical modeling of long-period variable star atmospheres. *The Astrophysical Journal*, 329:299–317, June 1988. 7

B. J. Boyle. The space distribution of DA white dwarfs. *Monthly Notices of the Royal Astronomical Society*, 240:533–549, Oct. 1989. 9

S. L. Casewell, P. D. Dobbie, S. T. Hodgkin, E. Moraux, R. F. Jameson, N. C. Hambly, J. Irwin, and N. Lodieu. Proper motion l and t dwarf candidate members of the pleiades. *Monthly Notices of the Royal Astronomical Society*, 378:1131–1140, July 2007. 67

S. L. Casewell, P. D. Dobbie, R. Napiwotzki, M. R. Burleigh, M. A. Barstow, and R. F. Jameson. High-resolution optical spectroscopy of praesepe white dwarfs. *Monthly Notices of the Royal Astronomical Society*, 395:1795–1804, June 2009. xiii, 12, 14, 82, 114

S. Catalan, J. Isern, E. Garcia-Berro, and I. Ribas. The initial-final mass relationship of white dwarfs revisited: effect on the luminosity function and mass distribution. *0804.3034*, Apr. 2008. 89

S. Chandrasekhar. The maximum mass of ideal white dwarfs. *The Astrophysical Journal*, 74:81, July 1931. 2

S. Chandrasekhar. *An introduction to the study of stellar structure*. 1939. 2

C. F. Claver, J. Liebert, P. Bergeron, and D. Koester. The masses of white dwarfs in the praesepe open cluster. *Astrophysical Journal*, 563:987–998, Dec. 2001. 14, 21, 80

P. D. Dobbie, D. J. Pinfield, R. Napiwotzki, N. C. Hambly, M. R. Burleigh, M. A. Barstow, R. F. Jameson, and I. Hubeny. Praesepe and the seven white dwarfs. *Monthly Notices of the Royal Astronomical Society*, 355:L39–L43, Dec. 2004. 14

P. D. Dobbie, R. Napiwotzki, M. R. Burleigh, M. A. Barstow, D. D. Boyce, S. L. Casewell, R. F. Jameson, I. Hubeny, and G. Fontaine. New praesepe white dwarfs and the initial mass-final mass relation. *Monthly Notices of the Royal Astronomical Society*, 369:383–389, June 2006. xvi, xvii, xviii, 14, 15, 78, 79, 80, 82, 83, 84, 87, 94, 95, 96, 98, 99, 100, 102, 103, 104, 119, 120, 427

P. D. Dobbie, R. Napiwotzki, M. R. Burleigh, K. A. Williams, R. Sharp, M. A. Barstow, S. L. Casewell, and I. Hubeny. A new detailed examination of white dwarfs in NGC 3532 and NGC 2287. *Monthly Notices of the Royal Astronomical Society*, 395:2248–2256, June 2009. xiv, 14, 15, 16, 89

S. Dreizler and K. Werner. Spectral analysis of hot helium-rich white dwarfs. *Astronomy and Astrophysics*, 314:217–232, Oct. 1996. 8

A. K. Dupree. Mass loss from cool stars. *Annual Review of Astronomy and Astrophysics*, 24:377–420, 1986. 5

D. J. Eisenstein, J. Liebert, H. C. Harris, S. J. Kleinman, A. Nitta, N. Silvestri, S. A. Anderson, J. C. Barentine, H. J. Brewington, J. Brinkmann, M. Harvanek, J. Krzesiski, E. H. Neilsen, D. Long, D. P. Schneider, and S. A. Snedden. A catalog of spectroscopically confirmed white dwarfs from the sloan digital sky survey data release 4. *The Astrophysical Journal Supplement Series*, 167:40–58, Nov. 2006. xix, 5, 32

L. Ferrario, D. Wickramasinghe, J. Liebert, and K. A. Williams. The open-cluster initial-final mass relationship and the high-mass tail of the white dwarf distribution. *Monthly Notices of the Royal Astronomical Society*, 361:1131–1135, Aug. 2005. xiv, 15, 17

D. S. Finley and D. Koester. PG 0922+162: Discovery of the youngest visual double degenerate. *Astrophysical Journal*, 489:L79, Nov. 1997. xiv, 18, 19, 20, 105

T. A. Fleming, J. Liebert, and R. F. Green. The luminosity function of DA white dwarfs. *The Astrophysical Journal*, 308:176–189, Sept. 1986. 9, 115

G. Fontaine, P. Brassard, and P. Bergeron. The potential of white dwarf cosmochronology. *Publications of the Astronomical Society of the Pacific*, 113:409–435, Apr. 2001. 61, 75, 425

R. H. Fowler. On dense matter. *Monthly Notices of the Royal Astronomical Society*, 87:114–122, Dec. 1926. 2

D. J. Frew. Planetary nebulae in the solar neighbourhood: Statistics, distance scale and luminosity function. http://adsabs.harvard.edu/abs/2008PhDT.....109F, July 2008. 9

B. Fuchs, C. Dettbarn, H. Rix, T. C. Beers, D. Bizyaev, H. Brewington, H. Jahrei, R. Klement, E. Malanushenko, V. Malanushenko, D. Oravetz, K. Pan, A. Simmons, and S. Snedden. The kinematics of Late-Type stars in the solar cylinder studied with SDSS data. *The Astronomical Journal*, 137:4149–4159, May 2009. 69

S. D. Gennaro, T. von Hippel, D. E. Winget, S. O. Kepler, A. Nitta, D. Koester, and L. Althaus. White dwarf luminosity and mass functions from sloan digital sky survey spectra. *The Astronomical Journal*, 135:1–9, 2008. 3

A. Gianninas, P. Bergeron, and M. T. Ruiz. Spectroscopic analysis of DA white dwarfs from the McCook & sion catalog. *Journal of Physics Conference Series*, 172:2021, June 2009. xviii, 116, 117

A. Gianninas, P. Bergeron, and M. T. Ruiz. A spectroscopic survey & analysis of bright, Hydrogen-Rich white dwarfs, Sept. 2011. 110, 114

L. Girardi, A. Bressan, G. Bertelli, and C. Chiosi. Evolutionary tracks and isochrones for low- and intermediate-mass stars: From 0.15 to 7 msun, and from z=0.0004 to 0.03. *Astronomy and Astrophysics Supplement Series*, 141:371–383, Feb. 2000. 5, 78, 79, 80, 82, 83, 427

J. Girven, B. T. Gnsicke, B. Klebi, D. Steeghs, S. Jordan, T. R. Marsh, and D. Koester. PG1258+593 and its common proper motion magnetic white dwarf counterpart. *Monthly Notices of the Royal Astronomical Society*, page 271, Feb. 2010. 18

J. L. Greenstein, N. Dolez, and G. Vauclair. Physical properties and evolution of the two white dwarfs in the Sanduleak-Pesch binary. *Astronomy and Astrophysics*, 127: 25–28, Oct. 1983. 18

J. Halbwachs. Statistical studies on wide pairs. *Astrophysics and Space Science*, 142: 237–244, Mar. 1988. 28

T. Hamada and E. E. Salpeter. Models for Zero-Temperature stars. *The Astrophysical Journal*, 134:683, Nov. 1961. 2, 3

R. S. Harrington. Stability criteria for triple stars. *Celestial Mechanics*, 6:322–327, Nov. 1972. 79

H. C. Harris, J. Liebert, S. J. Kleinman, A. Nitta, S. F. Anderson, G. R. Knapp, J. Krzesiski, G. Schmidt, M. A. Strauss, D. V. Berk, D. Eisenstein, S. Hawley, B. Margon, J. A. Munn, N. M. Silvestri, J. A. Smith, P. Szkody, M. J. Collinge, C. C. Dahn, X. Fan, P. B. Hall, D. P. Schneider, J. Brinkmann, S. Burles, J. E. Gunn, G. S. Hennessy, R. Hindsley, Z. Ivezi, S. Kent, D. Q. Lamb, R. H. Lupton, R. C. Nichol, J. R. Pier, D. J. Schlegel, M. SubbaRao, A. Uomoto, B. Yanny, and D. G. York. An initial survey of white dwarfs in the sloan digital sky survey. *Astronomical Journal*, 126:1023–1040, Aug. 2003. xiv, 25, 26

H. C. Harris, J. A. Munn, M. Kilic, J. Liebert, K. A. Williams, T. von Hippel, S. E. Levine, D. G. Monet, D. J. Eisenstein, S. J. Kleinman, T. S. Metcalfe, A. Nitta, D. E. Winget, J. Brinkmann, M. Fukugita, G. R. Knapp, R. H. Lupton, J. A. Smith, and D. P. Schneider. The white dwarf luminosity function from sloan digital sky survey imaging data. *The Astronomical Journal*, 131:571–581, Jan. 2006. 9

F. Herwig. Evolution of asymptotic giant branch stars. *Annual Review of Astronomy and Astrophysics*, 43:435–479, Sept. 2005. xiii, 6, 11

J. B. Holberg and P. Bergeron. Calibration of synthetic photometry using DA white dwarfs. volume 207, page 1158, Dec. 2005. 425

J. B. Holberg and P. Bergeron. Calibration of synthetic photometry using DA white dwarfs. *Astronomical Journal*, 132:1221–1233, Sept. 2006. 61, 75

J. B. Holberg, R. A. Saffer, R. W. Tweedy, and M. A. Barstow. The binary double-degenerate nature of the bright DAO white dwarf feige 55. *Astrophysical Journal*, 452:L133, Oct. 1995. 18

J. B. Holberg, T. D. Oswalt, and E. M. Sion. A determination of the local density of white dwarf stars. *The Astrophysical Journal*, 571:512–518, May 2002. 9

J. B. Holberg, E. M. Sion, T. Oswalt, G. P. McCook, S. Foran, and J. P. Subasavage. A new look at the local white dwarf population. *Astronomical Journal*, 135:1225–1238, Apr. 2008. 9

D. Homeier, D. Koester, H. Hagen, S. Jordan, U. Heber, D. Engels, D. Reimers, and S. Dreizler. An analysis of DA white dwarfs from the hamburg quasar survey. *Astronomy and Astrophysics*, 338:563–575, Oct. 1998. xiv, 19, 20

D. A. Howell, M. Sullivan, P. E. Nugent, R. S. Ellis, A. J. Conley, D. L. Borgne, R. G. Carlberg, J. Guy, D. Balam, S. Basa, D. Fouchez, I. M. Hook, E. Y. Hsiao, J. D. Neill, R. Pain, K. M. Perrett, and C. J. Pritchet. The type ia supernova SNLS-03D3bb from a super-Chandrasekhar-mass white dwarf star. *Nature*, 443:308–311, Sept. 2006. 11

I. Hubeny. A computer program for calculating non-LTE model stellar atmospheres. *Computer Physics Communications*, 52:103–132, Dec. 1988. 57, 58, 59

I. Hubeny and T. Lanz. Non-LTE line-blanketed model atmospheres of hot stars. 1: Hybrid complete linearization/accelerated lambda iteration method. *The Astrophysical Journal*, 439:875–904, Feb. 1995. 57, 59

I. Hubeny and T. Lanz. Synspec version fourty eight. 2001. URL `http://nova.astro.umd.edu/`. 57, 59

D. G. Hummer and D. Mihalas. The equation of state for stellar envelopes. i - an occupation probability formalism for the truncation of internal partition functions. *The Astrophysical Journal*, 331:794–814, Aug. 1988. 59

I. Iben. Stellar evolution. II. the evolution of a 3 m_sun star from the main sequence through core helium burning. *The Astrophysical Journal*, 142:1447, Nov. 1965. 5

I. Iben. Mass Transfer/Loss from AGB stars in close binaries. volume 199, page 107, 2000. 114

I. Iben and M. Livio. Common envelopes in binary star evolution. *Publications of the Astronomical Society of the Pacific*, 105:1373–1406, Dec. 1993. 5, 80, 83

I. Iben and A. Renzini. Asymptotic giant branch evolution and beyond. *Annual Review of Astronomy and Astrophysics*, 21:271–342, 1983. 11

I. Iben and A. V. Tutukov. On the evolution of close binaries with components of initial mass between 3 solar masses and 12 solar masses. *The Astrophysical Journal Supplement Series*, 58:661–710, Aug. 1985. 79, 84

J. Isern, R. Mochkovitch, E. Garcia-Berro, and M. Hernanz. The physics of crystallizing white dwarfs. *The Astrophysical Journal*, 485:308, Aug. 1997. 3

S. Jordan. Models of white dwarfs with high magnetic fields. *Astronomy and Astrophysics*, 265:570–576, Nov. 1992. 61

S. Jordan. Analysis of three magnetic DA white dwarfs. page 333, 1993. 61

S. Jordan, D. Koester, G. Vauclair, N. Dolez, U. Heber, H. Hagen, D. Reimers, M. Chevreton, and S. Dreizler. HS0507+0434: a double DA degenerate with a ZZCeti component. *Astronomy and Astrophysics*, 330:277–284, Feb. 1998a. 105

S. Jordan, D. Koester, G. Vauclair, N. Dolez, U. Heber, H. Hagen, D. Reimers, M. Chevreton, and S. Dreizler. HS0507+0434: a double DA degenerate with a ZZCeti component. *Astronomy and Astrophysics*, 330:277–284, Feb. 1998b. 18

J. S. Kalirai, H. B. Richer, D. Reitzel, B. M. S. Hansen, R. M. Rich, G. G. Fahlman, B. K. Gibson, and T. von Hippel. The Initial-Final mass relationship: Spectroscopy of white dwarfs in NGC 2099 (M37). *Astrophysical Journal*, 618:L123–L127, 2005a. 14

J. S. Kalirai, H. B. Richer, D. Reitzel, B. M. S. Hansen, R. M. Rich, G. G. Fahlman, B. K. Gibson, and T. von Hippel. Spectroscopy of faint white dwarfs - the DA/DB ratio and the Initial-Final mass relation. volume 334, page 9, July 2005b. 116

J. S. Kalirai, P. Bergeron, B. M. S. Hansen, D. D. Kelson, D. B. Reitzel, R. M. Rich, and H. B. Richer. Stellar evolution in NGC 6791: Mass loss on the red giant branch and the formation of Low-Mass white dwarfs. *Astrophysical Journal*, 671:748–760, Dec. 2007. 15

J. S. Kalirai, B. M. S. Hansen, D. D. Kelson, D. B. Reitzel, R. M. Rich, and H. B. Richer. The Initial-Final mass relation: Direct constraints at the Low-Mass end. *Astrophysical Journal*, 676:594–609, Mar. 2008. xvi, xvii, 15, 78, 79, 80, 81, 82, 83, 84, 87, 89, 94, 95, 96, 97, 98, 99, 100, 101

A. I. Karakas, J. C. Lattanzio, and O. R. Pols. Parameterising the third dredge-up in asymptotic giant branch stars. *Publications of the Astronomical Society of Australia*, 19:515–526, 2002. xiv, 16

S. O. Kepler, S. J. Kleinman, A. Nitta, D. Koester, B. G. Castanheira, O. Giovannini, A. F. M. Costa, and L. Althaus. White dwarf mass distribution in the SDSS. *Monthly Notices of the Royal Astronomical Society*, 375:1315–1324, Mar. 2007. xiii, xvii, 9, 10, 13, 25, 41, 105, 106, 108

R. Kippenhahn and A. Weigert. *Stellar Structure and Evolution*. 1990. 2

S. J. Kleinman, H. C. Harris, D. J. Eisenstein, J. Liebert, A. Nitta, J. Krzesiski, J. A. Munn, C. C. Dahn, S. L. Hawley, J. R. Pier, G. Schmidt, N. M. Silvestri, J. A. Smith, P. Szkody, M. A. Strauss, G. R. Knapp, M. J. Collinge, A. S. Mukadam, D. Koester, A. Uomoto, D. J. Schlegel, S. F. Anderson, J. Brinkmann, D. Q. Lamb, D. P. Schneider, and D. G. York. A catalog of spectroscopically identified white dwarf stars in the first data release of the sloan digital sky survey. *Astrophysical Journal*, 607:426–444, May 2004. 110

G. R. Knapp. Mass loss from evolved stars. VI - mass-loss mechanisms and luminosity evolution. *The Astrophysical Journal*, 311:731–741, Dec. 1986. 7

R. A. Knox, M. R. S. Hawkins, and N. C. Hambly. A survey for cool white dwarfs and the age of the galactic disc. *Monthly Notices of the Royal Astronomical Society*, 306: 736–752, July 1999. 9

D. Koester. White dwarf spectra and atmosphere models . *Memorie della Societa Astronomica Italiana*, 81:921, 2010. 76

D. Koester and D. Reimers. Spectroscopic identification of white dwarfs in galactic clusters. i NGC 2287 and NGC 2422. *Astronomy and Astrophysics*, 99:L8–L11, June 1981. 11

D. Koester, R. Napiwotzki, N. Christlieb, H. Drechsel, H. Hagen, U. Heber, D. Homeier, C. Karl, B. Leibundgut, S. Moehler, G. Nelemans, E. Pauli, D. Reimers, A. Renzini, and L. Yungelson. High-resolution UVES/VLT spectra of white dwarfs observed for the ESO SN ia progenitor survey (SPY). i. *Astronomy and Astrophysics*, 378: 556–568, Nov. 2001. xiv, 19, 20

M. B. N. Kouwenhoven, A. G. A. Brown, H. Zinnecker, L. Kaper, and S. F. Portegies Zwart. The primordial binary population. i. a near-infrared adaptive optics search for close visual companions to a star members of scorpius OB2. *Astronomy and Astrophysics*, 430:137–154, Jan. 2005. 114

J. Kubat. The sphericity effects in the NLTE model atmospheres of hot white dwarfs. *Astronomy and Astrophysics*, 299:803, July 1995. 57

B. Klebi, S. Jordan, F. Euchner, B. T. Gnsicke, and H. Hirsch. Analysis of hydrogen-rich magnetic white dwarfs detected in the sloan digital sky survey. *Astronomy and Astrophysics*, 506:1341–1350, Nov. 2009. 61

D. Q. Lamb and H. M. van Horn. Evolution of crystallizing pure c-12 white dwarfs. *The Astrophysical Journal*, 200:306–323, Sept. 1975. 3

S. K. Leggett, M. T. Ruiz, and P. Bergeron. The cool white dwarf luminosity function and the age of the galactic disk. *The Astrophysical Journal*, 497:294, Apr. 1998. 9

M. Lemke. Extended VCS stark broadening tables for hydrogen – lyman to brackett series. *Astronomy and Astrophysics Supplement Series*, 122:285–292, Apr. 1997. 59

J. Liebert, C. C. Dahn, and D. G. Monet. The luminosity function of white dwarfs. *The Astrophysical Journal*, 332:891–909, Sept. 1988. 9

J. Liebert, P. Bergeron, and J. B. Holberg. The formation rate and mass and luminosity functions of DA white dwarfs from the palomar green survey. *The Astrophysical Journal Supplement Series*, 156:47–68, Jan. 2005a. xiii, xiv, xvii, 9, 10, 17, 19, 20, 105, 107, 115

J. Liebert, D. T. Wickramasinghe, G. D. Schmidt, N. M. Silvestri, S. L. Hawley, P. Szkody, L. Ferrario, R. F. Webbink, T. D. Oswalt, J. A. Smith, and M. P. Lemagie. Where are the magnetic white dwarfs with detached, nondegenerate companions? *The Astronomical Journal*, 129:2376–2381, May 2005b. 84

C. W. H. D. Loore and C. Doom. Structure and evolution of single and binary stars. volume 179, 1992. 5, 6

J. Madej, M. Naleyty, and L. G. Althaus. Mass distribution of DA white dwarfs in the first data release of the sloan digital sky survey. *Astronomy and Astrophysics*, 419: L5–L8, May 2004. xiii, 10

P. Marigo and L. Girardi. Evolution of asymptotic giant branch stars. i. updated synthetic TP-AGB models and their basic calibration. *Astronomy and Astrophysics*, 469:239–263, July 2007. xiv, 11, 16

M. C. Marsh, M. A. Barstow, D. A. Buckley, M. R. Burleigh, J. B. Holberg, D. Koester, D. O'Donoghue, A. J. Penny, and A. E. Sansom. An EUV-selected sample of DA white dwarfs from the ROSAT All-Sky survey - i. optically derived stellar parameters. *Monthly Notices of the Royal Astronomical Society*, 286:369–383, Apr. 1997. xiv, 19, 20, 115

G. P. McCook and E. M. Sion. A catalog of spectroscopically identified white dwarfs. *The Astrophysical Journal Supplement Series*, 121:1–130, Mar. 1999. 4, 117

D. Mihalas. *Stellar atmospheres /2nd edition/*. 1978. 57, 58

E. Moraux, J. Bouvier, and J. R. Stauffer. Proper motion of very low mass stars and brown dwarfs in the pleiades cluster. *Astronomy and Astrophysics*, 367:211–217, Feb. 2001. 67

R. Napiwotzki, P. J. Green, and R. A. Saffer. A comparative study of the mass distribution of Extreme-Ultraviolet-selected white dwarfs. *The Astrophysical Journal*, 517:399–415, May 1999. 60, 77

M. S. O'Brien and S. D. Kawaler. Measurement of neutrino emission rates in cool Pre-White dwarfs. volume 189, page 1378, Dec. 1996. 3

M. A. C. Perryman, A. G. A. Brown, Y. Lebreton, A. Gomez, C. Turon, G. Cayrel de Strobel, J. C. Mermilliod, N. Robichon, J. Kovalevsky, and F. Crifo. The hyades: distance, structure, dynamics, and age. *Astronomy and Astrophysics*, 331:81–120, Mar. 1998. 80

S. R. Pottasch. Local space density and formation rate of planetary nebulae. *Astronomy and Astrophysics*, 307:561–578, Mar. 1996. 9

J. L. Provencal, H. L. Shipman, E. Hog, and P. Thejll. Testing the white dwarf Mass-Radius relation with HIPPARCOS. *The Astrophysical Journal*, 494:759, Feb. 1998. 4

D. Raghavan, H. A. McAlister, T. J. Henry, D. W. Latham, G. W. Marcy, B. D. Mason, D. R. Gies, R. J. White, and T. A. ten Brummelaar. A survey of stellar families: Multiplicity of solar-type stars. *The Astrophysical Journal Supplement Series*, 190: 1–42, Sept. 2010. 79

D. Reimers and D. Koester. Spectroscopic identification of white dwarfs in galactic clusters. II - NGC 2516. *Astronomy and Astrophysics*, 116:341–347, Dec. 1982. 11

A. Renzini and M. Voli. Advanced evolutionary stages of intermediate-mass stars. i - evolution of surface compositions. *Astronomy and Astrophysics*, 94:175–193, Jan. 1981. 7

K. H. R. Rubin, K. A. Williams, M. Bolte, and D. Koester. The white dwarf population in NGC 1039 (M34) and the white dwarf Initial-Final mass relation. *Astronomical Journal*, 135:2163–2176, June 2008. 14

M. Salaris, A. Serenelli, A. Weiss, and M. M. Bertolami. Semi-empirical white dwarf Initial-Final mass relationships: A thorough analysis of systematic uncertainties due to stellar evolution models. *Astrophysical Journal*, 692:1013–1032, Feb. 2009. 15, 19

E. E. Salpeter. Energy and pressure of a Zero-Temperature plasma. *The Astrophysical Journal*, 134:669, Nov. 1961. 2

E. L. Schatzman. *White dwarfs*, volume 5S. 1958. 7

G. D. Schmidt, H. C. Harris, J. Liebert, D. J. Eisenstein, S. F. Anderson, J. Brinkmann, P. B. Hall, M. Harvanek, S. Hawley, S. J. Kleinman, G. R. Knapp, J. Krzesinski, D. Q. Lamb, D. Long, J. A. Munn, E. H. Neilsen, P. R. Newman, A. Nitta, D. J. Schlegel, D. P. Schneider, N. M. Silvestri, J. A. Smith, S. A. Snedden, P. Szkody, and D. V. Berk. Magnetic white dwarfs from the sloan digital sky survey: The first data release. *Astrophysical Journal*, 595:1101–1113, Oct. 2003. 76

D. P. Schneider, G. T. Richards, P. B. Hall, M. A. Strauss, S. F. Anderson, T. A. Boroson, N. P. Ross, Y. Shen, W. N. Brandt, X. Fan, N. Inada, S. Jester, G. R. Knapp, C. M. Krawczyk, A. R. Thakar, D. E. V. Berk, W. Voges, B. Yanny, D. G. York, N. A. Bahcall, D. Bizyaev, M. R. Blanton, H. Brewington, J. Brinkmann, D. Eisenstein, J. A. Frieman, M. Fukugita, J. Gray, J. E. Gunn, P. Hibon, eljko Ivezi, S. M. Kent, R. G. Kron, M. G. Lee, R. H. Lupton, E. Malanushenko, V. Malanushenko, D. Oravetz, K. Pan, J. R. Pier, T. N. Price, D. H. Saxe, D. J. Schlegel, A. Simmons, S. A. Snedden, M. U. SubbaRao, A. S. Szalay, and D. H. Weinberg. The sloan digital sky survey quasar catalog. v. seventh data release. *The Astronomical Journal*, 139: 2360–2373, June 2010. 28

D. Schoenberner. Late stages of stellar evolution. II - mass loss and the transition of asymptotic giant branch stars into hot remnants. *The Astrophysical Journal*, 272: 708–714, Sept. 1983. 7, 8

L. Segretain, G. Chabrier, and R. Mochkovitch. The fate of merging white dwarfs. *The Astrophysical Journal*, 481:355, May 1997. 115

H. L. Shipman. Sirius b - a thermal soft x-ray source. *The Astrophysical Journal*, 206: L67–L69, May 1976. 58

E. M. Sion, J. L. Greenstein, J. D. Landstreet, J. Liebert, H. L. Shipman, and G. A. Wegner. A proposed new white dwarf spectral classification system. *The Astrophysical Journal*, 269:253–257, June 1983. 4

J. R. Stauffer, G. Schultz, and J. D. Kirkpatrick. Keck spectra of pleiades brown dwarf candidates and a precise determination of the lithium depletion edge in the pleiades. *The Astrophysical Journal*, 499:L199, June 1998. 21, 80

M. F. Sterzik and R. H. Durisen. Are binary separations related to their system mass? http://adsabs.harvard.edu//abs/2004RMxAC..21...58S, Aug. 2004. 114

R. B. Stothers and C. Chin. Tests of two convection theories for red giant and red supergiant envelopes. *The Astrophysical Journal*, 440:297, 1995. ISSN 0004-637X. doi: 10.1086/175270. URL http://adsabs.harvard.edu//abs/1995ApJ...440..297S. 5

I. Traulsen, A. I. D. Hoffmann, T. Rauch, K. Werner, S. Dreizler, and J. W. Kruk. HST and FUSE spectroscopy of hot Hydrogen-Rich central stars of planetary nebulae. volume 334, page 325, July 2005. xiii, 8

P. Tremblay, P. Bergeron, and A. Gianninas. An improved spectroscopic analysis of DA white dwarfs from the sloan digital sky survey data release 4. *The Astrophysical Journal*, 730:128, Apr. 2011. xvi, 76, 85, 86

P. G. van Dokkum. Cosmic-Ray rejection by laplacian edge detection. *Publications of the Astronomical Society of the Pacific*, 113:1420–1427, Nov. 2001. 49, 209

S. Vennes, P. A. Thejll, R. G. Galvan, and J. Dupuis. Hot white dwarfs in the extreme ultraviolet explorer survey. II. mass distribution, space density, and population age. *The Astrophysical Journal*, 480:714, May 1997. xiv, 19, 20, 115

S. Vennes, R. J. Smith, B. J. Boyle, S. M. Croom, A. Kawka, T. Shanks, L. Miller, and N. Loaring. White dwarfs in the 2dF QSO redshift survey - i. hydrogen-rich (DA) stars. *Monthly Notices of the Royal Astronomical Society*, 335:673–686, Sept. 2002. 9

A. Wachter, K. Schrder, J. M. Winters, T. U. Arndt, and E. Sedlmayr. An improved mass-loss description for dust-driven superwinds and tip-AGB evolution models. *Astronomy and Astrophysics*, 384:452–459, Mar. 2002. 7

G. Wegner, I. N. Reid, and R. K. McMahan. Gravitational redshift for the pleiad white dwarf LB 1497. *The Astrophysical Journal*, 376:186–189, July 1991. 80

V. Weidemann. Mass loss towards the white dwarf stage. *Astronomy and Astrophysics*, 59:411–418, Aug. 1977. 11

V. Weidemann. The initial/final mass relation for stellar evolution with mass loss. volume 89, pages 339–343, 1981. 11, 14

V. Weidemann. The initial-final mass relation - galactic disk and magellanic clouds. *Astronomy and Astrophysics*, 188:74–84, Dec. 1987. 13

V. Weidemann. Revision of the initial-to-final mass relation. *Astronomy and Astrophysics*, 363:647–656, Nov. 2000. xiv, 11, 13, 14, 16, 89

K. Werner. Construction of non-LTE model atmospheres using approximate lambda operators. *Astronomy and Astrophysics*, 161:177–182, June 1986. 57

F. Wesemael, J. L. Greenstein, J. Liebert, R. Lamontagne, G. Fontaine, P. Bergeron, and J. W. Glaspey. An atlas of optical spectra of white-dwarf stars. *Publications of the Astronomical Society of the Pacific*, 105:761–778, July 1993. 4

D. T. Wickramasinghe and L. Ferrario. The origin of the magnetic fields in white dwarfs. *Monthly Notices of the Royal Astronomical Society*, 356:1576–1582, Feb. 2005. 21

K. A. Williams. A new look at the empirical Initial–Final mass relation. volume 372, page 85, Sept. 2007. 21

K. A. Williams and M. Bolte. A photometric and spectroscopic search for white dwarfs in the open clusters NGC 6633 and NGC 7063. *Astronomical Journal*, 133:1490–1504, Apr. 2007. 14

K. A. Williams, M. Bolte, and J. W. Liebert. Spectroscopic identification of faint white dwarf candidates in the praesepe open star cluster. *Astronomical Journal*, 128:1784–1789, Oct. 2004. 14

K. A. Williams, M. Bolte, and D. Koester. Probing the lower mass limit for supernova progenitors and the High-Mass end of the Initial-Final mass relation from white dwarfs in the open cluster m35 (NGC 2168). *Astrophysical Journal*, 693:355–369, Mar. 2009. xvi, xvii, 11, 14, 78, 79, 80, 81, 82, 83, 84, 87, 88, 94, 95, 96, 97, 98, 99, 100, 101

D. E. Winget, C. J. Hansen, J. Liebert, H. M. van Horn, G. Fontaine, R. E. Nather, S. O. Kepler, and D. Q. Lamb. An independent method for determining the age of the universe. *The Astrophysical Journal*, 315:L77–L81, Apr. 1987. 8

M. A. Wood. Astero-archaeology: Reading the galactic history recorded in the white dwarf stars. http://adsabs.harvard.edu/abs/1990PhDT.....5W, 1990. 3, 8

M. A. Wood. Constraints on the age and evolution of the galaxy from the white dwarf luminosity function. *The Astrophysical Journal*, 386:539–561, Feb. 1992. 2, 14