

# A PIPELINE FOR THE ANALYSIS OF STELLAR SPECTRA

Robert Arfon Williams

A thesis submitted in partial fulfilment of the requirements of  
Liverpool John Moores University  
for the degree of  
Doctor of Philosophy.  
May 23, 2018

*"I must not fear.*

*Fear is the mind-killer.*

*Fear is the little-death that brings total obliteration.*

*I will face my fear.*

*I will permit it to pass over me and through me.*

*And when it has gone past I will turn the inner eye to see its path.*

*Where the fear has gone there will be nothing.*

*Only I will remain."*

- Dune by Frank Herbert

# Declaration

The work presented in this thesis was carried out at the Astrophysics Research Institute, Liverpool John Moore's University. Unless otherwise stated, it is the original work of the author.

While registered as a candidate for the degree of Doctor of Philosophy, for which submission is now made, the author has not been registered as a candidate for any other award. This thesis has not been submitted in whole, or in part, for any other degree.

Robert Arfon Williams  
Astrophysics Research Institute  
Liverpool John Moore's University  
IC2, Liverpool Science Park  
146 Brownlow Hill  
Liverpool  
L3 5RF  
UK

# Abstract

Understanding the formation and evolution of our galaxy, the Milky Way, has been an ongoing process, which with the development of large-scale surveys has picked up considerable pace. Together with these new surveys, pipelines have been constructed which allow for the rapid and automatic processing of this wealth of new data. These codes are able to turn raw data files into tables of stellar parameters and chemical abundances in far less time than if they were analysed by hand. The results from these surveys open new windows on to the history of our galaxy and other disk galaxies.

In this thesis, we present the development of a new pipeline, the STellAR Parameter AND Abundances pipeline (STARPANDA), which is able to rapidly derive stellar parameters, CNO abundances and other elemental abundances by utilising measurements of spectral features in both observed and synthetic spectra. We take the observed spectra, synthetic spectra and line lists employed by the APOGEE survey and produce new values for the stellar parameters, CNO abundances and  $\alpha$  abundances of the APOGEE stars. We then compare our results with those achieved by the APOGEE pipeline.



# Acknowledgements

Firstly, I would like to thank Ricardo Schiavon for all the advice and support he has given me during my PhD. I would also like to thank Phil James for his assistance and support at various points, but, especially during the writing-up phase.

Secondly, there are those people who have contributed to this work in various ways, directly and indirectly. A lot of the lines used in this work and all of the measurements of those lines have been done as part of undergraduate Masters projects and as part of PhD projects by several other students. So, to Ted Mackereth, Tiberiu Chirila Kleo Bosley, my thanks for their work and assistance, without which we would not have gotten to the stage we are at today. I would also like to thank the wonderful members of the APOGEE collaboration and especially those people who have worked so hard on the ASPCAP pipeline as well as those who worked on the model atmospheres and on line list used by both STARPANDA and ASPCAP.

And last, but by no means least, I would like to thank various family and friends for their support during the last several years. To my parents - thank you for your love and support; I wish my father could have seen me finally complete my PhD. To Lykke, Tom, Emma & Maya - thank you for your support and friendship. To Katie, Helen, Andy, Caroline, Scot, Rich, Kate, Rhana, Sam, Danielle and Elliott - my thanks for your help and support, and my apologies for the humming/singing and for my, may-haps occasional, outbursts of swearing when my code didn't behave.

*"No man is an island entire of itself; every man  
is a piece of the continent, a part of the main;  
if a clod be washed away by the sea, Europe  
is the less, as well as if a promontory were, as  
well as any manner of thy friends or of thine  
own were; any man's death diminishes me,  
because I am involved in mankind.  
And therefore never send to know for whom  
the bell tolls; it tolls for thee."*

- John Donne

# Contents

<b>Declaration</b>	<b>iii</b>
<b>Abstract</b>	<b>iv</b>
<b>Acknowledgements</b>	<b>v</b>
<b>Contents</b>	<b>vii</b>
List of Tables . . . . .	xi
List of Figures . . . . .	xiii
<b>1 The Milky Way</b>	<b>1</b>
1.1 Galaxy Formation Theory . . . . .	1
1.2 Formation & Structure of the Milky Way . . . . .	2
1.3 Galactic Surveys . . . . .	5
1.4 This Work . . . . .	8
<b>2 Spectral Analysis Codes</b>	<b>10</b>
2.1 The GAIA-ESO Pipeline . . . . .	11
2.2 The GALAH Pipeline . . . . .	12

2.3	The Cannon . . . . .	13
2.4	EZ_Ages . . . . .	14
2.5	APOGEE Stellar Parameter & Chemical Abundance Pipeline . . . . .	18
<b>3</b>	<b>Stellar Parameter &amp; Abundance Pipeline</b>	<b>24</b>
3.1	Structure of STARPANDA . . . . .	27
3.1.1	Input Files . . . . .	29
3.1.2	Output Files . . . . .	31
3.2	STARPANDA Algorithm . . . . .	32
3.2.1	Starting Tasks . . . . .	32
3.2.2	Parameter & CNO Abundance Analysis . . . . .	33
3.2.3	CNO Abundance Analysis . . . . .	47
3.2.4	Elemental Abundance Analysis . . . . .	51
3.2.5	Final Tasks . . . . .	55
3.2.6	Remarks . . . . .	56
3.3	Development of STARPANDA . . . . .	58
3.4	Testing STARPANDA . . . . .	62
3.4.1	Testing the RGI method . . . . .	63
3.4.2	Testing the interp1d method . . . . .	68
3.4.3	Testing the Pipeline . . . . .	69
<b>4</b>	<b>Results</b>	<b>84</b>
4.1	Data Preparation . . . . .	85

4.2	Indices . . . . .	88
4.3	Analysis of Stellar Parameters & CNO Abundances . . . . .	89
4.3.1	DR12 Analysis . . . . .	91
4.3.2	DR14 Analysis . . . . .	105
4.4	Analysis of CNO Abundances . . . . .	107
4.4.1	DR12 Analysis . . . . .	108
4.4.2	DR14 Analysis . . . . .	113
4.4.3	Comments on CNO Abundances . . . . .	117
4.5	Analysis of Elemental Abundances . . . . .	121
4.5.1	DR12 Analysis . . . . .	122
4.5.2	DR14 Analysis . . . . .	123
<b>5</b>	<b>Conclusions</b>	<b>127</b>
<b>6</b>	<b>Future Work</b>	<b>130</b>
6.1	Future Pipeline Development . . . . .	130
6.2	Future Science . . . . .	132
<b>A</b>	<b>How to use STARPANDA</b>	<b>134</b>
A.1	Download & Installation . . . . .	134
A.2	STARPANDA Readme File . . . . .	135
<b>B</b>	<b>Stellar Atmospheres &amp; Line Formation</b>	<b>142</b>
B.1	Model Stellar Atmospheres . . . . .	143
B.2	Line Formation . . . . .	144

B.3 Lines as Parameter Tracers . . . . .	145
<b>Bibliography</b>	<b>147</b>

# List of Tables

1.1	APOGEE Spectrograph Information . . . . .	6
2.1	GALAH - AMBRE Model Grid Parameters . . . . .	13
2.2	ASPCAP Model Grid Parameters . . . . .	20
3.1	Parameter Holes Used for RGI Testing . . . . .	64
3.2	Model atmosphere paramaters of the mock STARPANDA test obser- vations . . . . .	70
3.3	STARPANDA Parameters for Creating New Synthetic Spectra . . . .	74
3.4	STARPANDA Results of Analysing New Synthetic Spectra . . . . .	76
3.5	STARPANDA code running times . . . . .	79
3.6	Specifications of STARPANDA Development Hardware . . . . .	80
4.1	Model Atmosphere cuts for STARPANDA model inputs . . . . .	87
4.2	STARPANDA Indices for Stellar Parameters . . . . .	89
4.3	STARPANDA Indices for CNO . . . . .	90
4.4	STARPANDA Indices for Elements . . . . .	90
4.5	STARPANDA Best Indices . . . . .	91
4.6	STARPANDA Initial Parameters & Iteration Tolerances . . . . .	91

4.7	Comparison of 6 Parameter Analysis Results . . . . .	105
4.8	Comparison of CNO Abundances Analysis Results . . . . .	119
4.9	Analysis Results From Varying CNO Initial Abundances . . . . .	120
4.10	Comparison of CNO Abundances . . . . .	121



# List of Figures

1.1	Views of the Milky Way . . . . .	4
2.1	Table 1 from Graves & Schiavon (2008) . . . . .	16
2.2	Figure 1 from Graves & Schiavon (2008) . . . . .	17
2.3	ASPCAP Spectral Windows . . . . .	22
2.4	ASPCAP Process Flowchat . . . . .	23
3.1	STARPANDA Process Flowchart . . . . .	28
3.2	STARPANDA Algorithm Flowchart . . . . .	32
3.3	Example of an initial $T_{eff}$ -Fe grid . . . . .	36
3.4	Example of line comparisons . . . . .	37
3.5	STARPANDA Find_Params Algorithm Flowchart . . . . .	38
3.6	Example of a $T_{eff}$ -Fe grid . . . . .	41
3.7	Example of an interpolate parameter curve . . . . .	43
3.8	Example Find_Params FITS output from STARPANDA . . . . .	47
3.9	Example of Analysis Flag histograms from STARPANDA . . . . .	48
3.10	Example of Iteration Flag histograms from STARPANDA . . . . .	49
3.11	STARPANDA Find_CNO Algorithm Flowchart . . . . .	50

3.12	Example Find_CNO FITS output from STARPANDA . . . . .	51
3.13	STARPANDA Find_Abunds Algorithm Flowchart . . . . .	53
3.14	Example Find_Abunds FITS output from STARPANDA . . . . .	55
3.15	Example Logfile output from STARPANDA . . . . .	56
3.16	Histogram of residuals in RGI testing for $T_{eff}$ . . . . .	65
3.17	Histogram of residuals in RGI testing for Fe . . . . .	65
3.18	Histogram of residuals in RGI testing for $\log g$ . . . . .	66
3.19	Histogram of residuals in RGI testing for C . . . . .	66
3.20	Histogram of residuals in RGI testing for N . . . . .	67
3.21	Histogram of residuals in RGI testing for O . . . . .	67
3.22	Mock Observation Comparison on $T_{eff}$ . . . . .	71
3.23	Mock Observation Comparison on Fe . . . . .	71
3.24	Mock Observation Comparison on $\log g$ . . . . .	72
3.25	Mock Observation Comparison on C . . . . .	72
3.26	Mock Observation Comparison on N . . . . .	73
3.27	Mock Observation Comparison on O . . . . .	73
3.28	STARPANDA CNO Residual Histograms . . . . .	81
3.29	STARPANDA CNO Residual Comparisons . . . . .	82
3.30	STARPANDA CN Residual Comparison . . . . .	83
4.1	STARPANDA 6 Parameter Analysis (DR12) - Analysis Flags . . . . .	93
4.2	STARPANDA 6 Parameter Analysis (DR12) - Iteration Flags . . . . .	93
4.3	STARPANDA 6 Parameter Analysis (DR12) - Initial $T_{eff}$ -Fe Grid . . . . .	94

4.4	STARPANDA 6 Parameter Analysis (DR12) - $T_{eff}$ Comparison . . .	95
4.5	STARPANDA 6 Parameter Analysis (DR12) - Fe Comparison . . . .	95
4.6	STARPANDA 6 Parameter Analysis (DR12) - $\log g$ Comparison . . .	96
4.7	STARPANDA 6 Parameter Analysis (DR12) - C Comparison (1) . . .	96
4.8	STARPANDA 6 Parameter Analysis (DR12) - N Comparison (1) . . .	97
4.9	STARPANDA 6 Parameter Analysis (DR12) - O Comparison (1) . . .	97
4.10	STARPANDA Indicator Comparison - $T_{eff}$ . . . . .	99
4.11	STARPANDA Indicator Comparison - Fe . . . . .	99
4.12	STARPANDA Indicator Comparison - $\log g$ . . . . .	100
4.13	STARPANDA Indicator Comparison - C . . . . .	101
4.14	STARPANDA Indicator Comparison - N . . . . .	102
4.15	STARPANDA Indicator Comparison - O . . . . .	102
4.16	STARPANDA 6 Parameter Analysis (DR12) - C Comparison (2) . . .	103
4.17	STARPANDA 6 Parameter Analysis (DR12) - N Comparison (2) . . .	104
4.18	STARPANDA 6 Parameter Analysis (DR12) - O Comparison (2) . . .	104
4.19	STARPANDA 6 Parameter Analysis (DR14) - Analysis Flags . . . .	106
4.20	STARPANDA 6 Parameter Analysis (DR14) - Iteration Flags . . . .	106
4.21	STARPANDA 6 Parameter Analysis (DR14) - N Comparison . . . .	107
4.22	STARPANDA CNO Abundance Analysis (DR12) - Analysis Flags . .	109
4.23	STARPANDA CNO Abundance Analysis (DR12) - Iteration Flags . .	109
4.24	STARPANDA CNO Abundance Analysis (DR12) - O Comparison . .	111
4.25	STARPANDA CNO Abundance Analysis (DR12) - C Comparison . .	112

4.26	STARPANDA CNO Abundance Analysis (DR12) - N Comparison . .	114
4.27	STARPANDA CNO Abundance Analysis (DR14) - Analysis Flags . .	115
4.28	STARPANDA CNO Abundance Analysis (DR14) - Iteration Flags . .	115
4.29	STARPANDA CNO Abundance Analysis (DR14) - O Comparison . .	116
4.30	STARPANDA CNO Abundance Analysis (DR14) - C Comparison . .	117
4.31	STARPANDA CNO Abundance Analysis (DR14) - N Comparison . .	118
4.32	STARPANDA Elemental Abundance Analysis (DR12) - Al Compari- son (1) . . . . .	123
4.33	STARPANDA Elemental Abundance Analysis (DR12) - Al Compari- son (2) . . . . .	124
4.34	STARPANDA Elemental Abundance Analysis (DR14) - Al Compari- son (1) . . . . .	125
4.35	STARPANDA Elemental Abundance Analysis (DR14) - Al Compari- son (2) . . . . .	126

# Chapter 1

## The Milky Way

### 1.1 Galaxy Formation Theory

Modern cosmological models describe a universe which, after recombination, had an almost smooth distribution of matter, with only small fluctuations in density. Low amplitude over-densities were over cosmic timescales able to gravitationally attract more and more matter. These have led to the structures that we see in the local universe today. However there are still many questions about how these small over-densities became the stars and galaxies we see around us. In line with this, it is known that the baryon content of the early universe was made up of hydrogen (H), helium (He), a small trace of lithium (Li) and nothing else - no heavier elements such as oxygen, nitrogen or carbon. These are elements which are relatively abundant (at least in comparison to the early universe), in the world around us and without some of them there would be no life (at least not as we could imagine it). So how did the rich chemistry of the local universe come about? We know that all of these elements (and many more) are formed either in the life or death of stars - through nucleosynthesis in stellar cores and supernovae explosions.

Current galaxy formation theory (see Benson, 2010, for a review) holds that the first

stage of structure development is the formation of dark matter halos, in which all the galaxies we see live. These collapse from the over-densities under gravity and draw in the baryonic matter which makes up the luminous universe. The baryonic matter then collapses further and eventually becomes cold and dense enough to form the first stars. Over time more and more stars form, eventually leading to the first galaxies.

## 1.2 Formation & Structure of the Milky Way

Our galaxy, the Milky Way (MW), is a barred, spiral disk galaxy. Models of the formation of our Galaxy have put forward scenarios such as the monolithic collapse of gas (Eggen et al., 1962) or a hierarchical build up of structure (Searle & Zinn, 1978) and from these scenarios numerical simulations have been developed which have struggled to reproduce the structure of the MW (see Rix & Bovy, 2013, for a review). Also, from observations of the MW, we can see a structure which not only has a bulge and halo, but 2 disks, one thin and one thicker, and a bimodality in the distribution of  $[\alpha/\text{Fe}]$  as a function of  $[\text{Fe}/\text{H}]$  (e.g. Fuhrmann, 1998; Bensby et al., 2014; Mackereth et al., 2017).

The origin of this dual disk structure is uncertain, with simulations giving us many possibilities that will need to be constrained using observational data (e.g. Mackereth, in prep). The history of a galaxy is also heavily affected by the interactions it may have with smaller satellite galaxies and also its more massive neighbours. The structure of the galaxy may be heavily affected by close interactions with large neighbours, while the make-up of stars in the galaxy will be affected by the capture of stars from either the close interaction with satellites (where stars are stripped from the smaller galaxy) or by the merger of small galaxies with the larger. All of these processes would leave imprints in the dynamics and chemistry of stars (chemodynamical fingerprints), which can be statistically detected using large-scale surveys of stars in our galaxy.

The structure of the MW, as we currently understand it, is of 3 baryonic components: the Halo, the Bulge, the Disk. Figure 1.1 (NASA , 2015; ESO et al., 2015) shows in the top panel an artistic rendering of our current understanding of the structure of the MW, while the bottom panel shows a mosaic of the MW as seen from Earth.

The MW formed from material which was considerably poorer in metals than the current galaxy. Over cosmological time generations of stars together with material accreted from extra-Galactic sources have changed the chemical composition of the MW. By studying the chemical composition of individual stars and then analysing the distribution patterns of elemental abundances over different regions of the galaxy we can understand the evolution of the MW and the formation of the structures that we observe today. Work has been going on in this area for many years and there are several good reviews of the work which has been done (e.g., Freeman, 1987; Gilmore et al., 1989; Majewski, 1993; Rix & Bovy, 2013; Feltzing & Chiba, 2013).

The Apache Point Galaxy Evolution Experiment (APOGEE; Majewski et al., 2017), APOGEE-2, GALactic Archaeology with Hermes (GALAH; De Silva et al., 2015), the Gaia-ESO survey (Gilmore et al., 2012; Randich et al., 2013) and surveys that will use the William Herschel Telescope’s WEAVE instrument (Dalton et al., 2012) or the Multi Object Optical and Near-infrared Spectrograph for the VLT (MOONS; Cirasuolo et al., 2014), once they are built, are all examples of large-scale spectroscopic surveys which have been, are or will be targeting stars in the MW with the aims of understanding the formation and evolution of the Galaxy. APOGEE, having been completed and the final data released (Holtzman et al., 2015) has provided us with a wealth of new data to further this goal and will continue to provide insights for many years to come. So, by combining a detailed understanding of the ages, kinematics and abundances of a large sample of stars from all regions of our Galaxy it is hoped that we can understand the formation and evolution of not just the MW, but also, of all similar disk galaxies.



Figure 1.1: Views of the Milky Way. The top panel shows an artist's impression of the Milky Way; created for the 212th AAS meeting to highlight the best, then current, idea of what our galaxy would look like in a view from above (NASA , 2015). The bottom panel comes from ESO Gigagalaxy Zoom project and is a mosaic of the Bulge and disk of the Milky Way as seen from Earth (ESO et al., 2015)



## 1.3 Galactic Surveys

### SDSS & APOGEE

APOGEE is a spectroscopic survey which has obtained high resolution (with a resolving power of approximately 22,500) spectra of over 150,000 stars through the bulge, thin disk, thick disk and halo of the MW. It is one of the core surveys undertaken by the Sloan Digital Sky Survey. The APOGEE project was part of SDSS-III (Eisenstein et al., 2011) and this is continuing as APOGEE-2 in SDSS-IV (Blanton et al., 2017). All data gathered by SDSS projects are eventually made available to the whole astronomical community after a proprietary period; APOGEE data are available through periodic SDSS data releases (e.g., Alam et al., 2015).

The observations have been undertaken from the 2.5m telescope at Apache Point Observatory in New Mexico, USA. The APOGEE survey made use of a bench-mounted 300 fibre multi-object spectrograph (MOS) fed by fibres from the focal plane. The design of the instrument (Wilson et al., 2012) allows high signal-to-noise ( $S/N \geq 100$ ) observations in the near infra-red (NIR) range from 1.51-1.7 $\mu\text{m}$ , which lies within the H band. The reason for this choice of using the NIR for the survey is to provide access to dust-obscured regions in the Galactic bulge and also for disk stars (including those on the opposite side of the Galactic centre to us). In the optical this would not be possible due to obscuration and extinction by the high amount of dust present in the Galactic plane - with interstellar extinction due to dust being much larger in the optical than in the IR ( $A_V/A_H = 6$ ). However NIR observations do not suffer as badly from the presence of dust and so, spectra can then be obtained from representative samples of stars of all environments in the MW, thus giving a more complete survey. A summary of the APOGEE technical specification is provided in Table 1.1.

The scientific goals of the APOGEE survey are measuring the chemical abundances for different Galactic environments of stars; studying star formation, feedback and chem-

Resolving Power	$\sim 22500$
Nominal Spectral Coverage	$1.51\text{-}1.71\mu\text{m}$
Number of Fibres	300
Typical Integration Time	3hrs
S/N	$\gtrsim 100$
Limiting Magnitude	$H = 12.2$
Detector	Three 2048 HgCdTe Raytheon chips

Table 1.1: Some important technical details of the APOGEE spectrograph

ical mixing; studying the bar and spiral arms via dynamics of bulge and disk stars and probing the formation and evolution of the MW using chemodynamical information. In order to achieve these goals, much work went into the target selection for APOGEE (Zasowski et al., 2013) and the primary target were Red Giant (RG) stars. RG stars are evolved stars which have left the main sequence and entered the later stages of their life cycles. During this phase they grow larger and more luminous (than they were during the main sequence stage of their life cycle), making them observable over greater distances. They are found in all stellar populations and Galactic environments making them important tracers of Galactic chemodynamics.

APOGEE provides measurements of several stellar parameters: the effective temperature of the photosphere ( $T_{eff}$ ), the overall metallicity ( $[M/H]$ ), surface gravity ( $\log g$ ) and the micro-turbulent velocity,  $\xi$ . Detailed analysis of the spectral lines found in the H-band yields abundances for 20 elements: C, N, O, Mg, Al, Si, Ca, Ti, Cr, Fe, Ni, Na, S, V, Mn and Co (originally), plus Ce, Nd, Yb and  $C^{13}$  (recently added). This is achieved by passing all data taken through 2 pipelines and combining multiple observations of stars (as most targets are observed more than once to achieve the desired S/N as well as to identify RV variability from binary star systems). The first pipeline generates the calibrated spectra from the raw data and measures the radial velocity by cross-correlating with the rest frame spectral templates. The second pipeline, the APOGEE Stellar Parameters and Chemical Abundances Pipeline (ASPCAP), calculates the stellar parameters and abundances for each star - this is described in more detail in the next section. Work is ongoing to fully characterise the precision of the

results and improve them, where possible, in future data releases. The final output provides us with important information with which to probe the the formation and evolution of the Milky Way as well as providing insights on galaxy formation and evolution more generally.

### Other Surveys

The Gaia-ESO survey (Gilmore et al., 2012; Randich et al., 2013) started in 2011 with the aims of obtaining high signal-to-noise, high-resolution spectra for over  $10^5$  stars in all regions of the Milky Way (bulge, disk and halo). It is based at the VLT telescope in Chile and uses the GIRAFFE and UVES spectrographs on FLAMES (Pasquini et al., 2002). For the majority of stars spectra will be obtained at a resolution of  $R \sim 20,000$ , while for a smaller sample of stars (approximately 5,000 objects), spectra will be obtained at higher resolution,  $R \sim 47,000$ . As with APOGEE, the results are made public after a proprietary period, and the final data releases will include the stellar parameters ( $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$  & the micro-turbulence velocity,  $\xi$ ), CNO abundances and elemental abundances (Na, Mg, Al, Si, Ca, Sc, Ti, V, Cr, Mn, Fe, Co, Ni, Cu, Zn, Y, Zr, Mo, Ba, Nd & Eu) for all target stars. The raw data will be released immediately however.

GALactic Archaeology with Hermes (GALAH; De Silva et al., 2015) uses the HERMES fibre-fed multi-object spectrograph on the Anglo-Australian Telescope in Australia (3.9m). The survey is aiming to measure 29 abundances from  $10^6$  stars brighter than  $V=14$  with  $R \sim 28000$  over a 5 year period and obtain spectra from 4 CCDs in 4 wavelength regions, Blue (4718-4903 Å), Green (5649-5873 Å), Red (6481-6739 Å) and IR (7590-7890 Å).

In addition to the APOGEE surveys, GAIA-ESO and GALAH, there are other surveys such as ARGOS (Freeman et al., 2013) and LEGUE (Deng et al., 2012), either completed or ongoing. In the years ahead instruments such as 4MOST (de Jong et al.,

2016), MOONS (Cirasuolo et al., 2014) and WEAVE (Dalton et al., 2012) are being built, which will allow further opportunities for new spectroscopic surveys. Thus, the coming decades are sure to see great improvements in our understanding of the Milky Way, in particular, and galaxy formation, in general.

## 1.4 This Work

The work described here has several aims. Firstly, to develop a new pipeline which can be used to independently test the results produced by ASPCAP. To this end the code will use a different methodology to that employed by ASPCAP, but take the same line list and synthetic spectra. The code should be capable of matching the precision of the ASPCAP pipeline, but do so in significantly less time using standard desktop computer hardware. Secondly, the new pipeline should be capable of expansion to include the ability to analyse additional elements as well as being able to use different input models, line lists, wavelength regions, etc. In short, the final aim of this work is to start the development of a pipeline which is capable of providing fast, precise bulk spectroscopic analysis for any data set in any wavelength region which is able to meet the requirements of the methodology utilised.

The layout of the remainder of this work is as follows. Chapter 2 briefly reviews the methodology of several pipelines developed to analyse spectroscopic data sets. Chapter 3 describes in detail the structure and algorithms used by the new pipeline, details the testing undertaken and briefly discusses the development of the code. Except where noted, the work described in Chapter 3 has been undertaken by the author. Chapter 4 shows some of the current results obtainable by the pipeline with reference to results from ASPCAP. Again, except where noted, the work shown has been produced by the author. Chapter 5 contains concluding remarks. Chapter 6 briefly discusses work currently ongoing and ideas for further work either on expanding or improving the pipeline, or on what could be done with the results obtained using the pipeline. Fi-

---

nally, Appendix A contains information on how to use the STARPANDA pipeline, whilst Appendix B contains a brief description of Stellar Atmospheres & Line Formation as it applies to this work.

## Chapter 2

# Spectral Analysis Codes

Over most of the history of the spectroscopic study of stellar light, spectra have been analysed interactively, with experienced researchers painstakingly measuring features in spectra and comparing them to model predictions in order to calculate stellar parameters and find elemental abundances. With the advent of large-scale spectroscopic surveys, this method has become too time-consuming to allow the data to be analysed within the life-times of the researchers involved. If we are to fully explore this wealth of new data, tools must be developed to delegate this task to computers - to greatly speed up spectroscopic analysis and to provide more robust output with smaller and better quantified errors. This task is being undertaken by many groups already and many approaches are being utilised since there is no one method which clearly stands out above the rest.

In the following sections of this chapter various spectral analysis codes will be described. Brief descriptions will be given of codes developed specifically for the GAIA-ESO survey (Section 2.1) and GALAH survey (Section 2.2), as well as for the Cannon code (Section 2.3). A more detailed description is given of the EZ\_AGES code (Section 2.4) and of the pipeline used by the APOGEE survey (Section 2.5); the former is the inspiration for the new pipeline described in Chapter 3, while the latter is used as a comparison for our new pipeline.

## 2.1 The GAIA-ESO Pipeline

Since the GAIA-ESO survey makes use of two spectrographs on FLAMES, data reduction is handled separately for the output of Giraffe (Lewis et al. in prep) and UVES (Sacco et al., 2014). From there the reduced spectra are passed to individual working-groups which are responsible for deriving any further information which may be of interest to them.

One working group is focussed on deriving stellar parameters ( $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$  & the micro-turbulence velocity,  $\xi$ ), CNO abundances and elemental abundances (Na, Mg, Al, Si, Ca, Sc, Ti, V, Cr, Mn, Fe, Co, Ni, Cu, Zn, Y, Zr, Mo, Ba, Nd and Eu) for the  $10^4$  FGK-type stars observed by the UVES high-resolution spectrograph (Smiljanic et al., 2014). In order to achieve this, the team employed 13 separate, parallel pipelines using a variety of methodologies to ensure that they can get precise results for stars with a wide variety of parameter values; this approach bypasses the problem that single methodology pipelines can have where they work well for some regions of parameter space, but not as well for others. Each pipeline is tested against reference stars to determine precisely how well it performs in different parameter space regions.

While each pipeline is run on a separate node (a separate group using a separate computing cluster), there are some commonalities to which all adhere - there is a common line list (Heiter et al., 2015), though nodes do not all have to use the same lines from that list; a set of model atmospheres (MARCS grid; Gustafsson et al., 2008); and a set of calibration targets. Each node uses one of three general methodologies to then derive parameters, either based on equivalent width measurements, reference to a library of observed spectra or by computing synthetic spectra on-the-fly (full details of the working of each node see Appendix A of Smiljanic et al., 2014). Individual nodes then

derive stellar parameters for a target star and the results from all nodes are weighted and combined on a line-by-line basis. The weighting is done using the values of the parameters obtained and by reference to the results of each pipeline’s tests against the calibration targets. If there are less than 3 nodes delivering results for a line, the results are discarded (as there are too few nodes to allow an accurate characterisation of precision), otherwise the results are then combined to produce a final “best result” set of stellar parameters. CNO abundances are obtained via a single node using the best resulting stellar parameters, using C<sub>2</sub> lines for C, CN for N and for O, a forbidden [OI] line at  $\sim 6300\text{\AA}$ . Elemental abundances are then calculated separately on all nodes using the stellar parameters obtained from that node previously, rather than taking the best result set and these were then combined in the same manner as the stellar parameters. Some nodes were able to derive values for additional elements, but these were discarded as they were not available from 3 or more nodes.

## 2.2 The GALAH Pipeline

The data reduction and analysis pipeline for GALAH (Kos et al., 2017) makes considerable use of IRAF (Tody, 1986) routines. The code has been designed to work with observations made as part of the GALAH survey and replaces some of the data reduction code that would normally run on data obtained by HERMES (this was done in order to include solutions for specific problems encountered by this survey). Starting from raw data files, the IRAF pipeline takes approximately two weeks to deliver radial velocities and stellar parameters for the survey targets ( $\sim 10^5$  stars), running on a ‘high-end’ desktop PC. Further analysis of the spectra can then be carried out by individual teams as desired.

The pipeline has a mixture of manual and automated steps. The first stage is to use the Blue, Green and Red regions of each spectrum to derive the radial velocity,  $T_{eff}$ , [Fe/H] and  $\log g$  parameters. Radial velocity is derived by comparing the observed



spectra to a set of 15 AMBRE model spectra (de Laverny et al., 2012); these vary  $T_{eff}$  between 4000 and 7500K in steps of 250K, while fixing  $[Fe/H] = 0.0$  and  $\log g = 4.5$ . This is done separately for each region and then the results averaged to give a final radial velocity. Next, the stellar parameters are derived from a larger grid of 16783 AMBRE model spectra (see Table 2.1 for details on these). From this grid, the 10 closest matching spectra are found by computing the Euclidean norm for each model spectrum relative to the observed spectrum. These 10 closest matches are then linearly combined to produce a matching model spectrum, with the stellar parameters being derived by taking the model stellar parameters and combining them in the same manner.

Parameter	Range	Delta
$T_{eff}$	2500 - 4000 K	200 K
	4000 - 8000 K	250 K
$[Fe/H]$	-5.0 - -3.0 dex	1.0 dex
	-3.0 - -1.0 dex	0.5 dex
	-1.0 - +1.0 dex	0.25 dex
$\log g$	-0.5 - +5.5 dex	0.5 dex

Table 2.1: The stellar parameter ranges (and step sizes covered) by the 16782 AMBRE model spectra used by GALAH for deriving stellar parameters

The data are then passed to individual teams for further analysis and the derivation of elemental abundances.

## 2.3 The Cannon

The Cannon (Ness et al., 2015) has been developed to be a project-independent pipeline for analysing spectroscopic data from a wide variety of sources (e.g. supernovae, stars & integrated-light observations of clusters/galaxies). It is capable of being used to derive a wide range of stellar parameters and elemental abundances (called Labels) from spectra in any wavelength region.

It does not use line lists or grids of synthetic spectra, but uses a set of reference spectra to construct a generative model (called the training step) which can then be used to predict the parameters of a larger set of observed targets (called the test step).

The training step requires that the reference spectra be from objects for which the parameters of interest are well-known and cover a range of parameter space sufficient to the task of analysing the full set of observed targets. They should be taken from the same set as the targets of unknown parameters to ensure that any systematic errors are the same. From these the Cannon then takes the flux from each pixel and generates a polynomial function with coefficients given by all the parameters of interest. It is assumed that the parameters being used are sufficient to describe the flux at each pixel and that the flux changes smoothly as a function of the changing parameters.

The test step then takes the model constructed during the training step and applies it to the full set of spectra for which parameters are required. This step is extremely quick - the Cannon was able to derive 17 stellar parameters from the APOGEE data set (150,677 stars) running on a small computing cluster in  $\sim 0.01$  seconds per star (Casey et al., 2016). During this step the Cannon both interpolates between the reference set spectra and can, if necessary, extrapolate to areas outside of this parameter space.

## 2.4 **EZ\_Ages**

One method of analysing spectra is by measuring the strength of spectral features which are sensitive to either a single stellar parameter/elemental abundance, or to multiple parameters/abundances which can be derived by reference to other features. This method is described in Schiavon (2007) and from it a spectral analysis code was developed by Genevieve Graves (Graves & Schiavon, 2008), called *EZ\_Ages*.

*EZ\_Ages* looks at the integrated spectra from whole stellar populations in galaxies and clusters where individual stars cannot be resolved, rather than resolved stellar systems or individual stars. It was developed to quickly find the mean ages and abundances of C, Ca, Fe, Mg and N from medium (or higher) resolution spectra. This tool is available for download and can be used by those interested in the mean ages and abundances of whole stellar populations.

The method utilised in this code makes use of the Lick Indices (Burstein et al., 1984; Gorgas et al., 1993; Worthey et al., 1994), which specify a series of lines in optical spectra which are sensitive to mean stellar ages and chemical abundances. *EZ\_Ages* makes use of a set of these as described in Schiavon (2007) and the sensitivity of the indices to various stellar parameters and abundances has been shown by Tripicco & Bell (1995) and Korn et al. (2005). Figure 2.1 shows the Lick Indices from the Single Stellar Population models and some of these were used by *EZ\_Ages*. In this table,  $H\beta$  shows no sensitivities (or only a weak sensitivity) to Ni, but it is a good tracer of age. Indices which have multiple sensitivities can still be used, provided that one is able to break degeneracies by using other indices which have already been analysed to provide abundances (e.g. CN indices are used to determine the N abundance, but first C must be found using the  $C_2$  index and by assuming a value for O). In this way, many indices can be used together to obtain the mean abundances of all 5 elements as well as the mean stellar parameters. By iterating over this process several times (until a pre-defined convergence criterion is achieved) a self-consistent set of results can be obtained.

*EZ\_Ages* works by taking pairs of index measurements, the equivalent widths of the indices, from model spectra and plotting them together to form grids. An example of such a grid can be seen in Figure 2.2. The parameters used to create these grids include the age of a single stellar population, the overall metallicity (parameterised as  $[Fe/H]$ ) as well as the elemental abundances that are to be found by the code. Lines of constant age run quasi-horizontally, while lines of constant  $[Fe/H]$  run quasi-vertically. On top

TABLE 1  
LICK INDICES IN THE S07 SSP MODELS

Index	KMT Sensitivity <sup>a</sup>	SWB Sensitivity <sup>a</sup>
Balmer Indices		
H $\delta_A$ .....	Fe, C <sup>c</sup>	Fe, C, V
H $\delta_F$ .....	Fe, Mg <sup>b</sup>	Fe, C, Si
H $\gamma_A$ .....	C, Fe, Mg	C, Ti, (Mg)
H $\gamma_F$ .....	C, Fe	C, (Si), (Mg)
H $\beta$ .....	...	(Ni)
Fe Indices		
Fe4383.....	Fe, Mg, C <sup>c</sup>	C, Fe
Fe5015.....	Ti, Mg, Fe	C, Fe, (Ti)
Fe5270.....	Fe	C, (Fe)
Fe5335.....	Fe	Fe, C
Mg Indices		
Mg <sub>2</sub> .....	Mg, C	Mg, C, Fe
Mg <i>b</i> .....	Mg, Fe, C	Mg, Fe, (Cr)
C, CH, and CN Indices		
CN <sub>1</sub> .....	C, N, O	C, N, O
CN <sub>2</sub> .....	C, N, O	C, N, O
G4300.....	C, O, Fe <sup>b</sup>	C, Fe, Ti
C <sub>2</sub> 4668.....	C, O	C, O, (Si)
Ca Indices		
Ca4227.....	Ca, C	Ca, O, (CN)

NOTES.—Index sensitivities as given by KMT (Korn et al. 2005) and SWB (Sereno et al. 2005). Only the top three element sensitivities are given, and only those with significance above  $1\sigma$  in the model spectra. For the KMT sensitivities, only those of turnoff and giant branch stars are shown, as these two components comprise 90% of the light in the S07 models. For the SWB sensitivities, those under  $2\sigma$  are shown in parentheses.

<sup>a</sup> In addition to the reported sensitivities for individual element abundances, all indices listed here vary with total metallicity.

<sup>b</sup> Sensitivities that only appear at high metallicity.

<sup>c</sup> Sensitivities that only appear at low metallicity.

Figure 2.1: Table 1 from Graves & Schiavon (2008). This shows a number of the Lick Indices, some of which have been used in the EZ\_AGES code, and the element(s) to which they are sensitive.

of these grids the equivalent widths measured from observed objects can be plotted and by using linear interpolation, the value of the parameter or abundance can be found by reference to those values at the model indices.

EZ\_Ages starts by first looking at the H $\beta$  and  $\langle\text{Fe}\rangle$  (taken as the mean of the Fe5270 and Fe5335 indices). H $\beta$  is mostly sensitive to age and not to individual elemental abundances;  $\langle\text{Fe}\rangle$  is sensitive to the overall iron abundance, which scales with the metallicity of the star and the two chosen indices are dependent mostly on Fe and very little on other elements. If the measured equivalent widths do not fall on the grid at this stage, then there can be no calculation of the age and [Fe/H], and so no calculation of

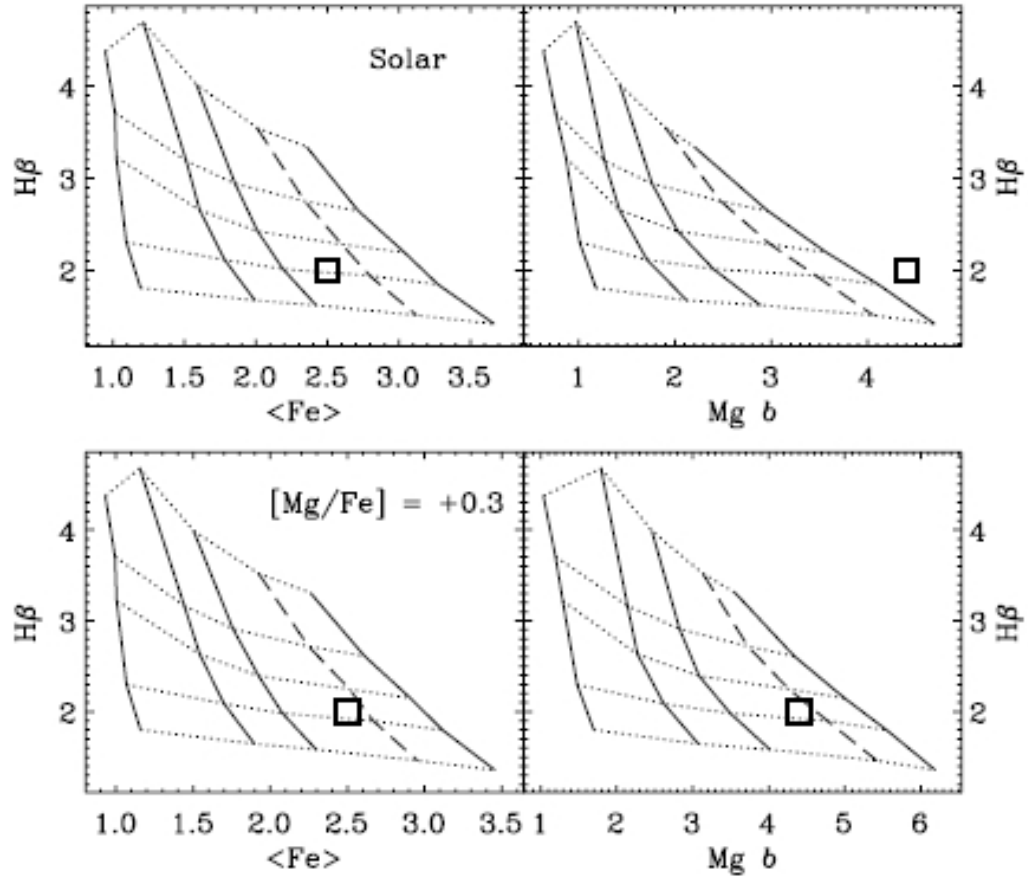


Figure 2.2: Figure 1 from Graves & Schiavon (2008). The upper panels show the model grids for H $\beta$  versus  $\langle\text{Fe}\rangle$  and H $\beta$  versus Mg b for a solar metallicity stellar model. The lower panels show the same, but for a stellar model with an enhanced Mg ( $[\text{Mg}/\text{Fe}] = +0.3$ ). In all panels, the solid quasi-vertical lines are lines of constant  $[\text{Fe}/\text{H}]$  with values of -1.3, -0.7, -0.4, 0.0 & +0.2 (going from left to right) and the dotted quasi-horizontal lines are lines of constant age with values of 1.2, 2.2, 3.5, 7.0 & 14.1 Gyr (going from top to bottom).

any further parameters or abundances is done since they rely on these being found first. If the measured values do fall within one of the grid boxes, then this relative position within the grid forms a reference from which further model grids of other parameters must match within a defined tolerance. From that point on an iterative process goes through the other elements and uses indices in the spectrum which are mostly dependent on the element being calculated or on elements already calculated. These are, C (using C<sub>2</sub>4668), N (using the CN1 and CN2 indices) and Ca (using Ca4227). Mg, using the Mg b index, can be calculated at any point as this index is only sensitive to  $[\text{Fe}/\text{H}]$ ,  $[\text{Mg}/\text{Fe}]$  and age; there is some slight sensitivity to  $[\text{C}/\text{Fe}]$ , but this is small

compared to the others and if C is found first, it can be used in the fitting of Mg. The benefit of this process is that through iteration and convergence any assumptions made at the start about the elemental abundances can be overcome. It is, of course, dependent on the quality of the spectra, the accuracy of the measurements of the equivalent widths and the models being used to compare to the observed objects to.

## 2.5 APOGEE Stellar Parameter & Chemical Abundance Pipeline

The APOGEE pipeline consists of 3 stages (Majewski et al., 2017): the first processes the raw data coming from the telescope; the second combines multiple observations (where they exist) of the same target into a single calibrated spectrum; the third derives stellar parameters and elemental abundances from the calibrated spectrum. The first 2 stages are handled by the *apred* and *apstar* codes (Nidever et al., 2015), while the third stage is dealt with by the APOGEE Stellar Parameter & Chemical Abundances Pipeline (ASPCAP; García Pérez et al., 2016).

The *apred* code deals with the data straight from observation, performing data reduction to turn the 3d data cubes into 1d spectra, and handles the following tasks: flat-fielding, sky subtraction, telluric correction, etc (for a full list, see, Nidever et al., 2015, their Figure 1). It can correct for known instrumental effects, such as differences in sensitivity between the 3 detector chips. At this stage, an initial value is derived for the radial velocity; this is done for each visit to the target. The reduced and calibrated 1d spectra are then passed to *apstar*, which combines spectra from several visits (in cases where they exist) into a single 1d spectrum - this allows a greater final signal to noise and also the identification of possible spectroscopic binary systems through the determination of radial velocity variability (which is then flagged as such in the final data product). A final value of the radial velocity can also be determined at this stage;

the values of radial velocity found by *apred* are not saved through to the final data product (the allStar file) as the values derived from the combined spectra were found to be more reliable.

The final part of the pipeline then takes the calibrated spectra and derives firstly stellar parameters ( $T_{eff}$ ,  $[M/H]$ ,  $\log g$  and a first guess on  $[\alpha/Fe]$ ,  $[C/Fe]$  &  $[N/Fe]$ ; since DR13,  $\xi$ ), then calculates the abundances of the chemical elements (C, N, O, Mg, Al, Si, Ca, Ti, Cr, Fe, Ni, Na, S, V, Mn and Co (originally); plus Ce, Nd, Yb and  $C^{13}$  (which have been recently added)). This is done by the ASPCAP code and it relies upon synthetic spectra from model atmosphere.

The model atmospheres used by ASPCAP have changed between the data releases used in this work. For DR12, the model atmospheres were generated using the ATLAS9 code (Kurucz, 1993). They are 1-d, LTE models and cover a wide range of stellar parameters as well as abundances of  $\alpha$  elements (varying all  $\alpha$  elements together; O, Mg, Si, S, Ca, Ti) and C. The ASS $\epsilon$ T code (Koesterke, 2009) was then employed to generate spectra from these model atmospheres. For DR14, a second set of MARCS model atmospheres (Gustafsson et al., 2008) have been used together with the Turbospectrum code (Alvarez & Plez, 1998; Plez, 2012); these cover the same range of parameters (with the same step values) as the ATLAS9/ASS $\epsilon$ T set, except in  $T_{eff}$ , where they start at 3500K and go in steps of 250K to 5500K. The resultant set of synthetic spectra,  $\sim 3.4$ million, form a 7 dimensional grid which cover a wide range of parameter space. The grid, in fact, consists of two overlapping sub-grids, one covering the 3500-6500K effective temperature range and one covering the 5500-8000K range, corresponding to GK and F type target stars. The details of the range of parameter space covered by the grid can be seen in Figure 2.2.

The synthetic spectra are then treated so that they match the observed APOGEE spectra in wavelength, resolution and sampling, which allows them to be properly used by ASPCAP. For DR12, it was decided that in order to reduce the work that needs to be

Parameter	Starting Value	Finishing Value	Delta
$T_{eff}$ [K]	3500	8000	250
[M/H] [dex]	-2.5	+0.5	0.5
$\log g$ [dex]	0.0	5.0	0.5
$\log \xi_t$ [Km s <sup>-1</sup> ]	-0.301	-0.903	0.301
[Alpha/M] [dex]	-1.0	1.0	0.25
[C/M] [dex]	-1.0	1.0	0.25
[N/M] [dex]	-1.0	1.0	0.5

Table 2.2: Table of stellar parameters, plus  $\alpha$ , C & N abundances found in the grid of synthetic spectra generated using ATLAS9/ASS $\epsilon$ T. This shows the ranges, plus the step sizes, covered by the synthetic spectra.

done by ASPCAP, and allow the analysis to be completed in an acceptable timescale, the dimensionality of the grid is reduced by ignoring micro-turbulence and then calculating these values from the surface gravity (assuming a linear relationship). This reduces the dimensionality to 6d and leaves  $\sim 350,000$  spectra for ASPCAP to work with. This was then changed for DR14, where micro-turbulence was varied (values were taken as either 0, 1, 2, 4 or 8 km s<sup>-1</sup>). Full details of the model atmospheres and synthetic spectra can be found in Mészáros et al. (2012) and Zamora et al. (2015).

ASPCAP consists of a core FORTRAN90 code, called FERRE (for examples of its previous adoption see Allende Prieto et al., 2006; Allende Prieto, 2016), and an IDL wrapper; it is more fully described in García Pérez et al. (2016). The IDL wrapper is responsible for reading in the observations, preparing and submitting jobs to FERRE, handling the output from FERRE and then writing the output files (in FITS format). FERRE uses OpenMP<sup>1</sup> to allow multiple spectral analyses processes to be run simultaneously, thereby reducing the overall code runtime.

ASPCAP adopts a starting point in the 6d/7d parameter space and then it searches for the best fit spectrum using the  $\chi^2$  test as a quality measure. In order to search the parameter space for the lowest  $\chi^2$ , the Nelder-Mead algorithm (Nelder and Mead, 1965) is employed. This algorithm can work in n-dimensional parameter spaces using a sim-

<sup>1</sup><http://www.openmp.org>



plex of  $n+1$  vertices (i.e. a triangle on a 2d plane is a 2d simplex with 3 vertices). The  $\chi^2$  is calculated at each vertex of the simplex. Then, either, the vertex with the highest  $\chi^2$  is discarded and that vertex moved through the opposite face of the simplex, before recalculating the  $\chi^2$  values again or, if no vertex is discarded, then a vertex is moved by some distance depending on a set of specified rules and pre-defined values. The algorithm continues to shrink/grow (depending on the movement of vertices) and move the simplex until a pre-defined convergence condition is met, which for ASPCAP, is that the standard deviation of the  $\chi^2$  values of the vertices falls below  $10^{-4}$ . The algorithm was found to take on the order of a few hundred iterations to reach convergence using the 6d ASPCAP synthetic grid. One potential pitfall of this method is the tendency of the algorithm to get stuck in local  $\chi^2$  minima and in order to circumvent this problem, the process is repeated 12 times using different starting positions in  $T_{eff}$ ,  $[M/H]$  and  $\log g$  and from these runs, the lowest overall  $\chi^2$  is taken as the best fitting spectrum.

Given the non-negligible distance between nodes in the spectral grid, it is necessary to use an interpolation scheme to allow for the derivation of stellar parameters off grid node points; this is faster than simply calculating more synthetic spectra (Mészáros & Allende Prieto, 2013). The stellar parameters are then obtained from the final best fitting spectrum as well as the first estimate of  $[\alpha/M]$ ,  $[C/M]$  and  $[N/M]$ .

The next step is to calculate the elemental abundances for each of the 16 elements, which includes re-calculating abundances for  $[C/M]$  and  $[N/M]$ , and all  $\alpha$  elements. Since the synthetic spectra include no information on how the spectra vary as functions of any element other than C or N, to find the elemental abundances of all 15 elements, windows are defined around lines which are known to be sensitive to the element being sought (the strength of the line being directly related to the abundance of the element). The windows for each of the elements are shown in Figure 2.3, as well as an example stellar spectrum with sky and telluric emissions/absorption. For Fe, C, N & O, there are many windows within the APOGEE spectral range, while for other elements there are significantly fewer (K having only 1 at the extreme blue end of the

range).

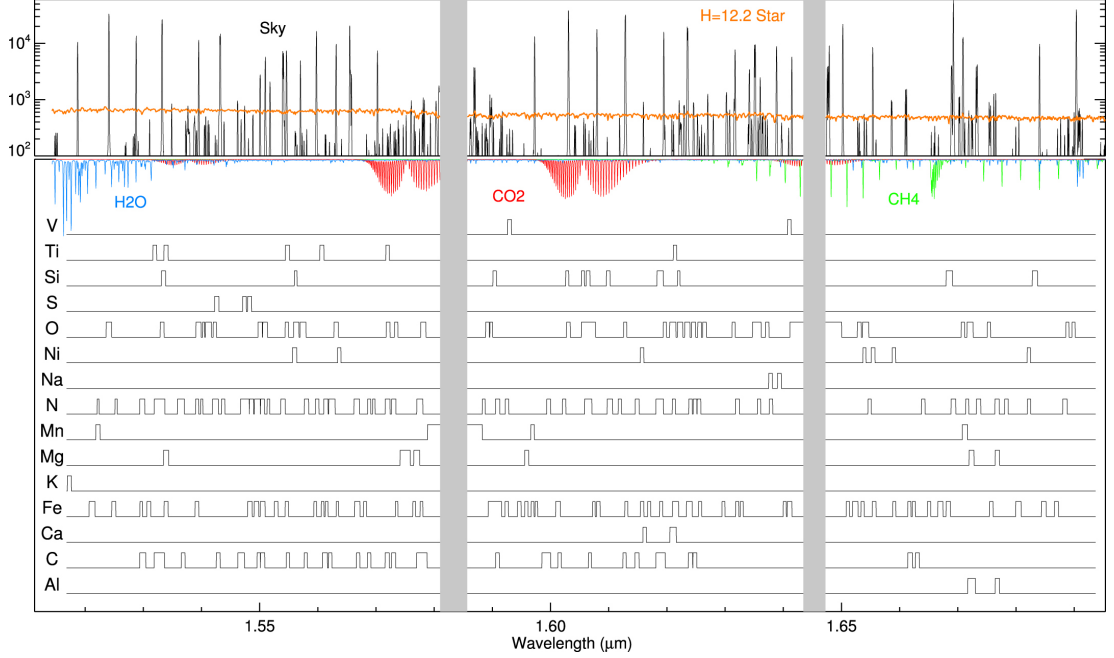


Figure 2.3: At the top, an example stellar spectrum is shown (orange), together with the sky (black) and telluric (blue, red & green) features. At the bottom, spectral windows used by ASPCAP in determining the 15 elemental abundances are shown; the spectral windows have been broadened by  $30 \text{ km s}^{-1}$ . The grey bands are the chip gaps. Taken from García Pérez et al. (2016, Figure 4)

The stellar parameters are assumed to be fixed at this point, taking the values found previously (with the exceptions described next). For determining C and N, the 5 remaining parameters are fixed while allowing the element being sought to vary. For the  $\alpha$  elements,  $[\alpha/M]$  is varied holding the other parameters fixed and for all other elements beyond CNO it is  $[M/H]$  which is varied. Again the grid of synthetic spectra, this time reduced by fixing the other parameters, is searched using  $\chi^2$  as a metric for the best fit, and, so the abundance value of the element being sought. An overview of the ASPCAP process can be seen in Figure 2.4.

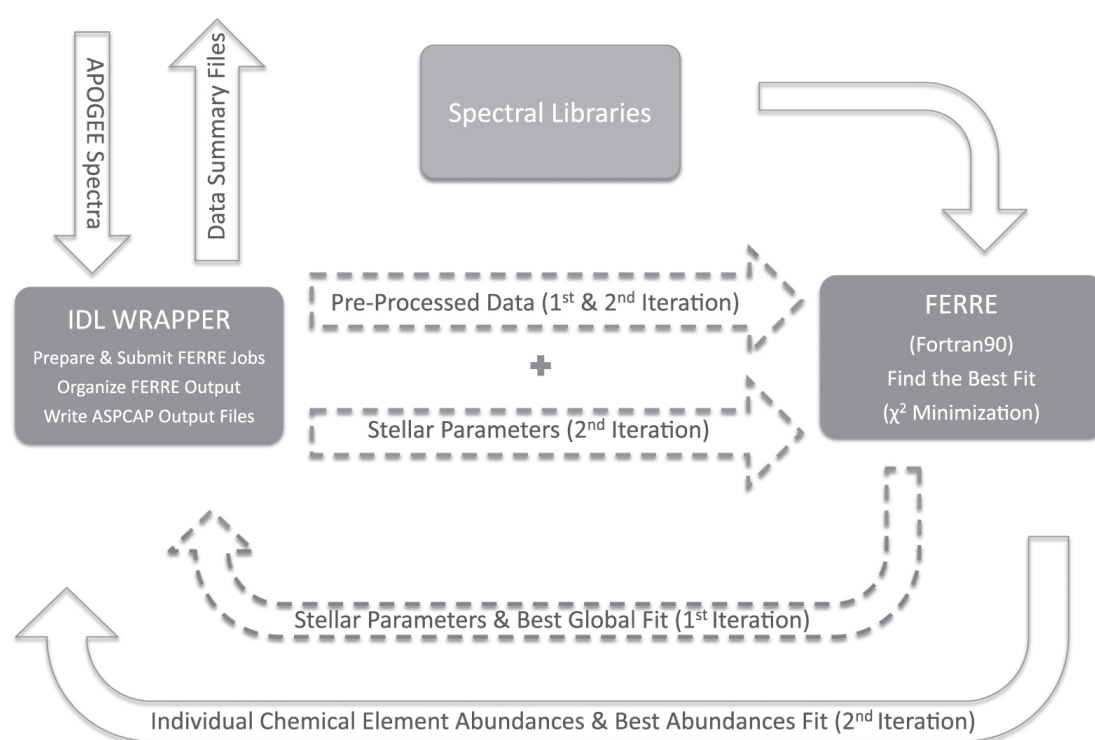


Figure 2.4: Flowchart showing the processes undertaken by ASPCAP. Taken from García Pérez et al. (2016, Figure 1)

## Chapter 3

# Stellar Parameter & Abundance Pipeline

Many current surveys are using custom pipelines which have been developed at great expense and this is likely to continue with those surveys being currently planned (see Sections 2.1, 2.2 & 2.5 for examples). Generic spectral analysis pipelines are being developed which can either act as a comparison to other pipelines or could even remove the need for their development in the first place (see Sections 2.3 & 2.4 for examples). Any generic pipeline would need to be able to take spectra from a wide wavelength and spectral resolution range, and would have to have flexibility in both the input models and indices to be used in the analysis; it would also need to be able to handle stars with a wide range of stellar parameters and elemental abundances. The methodology used in `EZ_Ages` is well suited to this purpose and an obvious extension of the pipeline is the development of it to work with individual stellar spectra and not just integrated spectra. Whilst it makes use of the Lick Indices, it could easily be used with other indices if they could be shown to reliably trace parameters and abundances. This chapter describes the development of just such a generic pipeline.

The new code is called `STARPANDA` (the `STellAR` Parameters `AND` Abundances pipeline). There are several benefits to developing a new analysis pipeline in addition

to ASPCAP and the others that either already exist or are being developed by different groups. Firstly, this will provide an alternative path for processing APOGEE data and allow the cross-checking of results being produced by ASPCAP; this has been discussed and agreed with other members of the APOGEE project. Secondly, another advantage of this method is that it should be much quicker than alternatives. Thirdly, the method is physical, in that the values of stellar parameters and abundances of individual elements are tied directly to the strength of spectral features which can be shown to be sensitive to changes in those values. Finally, whilst the code is currently being developed to take advantage of the results of the APOGEE survey and its high-resolution NIR spectra, it could in principle be extended to optical and near-UV wavelengths. This method is, therefore, only limited by the presence, within the spectral region being considered, of spectral features that can be shown to be reliable indicators of stellar parameters and elemental abundances, which can be probed by synthetic spectra from model atmospheres.

The methodology used in EZ\_Ages is taken as a starting point for the STARPANDA code. An overview of the process used by the code can be seen in Section 3.1. We use the same model atmospheres and synthetic spectra that ASPCAP relies on (see section 2.5 for details), however these could be easily replaced by other sets of synthetic spectra based on different line lists and/or model atmospheres. Most of the models used in codes such as this are based on LTE calculations of stellar atmospheres as the computational resources needed to perform NLTE stellar atmosphere calculations for large model grids are still prohibitively expensive. However it is possible to take advantage of a limited set of corrections to the models using NLTE calculations of hydrogen lines; this could then be used to produce better values for  $T_{eff}$  than can be produced by using just Fe lines. Other corrections will also have to be done to the values of C and N found by the code. This is because RG stars during this phase of their life cycles undergo periods of convection in their photospheres which modify the abundances of many elements, including C and N. Therefore, it is necessary to employ models of stellar evolution in order to correct the observed abundance of elements and calculate

the pre-mixed abundances, which can then be compared with the values calculated by GCE models.

Once a working code has been developed it can be tested against ASPCAP and by reference to literature values for stellar parameters and elemental abundances in well studied reference stars. When it has reached a stage where it can reliably calculate  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[C/Fe]$ ,  $[N/Fe]$ , and  $[O/Fe]$ , it can be released as a tool for the whole astronomical community which will hopefully lead to the quicker and better processing of spectroscopic data from the other current and upcoming large-scale resolved stellar surveys. Further development can continue to expand the number of elemental abundances calculated by STARPANDA so that in the end it will be able to match the output of the ASPCAP pipeline. Before this point it will need to be thoroughly tested and the results from a test set of APOGEE stars compared to the results, for the same stars, from the ASPCAP pipeline.

Before delving into the details of what STARPANDA does and how it does it, it is worth giving a quick description of the pipeline so that future sections can be placed in context. STARPANDA is designed to be a fast, accurate spectroscopic analysis tool, which works by comparing the equivalent width (EQW) measurements of lines in spectra to those measured from synthetic spectra generated from model stellar atmospheres with known stellar parameters, CNO & elemental abundances. The lines used are chosen for their sensitivity to the parameters of interest (i.e.  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[C/Fe]$ ,  $[N/Fe]$ , and  $[O/Fe]$ , plus additional elemental abundances) and may be combinations of several lines, either as averages, weighted averages or ratios. The model parameters form a multi-dimensional parameter space and a function is generated of how the EQW values of the lines vary with parameter values. From this, the parameter values of observed stars can be derived, evaluating these functions using the EQW measured from the observed spectra, and the associated errors are derived by considering the errors on the measured EQW values. The code iterates its analysis until a convergence point is reached. It then loops over all the observed stars.

The user can specify many configuration options by setting variables in a text file, including what type of analysis to run. Either Stellar Parameters & CNO Abundances, or just CNO Abundances using imported Stellar Parameters; Elemental Abundances can be calculated using either STARPANDA-derived parameters, or using imported parameters.

The code requires no training, but users do have to identify suitable lines for analysis (if the spectra being analysed are not taken from the APOGEE survey) and measure them in both observed and synthetic spectra. The input and output files are in FITS format, with the exception being a logfile, which is plain text.

In the following sections we present the structure of STARPANDA (Section 3.1), a detailed description of the algorithms used (Section 3.2), a brief overview of the development of the algorithms (Section 3.3) and tests used to show that the algorithms are working satisfactorily (Section 3.4).

## 3.1 Structure of STARPANDA

In order to aid in the development of the code, it has been broken down into several sections. This allows different functions to be isolated from each other and helps to reduce the number of lines of code being worked on at any one time. All user-defined options are placed in a single file, meaning that end-users only have to edit the contents of a single file before running the code. A flowchart showing the overall layout of the code blocks can be seen in Figure 3.1. In the following subsections, broken down by Input, Main Code and Output, the basic functions of each of the different code blocks are described.

## STARPANDA STRUCTURE OVERVIEW

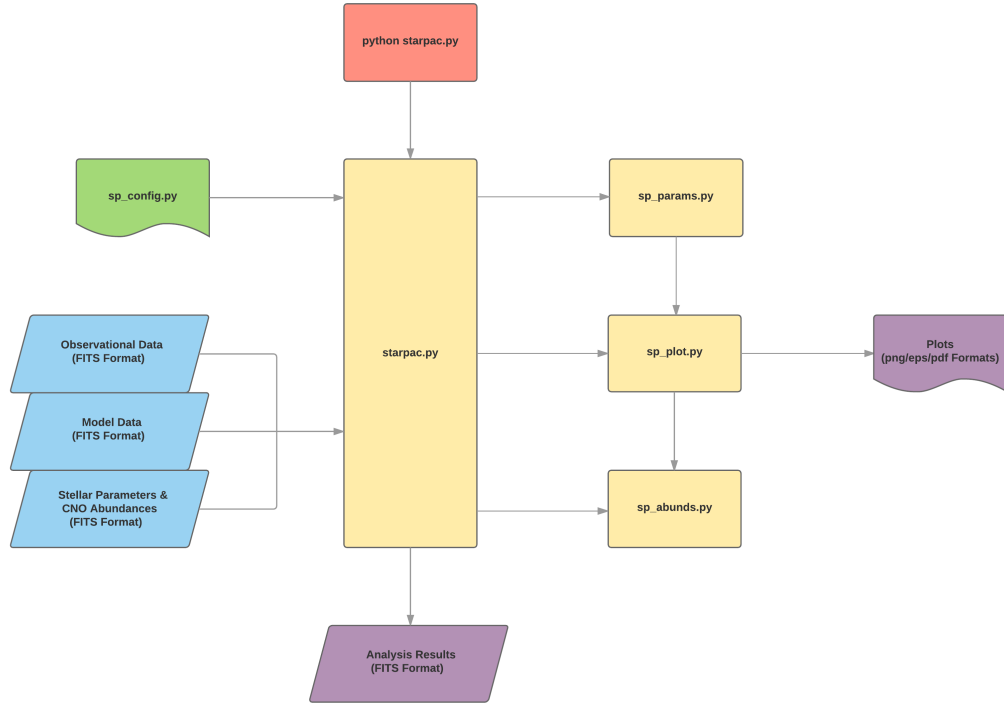


Figure 3.1: Overview flowchart of the STARPANDA code, showing the input files (Green for user-editable & Blue for FITS formatted data files, the internal code blocks (Cream) and the output files (Purple). The code is run from the command line by invoking “python starpac.py”

All of the code files are found in the main directory and, by default, there are 2 sub-directories, *Input* and *Output*, which hold the input and output files. The code comprises 5 Python files and a plain text file:

- *starpac.py* - Main code - handles data input & output
- *sp\_params.py* - Derives stellar parameters & CNO abundances
- *sp\_abunds.py* - Derives elemental abundances
- *sp\_plot.py* - Plotting routines
- *sp\_config.py* - User editable configuration file



- *sp\_readme.txt* - Readme text file

For a full description of how to install STARPANDA, its Python dependencies and how to run the code, see Appendix A.

### 3.1.1 Input Files

There are several input files which must be present for the code to run. If any of these are missing or are not correctly formatted then the code will fail to run successfully. Details of how the data these files contain are used can be found in Section 3.2.

One is the STARPANDA file, *sp\_config.py*, which contains all the user configurable options, including:

- Control options for what analysis code runs
- Control options for what plots the code produces
- Input file location & names
- Output file location & names
- Column names for stellar parameters, CNO & elemental abundances in model data input files
- Column names for stellar parameters, CNO & elemental abundances indicator lines
- Analysis order
- Column names for input parameters
- Initial surface gravity & CNO abundance values

- Iteration counter maximum & tolerances for the output parameters

Details for the formatting of each entry in this file can be found in the readme file (*sp\_readme.txt*), which is included in Appendix A.2. The values of the variables set in this file are directly referenced by STARPANDA during runtime and any change to variable names or the formatting of the data they contain will cause a fatal error.

Depending on what analysis options have been chosen, further input files may be required. While the number and contents of the files required will depend on the analysis option, in all cases there must be a FITS formatted file containing the observational data. This will need to have a column containing IDs for each target object and then columns containing the measured EQWs and the associated errors for each line used in the analysis. Additional columns may be present that are not used by STARPANDA for analysis and will not affect the functioning of the code. Also, it is vital that the column names used in files are consistent across input files (e.g. lines to be used in deriving [C/M] must have exactly the same names in both observed data and model data files).

If an analysis of stellar parameters and CNO abundances is requested, then a second file must be present which contains information on the models against which the observed data are compared. The model data file should have columns listing the stellar parameters and CNO abundances of the model atmospheres and measurements of the EQWs of lines from the synthetic spectra (generated from the model atmospheres). The lines should match those present in the observed data file.

If an analysis of just CNO abundances is requested, then in addition to the files needed for a full stellar parameter and CNO abundance analysis, an additional file is needed which contains the IDs of the target objects and stellar parameter values for each.

If an analysis of elemental abundances is requested, then in addition to the file containing the observed EQWs, there should be a second model data file containing information similar to that which is needed for stellar parameter and/or CNO abundance analysis. The difference between the two files is that in this case additional columns should be present giving elemental abundances for each element of interest from the model atmosphere and the measured EQWs from the synthetic spectra for each line to be used in analysis. These should be given as floating point arrays in each case, e.g. for element [X/M] there should be a column containing an array of abundances in the model atmosphere and a column containing an array of EQW values from a line in the synthetic spectra.

The code is able to use multiple lines from a spectrum to derive stellar parameter and/or elemental abundance values, but only by taking an average of the values of all lines specified in the *sp\_config.py* file. It does this when data are read in from the observed and model data files before they are stored in working data structures prior to analysis. This could also be done outside of the code by using other code or tools to combine values from multiple columns before saving them as a new column. If the user requires multiple lines to be combined in any other method than a straight average, such as a weighted mean, then this would have to be done prior to running STARPANDA.

### 3.1.2 Output Files

The files produced by STARPANDA are dependent on the control options chosen by the user and specified in the configuration file, *sp\_config.py*. A description of these files and what they contain can be seen in Section 3.2 and subsections. However, in all cases running STARPANDA will result in a logfile being produced, even if no analysis options have been chosen.

## 3.2 STARPANDA Algorithm

STARPANDA comprises 4 software blocks, with each block being responsible for specific functions. A schematic representation of the working of the STARPANDA algorithm can be seen in the Figures 3.2, 3.5, 3.11 & 3.13, and the details of how it accomplishes this is described more fully in the following sections.

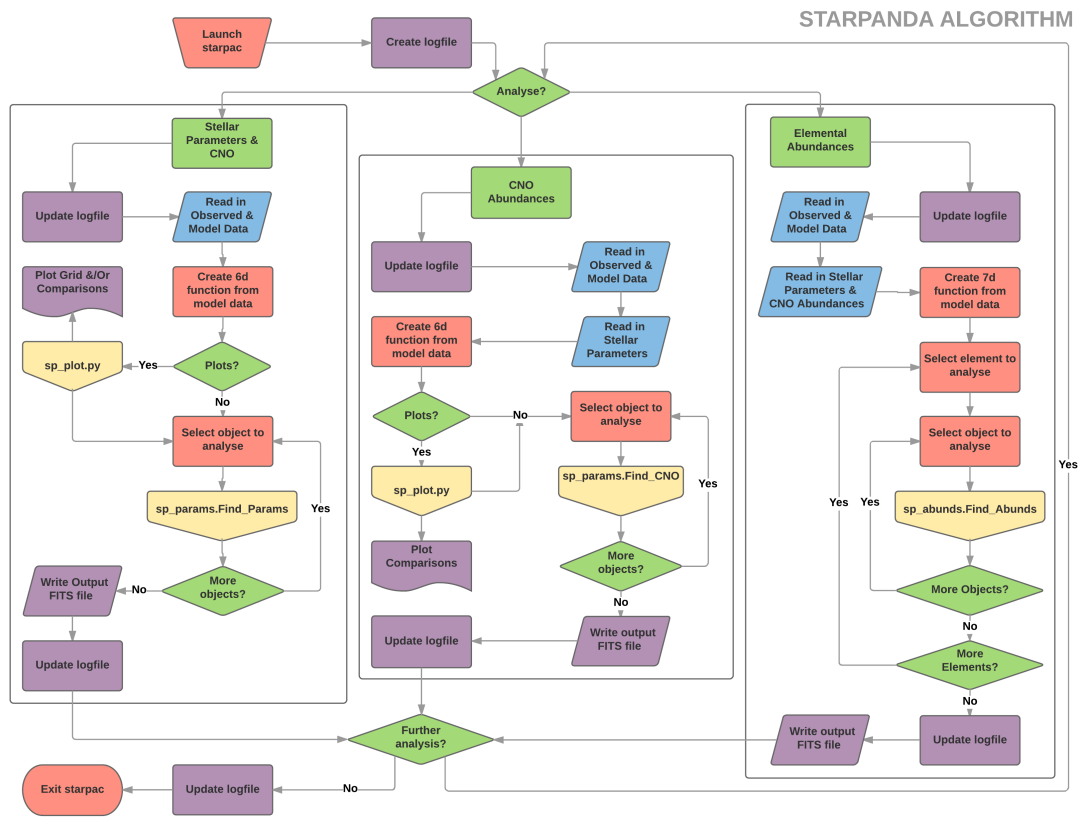


Figure 3.2: Flowchart of the STARPANDA code, showing the overall structure of the code (Red denotes internal code processes, Green are decisions & analysis options, Blue are data read-in processes, Cream are calls to analysis methods in other code blocks, and Purple are output processes).

### 3.2.1 Starting Tasks

The code is started from the command line by the user invoking Python together with the main code file, e.g. “python starpac.py” - if run from the directory containing the

STARPANDA code files. Immediately upon launch, the code reads some of the variables set in *sp\_config.py*, notes the start time, and creates the logfile (with the filename of *splog\_output\_name.txt* in the directory specified by **output\_dataloc**). The logfile at this stage just records the current date.

The next step is to decide what analysis options have been selected by checking the **param\_analysis**, **cno\_analysis** & **abund\_analysis** boolean variables in *sp\_config.py*. **param\_analysis** is the derivation of stellar parameters and CNO abundances, while **cno\_analysis** is the derivation of CNO abundances using stellar parameters imported from elsewhere - both of these can be run either with or without iterating. **abund\_analysis** then is elemental abundance analysis for all the elements listed in **abund\_labels** and does not iterate - stellar parameters and CNO abundances are provided, either by a previous run of STARPANDA or from elsewhere. The options are checked in this order and if **True**, then the specified analysis is run; if **False**, then for **param\_analysis** and **abund\_analysis**, a note is made in the logfile to indicate that these options have not been selected.

### 3.2.2 Parameter & CNO Abundance Analysis

If the user has selected **param\_analysis**, then the first action is to update the logfile to indicate this and to record the lines being used for analysis (**temp\_label**, **feh\_label**, **logg\_label**, **cm\_label**, **nm\_label** & **om\_label**) together with the filenames of the observed and model data files, **obs\_datafile** and **model\_datafile** respectively. Next the structures used to hold the Observational, Model and Output data are created and the observation and model data are read in using **astropy.io.fits**<sup>1</sup> methods (astropy; Astropy Collaboration et al., 2013). The reading in of the observational data requires that the first column in the data file be the unique ID of the target object and from this the number of objects to be analysed can be found. The EQW and EQW error values are

<sup>1</sup><http://docs.astropy.org/en/stable/io/fits/index.html>

read in using the values in the label variables to identify the columns of interest - each column is multiplied by one over the number of labels for each parameter before being added together and saved in the observational data structure. The model data can then be read in, starting with the construction of sets of values for the stellar parameters and CNO abundances used to generate the model atmospheres, before copying these values into the model data structure and then, as with the observed data, the model EQW values are, for each parameter, multiplied by 1 over the number of labels used by that parameter before being summed and then saved into the model data structure. Finally, in order to meet the requirements of an interpolation scheme used later, the model data gets sorted in ascending order with the priority order being:  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[O/Fe]$ ,  $[C/Fe]$  and  $[N/Fe]$  (so the first row contains the lowest value for each parameter and the last row contains the highest for each).

Once the data are read into the working structures, the code then creates a function to interpolate the model data. The model data, then, represents a 6 dimensional parameter space. Each node point within this parameter space is defined by the values of the stellar parameters and CNO abundances used to create the synthetic spectra, and additionally each node point has associated with it the EQW values of the lines being used to trace those 6 parameters. The whole structure forms a regularly gridded hyperrectangle, with well-defined start and end points for each dimension and nodes labelled with a 1 dimensional array of EQW values (even if multiple lines are specified for tracing a parameter, only the average EQW value is stored in the model data structure, hence the labelling is always a 1d array). Thanks to the regular nature of the parameter space being used, we can make use of a SciPy interpolation method called **RegularGridInterpolator**<sup>2</sup> [RGI].

**RGI** takes arrays of points representing positions in N dimensional parameter space, together with arrays of values representing data at those positions. The method re-

---

<sup>2</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.RegularGridInterpolator.html>

quires that the position arrays form a regular grid with no missing points. This allows it to save computational time in generating a function to describe the input data by dispensing with the triangulation of input positions which would be necessary for non-regularly structured data sets. **RGI** is only able to use nearest neighbour or linear interpolation. Any attempt to evaluate positions outside of the input data ranges can be set to either return a pre-defined value or raise an error.

For STARPANDA the RGI is constructed such that any attempt to evaluate the EQW values of a point outside of the defined parameter ranges will return a NaN value; this was decided upon as using this interpolator to extrapolate values outside of model parameter space would result in errors which are relatively large and hard to fully quantify. Linear interpolation was chosen as we required some form of interpolation, which cannot be satisfied by nearest neighbour. For linear interpolation, a multi-dimensional linear interpolation scheme (Weiser & Zarantonello, 1988) across the  $N$  dimensions of the input data is utilised. For simplicity, imagine we are working with only 3 dimensions  $(x, y, z)$  and that our data forms a regular lattice, while we wish to evaluate a point  $p$  at some position within this parameter space. Starting from the nearest nodes in each dimension, a single “cube” (in reality a 6-sided irregular polyhedron) is defined surrounding the point  $p$ . Considering the  $x$  dimension first (though the order of interpolations is actually unimportant), the value is interpolated along  $x$  at each of the 4 points defined by the variations of  $y, z$ . Next, interpolation is considered along the  $y$  dimension between each pair of values created previously holding the  $z$  value fixed; this creates 2 new points directly above and below  $p$  in  $z$ . Finally, interpolation is considered between the final 2 points, giving the value at point  $p$ . As can be seen from this method, it is easily scaleable to an arbitrary number of dimensions as long as the condition that the data form a regular grid is met.

Next, a check is made to see if 2 diagnostic plots are required, as specified by the boolean variables **full\_grid\_plot** and **comp\_plot** in *sp\_config.py*. The first of these 2 plots generates an initial  $T_{eff}$ -[Fe/H] grid (Figure 3.3) using the initial values of sur-

face gravity and CNO abundance specified by the user and on to this plots all the target objects. From this plot it is possible to get a rough (by eye) estimation on the upper limit of the number of target objects for which stellar parameters and CNO abundances can be derived. The actual number of successful derivations is usually lower than this as it is possible that objects which are within parameter space ranges for  $T_{eff}$  and  $[Fe/H]$ , will be out of the range covered by the models for the other parameters. Additionally, it is also possible that during iteration objects will move out of model parameter ranges. The second plot (Figure 3.4) then shows histograms of the EQW values of the tracer lines for both the model and observed data, which provides another way of checking the compatibility of the model measurements to interpolate the observed data.

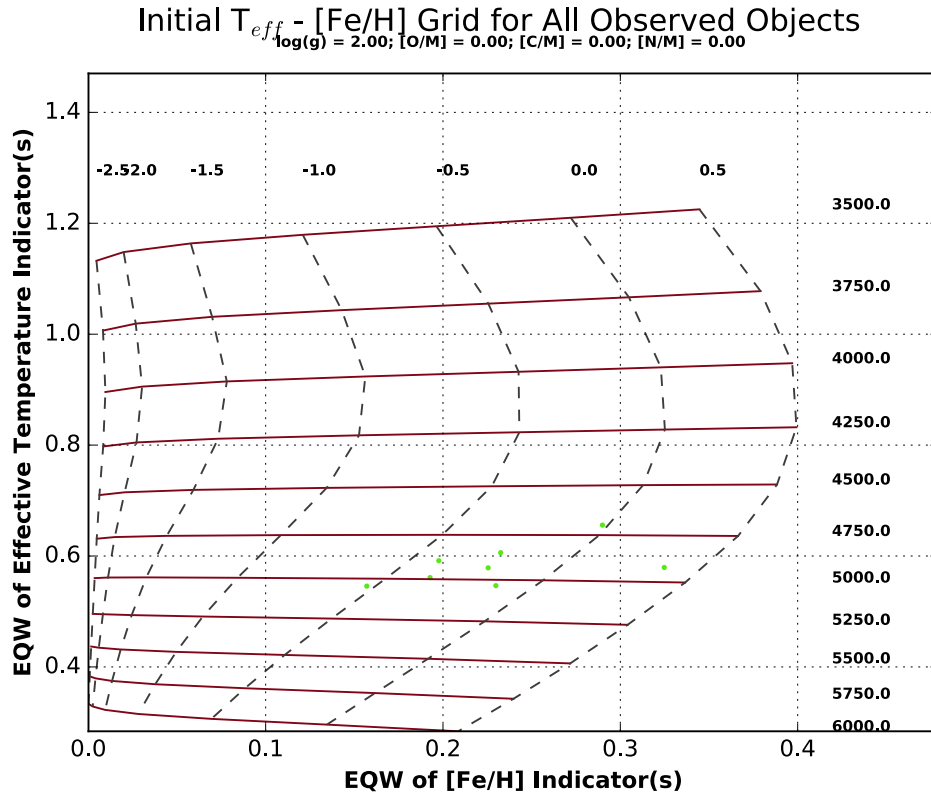


Figure 3.3: An example of an initial  $T_{eff}$ - $[Fe/H]$  grid produced, assuming initial values for the other parameters (shown in the sub-title), by **Find Params** showing the position of all target objects (green dots) in model parameter space. The quasi-vertical lines (dashed grey) are lines of constant  $[Fe/H]$ , with values shown above the grid; the quasi-horizontal lines (solid red) are lines of constant  $T_{eff}$ , with values shown to the right of the grid.



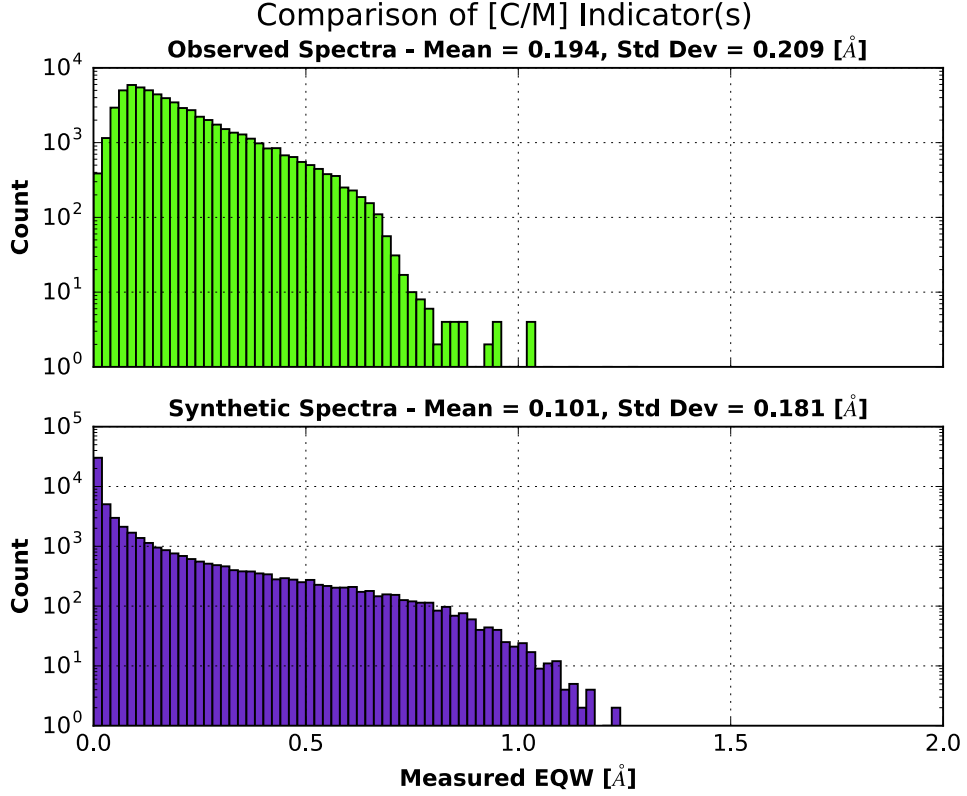


Figure 3.4: An example of the comparison of indices used to evaluate the suitability of specific indices to trace stellar parameters & CNO abundances. EQW values of indices are compared between the observed & model data (top and bottom respectively). Here the indices being used are: ‘CO\_15997’, ‘CO\_15582’, ‘CO\_15982’ & ‘CO\_16620’; these are averaged and then used to derive [C/M].

The code then loops over all target objects and calls the **Find Params** method in *sp\_params.py* passing the observed, model and output data structures together with the interpolated model function, logfile and a house-keeping iteration counter. The overall structure of the **Find Params** algorithm can be seen in Figure 3.5.

The **Find Params** method starts by saving the ID of the object being analysed to the output data structure. It then reads initial values for  $\log g$ , [C/Fe], [N/Fe], and [O/Fe] from *sp\_config.py* (**model\_init\_logg**, **model\_init\_cm**, **model\_init\_nm** & **model\_init\_om** respectively). With those values it can then start deriving the stellar parameters and CNO abundances; if iteration has been selected, by setting a non-zero value, in the variable **iter\_max**, then a loop is started that will continue until one of the following

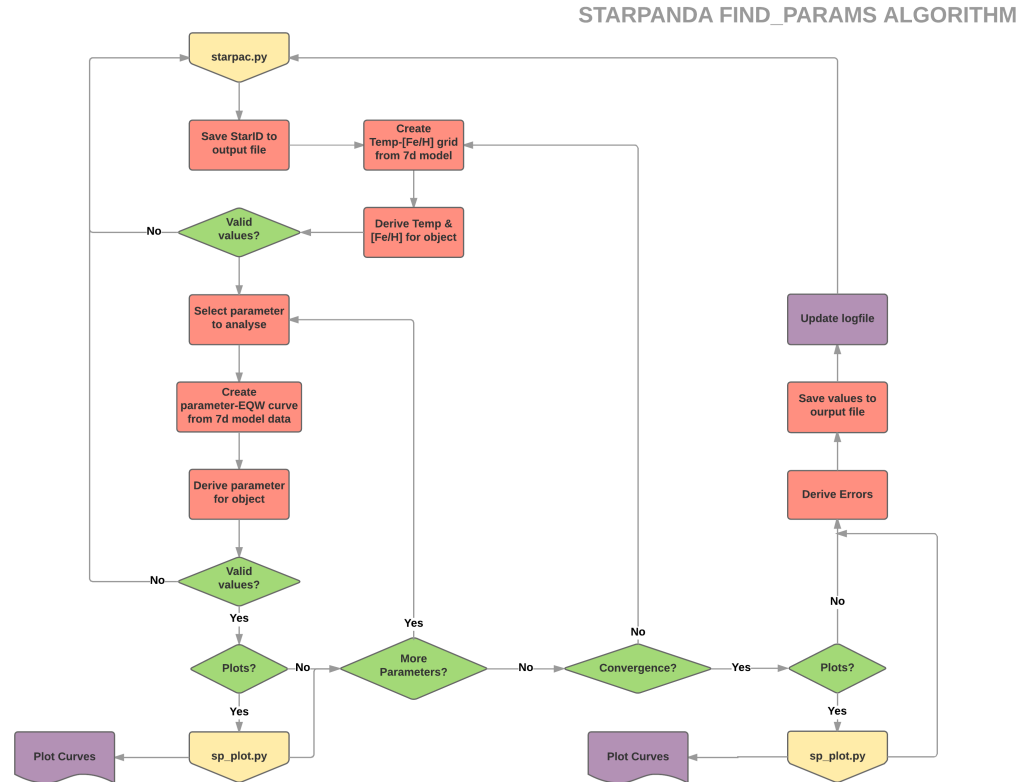


Figure 3.5: Flowchart of the STARPANDA code, showing how the Find\_Params method (part of *sp\_params.py*) works. Objects coloured cream are calls from/to other code blocks, red represents internal code processes, green shows decision points and purple relates to output files.

criteria has been met:

- Convergence - all parameters reach a point during iteration where they change by less than specified values between iterations
- The iteration limit is reached
- Divergence - one of the parameters moves outside of model parameter space

For STARPANDA, convergence of all 6 parameters is determined by the difference between the current iteration and the previous iteration, with convergence being reached if these values are less than those specified by the user in the variables **tol\_temp**, **tol\_feh**, **tol\_logg**, **tol\_cm**, **tol\_nm** and **tol\_om**. The iteration limit, specified by a non-zero value for **iter\_max**, prevents issues where the code could become stuck in an

infinite loop where the parameter values fail to either converge or diverge (the most often observed example being when the code bounces between 2 values of one or several parameters which have differences greater than the convergence tolerances set by the user; in this case, the logfile is updated to indicate that the analysis failed to converge within the iteration limit. Divergence, then, is the case where a parameter moves outside of the range of values covered by the model data; in these cases, rather than either extrapolating values or discarding the “bad” value and returning to a previous “good” value to try again, we simply end the analysis and indicate in the logfile that a parameter moved out-of-bounds and which parameter it was.

Effective temperature and metallicity are always the first parameters to be derived and are done together by interpolating the target on a 2 dimensional model  $T_{eff}$ -[Fe/H] grid (e.g. Figure 3.6). If this is the first iteration then the initial values of  $\log g$ , [C/Fe], [N/Fe], and [O/Fe] are used to generate the  $T_{eff}$ -[Fe/H] grid; in subsequent iterations, values derived by STARPANDA are used. These fixed values are then used to evaluate the model RGI function created in **starpac.py** and evaluate the EQW of the  $T_{eff}$  and [Fe/H] lines for all the  $T_{eff}$  and [Fe/H] value combinations. This forms a 2 dimensional grid, which, using the EQW values of the target object for the same lines allows the  $T_{eff}$  and [Fe/H] values to be derived. This second interpolation is done using the SciPy interpolation scheme **griddata**<sup>3</sup>.

The **griddata** method takes arrays of positions in N dimensions together with an array of values associated with those positions and from these it creates a function. The input positions do not need to form a regular grid and can be randomly distributed, though each point in the parameter space can only be represented by a single value. The method is able to utilise different kinds of interpolation, including nearest neighbour, linear and cubic (though the latter is only available for 1d or 2d data sets) and any attempt to evaluate the function outside of the range of input positions will either

<sup>3</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.griddata.html>

return a NaN or pre-defined floating-point value (no extrapolation is possible).

For STARPANDA we make use of the 2d cubic interpolation option and specify that any attempt to evaluate out-of-bounds points return a NaN value. In the 2d case, where the positions are represented by the coordinates  $x$  and  $y$ , with values  $z$  at those positions, **griddata** would create a function of the form  $z = f(x, y)$ . The first task in generating this function is to triangulate the positions of the input data (this is done using **Qhull**<sup>4</sup>) on to a 2d surface of Clough-Tocher triangles (Clough & Tocher, 1965), which are then further subdivided into 3 sub-triangles. Each Clough-Tocher triangle is defined by 12 parameters, including the positions of the vertices and the gradients of each sub-triangle. A cubic interpolating Bezier polynomial is fitted to each sub-triangle. The final 2d surface is smooth and minimises changes of gradients between triangles. While **griddata** is a slower interpolation scheme than some others (due to its use of triangulation of input data to create the function), it is still relatively fast and it does enable a non-linear interpolation of the data. In our case the increase in time needed for generating the function and evaluating it is not significant, so it is worth using given the increased precision gained in  $T_{eff}$  and  $[Fe/H]$  when using a cubic interpolation scheme.

A check is then made to see if debugging plots are required (as specified by the boolean variable **debug\_plots**), and if so, a  $T_{eff}$ - $[Fe/H]$  grid showing the target object, the derived  $T_{eff}$  and  $[Fe/H]$  values, together with the current assumed values of  $\log g$ ,  $[C/Fe]$ ,  $[N/Fe]$ , and  $[O/Fe]$  is produced; if iteration is specified, plots will be produced each time new  $T_{eff}$  and  $[Fe/H]$  values are derived. Finally, a check is made to see if the derived values are out-of-bounds; if so, analysis ends, otherwise the analysis continues.

After  $T_{eff}$  and  $[Fe/H]$ , the order of derivation is flexible and determined by the user through the **sp\_order** string variable. The permitted values in this variable are: ‘log  $g$ ’, ‘C’, ‘O’ & ‘N’. The code loops over the entries and runs each derivation as specified.

---

<sup>4</sup><http://www.qhull.org/>

## Temperature-Metallicity Grid for StarID 2M12003647+1702176 (Iter: 6)

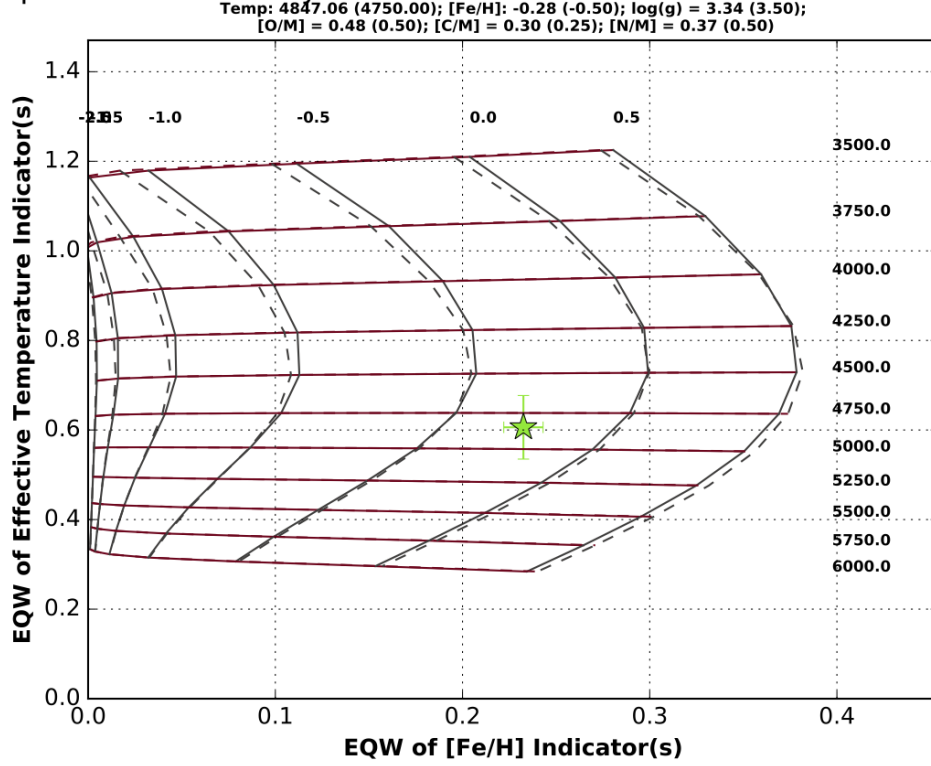


Figure 3.6: An example of a  $T_{eff}$ -[Fe/H] grid showing the position of a target object in parameter space, with specific values of the other parameters. The title shows the ID of the star being analysed and the number of the current iteration. The sub-title shows the values for  $T_{eff}$  and [Fe/H] interpolated from this plot, together with the values of the other parameter from the previous iteration; the values shown in parentheses are those of the nearest model node values in each dimension. The quasi-vertical lines (grey) are lines of constant [Fe/H], with values shown above the grid and the quasi-horizontal lines (red) are lines of constant  $T_{eff}$ , with values shown to the right of the grid; solid lines show the interpolated grid derived from the values found in the previous iteration, while the dashed lines show the grid for the nearest node. The star being analysed is shown as a green star, with error bars within the grid.

The details of the derivation for the remaining parameters are essentially the same in each case with just the actual numbers in variables/arrays changing, so assuming log  $g$  is the first parameter to be derived the following description can be re-read replacing that parameter with [C/Fe], [N/Fe] and [O/Fe] as required. The model RGI function is evaluated using the initial/previously derived parameters for all but log  $g$ , giving a set of model log  $g$  values and the EQW of the line(s) tracing that parameter at those values. This is then interpolated further using the SciPy interpolation scheme `interp1d`<sup>5</sup>.

<sup>5</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.interp1d.html>

The **interp1d** method takes two arrays of  $x$  &  $y$  coordinates and creates a function of the form  $y = f(x)$  to describe the data. The kind of interpolation performed by the scheme is flexible and can be a nearest neighbour interpolation (e.g. a step function between points), a simple point-to-point linear interpolation, or a spline interpolation of either first, second or third order (i.e. slinear, quadratic and cubic respectively). Through this method, users are also able to specify how to handle attempts to evaluate points outside of the input data ranges, with the options being either to return a pre-defined value (such as an integer or NaN), raise an error or extrapolate to the new point (this only works when using nearest neighbour or linear interpolation).

The kind of interpolation chosen for this work is a cubic spline and any attempt to evaluate a value out-of-bounds returns a NaN value. The cubic spline is a series of piecewise third order polynomials (of the form  $y = ax^3 + bx^2 + cx + d$ ) which are forced to go through all input data points. The resultant function is smooth, easy to evaluate and bounded by the range of the input data. Evaluating this function at the EQW of the target then gives a new value for  $\log g$ .

Next a check, again **debug\_plots**, is made to see if a plot of the interpolated variation in model EQW value with  $\log g$  is required. Finally, the value is checked to ensure that it is within model parameter ranges before any further analysis is undertaken. The new value for  $\log g$  can then be used in future parameter derivations, both in this and the next iteration. The code then continues on to the next parameter specified in **sp\_order**. Figure 3.7 shows an example of the plot that is generated for debugging  $\log g$  or CNO abundances. It shows the interpolated variation of the EQW of the indicator line(s) against the parameter value and in addition to the interpolated curve, the target EQW value is plotted as a horizontal line together with the nearest node curve line (as a check of the interpolation's accuracy).

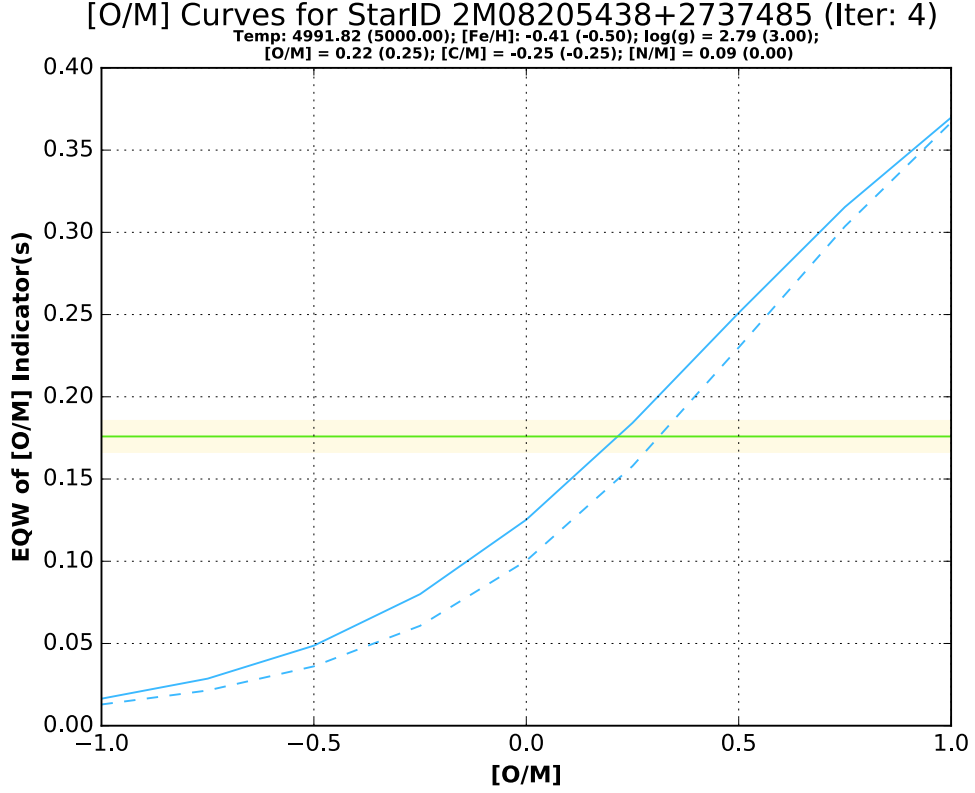


Figure 3.7: An example of an interpolated [O/M] curve showing the position of a target object in parameter space, with specific values of the other parameters. The title shows the ID of the star being analysed and the number of the current iteration. The sub-title shows the value for [O/M] interpolated from this plot, together with the values of the other parameter from the previous iteration; the values shown in parentheses are those of the nearest model node values in each dimension. The solid blue line shows the interpolated variation of [O/M] with EQW of the indices being used at the values of the parameters found in the previous iteration; the dashed blue line is the same relationship at the nearest node point. The horizontal green line is the measured EQW for the star being analysed and the yellow shaded regions denotes the error range on this measurement.

Once values for all 6 parameters have been obtained, and if they are all in-bounds, the code then checks to see what iteration it is on and what value for the maximum number of iterations has been specified by the user in `iter_max`. If this is the first iteration and no iterations are required or if this iteration is the same as the number set in `iter_max`, then again the analysis ends here and flags are set to either indicate the iteration was not required or that analysis failed to converge within the iteration limit. If this is the first iteration of  $n$ , then the analysis returns to derive new values using those found in this iteration. If, however, this is not the first iteration of  $n$ , then before returning to make

another derivation of the parameters, a check is made to see if the values have converged according to the limits set by the user (in **tol\_temp**, **tol\_feh**, **tol\_logg**, **tol\_cm**, **tol\_nm** and **tol\_om**); if the convergence criteria have been met then analysis ends.

Regardless of how the analysis ends (convergence, divergence, etc), the code then checks if a set of final analysis plots (e.g. Figures 3.6 & 3.7) are required by testing the boolean variable **final\_plots**. These are the same plots that **debug\_plots** produces, but this just does the last iteration of analysis, so producing only 5 plots per target object.

Next a check is made to see what the outcome of analysis was. If the result was divergence, then null values (currently set at  $-999.99$ ) are saved into the output structure for parameter and parameter error values, parameter and parameter error flags are set to show that values are untrustworthy, the number of iterations completed, and the **STAR\_FLAG** flag is set to show which parameter moved out-of-bounds first. If the result was that the iteration limit was reached before convergence, then, the same values are set in the output structure, with the exception of the **STAR\_FLAG** flag, which is now set to show a failure to converge within the iteration limit. If the result was that analysis was successful, then the converged parameter values are saved to the output structure together with flags to show that they are considered trustworthy, and the **STAR\_FLAG** flag is set to show successful analysis; the code can then proceed to derive error values for each parameter.

Errors are derived in the same manner as the actual parameter values. The order of derivation is not important and is fixed as,  $T_{eff}$ -[Fe/H],  $\log g$ , [C/Fe], [N/Fe], and [O/Fe]. Currently the errors are not treated as being dependent on anything other than the errors in the measured observed EQW values of the line(s) being used for each parameter; see Chapter 6 for a discussion on how this may be improved in future. Since the relationship between a parameter and EQW of the line(s) used is rarely linear, it is necessary to consider the positive and negative error on the EQW separately and since a target may fall on the edge of model parameter ranges in any of the 6 parameters, it



is possible that while one of the errors is in-bounds, the other may be out-of-bounds. Therefore, the positive parameter error is derived by adding the EQW error to the EQW value and then re-deriving a value for that parameter (using the same methods as described above) with the new EQW value; the negative parameter error is derived by subtracting the EQW error from the EQW value. After each pair of parameter error values are derived, they are checked to see if they are in-bounds or out-of-bounds and a flag is set to indicate the reliability of the pair. This continues for all 6 parameters.

The final step undertaken by *sp\_params.py* is to update the logfile with the results of the analysis for this target. Regardless of the success or failure of the analysis attempt, the following details are always saved to the logfile:

- STAR ID
- STAR EQW - The EQW values used for each of the 6 parameters
- $T_{eff}$  - An array of all  $T_{eff}$  values derived at each iteration, plus the final  $T_{eff}$  error values for both directions
- [Fe/H] - An array of all [Fe/H] values derived at each iteration, plus the final [Fe/H] error values for both directions
- $\log(g)$  - An array of all  $\log g$  values derived at each iteration, plus the final  $\log g$  error values for both directions
- [C/M] - An array of all [C/M] values derived at each iteration, plus the final [C/M] error values for both directions
- [N/M] - An array of all [N/M] values derived at each iteration, plus the final [N/M] error values for both directions
- [O/M] - An array of all [O/M] values derived at each iteration, plus the final [O/M] error values for both directions

Then, depending on the success of the analysis, a note is made to indicate either that the analysis was completed successfully, that analysis failed due to the iteration limit being reached before convergence or that one of the parameter moved out-of-bounds and at which iteration that occurred (the actual parameter which caused the failure can be seen by looking at the parameter array values for the first appearance of a NaN value). Once this has been done, control passes back to *starpac.py* to either begin analysis of the next target object or if that was the last/only object, then to start outputting the analysis results.

Once analysis of all input objects has been completed, the task of writing out the results can begin. First, the logfile is updated to show that analysis was completed, how long it took and how many target objects were successfully analysed, how many failed to converge and how many failed. Next the output FITS table is constructed using the `astropy.table` method and the following columns are created:

- STAR\_ID - The ID of the target object
- STAR\_FLAG - The flag indicating the analysis result
- ITER\_FLAG - The flag indicating the number of iterations required
- [PARAM] - The final derived parameter value or a null value
- [PARAM]\_FLAG - The flag indicating the trustworthiness of the parameter value
- [PARAM]\_ERR\_NEG - The parameter error value with the EQW error subtracted
- [PARAM]\_ERR\_POS - The parameter error value with the EQW error added
- [PARAM]\_ERR\_FLAG - The flag indicating the trustworthiness of the parameter error values

The last 5 columns are repeated for each of the 6 parameters (TEMP, Fe\_H, log\_g, O\_M, C\_M & N\_M). The file is then written out with the filename *spdata\_output\_name\_params.fits*.

An example of the table can be seen in Figure 3.8.

TOPCAT(6): Table Browser

Table browser for 6: spdata\_T5\_S6\_TP\_params.fits

	STAR_ID	STAR_FID	ITER_FLAG	TEFF	TEFF_FLAG	TEFF_ERR_NEG	TEFF_ERR_POS	TEFF_ERR_FID	Fe_H	Fe_H_FLAG	Fe_H_ERR_NEG	Fe_H_ERR_POS	Fe_H_ERR_FID	log_g	log_g_FLAG
1	2M19121933+3848348	2	0	-999.99	1.	-999.99	-999.99	3.	-999.99	1.	-999.99	-999.99	3.	-999.99	1.
2	2M03201522+6139014	0	7	4924.27111	0.	4555.59697	5341.96104	0.	0.13805	0.	0.	0.	0.	3.	4.85396
3	2M00370524+5005023	0	4	4893.80481	0.	4497.76443	5347.05297	0.	-0.44306	0.	-0.50226	-0.38744	0.	2.6726	0.
4	2M0205438+2737485	0	4	4991.81748	0.	4953.27262	5030.91986	0.	-0.41204	0.	0.	0.	0.	3.	2.78626
5	2M18031259+0150582	0	4	4697.99135	0.	4248.7966	5229.33635	0.	-0.05322	0.	-0.11478	0.01065	0.	2.83354	0.
6	2M04315025+3704126	0	4	4932.94257	0.	3934.54279	1.	-0.28351	0.	-999.99	-999.99	-999.99	3.	2.65492	0.
7	2M17323218+2349164	2	0	-999.99	1.	-999.99	-999.99	3.	-999.99	1.	-999.99	-999.99	3.	-999.99	1.
8	2M12003647+1702176	0	6	4847.05514	0.	4636.86484	5073.45383	0.	-0.28139	0.	-0.33792	-0.22744	0.	3.34843	0.
9	2M09323469+2658057	0	4	5045.75376	0.	4747.68482	5374.99965	0.	-0.55567	0.	0.	0.	0.	3.	3.49114
10	2M03234621+5150557	0	4	5035.82754	0.	3990.21917	1.	-0.28097	0.	0.	0.	0.	0.	3.	2.7516

Figure 3.8: An example of a FITS file generated when using **Find\_Params** from the STARPANDA code showing some of the values (given as separate columns, e.g. Success flag, Iteration flag,  $T_{eff}$  value,  $T_{eff}$  success flag,  $T_{eff}$  errors, etc) derived for each star (given as separate rows and identified by the STAR.ID).

Finally, a check is made to see if results histograms are required, using the boolean variable **histo\_plot**. These 2 plots (Figures 3.9 & 3.10) show, firstly, the number of successfully analysis cases, failures due to divergence and failures due to either  $T_{eff}$ -[Fe/H],  $\log g$ , [C/Fe], [N/Fe] or [O/Fe] values moving out-of-bounds; secondly, the number of iterations required in all cases (including divergence and out-of-bounds failures).

### 3.2.3 CNO Abundance Analysis

If CNO abundance analysis has been selected (**cno\_analysis**), then a check is immediately made to see if full **param\_analysis** has just been made, and if so, the CNO analysis will be skipped, with a note being made in the logfile to indicate this. This is to prevent accidental running of both analyses. Assuming that this is not the case, then the code updates the logfile with details of this analysis run, including the labels of the lines to be used in analysis and the filenames of the observed, model and stellar parameter data files (**cm\_label**, **nm\_label**, **om\_label**, **obs\_datafile**, **model\_datafile** and **param\_datafile** respectively). Data structures are then created to hold the observed, model and output data. The stellar parameters are copied from the **param\_datafile** directly to the output structure; thus they are available to be used for further analysis together with the CNO abundances derived during this analysis. The observational

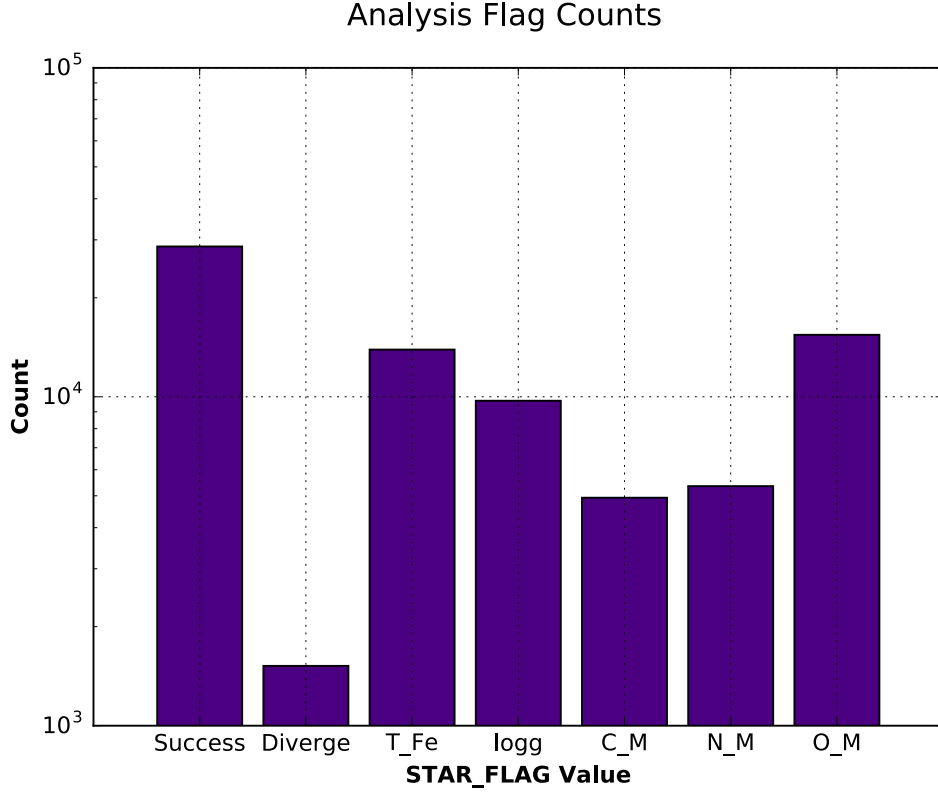


Figure 3.9: An example of an Analysis Flag Count histogram generated when using **Find\_Params** from the STARPANDA code. This shows the number of successful code completions, the number of times analysis failed due to divergence and the number of times that analysis failed due to one of the parameters being sought moving out of bound (represented by the bars labelled with those parameters).

and model data are read in exactly as described previously, with the addition this time of stellar parameters being read in from a separate file, **param\_datafile**, where the string variables **io\_temp\_label**, **io\_feh\_label** and **io\_logg\_label** specify the column names containing the parameter values.

Once the data have been read in, the SciPy method **RegularGridInterpolator** is again used to construct a 6 dimensional function describing the model data, though this time each node point in parameter space is only labelled with the 3 EQW values for the CNO abundance tracers. The code then loops over all target objects in the observed data file and calls the **Find\_CNO** method in *sp\_params.py*. The **Find\_CNO** method is identical to the **Find\_Params** method, with the exception of the removal of code de-

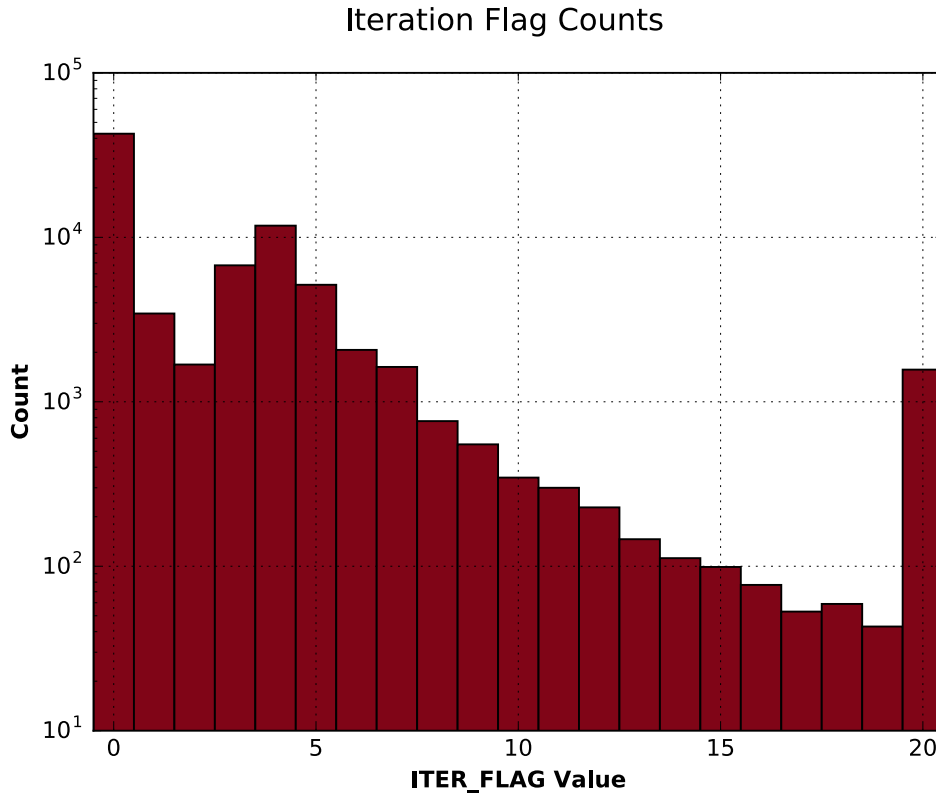


Figure 3.10: An example of an Iteration Flag Counts histogram generated when using **Find\_Params** from the STARPANDA code. This shows the number of iterations completed by the code during each analysis, regardless of whether analysis was successful or failed, either due to lack of convergence or by a parameter moving out of bounds.

signed to derive the parameter and error values for the stellar parameters. The overall structure of the **Find\_CNO** algorithm can be seen in Figure 3.11.

The code iterates, if required over the CNO abundance, in the order specified in **cno\_order** and evaluates the interpolated model data to derive CNO abundances. Then if those values are in-bounds, error values are derived for both positive and negative EQW error cases. Finally the details are written to the logfile, before control passes back to *starpac.py* to either start analysis of a new target object or write out the results. The logfile is again updated to indicate the analysis was completed, how long it took and how many target objects were successfully analysed and how many failed. The FITS output table is similar to that generated at the end of **Find\_Params**, but this time has slightly fewer columns, with the last 5 only being repeated for the 3 CNO abundances.

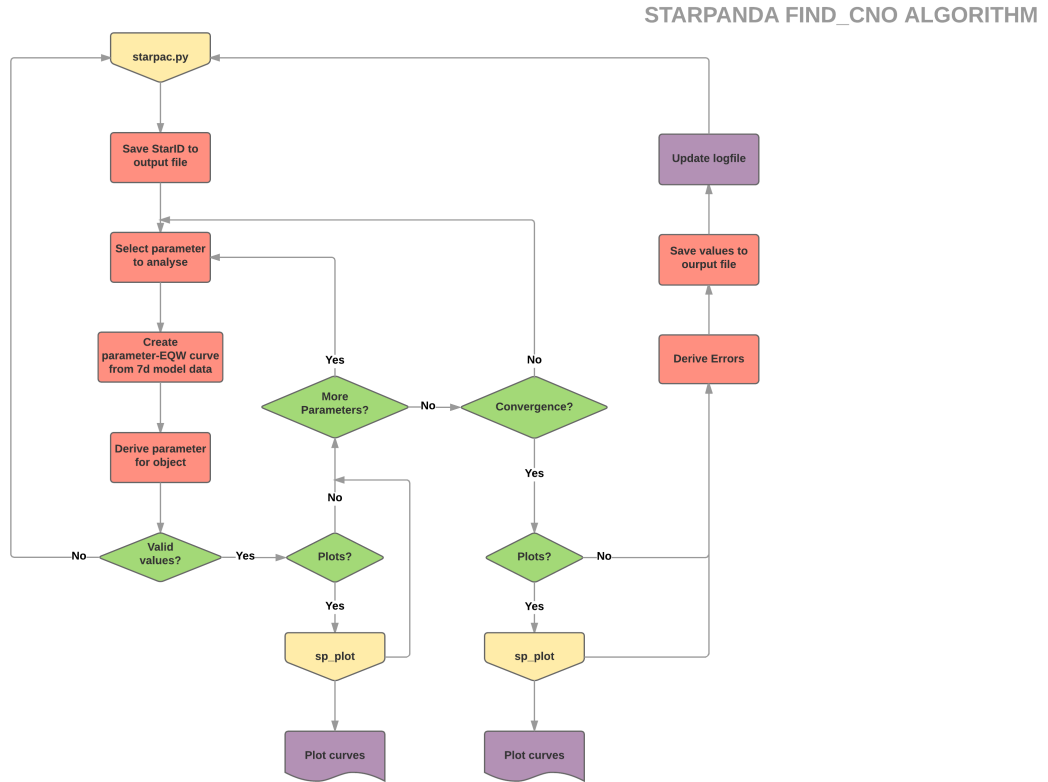


Figure 3.11: Flowchart of the STARPANDA code, showing how the Find\_CNO method (part of *sp\_params.py*) works. Objects coloured cream are calls from/to other code blocks, red represents internal code processes, green shows decision points and purple relates to output files.

An example of this table can be seen in figure 3.12.

- STAR\_ID - The ID of the target object
- STAR\_FLAG - The flag indicating the analysis result
- ITER\_FLAG - The flag indicating the number of iterations required
- TEMP -  $T_{eff}$  copied from the input stellar parameter file
- Fe\_H - [Fe/H] copied from the input stellar parameter file
- log (g) - log  $g$  copied from the input stellar parameter file
- [CNO] - The final derived CNO abundance value or a null value

- [CNO].FLAG - The flag indicating the trustworthiness of the CNO abundance value
- [CNO].ERR\_NEG - The CNO abundance error value with the EQW error subtracted
- [CNO].ERR\_POS - The CNO abundance error value with the EQW error added
- [CNO].ERR\_FLAG - The flag indicating the trustworthiness of the CNO abundance error values

STAR_ID	T <sub>eff</sub>	Fe/H	log g	O/M	O_M_FL	O_M_ERR_N	O_M_ERR_P	O_M_ERR_F	C/M	C_M_FL	C_M_ERR_N	C_M_ERR_POS	C_M_ERR_FL	C_M_ERR_N_M
1 2M19121933+3848348	5021.28	-0.021604	3.16446	0.061	0	0.03538	0.08659	0.	-0.27823	0	-0.35705	-0.20268	0.	0.
2 2M03201522+6139014	4954.31	0.283272	3.67736	0.14381	0	0.12194	0.16521	0.	0.14365	0	0.07843	0.20687	0.	-0.
3 2M00370524+5005023	4929.49	-0.210687	2.56123	0.10364	0	0.06224	0.14497	0.	-0.44754	0	-0.5554	-0.34799	0.	0.
4 2M08205438+2737485	4996.76	-0.30356	3.01402	0.09973	0	0.06249	0.1368	0.	-0.327	0	-0.42509	-0.23238	0.	-0.1
5 2M18031259+0150582	4548.78	0.019783	2.29035	0.13387	0	0.10902	0.15719	0.	0.08087	0	0.00313	0.15736	0.	0.1
6 2M04315025+3704126	4883.75	-0.174687	2.94536	0.0465	0	0.01225	0.08094	0.	-0.30389	0	-0.41184	-0.20133	0.	0.
7 2M17323218+2349164	4705.85	0.330903	2.55461	0.1964	0	0.15654	0.23628	0.	0.34188	0	0.26043	0.42114	0.	-0.
8 2M12003647+1702176	4719.1	-0.250001	2.88537	0.37551	0	0.35195	0.39856	0.	0.2192	0	0.13674	0.29378	0.	0.
9 2M09323469+2658057	4757.57	-0.591305	2.66821	0.39284	0	0.35618	0.42951	0.	0.15406	0	0.05725	0.25072	0.	-0.
10 2M03234621+5150557	4980.82	-0.103856	2.92894	0.11207	0	0.08727	0.13643	0.	-0.18467	0	-0.25744	-0.10817	0.	0.

Figure 3.12: An example of a FITS file generated when using **Find.CNO** from the STARPANDA code showing some of the values (given as separate columns, e.g. Success flag, Iteration flag, imported  $T_{eff}$  value, imported [Fe/H] value, imported  $\log g$  value, [O/M] value, [O/M] success flag, [O/M] errors, etc) derived for each star (given as separate rows and identified by the STAR\_ID).

Finally, **histo\_plot** is checked to see if results histograms are required. The results histogram will be essentially the same as for the **Find.Params** method (see Figures 3.9 & 3.10), but there will be no entries in the columns indicating that the  $T_{eff}$ -[Fe/H] or  $\log g$  values moved out-of-bounds since these are not analysed here.

### 3.2.4 Elemental Abundance Analysis

If elemental abundance analysis has been selected via **abund.analysis**, then the logfile is updated to indicate this and includes a list of the elements to be analysed (**model\_abund\_labels**), the observed data file (**obs\_datafile**), the abundances model data file (**abunds\_datafile**), and the stellar parameter & CNO abundances data file, which if parameter analysis

has been run will be *spdata\_output\_name\_params.fits*, and if not, will be **param-cno\_datafile**. The structures for the observed, model and output data are created and the observed data are read in exactly the same manner as for other analyses. The handling of the values for stellar parameters and CNO abundances of the target objects will depend on whether parameter analysis has been carried out already this run and if so, the values are used directly from the output structure used then; if parameter analysis has not been carried out, then this information will be read in from an input file and copied to the abundance output structure, so that this information is preserved for future further analysis.

In order to minimise the workload done by the code, each elemental abundance is derived separately, for each target object. The code loops over the elements to be analysed and reads in the model data to be used in deriving the elemental abundance values and errors for each object. The data file containing information on elemental abundances is treated a little differently - now, together with the values for the stellar parameters and CNO abundances used in the model atmospheres, it also has values of abundances for the elements to be analysed and values of the EQW measurements for the lines being used to trace the elements. These last 2 sets of values are saved as columns of arrays, e.g. the column containing the EQW measurements will have an n-element array for each combination of stellar parameter and CNO abundance values, where n is the number of abundances for that element for which synthetic spectra were generated and EQWs measured.

Again, the model data are interpolated using RGI, but this time it is a 7-dimensional parameter space with each node labelled with a single value describing the EQW of the line(s) being used to trace the elemental abundance. Values outside of this parameter space are set to deliver NaN values when evaluated. The code then loops over all targets in the input data and calls the method **Find\_Abunds** in *sp\_abunds.py* to derive the abundances and errors for each target object. The overall structure of the **Find\_Abunds** algorithm can be seen in Figure 3.13.



## STARPANDA FIND\_ABUNDS ALGORITHM

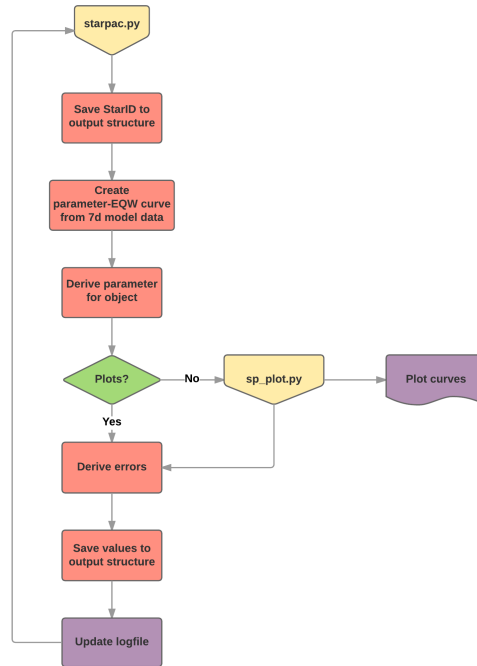


Figure 3.13: Flowchart of the STARPANDA code, showing how the Find\_Abunds method (part of *sp\_abunds.py*) works. Objects coloured cream are calls from/to other code blocks, red represents internal code processes, green shows decision points and purple relates to output files.

The **Find\_Abunds** method has been adapted from the algorithm used in determining the values and errors of  $\log g$ , [C/M], [N/M] and [O/M]. In order to future-proof the code as much as possible, the method is written in such a way that it can be used for any element that has lines present in the spectra, tracers reliable enough to be used in deriving the abundance values and model data available to do the derivation.

The first step is to evaluate the RGI function, using the stellar parameters and CNO abundance values supplied, to derive the EQW values of the tracer line at the model elemental abundance values. These are then interpolated, again using the method **interp1d**, with all values outside of the model parameter ranges set to return NaN values.

This is then evaluated at the measured EQW values to give the elemental abundance value. The result is then tested to see if it is valid (i.e. the result does not contain a NaN value) and the value is saved to the output structure together with the appropriate flag value. If the abundance value is in-bounds, then the error values can be derived next, using the same method as previously. Abundance errors are derived asymmetrically with the error on the measured EQW values being first added to, then subtracted from the value before re-evaluating the **interp1d** function; the error values are tested to see if they are in-bounds and then saved, with appropriate flags to the output structure.

Since we have generated model spectra with varying elemental abundances, stellar parameters and CNO abundances, then the derived elemental abundance is dependent solely on the amount of element present, and not on the values the stellar parameters or CNO abundances. Thus, no iteration is required at this stage and so the values derived above are taken to be the final values. Also, in order to prevent the logfile from becoming too large, no notes are made to the logfile during this analysis. Control then passed back to *starpac.py* to either continue on to the next target object, move to the next element to be analysed, or write out the results to file.

The logfile is then updated to show that elemental abundance analysis has been completed, the runtime duration and how many objects were analysed. Finally, the FITS output table is constructed (in the same manner as for previous options), this time with the stellar parameters and CNO abundance copied directly from the input file, and with the last 5 columns being repeated for each element included in **model\_abund\_labels**. An example of the output table can be seen in figure 3.14.

- STAR\_ID - The ID of the target object
- TEMP -  $T_{eff}$  copied from the input stellar parameter file
- Fe\_H - [Fe/H] copied from the input stellar parameter file

- $\log(g)$  -  $\log g$  copied from the input stellar parameter file
- $C_M$  -  $[C/M]$  copied from the input stellar parameter file
- $N_M$  -  $[N/M]$  copied from the input stellar parameter file
- $O_M$  -  $[O/M]$  copied from the input stellar parameter file
- $[ABUND]$  - The final derived elemental abundance value or a null value
- $[ABUND]_{FLAG}$  - The flag indicating the trustworthiness of the elemental abundance value
- $[ABUND]_{ERR\_NEG}$  - The elemental abundance error value with the EQW error subtracted
- $[ABUND]_{ERR\_POS}$  - The elemental abundance error value with the EQW error added
- $[ABUND]_{ERR\_FLAG}$  - The flag indicating the trustworthiness of the elemental abundance error values

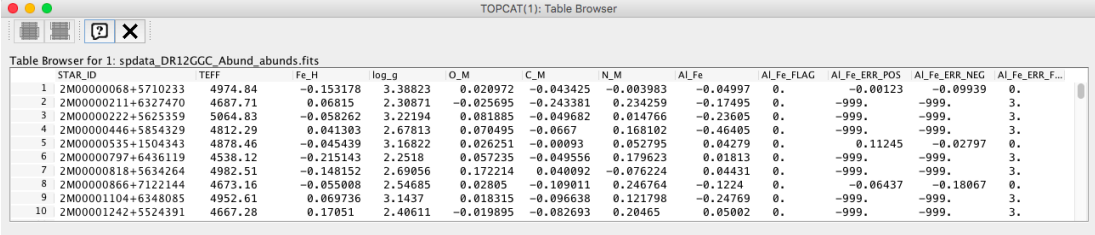


Table Browser for 1: spdata\_DR12GGC\_Abund\_abunds.fits

STAR_ID	Teff	Fe_H	log_g	O_M	C_M	N_M	Al_Fe	Al_Fe_FLAG	Al_Fe_ERR_POS	Al_Fe_ERR_NEG	Al_Fe_ERR_FLAG
1 2M00000068+5710233	4974.84	-0.153178	3.38823	0.020972	-0.043425	-0.003983	-0.04997	0.	-0.00123	-0.09939	0.
2 2M00000211+6327470	4687.71	0.06815	2.30871	-0.025695	-0.243381	0.234259	-0.17495	0.	-999.	-999.	3.
3 2M00000222+5625359	5064.83	-0.058262	3.22194	0.081885	-0.049682	0.014766	-0.23685	0.	-999.	-999.	3.
4 2M00000446+5854329	4812.29	0.041303	2.67813	0.070495	-0.0667	0.168102	-0.46405	0.	-999.	-999.	3.
5 2M00000535+1504343	4878.46	-0.045439	3.16822	0.026251	-0.00893	0.052795	0.04279	0.	0.11245	-0.02797	0.
6 2M00000797+6436119	4538.12	-0.215143	2.2518	0.057235	-0.049556	0.179623	0.01813	0.	-999.	-999.	3.
7 2M00000818+5634264	4982.51	-0.148152	2.69056	0.172214	0.040092	-0.076224	0.04431	0.	-999.	-999.	3.
8 2M00000866+7122144	4673.16	-0.055008	2.54685	0.02805	-0.109011	0.246764	-0.1224	0.	-0.06437	-0.18067	0.
9 2M00001104+6348085	4952.61	0.069736	3.1437	0.018315	-0.096638	0.121798	-0.24769	0.	-999.	-999.	3.
10 2M00001242+5524391	4667.28	0.17051	2.40611	-0.019895	-0.082693	0.20465	0.05002	0.	-999.	-999.	3.

Figure 3.14: An example of a FITS file generated when using **Find\_Abunds** from the STARPANDA code showing some of the values (given as separate columns, e.g. Success flag, Iteration flag, imported  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[O/M]$ ,  $[C/M]$ , &  $[N/M]$  values,  $[Al/Fe]$  value,  $[Al/Fe]$  success flag and  $[Al/Fe]$  errors) derived for each star (given as separate rows and identified by the STAR\_ID).

### 3.2.5 Final Tasks

Once all selected analysis options have been successfully completed, the code calculates the total runtime of those options and makes a note in the logfile to indicate that

analysis has finished, how long it took and how many spectra were analysed. The code then ends.

The contents of the logfile will depend on what analysis has been run, but figure 3.15 shows an example of a logfile generated during a run where stellar parameters and CNO abundances have been derived for a single target object. The analysis was successful after 7 iterations, though some error values could not be calculated due to them being out-of-bounds of model parameter ranges.

```

STARPANDA Logfile - 24/02/2017
=====
Input values for parameter & CNO abundance analysis:
=====
Temp Labels: ['J_K']
[Fe/H] Labels: ['FeI_15969', 'FeI_15399']
log(g) Labels: ['MgD', 'MgH']
[O/M] Labels: ['OH_16389']
[C/M] Labels: ['CII_16089']
[N/M] Labels: ['CN_15226', 'CN_15232', 'CN_15518']
Input Observed Data: 20161115 - ObsEWTest - TestElen - 1.fits
Input Model Data: 20161115 - ModelEW.fits
=====
Starting Analysis of Spectra
=====
Analysing Spectra: 1 of 1
Star ID: 2M03281522+6139014
Star EQWs: (0.579315368635519, 0.3247689986830887, 4.021813381101437, 0.33549446132544497, 0.17588368614658573, 0.18211366081892408)
Temp: (15800.0, 4919.766336140784, 4925.5311548422305, 4922.225303100312, 4925.117571153993, 4923.6724681925835, 4924.588482772128, 4924.271112993643) (5341.961038, 4555.596971)
[Fe/H]: [-2.5, 0.3599463349494333, 0.0663493528311834, 0.2488515589624439, 0.09201745477318794, 0.16933963814283848, 0.12123911218173336, 0.13885464716611376] (nan, nan)
log(g): (2.0, 3.7843177893156817, 3.3808880361493437, 4.112617181752848, 3.8112268337677495, 4.07748237537931, 4.010563757519183, 4.053956618110879) (4.991968, nan)
[C/M]: [0.0, 0.06892736748494633, 0.25425823804101585, 0.3714738084614848, 0.48558993814433797, 0.4358879848428975, 0.4589729288958755, 0.4578731427568616] (0.51228, 0.396041)
[N/M]: [0.0, -0.17024265663581115, 0.1483161136176718, -0.1744713174794853, 0.084948228959688882, -0.12501314532579336, -0.06961162515830037, -0.08947483184440769] (nan, nan)
[O/M]: [0.0, 0.06932786814766229, 0.30936514389392845, 0.29820841137106596, 0.3489399291828641, 0.343351524734985, 0.35894574219530756, 0.36180743682639804] (0.380289, 0.342348)
Successfully derived stellar parameters
Iteration Count: 7
Successfully completed stellar parameter & CNO abundance analysis
Analysed 1 objects in 3.003 seconds
Found Parameters for 1 objects
Unable to achieve convergence for 0 objects
Unable to find parameters for 0 objects
=====
Abundance Analysis not selected
=====
Completed successfully
Code Ran For 3.108 seconds
Analysed 1 spectra

```

Figure 3.15: An example of a logfile output from the STARPANDA code where only 1 target object is being analysed for stellar parameters & CNO abundances, with iteration. The first section shows the date; the second section shows the indices being used for deriving each parameter and the input filenames; the third section shows the analysis results for each object, including its name, the EQW values from observed spectra, the parameter values at each iteration (in square brackets), the errors on the final parameter values (in parentheses), a success/failure notice and the number iterations needed; the final section shows the results of the analysis overall, including the numbers of objects analysed, details of how many were successful/failed, and how long analysis took and the code ran for.

### 3.2.6 Remarks

As mentioned at several points, the code checks if **debug\_plots** is set and if so produces a diagnostic plot. If this is set and if iteration is also specified, then it is possible for the code to generate a very large number of plots. At the minimum, assuming no pa-

parameter value is out-of-bounds, it would generate 5 plots per target object if there is no iteration. So, for a data set like APOGEE with  $\sim 10^6$  objects and assuming an average of 5 iterations to reach convergence (this will be shown later to be a good assumption), the code would generate a not insignificant number of plots! In addition, it takes non-zero time to generate each plot and so the code will be significantly slowed by having this option selected when running on anything other than a small set of targets. This option should be used with care.

Starting values for  $\log g$  and CNO abundances should be chosen such that they are within model parameter ranges and not either out-of-bounds (which will cause a code error) or right at the edges of the model parameter space. The value specified for the maximum number of iterations to allow in the search for convergence should be set at a reasonable limit. As mentioned in the previous paragraph, it was found during testing that the mode of an iteration histogram was 5, though convergence was still seen, though infrequently, at iteration numbers greater than 10.

The analysis order would be unimportant if all tracers used in deriving parameter values were independent of everything other than the parameter they are tracing. However this is not the case for all of indices that have been adopted during this work and care should be taken to ensure that parameters are analysed in an appropriate order, e.g. if C indices are being used to trace [C/M] and CN indices for [N/M], then it is obvious that C should precede N in the analysis order.

### 3.3 Development of STARPANDA

#### Interpolating Model Data

At its core, STARPANDA operates by first interpolating the model data and generating a function to describe it, and then evaluating this for the target object at the measured EQW for a specific line (or an average of a group of lines). When running analysis for stellar parameters and/or CNO abundances, the model data will be 6-dimensional in nature and each node is labelled with the EQW values of the lines chosen as indicators for each of the 6/3 parameters; for elemental abundance analysis, the model data now becomes 7-dimensional, but with only a single value labelling each node. While the code is designed such that the interpolation of the model data need only happen once, the evaluation will be repeated once per parameter, per iteration.

Therefore, the choice of interpolation methods is of vital importance to the successful running of the code. The methods used must be reliable, accurate and quick - this last part is doubly important for the evaluation of the function which will happen over and over again during analysis. During the development of the code, several methods were tried and discarded in the search for a suitable interpolation scheme. Described below are some of these other methods and reasons for their eventual rejection.

Initially, the algorithm was designed more along the lines of EZ\_AGES (see Section 2.4) - it would take a 2-dimensional grid of  $T_{eff}$  and  $[Fe/H]$ , and check each box to find the one containing the observed point (defined by the EQW measurements of the  $T_{eff}$ - and  $[Fe/H]$ -sensitive lines). From this it would then interpolate the position of the point within the box and therefore derive  $T_{eff}$  and  $[Fe/H]$ . Errors could then be obtained by adding and subtracting the EQW errors from the central one, searching the grid for the box which contains the point and interpolating new values for  $T_{eff}$  and  $[Fe/H]$ . This would be repeated for a grid of  $T_{eff}$  and  $\log g$ , and so on for other parameters. However, it quickly became apparent that this method suffered from a number

of problems. Firstly, identifying if a point is inside a box, while apparently a simple task, is not so straightforward computationally, especially given some of the irregular shaped boxes that are found within the model grids used in this work. EZ\_AGES works using the equation of a straight line to draw boundary lines between nodes to form the box surrounding the point, then derives the horizontal and vertical transects and if the point had a value between the vertical and horizontal transect values, then it was inside the box. However, this does not generalise for an arbitrary 4-sided polygon and there are many examples where a point can be inside a box, but fail this check. After a considerable period of trial and error, a possible solution was found, whereby you can test the membership of a point in a box, by considering the number of times a vertical or horizontal line drawn from the point crosses a vertical and horizontal boundary line.

Take a line (vertical or horizontal) from a point at the centre, and in order to leave the square it would have to cross a boundary line once in either direction. If the point is outside the box, then it would cross either 0 or 2 boundary lines, depending on its offset in the other plane. Generalising this, if a point is inside an arbitrarily-shaped polygon, then it would have to cross boundary lines an odd number of times in both planes, otherwise it would be outside the polygon. However, this solution is not straight forward to implement and alternate solutions were still being sought. At this stage, we dealt with only a 2-dimensional parameter space (the other parameters had not been implemented yet as we had no tracers for these) and there were possible solutions utilising SciPy interpolation schemes. Several of these were investigated and tested, and **grid-data** was found to offer a faster and more easily implemented solution to this problem.

After lines suitable for deriving  $\log g$  were identified, the plan for using the same method for this parameter as for  $T_{eff}$  and  $[Fe/H]$  was revisited and it was thought that a better method for deriving a  $\log g$  value would be to use the, now 3-dimensional model data, to interpolate the variation in EQW values for the  $\log g$  line(s) as a function of the model  $\log g$  parameter value. Investigating interpolation schemes for n-dimensional data (to allow for future expansion of the model data set), several possible

schemes were identified, **Rbf**<sup>6</sup> (Radial Basis Functions), **RegularGridInterpolator**<sup>7</sup> [RGI], **interp**<sup>8</sup> and **griddata**<sup>9</sup>. **RegularGridInterpolator** was chosen as it appeared to offer the fastest method, both in terms of implementation and in terms of the runtime of the code; it also offered expandability, allowing us to use this for the 6 and 7 dimensions that we would end up with. In comparison, while the **Rbf** method could match the speed of **RGI**, this was only when using a linear interpolation scheme and higher order interpolation schemes were significantly slower on the hardware being used for development and testing of the code; **griddata**, even when considering linear interpolation, was too slow beyond the 2-dimensional case of the  $T_{eff}$ -[Fe/H] grids to consider. After deciding to use **RGI** in deriving  $\log g$ , it was realised that this method could also be used to generate a  $T_{eff}$ -[Fe/H] grid for further interpolation by **griddata**.

With the implementation of the new interpolation scheme, we are now able to evaluate the 6/7-dimensional model data, holding 5/6 parameters fixed and derive how the EQW values of the remaining parameter would vary as a function of the parameter value. This would give us a two dimensional array, that would need to be interpolated and then evaluated at the measured EQW of the target object. SciPy offers many options for univariate interpolation; **interp1d**<sup>10</sup> appeared to offer the simplest solution and in testing worked well.

---

<sup>6</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.Rbf.html#scipy.interpolate.Rbf>

<sup>7</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.RegularGridInterpolator.html>

<sup>8</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.interpn.html#scipy.interpolate.interpn>

<sup>9</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.griddata.html#scipy.interpolate.griddata>

<sup>10</sup><https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.interpolate.interp1d.html#scipy.interpolate.interp1d>



### Diagnostic Tools

During development of the code, it was necessary to test the output of the many functions and methods used to ensure that they are returning reliable values and working appropriately. Therefore, it was necessary to develop procedures to assist with understanding what was happening during each stage of analysis and iteration. It is also useful to have access to interim parameter values, but not at the expense of cluttering the FITS output files. Therefore, it seemed reasonable to have the code produce a ‘log-file’ containing additional information that could be checked by eye or by script (see Figure 3.15). This would provide a way of monitoring convergence or divergence of parameters, give details on the runtime of the various stages of the code, ensure that the read in and write out of data are being done correctly and provide a record of the details (e.g. the date, input files, etc) of the analysis run.

The logfile, by itself, is a useful addition to the FITS output data files, but it does not provide a way to ensure that the interpolation and evaluation of the model data are accurate. To this end a more visual set of tools would be required and plots of the model data together with the observed data seemed appropriate. To that end, plots would be needed for each parameter (or pair in the case of  $T_{eff}$  and  $[Fe/H]$ ) for each iteration and it would be useful if they could show the interpolated model data, the observed EQW of the line(s) being used for this parameter, and the nearest model data node points (see Figures 3.6 & 3.7 for examples); the first two points allow a visual check that the derived parameter value does appear to be correct, while the latter point allow a quick and dirty check that the interpolated model data are ‘reasonable’. A final addition to these plots can be made in cases where the analysis reaches successful convergence and this is the addition of the error values from the observed EQW data and the derived error values, once again allowing a visual check of these values.

Another way of checking if lines are a good tracer of a parameter is to compare the EQW values in the observed and model data in bulk, rather than individually (which

is effectively what STARPANDA does). To this end, histograms can be plotted with the values from both data sets and the results compared. To be useful as a tracer, the observed EQW values must occupy the range, or some subset thereof, of model EQW values for that index or average of several indices, assuming, of course, that the synthetic spectra have been generated using model parameters which cover the expected range of the stars being observed.

Finally, a useful addition to the above are the results histograms (see Figures 3.9 & 3.10), which allow us to see the overall success and failure cases of an entire analysis run. The ‘Analysis Flag Counts’ histogram can be useful in identifying if analysis is failing because of divergence due to a particular parameter, which could be the result of a problem with the indices being used to trace that parameter. The ‘Iteration Flag Counts’ shows how many iterations the code has run through before either succeeding or failing, or how many reach the iteration limit without either succeeding or failing.

The above diagnostic tools can be used to study the effects of varying the indices used to trace parameters, the effect of altering the initial parameter values for  $\log g$ , [C/M], [N/M] and [O/M], the effect of tightening/relaxing the convergence values, etc. It is worth mentioning again (see Section 3.2.6), that while these plots are useful, they should be switched on with caution, as if used during a large, iterating data analysis run, then the number of plots that can be produced will be extremely large, causing the code to slow down drastically and filling the *Output* directory with many files.

## 3.4 Testing STARPANDA

In this section tests highlighting the suitability and reliability of the algorithms used in the STARPANDA code are discussed. Firstly, tests of the interpolation schemes are shown - interpolation methods are central to the algorithms behind STARPANDA

and ensuring their suitability to this work is of vital importance. So, the interpolation methods are tested (**RegularGridInterpolation** and **interp1d**), in Sections 3.4.1 and 3.4.2 respectively.

Finally, the pipeline as a whole is tested using a set of mock observations (Section 3.4.3). Since one of the design goals of this work has been to develop a pipeline which is both fast and capable of running on comparatively low-powered computing hardware (when compared with other pipelines), it is necessary to run some general tests showing the overall working of the code and its speed. The results of these tests and details of the hardware on which the code has been run is included here also.

### 3.4.1 Testing the RGI method

The **RGI** method works by taking an  $n$ -dimensional set of parameter space coordinates together with a  $(n, m)$  array of values corresponding to the node points of that parameter space and constructing a function to describe how the values change as a function of the coordinates. It can work in nearest and linear interpolation methods only, but due to the restriction that the parameter space coordinates must form regular grid, it is much quicker than other multi-dimensional interpolation methods such as **griddata** or **Rbf**. The function can then be evaluated to give a set of values corresponding to a new set of parameter space coordinates.

In order to test the **RGI** method, we have removed nodes from the model data prior to interpolation and then fill these holes by evaluating the function at those positions. A comparison between the actual model values and the evaluated values at the missing nodes gives a good sense of how well the method is able to interpolate positions between nodes in the input data, though the errors associated with this method will be overestimated in this test.

To accomplish this, we started by removing one value from each of the 6 parameter ranges to create a large number of holes in the model grid. Table 3.1 shows the values of each of the 6 parameters which were removed from the model parameter space. The full model grid has 343,035 nodes covering the full range of all 6 parameters. With the values in table 3.1 removed, this new test grid has only 189,435 nodes, which is  $\sim 55\%$  of the full model grid. Removing this many nodes from the full grid should give a good test of the methods ability to accurately interpolate values between those that we input. It is important to note here that we cannot simply remove a single node from the grid, as the **RGI** method requires that there be a node point at each combination of specified parameters and any holes in this grid would cause the method to fail.

Parameter	Value
$T_{eff}$	4250.0 K
[Fe/H]	-0.5
$\log g$	2.5
[C/M]	0.0
[N/M]	0.0
[O/M]	0.0

Table 3.1: The values of each parameter removed from each dimension of the model grid in order to create holes in the grid to use for testing the **RGI** interpolation method.

Once the chosen values have been removed from the model grid, the reduced test grid can be interpolated and the resultant function can be evaluated at the positions of the removed nodes. The values that **RGI** is able to interpolate can then be directly compared to those removed, which are the EQW values measured from the model spectra. To see how well the **RGI** method is able to deal with interpolating the model data, the residuals between the full and test model grids have been calculated and for each parameter, these have been plotted as histograms, together with the mean and standard deviations for each set of residuals. The following 6 figures show these histograms for each parameter, Figure 3.16 for the  $T_{eff}$  tracers, Figure 3.17 for the [Fe/H] tracers, Figure 3.18 for the  $\log g$  tracers, Figure 3.19 for the [C/M] tracers, Figure 3.20 for the [N/M] tracers and finally Figure 3.21 for the [O/M] tracers.

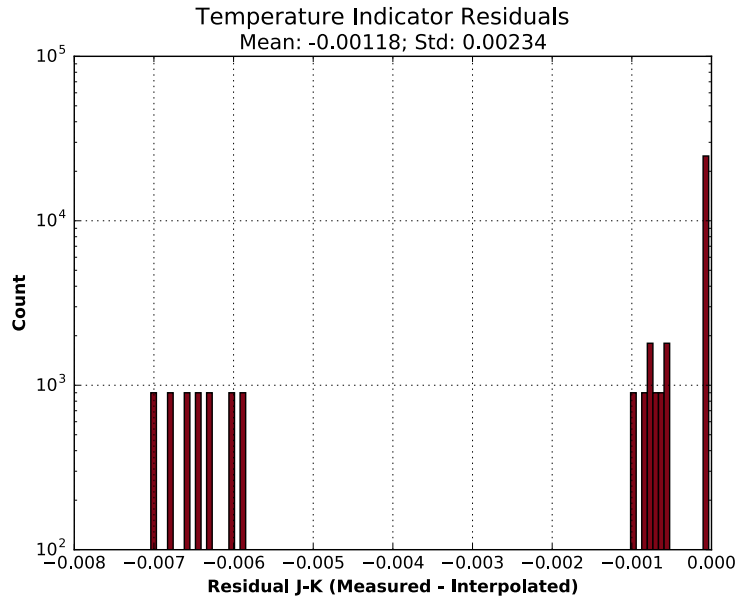


Figure 3.16: The residual J-K values for the of the  $T_{eff}$  tracers between those measured in the full model grid and those interpolated from the reduced test grid; values for the mean and standard deviation of this distribution are shown above the plot. The count shown on the y-axis is logged. The subtitle gives the mean and standard deviation.

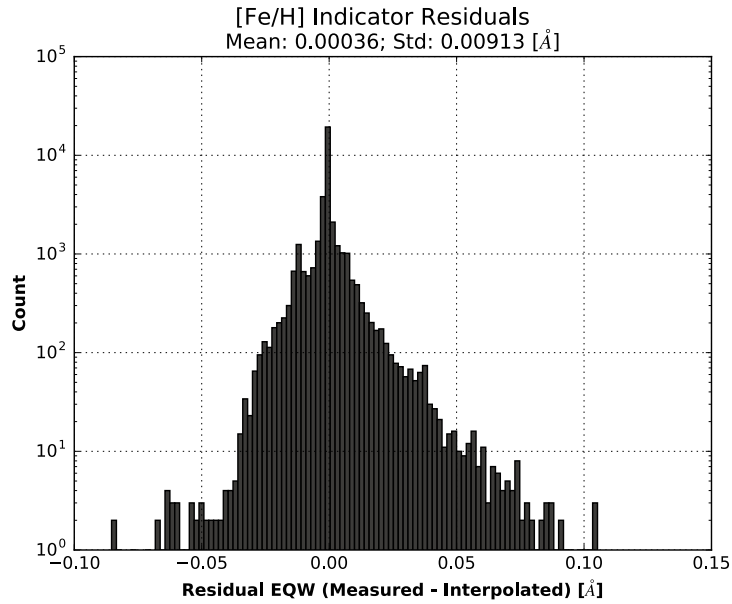


Figure 3.17: The residual values for the EQW (in Å) of the [Fe/H] tracers between those measured in the full model grid and those interpolated from the reduced test grid; values for the mean and standard deviation of this distribution are shown above the plot. The count shown on the y-axis is logged. The subtitle gives the mean and standard deviation.

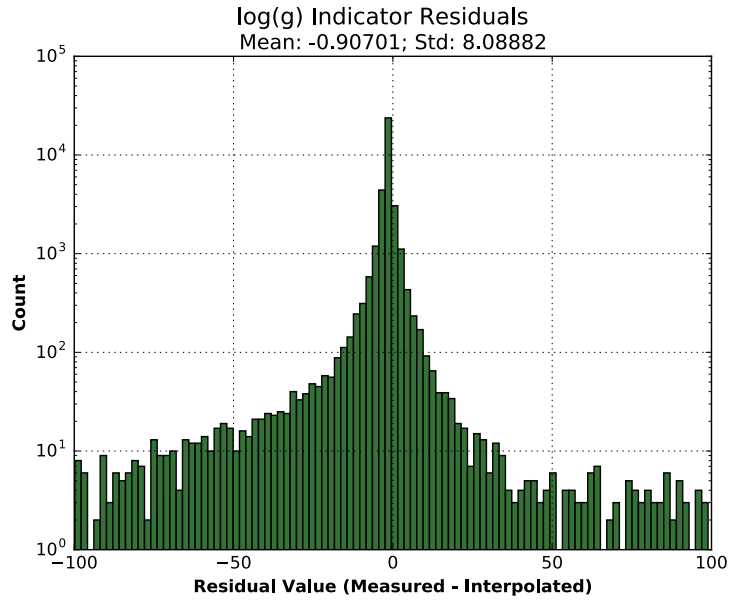


Figure 3.18: The residual values of the  $\log g$  tracers between those measured in the full model grid and those interpolated from the reduced test grid; values for the mean and standard deviation of this distribution are shown above the plot. Only residuals with values less than 100 have been plotted. The count shown on the y-axis is logged. The subtitle gives the mean and standard deviation.

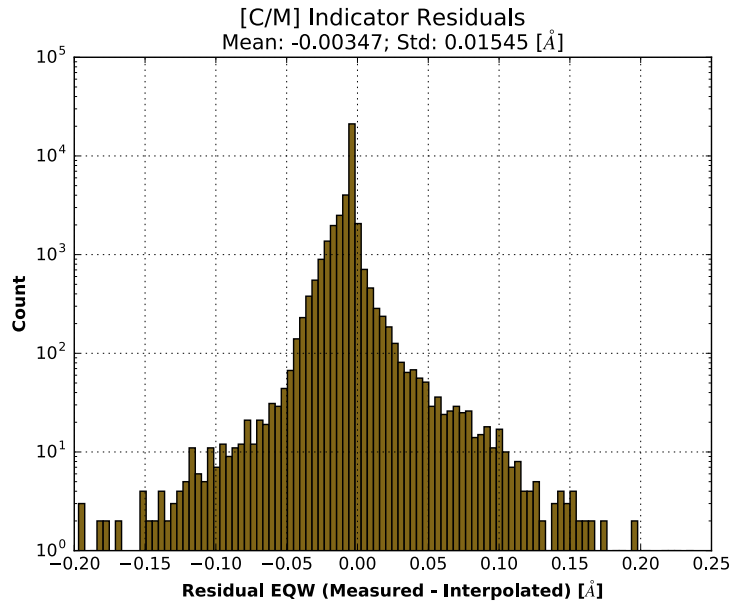


Figure 3.19: The residual values for the EQW (in Å) of the [C/M] tracers between those measured in the full model grid and those interpolated from the reduced test grid; values for the mean and standard deviation of this distribution are shown above the plot. The count shown on the y-axis is logged. The subtitle gives the mean and standard deviation.

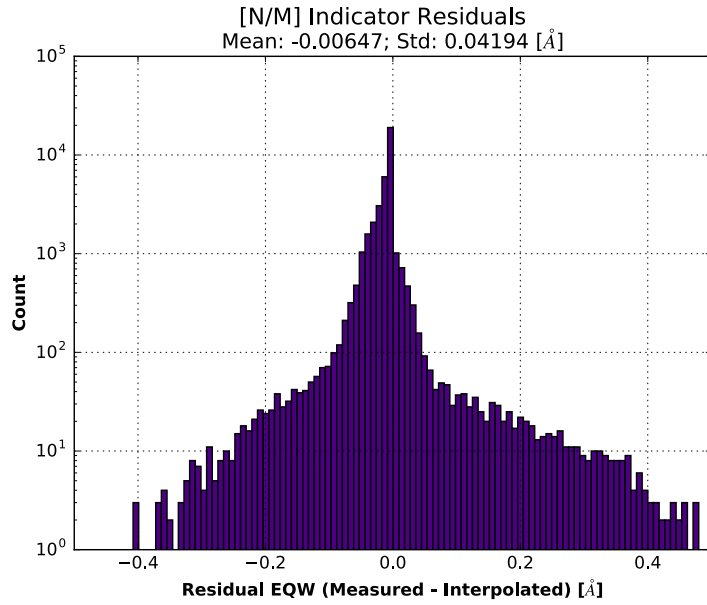


Figure 3.20: The residual values for the EQW (in Å) of the [N/M] tracers between those measured in the full model grid and those interpolated from the reduced test grid; values for the mean and standard deviation of this distribution are shown above the plot. The count shown on the y-axis is logged. The subtitle gives the mean and standard deviation.

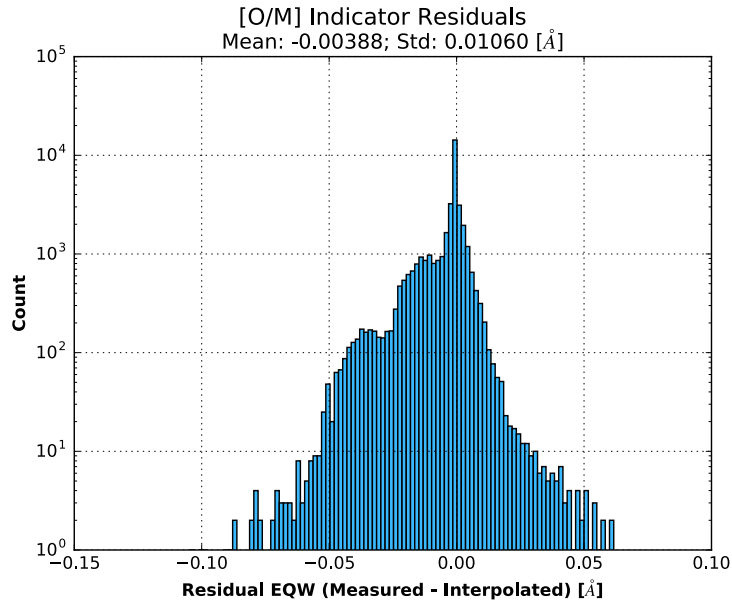


Figure 3.21: The residual values for the EQW (in Å) of the [O/M] tracers between those measured in the full model grid and those interpolated from the reduced test grid; values for the mean and standard deviation of this distribution are shown above the plot. The count shown on the y-axis is logged. The subtitle gives the mean and standard deviation.

As can be seen from all but Figure 3.18, the values recovered by the **RGI** method are in fairly good agreement with the measured values present in the full model grid. This can be taken as showing that even when using a coarse model grid, one with only  $\sim 55\%$  of the nodes of our actual model grid, the interpolation method is able to successfully recover values which show only small differences relative to those which can be measured directly in the missing spectra.

The values plotted in Figure 3.18 are only those for which the residual value is less than 100. If we plot the full range of values recovered, then the results are significantly affected by outliers. The results for  $\log g$ , appear at first sight to present a significant problem: Why are the values for the  $\log g$  tracers not showing the same accuracy as those for the other parameters? The answer for this appears to lie in the lines (see Tables 4.2 & 4.5 in Section 4.2) that we are using to derive  $\log g$  and if we look at the range of measured EQW values for these lines, which is shown in Figure 4.12 (in Section 4.3), we can see that the lines chosen have a large range of values. As we are using the average of two ratios of lines, if the line used in the denominator approaches zero, which it does, then the final value of the indicator can become quite large. This in turn gives rise to a large range of residual values and in addition it appears that such variations are proving more difficult for the **RGI** method to accurately interpolate.

### 3.4.2 Testing the interp1d method

The `interp1d` method is a simple 1-dimensional interpolation scheme which takes two arrays (e.g.  $x$  and  $y$ ) and then generates a function of the form,  $y = f(x)$  to describe them. It can perform several kinds of interpolation (linear, nearest, zero, linear spline, quadratic or cubic); perform extrapolation beyond the original array ranges or return specified values outside of these ranges; and be evaluated quickly to find new values of  $y$  given new values of  $x$ .



In order to test the **interp1d** method, we employ one of the diagnostic plots used earlier in this chapter - see Figure 3.7.

In this plot, the solid line is the interpolated variation of the EQW value of the lines being used to probe  $[O/M]$  and the value of the  $[O/M]$  parameter; the values of the 5 other parameters used to generate this curve can be seen in the sub-heading. The dashed line is then the same relationship, but for an actual node point in the 6-d parameter space, with the values of this node point shown in parenthesis. It can be clearly seen how close the 2 lines are for the majority of the range of  $[O/M]$ .

### 3.4.3 Testing the Pipeline

#### Mock Observation Tests

In order to get a feel for how well the code is able to derive stellar parameters and CNO abundances, we have taken two sets of synthetic spectra and run them through the pipeline before comparing the derived parameter values against the model parameter values used to generate the synthetic spectra. We use the same grid of model spectra to interpolate our the parameters for our set of “observed” spectra as we use throughout this work.

So, to start, we took a sub-sample of the model spectra used in the pipeline and added noise to them consistent with the signal-to-noise ratios typical of the APOGEE observations. These mock observations are then treated as observational measurements, running them through the code using a full analysis option. The results can then be compared to the parameters used to generate the model spectra to see how well the code is able to recover model parameters, around node-points in the model grid, given the noise which has been added.

Starting from spectra generated from model atmospheres with parameters shown in

Table 3.2, the above procedure gives 144 baseline spectra. To each of these spectra, noise has been added by multiplying the flux value of each pixel by a gaussian distribution with a  $\mu$  of 1.0 and  $\sigma$  of 0.01. This matches the cut made in the APOGEE data for signal-to-noise of 100 or higher. We repeated this 100 times for each spectrum, to produce a set of mock observed spectra with the same underlying stellar parameters, but slightly different EQWs for the lines we are using; this allows us to ensure that the pipeline returns parameter values in line with the high signal-to-noise ratios. In the end, there are 14400 spectra being analysed by STARPANDA; these mock observations were generated by Mackereth.

Parameter	Values
$T_{eff}$	4000, 4500 & 5000 K
[Fe/H]	-1.0, -0.5 & 0.0 dex
$\log g$	1.0 & 2.5 dex
[C/M]	-0.25 & 0.0 dex
[N/M]	0.0 & 0.5 dex
[O/M]	0.0 & 0.25 dex

Table 3.2: The model atmosphere parameters used as the starting point for the set of mock observations used to test the STARPANDA pipeline.

The results of the analysis of the mock observations can be seen in the figures below - Figures 3.22, 3.23, 3.24, 3.25, 3.26 and 3.27 show the difference between STARPANDA and the input model parameter as a function of the model parameter for, respectively, the Effective Temperature, [Fe/H],  $\log g$ , [C/M], [N/M] & [O/M].

In all six cases, the mean difference between the STARPANDA derived parameter and the input model atmosphere parameter is small compared with the input parameter, and the standard deviation is of the order expected given that the noise has been generated to mimic a signal-to-noise of  $\sim 100$ . Furthermore, in all cases but one, the spread of differences between STARPANDA and model input parameters is symmetrical about 0, which can be expected given the random gaussian nature of the noise generation. The only exception is for  $T_{eff}$ , where at 4000K, the mean residual is 60K, with a

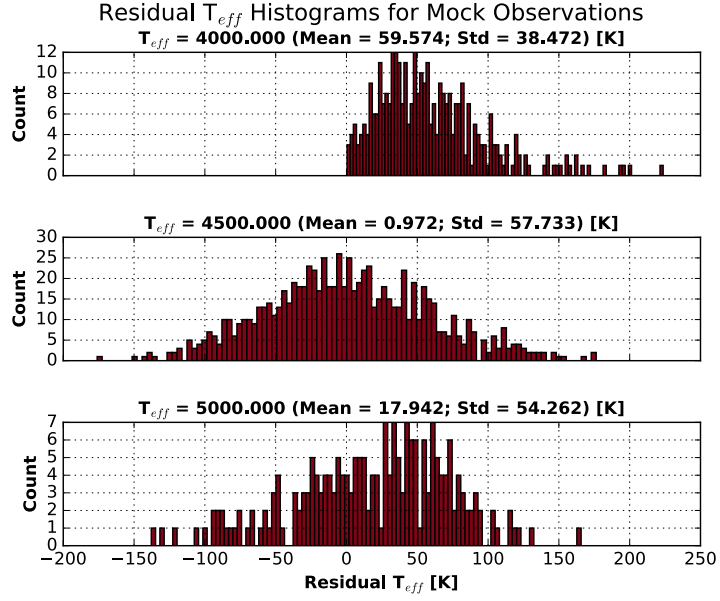


Figure 3.22: The Effective Temperature used to generate the model atmospheres and the results of the STARPANDA analysis using mock observations based on the same model atmospheres for 3 different temperatures. The x axis is shared by all 3 subplots. The parameter value, together with the mean and standard deviation are shown above each plot.

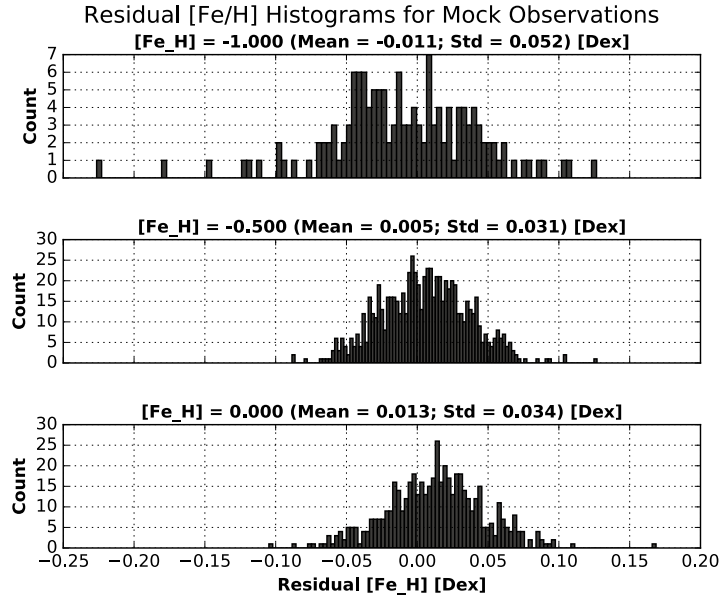


Figure 3.23: The  $[Fe/H]$  used to generate the model atmospheres and the results of the STARPANDA analysis using mock observations based on the same model atmospheres for 3 different  $[Fe/H]$  values. The x axis is shared by all 3 subplots. The parameter value, together with the mean and standard deviation are shown above each plot.

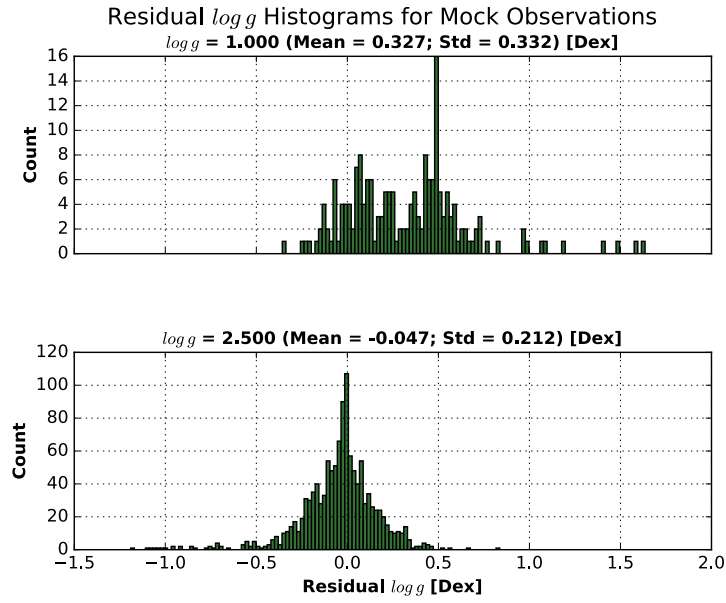


Figure 3.24: The  $\log g$  used to generate the model atmospheres and the results of the STARPANDA analysis using mock observations based on the same model atmospheres for 2 different  $\log g$  values. The x axis is shared by both subplots. The parameter value, together with the mean and standard deviation are shown above each plot.

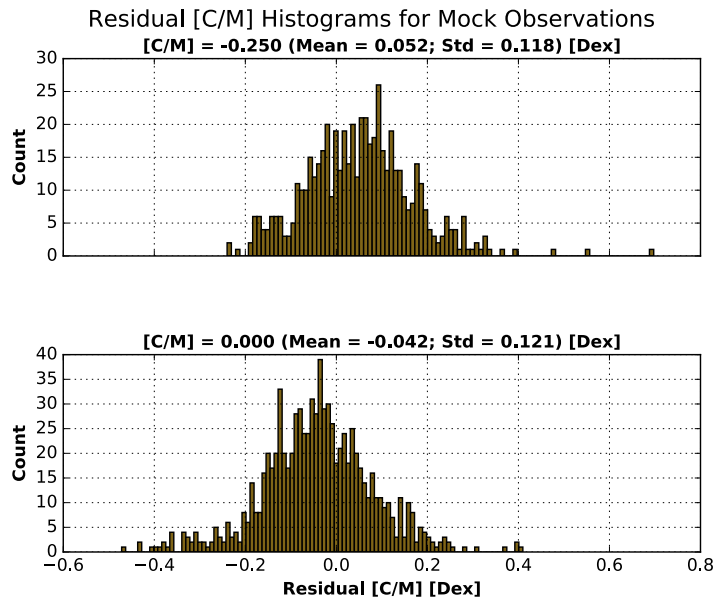


Figure 3.25: The [C/M] used to generate the model atmospheres and the results of the STARPANDA analysis using mock observations based on the same model atmospheres for 2 different [C/M] values. The x axis is shared by both subplots. The parameter value, together with the mean and standard deviation are shown above each plot.

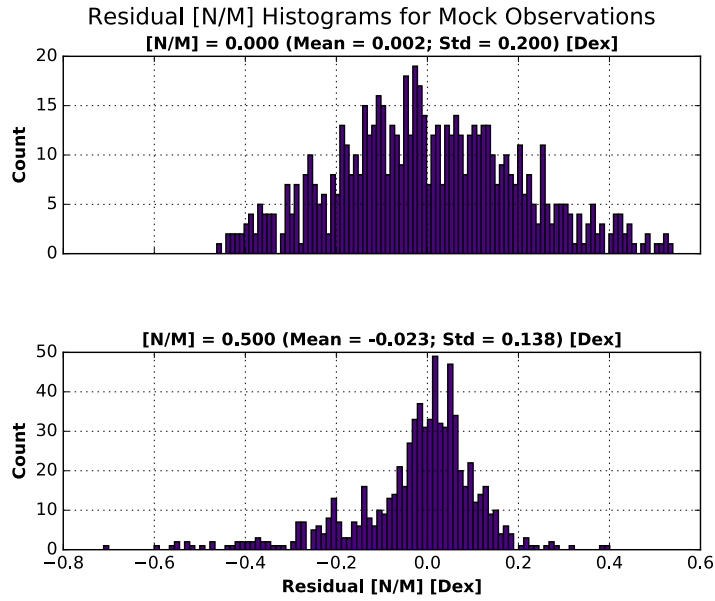


Figure 3.26: The  $[N/M]$  used to generate the model atmospheres and the results of the STARPANDA analysis using mock observations based on the same model atmospheres for 2 different  $[N/M]$  values. The x axis is shared by both subplots. The parameter value, together with the mean and standard deviation are shown above each plot.

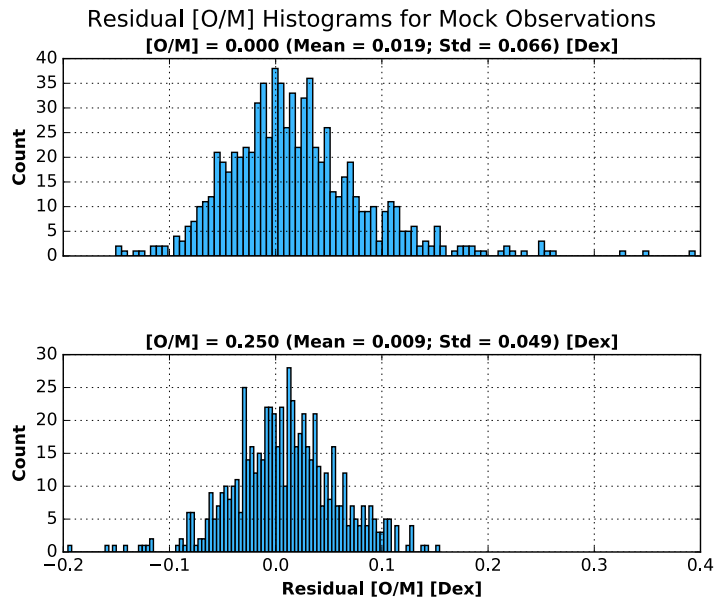


Figure 3.27: The  $[O/M]$  used to generate the model atmospheres and the results of the STARPANDA analysis using mock observations based on the same model atmospheres for 2 different  $[O/M]$  values. The x axis is shared by both subplots. The parameter value, together with the mean and standard deviation are shown above each plot.

standard deviation of 38K; this is something that will require further analysis to try to understand why this occurs in only this test.

So, from these tests, it can be seen that the pipeline is able to accurately recover the model atmosphere parameters without introducing any significant systematic errors and within expected accuracy given the level of noise added prior to analysis. While this test verifies the performance of STARPANDA at grid nodes, it is vital that it performs well more generally, i.e. at positions off the grid nodes.

In order to test the general performance of STARPANDA, a second set of synthetic spectra was generated using FERRE through interpolation within the APOGEE grid of model spectra (C. Allende Prieto 2017, priv. comm.), but at points away from the nodes in our model grid. The stellar parameters and CNO abundances used to create these new synthetic spectra are shown in Table 3.3, and comprise 216 objects.

Parameter	Values
$T_{eff}$	4100, 4700 & 5300 K
[Fe/H]	-1.3, -0.8 & -0.3 dex
$\log g$	1.2, 2.2 & 3.2 dex
[C/M]	-0.4 & 0.2 dex
[N/M]	0.2 & 0.7 dex
[O/M]	-0.2 & 0.4 dex

Table 3.3: The stellar parameter and CNO abundance values used to create 216 new synthetic spectra for testing the overall effectiveness of the STARPANDA pipeline.

This set of new synthetic spectra have been run through the STARPANDA pipeline twice. Firstly, we have used them as they are, so giving us 216 input objects, and the results of their analysis should tell us about the reliability of the interpolations scheme used in STARPANDA. Secondly, we have added noise to them (in the same manner as earlier), resulting in 21,600 simulated spectra, with  $S/N = 100$ .

So, when using these spectra without added noise, we would hope that the results derived by STARPANDA are very close to the parameter values used to create the synthetic spectra. Running a 6 parameter analysis of these 216 objects we find that STARPANDA is only able to successfully derive parameters in 15 cases and that analysis most frequently appears to fail due to problems converging on a  $\log g$  solution. While the point of failure is in derivation of  $\log g$ , the cause appears to be due to difficulties in deriving C and N abundances at higher  $T_{eff}$  (4700K & 5300K) together with lower metallicities and C or N abundances (e.g.  $[Fe/H] = -1.3$  &  $[C/M] 0.4$ ), as the indication lines we use become very weak in these regions. This in turn will effect the derivation of  $\log g$  during the next iteration of analysis. So, in an effort to increase the number of successful analysis cases, we chose to run a CNO abundance analysis option (using the model parameter values for the stellar parameters). Now, 143 objects successfully complete analysis ( $\sim 66\%$ ), with the failures mainly occurring when attempting to derive C at iteration counts greater than 1 (again these failures are much more common at high  $T_{eff}$  and low metallicity). We do the same with the noisy spectra and out of the 21,600 input objects, 12,902 successfully complete analysis ( $\sim 60\%$ ), and again the bulk of the failures are when attempting to derive C.

Since we now have CNO abundances for both the noiseless and noisy synthetic spectra, we can compare the results obtained for the different input sets. In order to compare the quality of the parameters obtained, we start by calculating the difference between the STARPANDA results and the input model parameter for each of CNO; this is a similar analysis to that performed in Chapter 4 (c.f. Section 4.4). We then calculate the mean difference and standard deviation of objects in each sub-set (we only do this for objects for which STARPANDA was able to derive parameters). Additionally we have done the same for the APOGEE observations which are shown in Section 4.4, where we compare the STARPANDA results to those obtained in ASPCAP DR12 (taking those objects in the range  $-0.05 \leq [x/M] \leq +0.05$ ). This comparison is then shown in Table 3.4.

Input Parameter	Set	Mean [Dex]	Std Dev [Dex]	Number of Objects
[C/M] = -0.4	SYN	0.00285	0.07966	61
	SYN+N	0.05473	0.13879	5484
	ASPCAP	0.06507	0.17509	203
[C/M] = 0.2	SYN	-0.09204	0.14753	82
	SYN+N	-0.04464	0.13930	7418
	ASPCAP	-0.01530	0.13112	1918
[N/M] = 0.2	SYN	0.07321	0.13694	62
	SYN+N	0.01511	0.23587	5882
	ASPCAP	0.08894	0.12619	17522
[N/M] = 0.7	SYN	-0.19071	0.17187	81
	SYN+N	-0.17033	0.14687	7020
	ASPCAP	-0.28593	0.22233	47
[O/M] = -0.2	SYN	-0.02991	0.03253	61
	SYN+N	-0.02224	0.06424	5678
	ASPCAP	0.25094	0.06995	22
[O/M] = 0.4	SYN	0.01472	0.02408	82
	SYN+N	0.01611	0.03793	7224
	ASPCAP	0.00998	0.08522	733

Table 3.4: Comparing the differences in results obtained by STARPANDA for 3 sets of objects - Noiseless synthetic spectra (SYN), Synthetic spectra with added noise (SYN+N) and APOGEE observations with ASPCAP DR12 results (ASPCAP). The first column shows the input parameter and its value being marginalised over, the Set shows which set the Mean and Std Dev (both in Dex) are being shown for and finally the last column shows the number of objects in each group.

We can see that in most cases the offset is quite small; the exception being when considering  $[N/M] = 0.7$ . The reason for this is unclear at this point. It is also worth noting the values obtained for  $[O/M] = -0.2$ , where STARPANDA achieves a much smaller offset than ASPCAP, which is in part due to problems encountered by the latter when working with low  $[O/Fe]$  values. Of greater interest are scatter results (given by the Std Dev values), which gives us an idea of the precision of the interpolation schemes used in different regions of parameter space. It is clear that while in some areas of parameter space, and for some parameters, the interpolation schemes are doing a good job at recovering values close to those used to create the input spectra, there are many cases where this is not true. This will be mainly due to the piecewise linear interpolation employed by the **RGI** interpolator, which will obviously return poorer results in areas of parameter space where the EQWs of indicators do not vary linearly with the



parameters they are tracing (such as towards the edges of the model grid). It would, therefore, be a significant improvement if a new scheme could be found, which uses higher-order interpolation, but can still deliver results quickly in multidimensional parameter space when using low-powered computer hardware - not a trivial challenge!

Finally, we need to understand why so few objects successfully completed analysis when the full 6 parameter option was chosen. Close inspection of the log file shows that the problem starts with  $\log g$  and that during iteration the values of this parameter move towards either the upper or lower boundaries of the parameter range, which in turn is taking us toward the edges of our parameter space. While we may still be just inside parameter space for  $\log g$ , when either C, N or O come to be derived we find that it is one of these parameters which have moved outside of parameter space first. This, again, shows the need for new tracers for  $\log g$ .

### Exploring Degeneracies

While it is known that there are degeneracies between parameters in stellar atmospheres, it is necessary to explore potential degeneracies added by the methods employed by STARPANDA. To that end, we can use the synthetic spectra that were described in the previous section. Now, we plot histograms for each of the 3 CNO parameters as well as plots comparing parameters (C vs N, C vs O and C vs N). The former can be seen in Figure 3.28, while the latter is shown in Figure 3.29.

So, starting with Figure 3.28, we can see that that for C, most of the results obtained show good agreement with the model C value, though there are some cases where STARPANDA is not able to achieve such good results. For N, the results are not quite as good and it appears that STARPANDA has a tendency to underestimate the N value. Finally, for O, we see very good agreement between the values of O obtained by STARPANDA and the model value. Looking at Figure 3.29, we can explore the

degeneracies between the parameters. Comparing O and C (top panel), we can see no significant degeneracy; this is also the case for O and N (middle panel). However, when we look at C and N (bottom panel), we do see a problem, especially at higher N values.

In order to understand why we get these results, we explored the model stellar parameters for those objects which showed residual values greater than 0.2. While O shows no objects with residuals this large, in all the other cases, the majority of objects with high residuals also have  $T_{eff}$  of 5300K. We can illustrate this by re-plotting the bottom panel from Figure 3.29, but adding the  $T_{eff}$  values from the model; objects with large residuals tend to be warmer, while those with small residuals tend to be cooler. This can be explained by our use of molecular lines, which become weak and disappear at higher temperatures; here we see how this affects the quality of the results that STARPANDA is able to achieve. In order to derive parameters for stars with higher effective temperatures we will need to exploit atomic lines.

### Speed Tests

One of the prime aims for developing this code has been to deliver a method which is able to very quickly analyse spectroscopic measurements and derive stellar parameters, CNO abundances and/or elemental abundances. To this end, one of the criteria used in evaluating how well the pipeline works is the runtime of each analysis option. Table 3.5 shows both the total runtime and the runtime per target object analysed, for the same input list of objects (a subset of the DR12 APOGEE data release), when analysed using each of the 3 different analysis options in STARPANDA. Even when deriving both stellar parameters and CNO abundances, the code was able to complete its analysis in less than a tenth of a second per target, while just CNO abundances can be derived in approximately five hundredths of a second per target and elemental abundances in less than a hundredth of a second per target. From this it can be seen that we have been more than successful in achieving this aim, even with the pipeline

being run on a relatively modest computer, without any serious attempt at optimisation.

Analysis Option	Runtime (s)	
	Total	Per Target
Stellar Parameters & CNO Abundances	6758.348	0.085
CNO Abundances	3891.926	0.049
Elemental Abundances	584.308	0.007

Table 3.5: The time taken to complete the 3 different STARPANDA analysis options (without plotting options enabled), shown first as the total running time and secondly as the running time per target object (assuming zero time associated with data read in and read out). These values are based on a subset of the DR12 APOGEE data release, containing 79519 stars.

As a comparison to the times recorded in Table 3.5, a run of the code with all plotting options switched on and analysing stellar parameters & CNO abundance values, completed to generate the plots showing in this work, took approximately 16 times as long as those with all plotting options switched off. The **debug\_plots** option is the biggest cause of this increase due to the large number of plots that that option generates.

The overhead time for the code, i.e. that time needed to read in the data files and read out the results files, is small compared to the time taken by the analysis options. Additionally, the time needed to analyse elemental abundances will scale as the number of abundances, so as more lines are identified which can be used to derive additional elemental abundances, the code will remain as efficient as it is for a single element i.e. the per abundance per target computational time required is the same regardless of the number of abundances or targets.

During development and testing, the code has mostly been run on a fairly modest computer, namely a MacBook Air. The technical specifications of this machine can be seen in Table 3.6. The code has also been run on other hardware and will operate as long as the Python dependencies are met - see the *sp\_readme.txt* file for details of these (Appendix A.2).

<b>Computer</b>	MacBook Air
<b>Model</b>	13-inch, Mid 2013
<b>Operating System</b>	10.11.6
<b>Processor</b>	1.3 GHz Intel Core i5 (Duel Core)
<b>Memory</b>	8 GB 1600 MHz DDR3

Table 3.6: The technical specifications of the primary computer used to develop and test STARPANDA.

We can compare the hardware and runtime of our pipeline against the ASPCAP pipeline (Allende Prieto & Holtzmann, priv. comm.). ASPCAP utilises a 27 node cluster, each of which has 16 cores (Intel Xeon CPU E5-2650 v2, 2.60GHz). The pipeline then derives stellar parameters and elemental abundances in  $\sim 100$  sec/star on a single core.

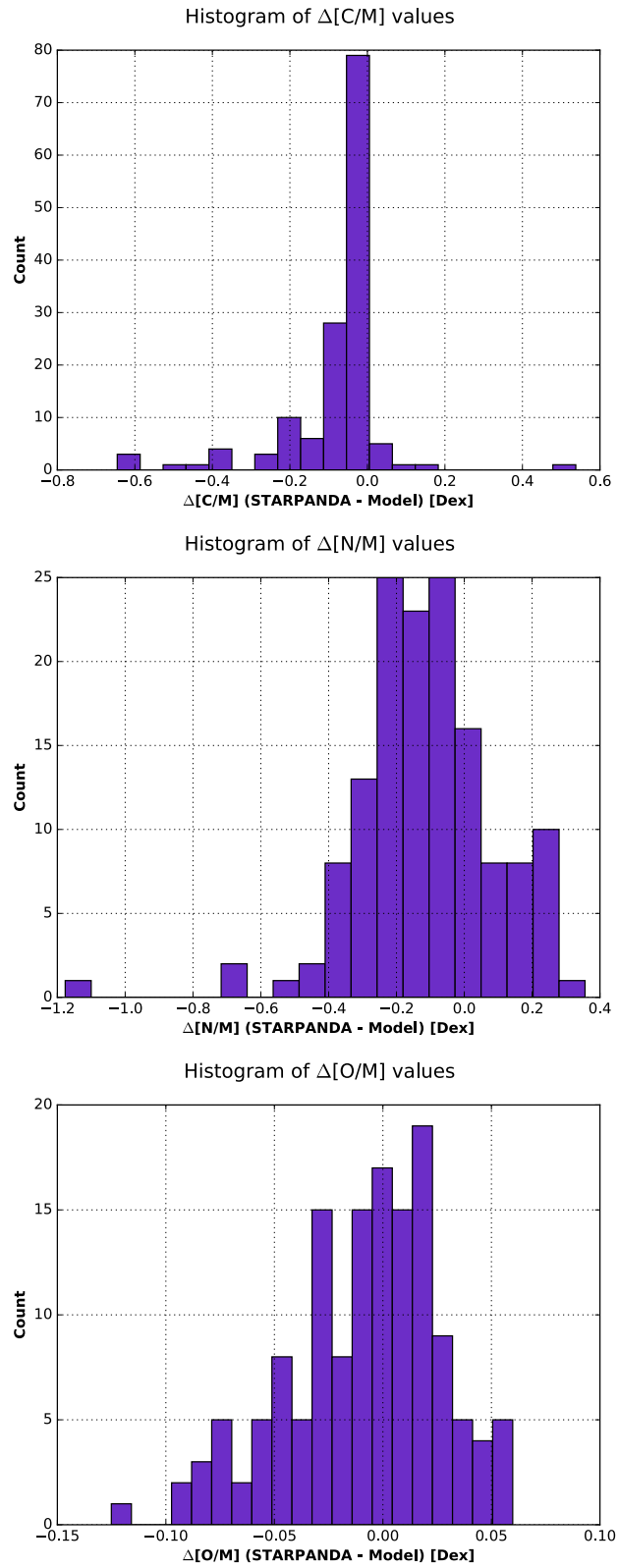


Figure 3.28: Histograms showing the residual values from a comparison between the results obtained by STARPANDA and the CNO values used to create the synthetic spectra being analysed. The top panel shows the histogram for C, the middle for N and the bottom for O. In all 3 cases, the x axis has the residual (STARPANDA - Model Input) in Dex and the y axis has the count.

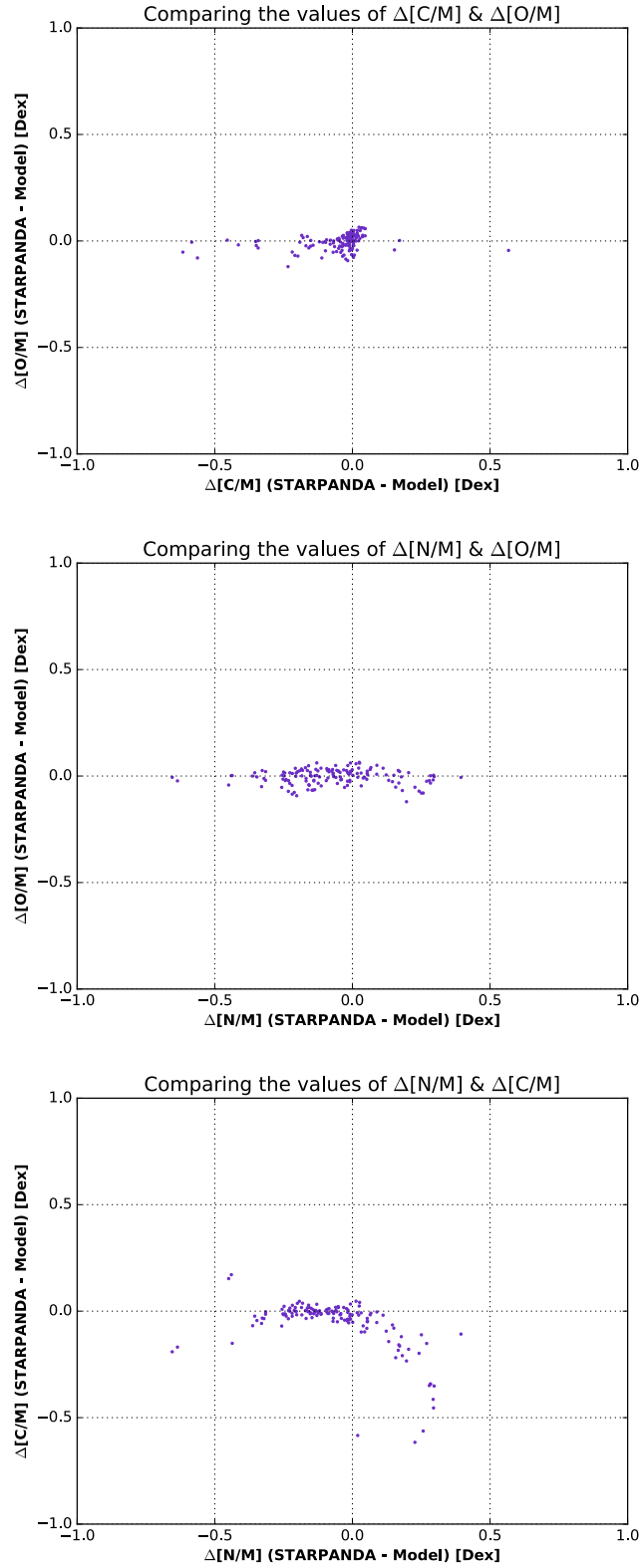


Figure 3.29: Plots showing the comparisons of residual values from a comparison between the results obtained by STARPANDA and the CNO values used to create the synthetic spectra being analysed. The top panel shows the plot for [O/M] vs [C/M]. The middle for [O/M] vs [N/M]. The bottom for [C/M] vs [N/M]. In all 3 cases, the x and y axes have the residual (STARPANDA - Model Input) in Dex.

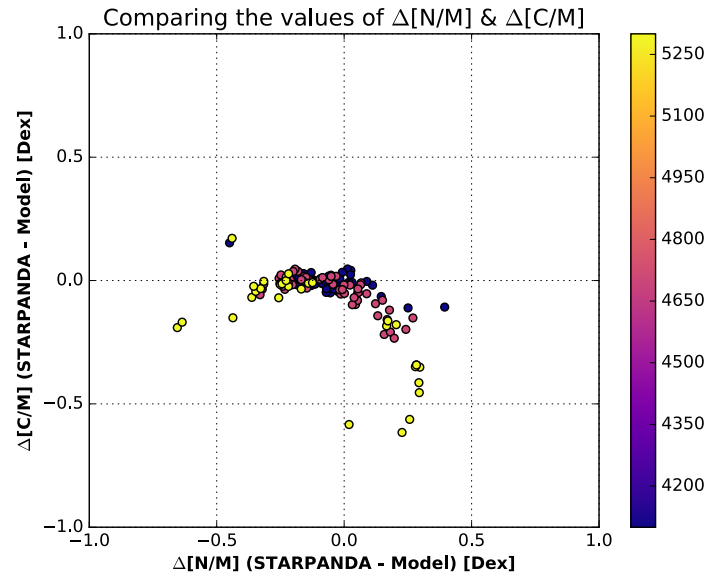


Figure 3.30: Plot showing the comparisons of residual values from a comparison between the results obtained by STARPANDA and the CNO values used to create the synthetic spectra being analysed. This is a repeat of the bottom panel of Figure 3.29, but with the  $T_{eff}$  of the . In all 3 cases, the x and y axes have the residual (STARPANDA - Model Input) in Dex.

# Chapter 4

## Results

In this chapter, the results from the first application of STARPANDA to the APOGEE data are presented. As a comparison, the APOGEE results derived from the ASPCAP pipeline are presented too.

The following sections describe the work that was undertaken in preparing the input data files (Section 4.1), the lines that have been found over the course of this work with commentary on their suitability as tracers of the relevant parameters (Section 4.2), results of a full stellar parameter and CNO abundance analysis (Section 4.3), results of CNO abundance analysis using stellar parameters derived by ASPCAP (Section 4.4) and finally the results of elemental abundance analysis for Al (Section 4.5) using stellar parameters and CNO abundances derived by ASPCAP. Separate analyses are shown in the following sections to assist in our understanding of where the pipeline is getting good parameter values and where it may be obtaining poor values. Therefore, running a six parameter analysis separately from a CNO abundance analysis shows how the stellar parameters obtained by STARPANDA affect the CNO abundances derived with those values. By using just ASPCAP parameters for the Al abundance analysis, we can directly compare the results obtained by both pipelines using different methods.



## 4.1 Data Preparation

The starting point for the input files used during this work is the model atmospheres used by ASPCAP; generated using ATLAS9 (Kurucz , 1993) and ASS $\epsilon$ T (Koesterke, 2009), see Chapter 2.5 for more information. The synthetic spectra were downloaded for a wide range of model atmosphere parameters (see the Initial Ranges in Table 4.1) together with the combined visit spectra for all targets observed by APOGEE. A custom code was then used to measure the EQW of all the indices that have been identified as being of interest in this work - see Section 4.2 for more details on the lines in both the synthetic and observed spectra; the code was developed as part of the Masters work undertaken by Mackereth, who subsequently made all the line measurements used in this work. Finally, the SDSS DR12 APOGEE allStar file<sup>1</sup> and the internal DR14 allStar file<sup>2</sup> were downloaded to be used for comparison to the results produced by STARPANDA and were also used in preparing the input files prior to running STARPANDA on a large dataset (the reasons for this are explained below). The allStar files contain 1 row for each target observed as part of the APOGEE surveys. The columns then list, for each target, a variety of identifying information, photometric data, astrometric data, APOGEE survey data, and the results of ASPCAP analysis.

Firstly, due to the use of the **RGI** method in the STARPANDA code, it was necessary to reduce the range of model parameters from those available (and used by ASPCAP) to a smaller subset. This is because RGI requires there to be no missing node points in model parameter space, so in cases where there is no model atmosphere for one combination of parameters, an entire row of one dimension of the model parameter space must be eliminated in order to satisfy the RGI method. The missing node points are due to the failure of the model atmospheres to reach convergence for a specified set of stellar parameters; this means that no synthetic spectra are available. These holes appear at the edges of our model parameter space, in areas where we do not expect to

<sup>1</sup><https://data.sdss.org/sas/dr12/apogee/spectro/redux/r5/allStar-v603.fits>

<sup>2</sup><http://data.sdss.org/sas/dr14/apogee/spectro/redux/r8/stars/l31c/l31c.2/allStar-l31c.2.fits>

find many stars, and necessitated cuts to the lowest and highest effective temperatures and surface gravities only. The synthetic spectra were generated and downloaded using the APOGEE code (Bovy, 2016). Secondly, more cuts to the input range of the model parameters were needed to deal with cases where the model inputs gave rise to non-unique solutions when interpolating parameters. This happens, again, at the edges of our model parameter space, and is only partially solved by the previous cuts necessitated by the use of **RGI** method. So, further cuts were made to the remaining parameters (no additional cuts were needed in effective temperature or surface gravity). The combined effect of these cuts can be seen in Table 4.1.

Finally, to ensure that we are only dealing with target stars which have no identified observational problems, those targets with poor signal-to-noise or with specific error flags set by the ASPCAP pipeline should be removed from the analysis inputs. To that end we removed stars for which the signal-to-noise ratio was less than 100, where the ASPCAP STARFLAGS identifier indicated that there were either significant numbers of bad pixels in the spectrum (BAD\_PIXELS) or that there was a very bright close neighbour to the target (VERY\_BRIGHT\_NEIGHBOR), and finally where the ASPCAP TARGFLAGS identifier indicated that there was significant telluric contamination (APOGEE\_TELLURIC) in the spectrum.

Thus, the above cuts were applied to the model input file and also to the ASPCAP results file, which was then cross-matched with our Observed data file to remove target stars which were removed by the ASPCAP cuts described above. This allowed us to focus only on stars for which we expected to be able to derive results and also stars for which our results could be compared with the ASPCAP values, as those targets for which ASPCAP failed to derive stellar parameters and CNO abundances are eliminated from the input files.

Following the cuts outlined above, and for the DR12 set of ASPCAP results, this reduced the 163,278 observed stars to 79,519 (49%). For the DR14 results, 277,371

Parameter	Initial Range	Final Range
$T_{eff}$ [K]	3500 - 6500	4000 - 6000
[Fe/H] [dex]	-2.5 - +0.5	-2.0 - +0.5
$\log g$ [dex]	0.0 - 5.0	0.5 - 4.5 dex
[C/M] [dex]	-1.0 - +1.0	-0.5 - +0.5
[N/M] [dex]	-1.0 - +1.0	-0.5 - +1.0
[O/M] [dex]	-1.0 - +1.0	-0.5 - +0.5

Table 4.1: The initial value ranges of the model atmospheres downloaded for use by STARPANDA and the final model parameter ranges after the outlined cuts have been applied.

observed stars were reduced to 105,823 (38%) initially, however this was further reduced to 63,795 objects; the additional stars were observed as part of the APOGEE-2 survey, for which we have not downloaded the spectra.

For the analysis of the elemental abundances, while ASPCAP uses a wide range of stellar parameters and CNO abundances in the synthetic spectra model grid, it does not use any spectra where individual elemental abundances are varied, instead varying overall metallicity and assuming a relationship between the abundance of a given element, the strength of the line(s) chosen and the overall metallicity of the star. Instead of following this path, Mackereth used Bovy’s APOGEE code to create model mini-grids, where synthetic spectra were calculated for varying individual abundances while all other parameters remain fixed. For Al, this has been done for values between -0.5 and +1.0, in 0.25 dex steps, and this is repeated for the range of model atmosphere parameters used in deriving stellar parameters and CNO abundances. In order to keep the mini-grids as small as possible, the step size has been altered for CNO values to be 0.5 dex steps between -1.0 and +1.0, while keeping the stellar parameter steps size the same.

## 4.2 Indices

Over the course of this work, many indices have been identified as being potential probes of parameters. These have come from the line list used by ASPCAP (Shetrone et al., 2015) and then checked by studying differential spectra (over a range of values of  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[C/M]$ ,  $[N/M]$  &  $[O/M]$ ) and looking for indices which only vary significantly as a function of one parameter and remain usable across a wide range of other parameters. This work was carried out by several students as part of their Masters projects. Presented in Tables 4.2, 4.3 and 4.4 are the indices which have been identified and tested by STARPANDA for use in deriving stellar parameters, CNO abundances and elemental abundances respectively. These are broken down, in each case, by the parameter being probed and then sorted alphanumerically, with the index shown as a combination of species and wavelength (given in Å). Comments on the suitability of each index to the parameter in question is shown in the final column of the table. In terms of the quality of a index, this is judged on the basis of how well the parameter values derived by STARPANDA match those obtained by ASPCAP.

In addition to the indices identified as described above, the de-reddened J-K colour has been used in this work as a probe of effective temperature and is therefore included in this table and shown in the following sections. For the observed stars the de-reddened J-K colour has been calculated using the values provided in the allStar files. The model J-K colour was calculated by Mackereth as part of his Masters work, and is based on the Colour- $[Fe/H]$  relationship used in González Hernández & Bonifacio (2009) which can be solved to give the J-K colour (see Equation 10 and Table 5 of that paper). This is due to our current lack of a good single index or combination of indices suitable for probing this parameter.

In many cases it is advantageous to combine two or more indices and use the average EQW values during the analysis. This has been done extensively during the development of STARPANDA and, currently, the best results obtained by the code utilise

Parameter	Index	Designation	Quality
$T_{eff}$	J-K	$J_K$	Ok
	FeI_16230 / FeI_16522	T1	Poor
[Fe/H]	(FeI_15969 + FeI_15399) / 2	Fe_IR	Uncertain
log $g$	MgI_15753 / MgI_15917	MgA	Poor
	MgI_15770 / MgI_15917	MgB	Poor
	MgI_15753 / MgI_15959	MgC	Ok
	MgI_15770 / MgI_15959	MgD	Good
	MgI_15753 / (MgI_15917 + MgI_15959)	MgE	Poor
	MgI_15770 / (MgI_15917 + MgI_15959)	MgF	Ok
	(MgI_15753 + MgI_15770) / MgI_15917	MgG	Poor
	(MgI_15753 + MgI_15770) / MgI_15959	MgH	Good
	(MgI_15753 + MgI_15770) / (MgI_15917 + MgI_15959)	MgJ	Poor

Table 4.2: The indices identified for use in STARPANDA to probe stellar parameters, broken down by parameter and sorted alphanumerically. Designations are given for cases where indices are combined as ratios, etc. Comments on the suitability of each index are given in the Quality column.

multiple indices for almost all parameters. The results shown in the following sections of this chapter use the indices or combination of indices shown in Table 4.5.

## 4.3 Analysis of Stellar Parameters & CNO Abundances

In this section, we show the results produced when running STARPANDA on the input files described in Section 4.1, using all the indices listed in Table 4.5 and the code set to analyse all 6 parameters. Firstly we derive the stellar parameters and CNO abundances using STARPANDA and then compare them to those derived by ASPCAP as reported in the DR12 data release; secondly we compare our results to those reported in the DR14 data release.

In addition, for this analysis option and all the others shown in this chapter, we have used the initial parameters and iteration tolerances shown in Table 4.6. We chose a starting value of 2.0 for log  $g$  and 0.0 for CNO abundances. No initial values are re-

[C/M]		[N/M]		[O/M]	
Index	Quality	Index	Quality	Index	Quality
CI_16009	Bad	CN_15226	Ok	CO_15582	Untested
CI_16026	Bad	CN_15232	Good	CO_15982	Untested
CI_16859	Poor	CN_15290	Bad	CO_15990	Untested
CI_16895	Poor	CN_15290	Bad	CO_15997	Untested
CO_15582	Ok	CN_15518	Ok	CO_16007	Untested
CO_15982	Ok	CN_16388	Bad	CO_16020	Untested
CO_15990	Poor	CN_16564	Bad	CO_16030	Untested
CO_15997	Ok	CN_16586	Poor	CO_16190	Untested
CO_16007	Bad			CO_16386	Untested
CO_16020	Very Bad			CO_16620	Untested
CO_16030	Bad			CO_16840	Untested
CO_16190	Poor			OH_15395	Poor
CO_16386	Very Bad			OH_15413	Poor
CO_16620	Ok			OH_15573	Poor
CO_16840	Poor			OH_15576	Poor
				OH_15724	Poor
				OH_15761	Poor
				OH_16057	Poor
				OH_16369	Good
				OH_16877	Poor
				OH_16877	Very Bad
				OH_16719	Very Bad

Table 4.3: The indices identified for use in STARPANDA to probe CNO abundances, broken down by parameter and sorted alphanumerically. Comments on the suitability of each index are given in the Quality column.

Parameter	Index	Quality
[Al/M]	Al_16723	Ok
	Al_16755	Unmeasured
	Al_16767	Good

Table 4.4: The indices identified for use in STARPANDA to probe elemental abundances, broken down by parameter and sorted alphanumerically. Comments on the suitability of each index are given in the Quality column.

quired for  $T_{eff}$  and  $[Fe/H]$  as they are derived first under the assumption of the other values. Likewise for  $[Al/M]$ , no initial value is required, just the values of the stellar parameter and CNO abundances supplied via input file. The iteration tolerances are set for the stellar parameters and CNO abundances and the iteration limit is set to 20 iterations. As no iteration is performed during elemental abundance analysis, there is

Parameter	Index/Indices Used
$T_{eff}$	J_K
[Fe/H]	Fe_IR
$\log g$	MgD, MgH
[C/M]	CO_15997, CO_15582, CO_15982, CO_16620
[N/M]	CN_15226, CN_15232, CN_15518
[O/M]	OH_16369
[Al/M]	Al_16767

Table 4.5: These are the best indices, or combination of indices, found during the development of STARPANDA. These have been used (by taking an average of their values) in producing the results shown elsewhere in this chapter.

no requirement to set a tolerance for [Al/M].

Parameter	Initial Parameter Value	Iteration Tolerance
$T_{eff}$	-	20 K
[Fe/H]	-	0.02 dex
$\log g$	2.0	0.1 dex
[C/M]	0.0	0.02 dex
[N/M]	0.0	0.02 dex
[O/M]	0.0	0.02 dex
[Al/M]	-	-

Table 4.6: The initial parameter & iteration tolerance values used in STARPANDA. The initial parameter values for  $\log g$  and CNO abundances are used to derive values during the first iteration;  $T_{eff}$ , [Fe/H] & [Al/M] do not require initial parameter values to be set. The iteration tolerances are used to determine when iteration should cease due to successful convergence of analysis; since no iteration is performed during elemental abundance analysis, then no iteration tolerances are needed for [Al/M].

### 4.3.1 DR12 Analysis

STARPANDA took 6758 seconds to analyse 79519 stars (therefore averaging  $\sim 0.08$  sec/star) and it achieved results for 28642 of these. As can be seen in Figure 4.1, the number of stars that failed due to lack of convergence was small (1520 stars), with the bulk of the failures being due to the star falling outside of model parameter space either on the first or subsequent iterations and most commonly while either deriving

the  $T_{eff}$  and  $[Fe/H]$  (since they are derived together) or  $[O/M]$ . An idea of the number of stars which fall outside of the model parameter space in the  $T_{eff}$ - $[Fe/H]$  plane on the first iteration (where initial values of  $\log g$ ,  $[C/M]$ ,  $[N/M]$  and  $[O/M]$  are taken from the configuration file (*sp\_config.py*) can be seen from Figures 4.2 and 4.3. Figure 4.2 shows the results of the iteration counter, where there are a large number of stars which end analysis on the first iteration - given the design of the code, where at the end of an iteration the derived values are compared to values from the previous iteration or initial values, it is not possible to successfully derive parameters in less than 1 iteration. Figure 4.3 confirms this, showing a large number of points outside of the initial  $T_{eff}$ - $[Fe/H]$  grid, which would fail analysis at the first attempt to derive  $T_{eff}$  and  $[Fe/H]$  parameter values. The other prominent features of this plot is the strong horizontal cutoff at a J-K value of 0.5. This can be explained by the selection criteria imposed by the APOGEE survey, where only stars having  $J-K > 0.5$  were selected for observation, except for those in clusters which were selected for other reasons.

For the  $\sim 28000$  stars for which STARPANDA has been able to derive results for all 6 parameters, we have compared the values obtained by our code to those obtained by the ASPCAP pipeline. This is done by cross-matching the STARPANDA output with the APOGEE DR12 allStar file and then subtracting the ASPCAP result from the STARPANDA result, while discarding any stars for which ASPCAP failed to derive parameters. Therefore, in the following figures we plot the difference between the STARPANDA result and the ASPCAP result (STARPANDA - ASPCAP) on the y-axis and the ASPCAP result on the x-axis for  $T_{eff}$  (Figure 4.4),  $[Fe/H]$  (Figure 4.5),  $\log g$  (Figure 4.6),  $[C/M]$  (Figure 4.7),  $[N/M]$  (Figure 4.8) and  $[O/M]$  (Figure 4.9) respectively.

What then do these plots tell us about the quality of the results produced by STARPANDA?

First, it is obvious that the results obtained for all 6 parameters are still significantly different from those obtained by ASPCAP, and even though STARPANDA is very fast, it counts for little if the results are not reliable. However, there is still cause for opti-



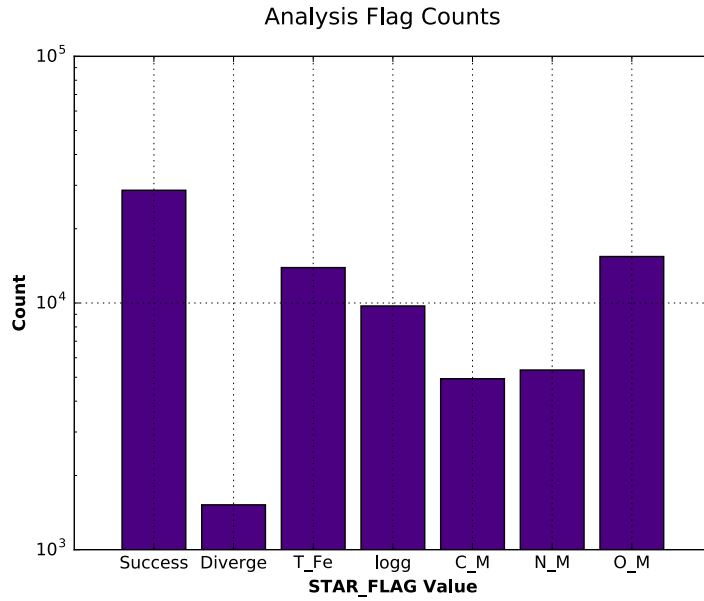


Figure 4.1: Analysis Flags for a 6 parameter run of STARPANDA. The x-axis shows, Successful Analysis, Failure due to Divergence, Failure at  $T_{eff}$ -[Fe/H] analysis, Failure at  $\log g$  analysis, Failure at [C/M] analysis, Failure at [N/M] analysis or Failure at [O/M] analysis. The y-axis is then the log of the number of occurrences.

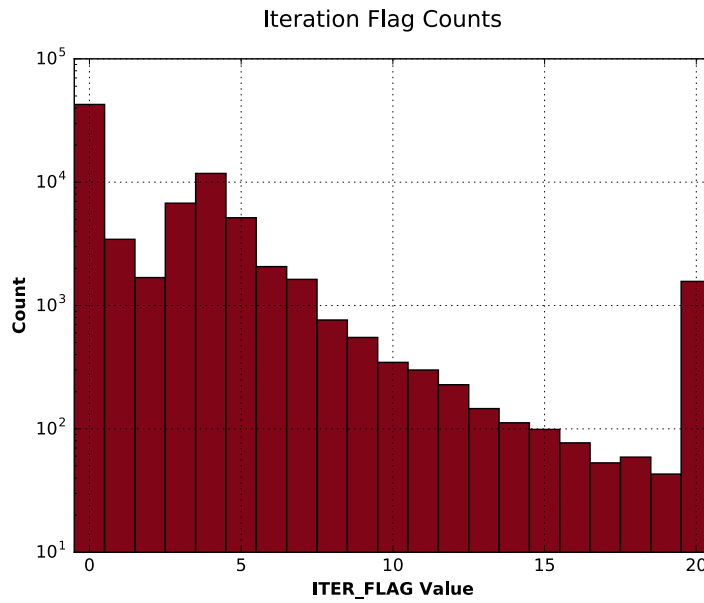


Figure 4.2: Iteration Flag results for a 6 parameter run of STARPANDA. This histogram shows the number of iterations completed during analysis, with the count on the y-axis being logged. There is a large peak at 0 iterations, showing a failure of analysis due to the star being outside of model parameter space and a second smaller, but still sizable, peak at 20 iterations, showing the number of stars for which convergence was not possible.

mism when considering the plots in more detail.

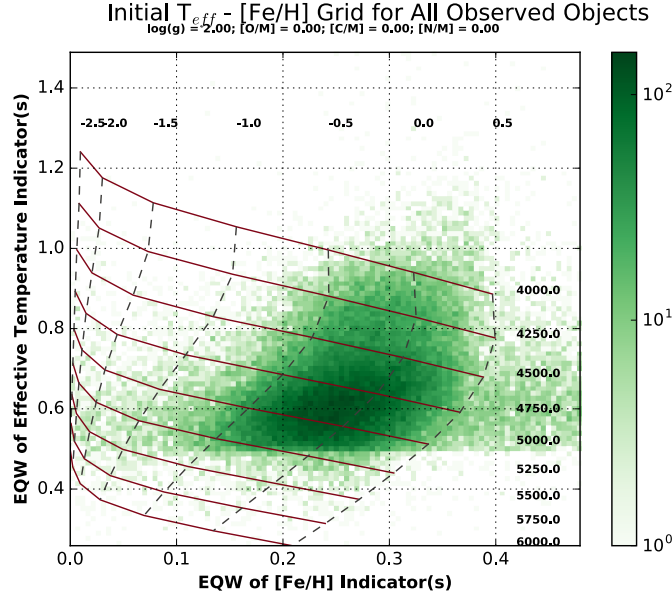


Figure 4.3: Initial  $T_{eff}$ -[Fe/H] grid (constructed using the initial  $\log g$ , [C/M], [N/M] & [O/M] values specified) highlighting the number of stars which do not fall within model parameter space under initial conditions. On the x-axis is the EQW value of the [Fe/H] indicators, while on the y-axis are the J-K colours being used as an Effective Temperature indicator. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

First, we look at the  $T_{eff}$  and [Fe/H] results, shown in Figures 4.4 and 4.5 respectively. The mean difference between the two pipelines is small compared with the values obtained for individual stars, even while there is significant scatter on the individual stars. The mean offset between the two is 13 K and 0.06 dex, while the standard deviation is 135 K and 0.11 dex. To help in understanding why there are differences between the pipelines, especially given the tests which appear to show the reliability of STARPANDA, it is necessary to look at the lines which are currently being used. Figures 4.10 and 4.11 compare the EQW values of the lines used for  $T_{eff}$  and [Fe/H] (respectively) in both the synthetic and observed spectra. Firstly, the nature of the synthetic distribution can be explained by the method used to derive these values and the limited number of model temperatures and [Fe/H] values (see Section 4.2). Secondly, it is immediately clear that the range of values in the model data for both parameters is small compared with those measured in the observed spectra. This would suggest that using J-K as a probe of effective temperature is not a good idea in this pipeline and that

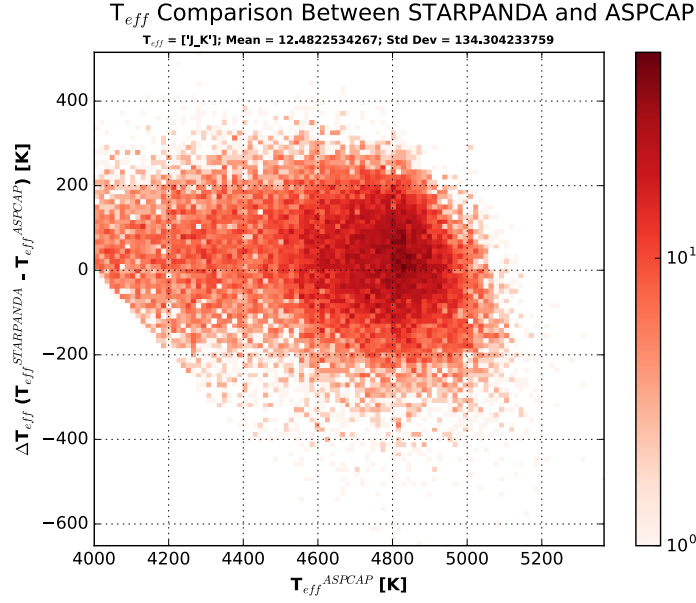


Figure 4.4: The results of a 6 parameter STARPANDA analysis with DR12 ASPCAP results for effective temperature. On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP effective temperature. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

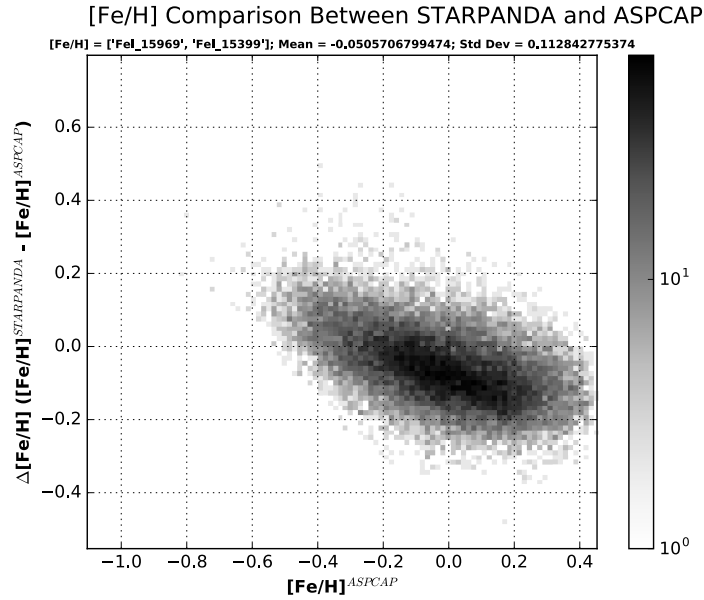


Figure 4.5: The results of a 6 parameter STARPANDA analysis with ASPCAP DR12 results for [Fe/H]. On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP [Fe/H]. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

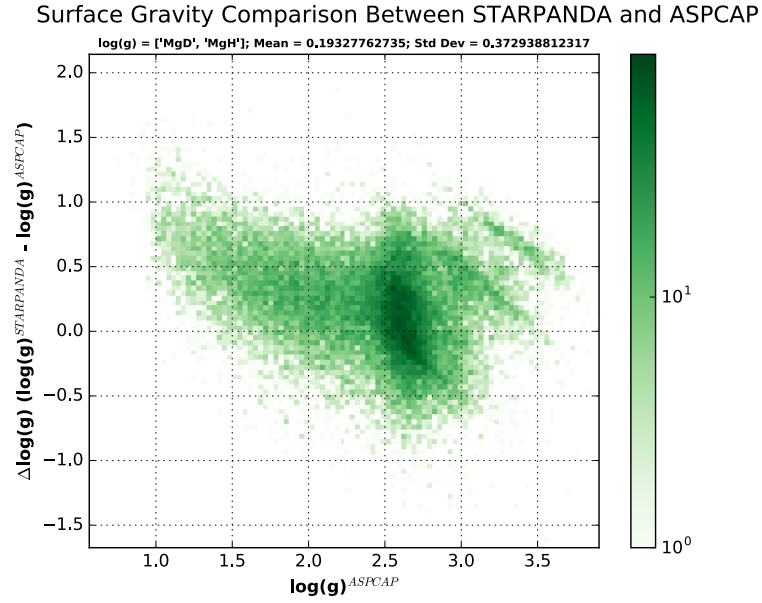


Figure 4.6: The results of a 6 parameter STARPANDA analysis with ASPCAP DR12 results for  $\log g$ . On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP  $\log g$ . The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

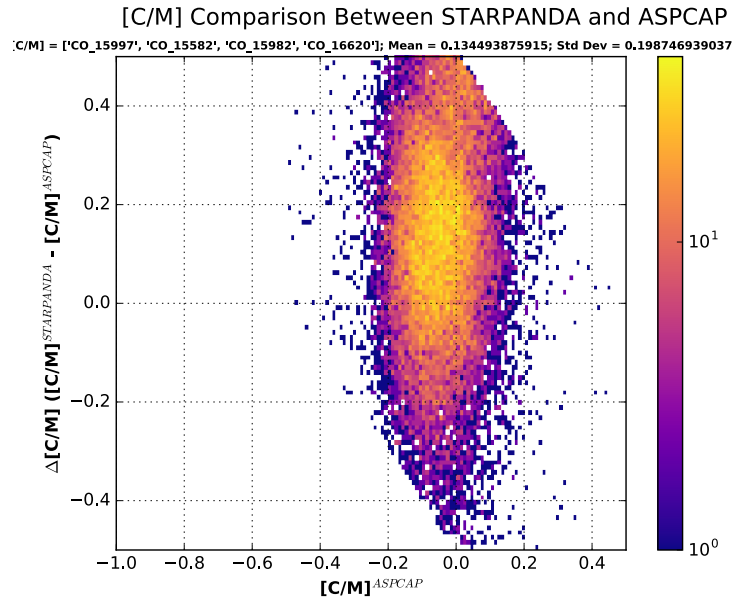


Figure 4.7: The results of a 6 parameter STARPANDA analysis with ASPCAP DR12 results for [C/M]. On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP [C/M]. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

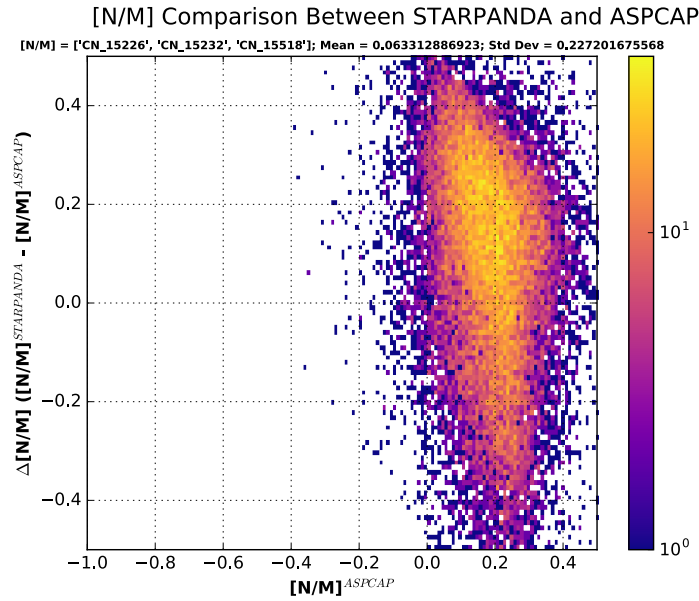


Figure 4.8: The results of a 6 parameter STARPANDA analysis with ASPCAP DR12 results for [N/M]. On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP [N/M]. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

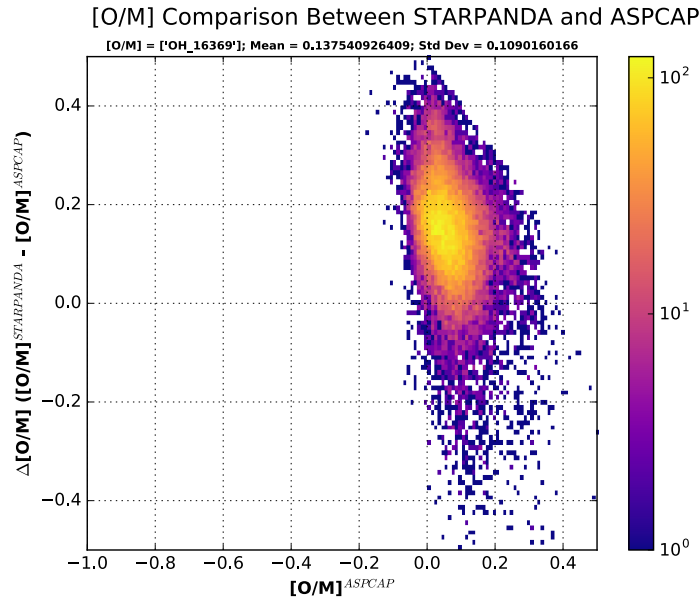


Figure 4.9: The results of a 6 parameter STARPANDA analysis with ASPCAP DR12 results for [O/M]. On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP [O/M]. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

the lines we use for deriving  $[\text{Fe}/\text{H}]$  could be improved upon. Since  $T_{\text{eff}}$  and  $[\text{Fe}/\text{H}]$  are derived together, errors stemming from the use of J-K will cause further errors in  $[\text{Fe}/\text{H}]$  (in addition to those from the errors on the measured Fe lines) and vice versa. It would be of significant benefit to this pipeline if new lines were identified for use in deriving  $T_{\text{eff}}$  and  $[\text{Fe}/\text{H}]$ , and until that happens it is difficult to further speculate on the reliability of these results.

One point to note regarding the use of comparisons between the distributions of values in observed and synthetic data is that we are not looking for indicators where both distributions are similar. We are only looking for indicators which show a greater range of values in the synthetic data than in the observed data. This is because we expect objects to be more commonly found in certain areas of parameter space than in others, while the synthetic data will show indicator values from a much wider range of parameter space; for large swathes of parameter space we would very rarely or never expect to see objects. Therefore, the range of observed values should generally appear as a subset of the synthetic range.

The results for  $\log g$ , shown in Figure 4.6, show the same large scatter when compared with ASPCAP results. Here the mean difference between the two pipelines is 0.14 dex with a standard deviation of 0.38 dex. This is worse than the results for the other two stellar parameters. Looking at Figure 4.12, which shows the comparison between the EQW values measured in the synthetic and observed spectra, it is clear that there is a discrepancy between the two distributions; for clarity the plot is shown with just the central range of measured values. The range of values obtained is testament to the complexity of the line combinations used in deriving  $\log g$  (see Tables 4.2 and 4.5) and that since we are using the average of ratios of two different lines in different combinations, any problem measuring a line is going to result in a very wide range of final values. In this case it appears that there are problems measuring the lines in the synthetic spectra, possibly due to the line becoming very weak towards the edges of the model parameter space. Perhaps a new set of lines would allow the derivation

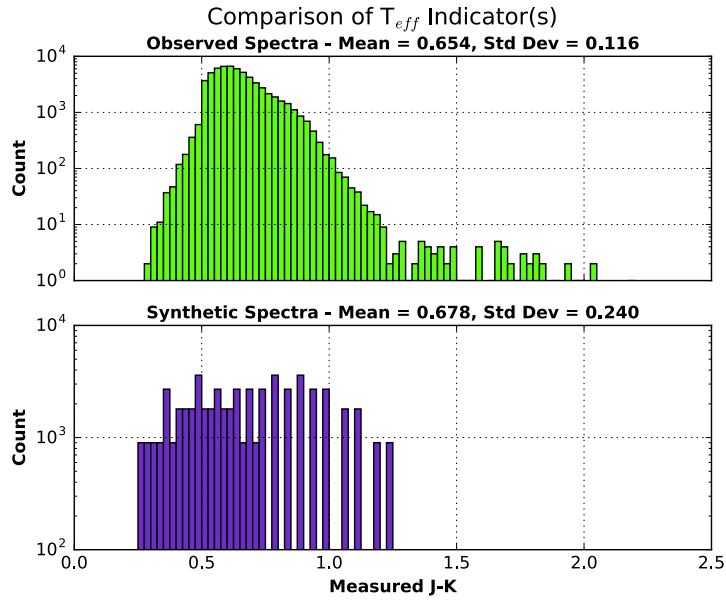


Figure 4.10: The measured J-K values in the model and observed data used to probe  $T_{eff}$ . The top panel shows the J-K values for the observed data, while the bottom panel shows the values for the model data. The mean and standard deviation for each distribution are shown above each panel. The plots share a common x-axis and have the y-axis has been shows the log of the counts.

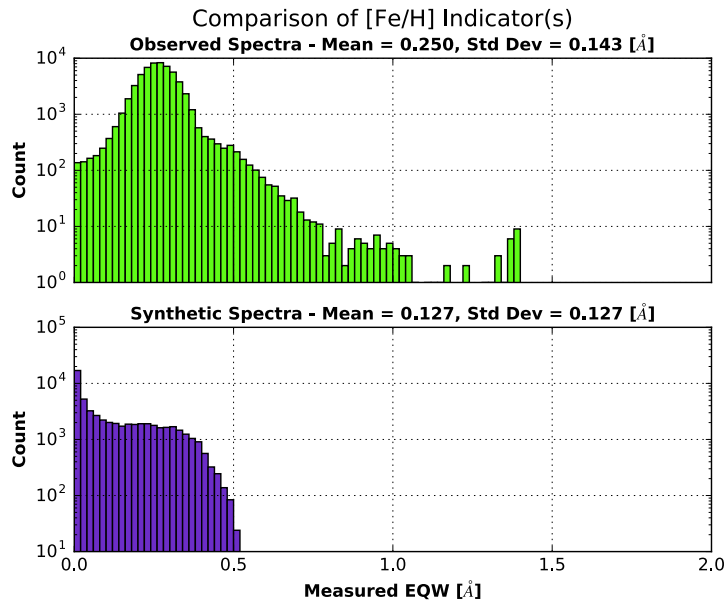


Figure 4.11: The measured indicator EQW values in the model and observed data for the lines used to probe  $[Fe/H]$ . The top panel shows the EQW values for the observed data, while the bottom panel shows the values for the model data. The mean and standard deviation for each distribution are shown above each panel. The plots share a common x-axis and have the y-axis has been shows the log of the counts.

of more reliable  $\log g$  values by STARPANDA. The most striking feature of this plot are the strong diagonal lines seen to the right hand side of the plot. These occur at  $\log g$  values of 3.5 and 4.0 (as derived by STARPANDA); others may be present at 2.5 and 3.0, though this is harder to see amidst the bulk of the data. At first, one might assume that these are some artefact of the interpolation method since they appear to correspond to node points in the model data. However, this does not appear to be the case, as these features are not present at all node points and are absent from the CNO results entirely (which uses the same code as the  $\log g$  analysis). In order to try to uncover the origin of these features we have considered stars both clustered around node values and stars between node values, looking at the distributions of EQWs of the lines used, and the stellar parameters derived by both STARPANDA and ASPCAP. However, there is no discernible difference in the distribution of EQW values between stars clustered at or between node values. Thus far, we cannot ascertain the origin of these features.

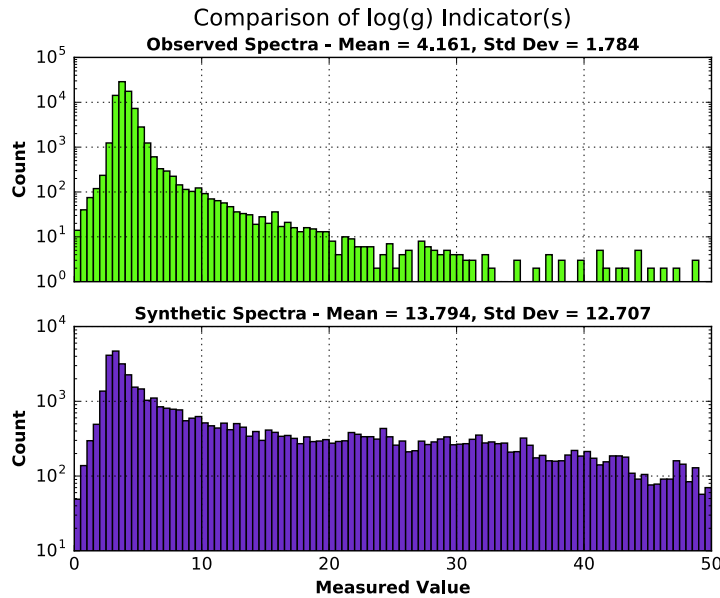


Figure 4.12: The measured indicator values in the model and observed data for the lines used to probe  $\log g$  in the range 0 - 50. The top panel shows the indicator values for the observed data, while the bottom panel shows the values for the model data. The mean and standard deviation for each distribution are shown above each panel. The plots share a common x-axis and have the y-axis has been shows the log of the counts.

Lastly, the results for the CNO abundances (Figures 4.7, 4.8 & 4.9) show a small offset



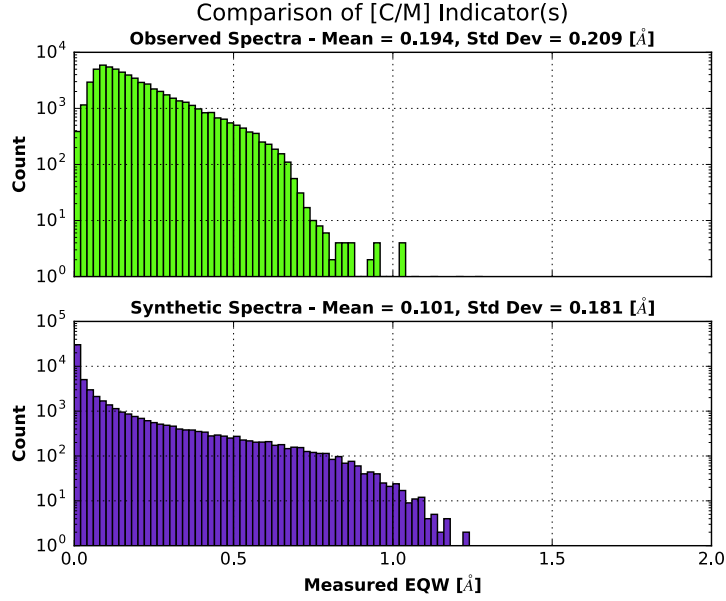


Figure 4.13: The measured indicator EQW values in the model and observed data for the lines used to probe  $[C/M]$ . The top panel shows the EQW values for the observed data, while the bottom panel shows the values for the model data. The mean and standard deviation for each distribution are shown above each panel. The plots share a common x-axis and have the y-axis has been shows the log of the counts.

when compared to ASPCAP and large scatter - the means being 0.10, 0.09 & 0.13 dex respectively, while the standard deviations are 0.20, 0.22 & 0.10 dex respectively. The  $[O/M]$  values appear the most reliable here, with  $[C/M]$  &  $[N/M]$  showing much higher scatter. The plots for C and O also show the effect of our cuts in the model parameter ranges (see Section 4.1), with clear diagonal lines showing the model cuts at  $\pm 0.5$  dex (this is especially clear on the C plot). Again, looking at the comparison between the EQW values measured in both synthetic and observed spectra (Figures 4.13, 4.14 & 4.15), we can see that for C and N, the distributions are fairly well matched. For O, while the bulk of observed stars fall within the range of synthetic spectra values, there are a number of stars with higher EQW values; this gives a partial explanation for the increased number of analysis failures at the  $[O/M]$  stage.

One point to note is that in producing the Figures 4.10 - 4.15, it was necessary to remove  $\sim 1,500$  stars from the input file. This was due to those stars either having unknown K-band extinction (given by a  $-9999.99$  value set in the **AK\_TARG** column in

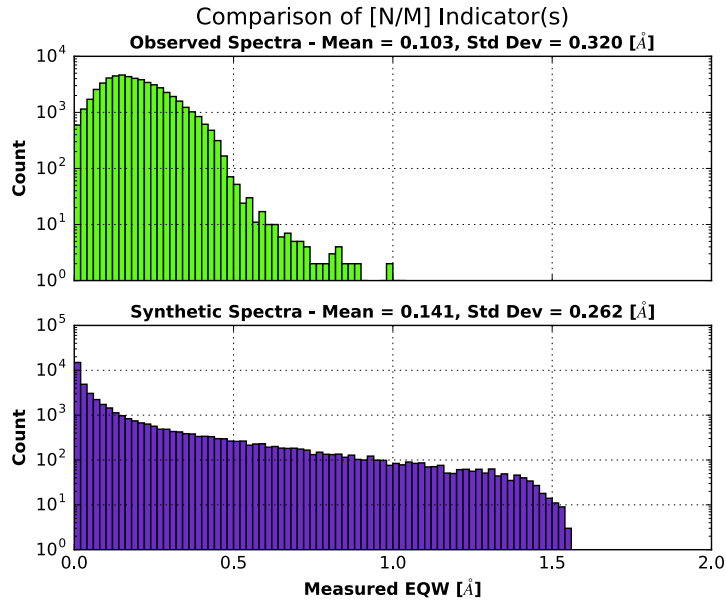


Figure 4.14: The measured indicator EQW values in the model and observed data for the line used to probe [N/M]. The top panel shows the EQW values for the observed data, while the bottom panel shows the values for the model data. The mean and standard deviation for each distribution are shown above each panel. The plots share a common x-axis and have the y-axis has been shows the log of the counts.

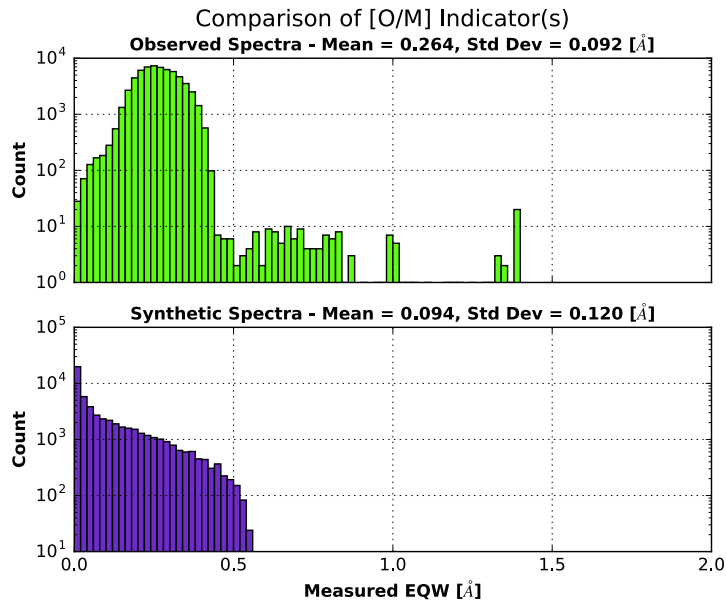


Figure 4.15: The measured indicator EQW values in the model and observed data for the lines used to probe [O/M]. The top panel shows the EQW values for the observed data, while the bottom panel shows the values for the model data. The mean and standard deviation for each distribution are shown above each panel. The plots share a common x-axis and have the y-axis has been shows the log of the counts.

the allStar files) or there being missing values for one or more lines used in calculating the  $[O/M]$  abundances. These are stars for which STARPANDA is unable to successfully complete analysis.

An alternative way of looking at these results is to plot the distributions of each parameter as a function of  $[Fe/H]$  in two separate panels and this is shown in Figures 4.16, 4.17 & 4.18 for  $[C/M]$ ,  $[N/M]$  &  $[O/M]$ . C shows the much larger scatter of values that were obtained by STARPANDA and the effects of the model parameter cuts. N shows a similar situation, though with an offset compared to ASPCAP results. Finally the STARPANDA results for O appear to be much more in line with those for ASPCAP.

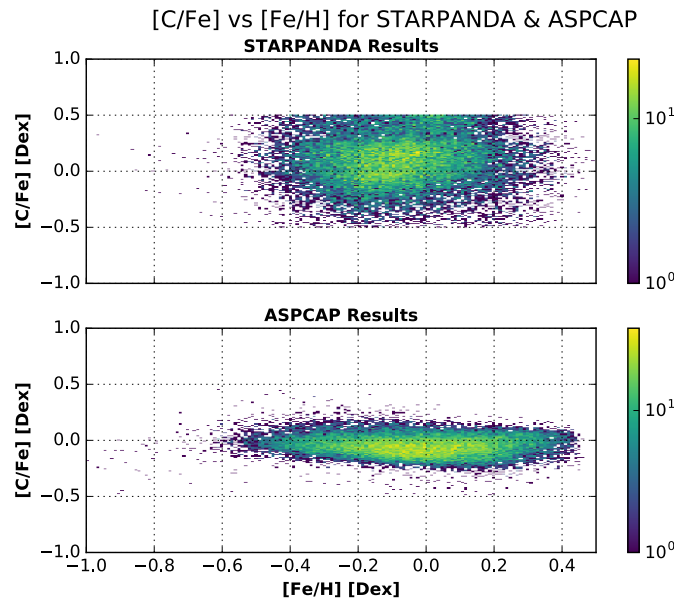


Figure 4.16: The results of a CNO abundance STARPANDA analysis and ASPCAP DR12 results for  $[C/M]$ . On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP  $[Fe/H]$ . The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

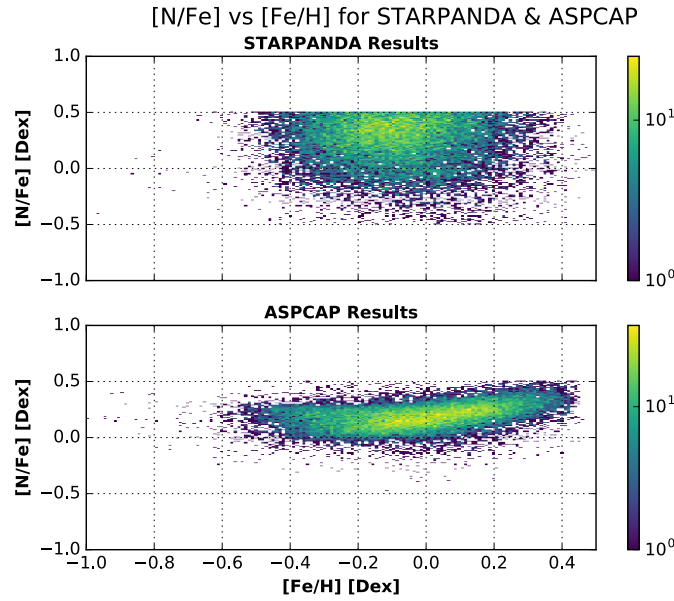


Figure 4.17: The results of a CNO abundance STARPANDA analysis and ASPCAP DR12 results for  $[N/M]$ . On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP  $[Fe/H]$ . The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

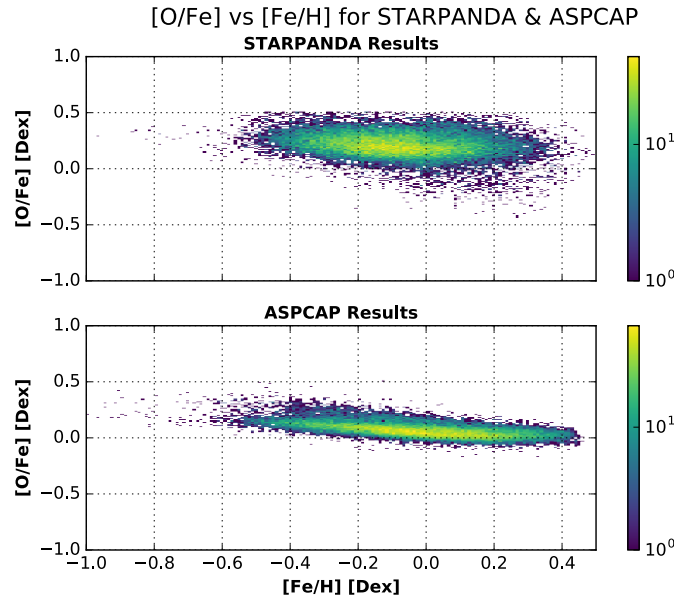


Figure 4.18: The results of a CNO abundance STARPANDA analysis and ASPCAP DR12 results for  $[Fe/H]$ . On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP  $[Fe/H]$ . The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

### 4.3.2 DR14 Analysis

For DR14, STARPANDA took 6468 seconds to analyse 63795 stars (therefore averaging  $\sim 0.10$  sec/star) and it achieved results for 26649 of these. As can be seen in Figure 4.19, the number of stars that failed due to lack of convergence was small (1404 stars), with the bulk of the failures, again, being due to the star falling outside of our model parameter space either on the first or subsequent iterations and most commonly while either deriving the  $T_{eff}$  &  $[Fe/H]$  or  $[O/M]$ .

We then compared the STARPANDA results with those given in the DR14 ASPCAP data release in the same manner as we did for the DR12 ASPCAP results and produced plots similar to those shown in Figures 4.10 - 4.15. With the exception of  $[N/M]$ , there is no significant difference between the plots produced using DR12 and DR14 data (and so we have omitted them here). For  $[N/M]$ , DR14 ASPCAP results saw a  $\sim 0.15$  dex change in abundance when compared with the DR12 values. This can be seen in our results by comparing the Figure 4.21 (produced using the DR14 data) with Figure 4.8 (produced using the DR12 data). Table 4.7 shows the mean and standard deviation values for the comparison between STARPANDA and ASPCAP using both DR12 and DR14 data; again the effect of the shift in ASPCAP  $[N/M]$  values can be clearly seen.

Parameter	DR12		DR14	
	Mean	Std	Mean	Std
$T_{eff}$ [K]	12.482	134.304	20.969	129.531
$[Fe/H]$ [Dex]	-0.051	0.113	-0.058	0.103
$\log g$ [Dex]	0.193	0.373	0.278	0.339
$[C/M]$ [Dex]	0.135	0.199	0.153	0.198
$[N/M]$ [Dex]	0.063	0.227	-0.060	0.205
$[O/M]$ [Dex]	0.138	0.109	0.178	0.108

Table 4.7: The mean and standard deviation for each parameter obtained from both STARPANDA & ASPCAP for DR12 & DR14 results. The STARPANDA results come from 6 parameter analysis runs.

As will be seen in the next section, the results for CNO abundances can be dramatically

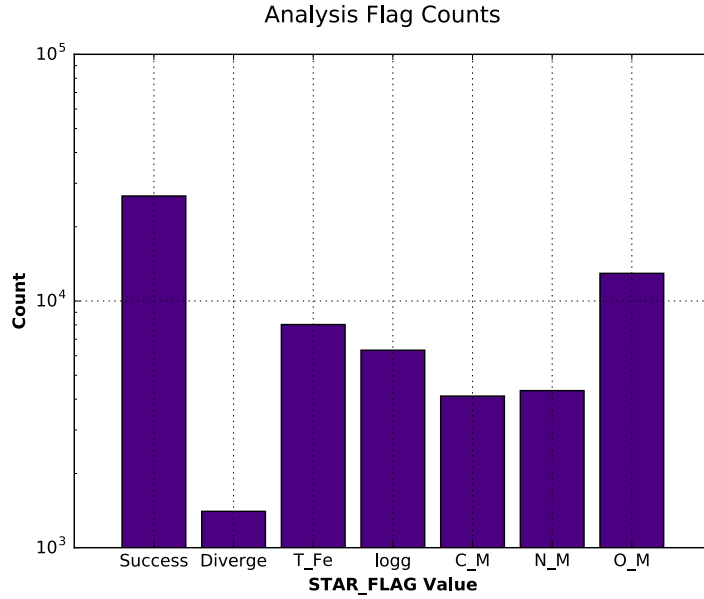


Figure 4.19: Analysis Flags for a 6 parameter run of STARPANDA. The x-axis shows, Successful Analysis, Failure due to Divergence, Failure at  $T_{eff}$  &  $[Fe/H]$  analysis, Failure at  $\log g$  analysis, Failure at  $[C/M]$  analysis, Failure at  $[N/M]$  analysis or Failure at  $[O/M]$  analysis. The y-axis is then the log of the number of occurrences.

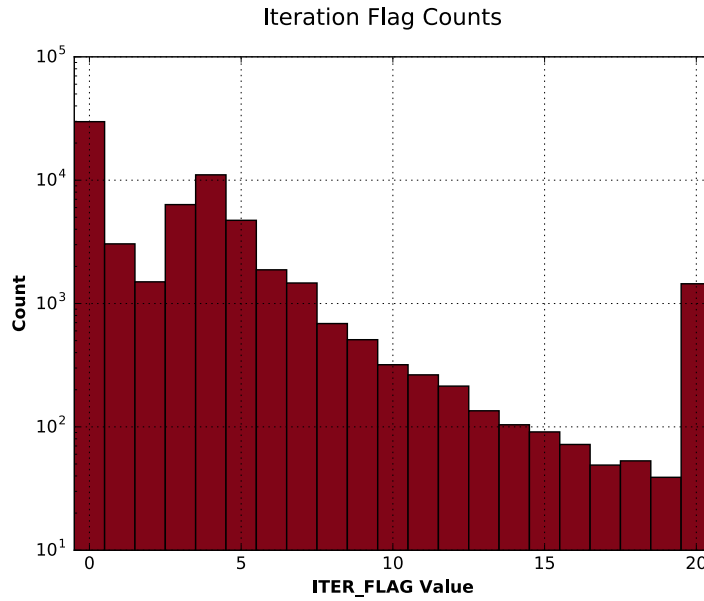


Figure 4.20: Iteration Flag results for a 6 parameter run of STARPANDA. This histogram shows the number of iterations completed during analysis, with the count on the y-axis being logged. There is a large peak at 0 iterations, showing a failure of analysis due to the star being outside of model parameter space and a second smaller, but still sizable, peak at 20 iterations, showing the number of stars for which convergence was not possible.

improved if we can first obtain good results for the stellar parameters. So it appears that, while the results currently are not as good as we could hope for, there is still hope

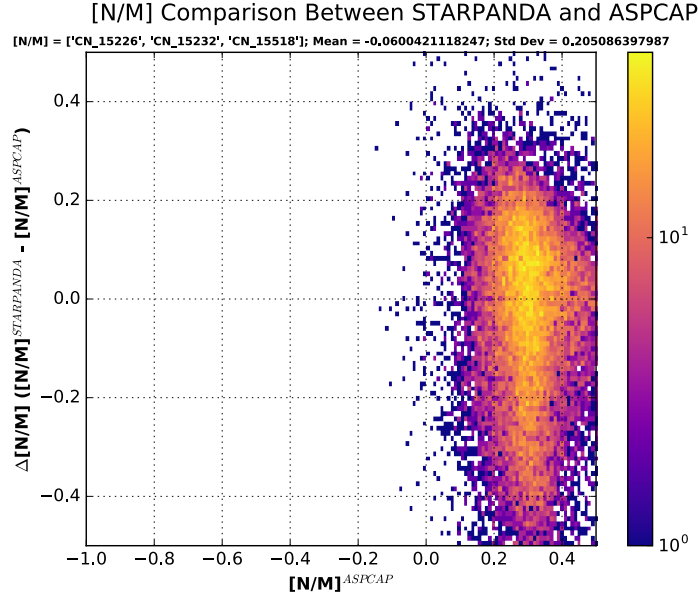


Figure 4.21: The results of a 6 parameter STARPANDA analysis with ASPCAP DR14 results for  $[N/M]$ . On the y-axis is the difference between the results obtained by STARPANDA and ASPCAP, while the x-axis shows the ASPCAP  $[N/M]$ . The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

and that if we would be able to identify better lines for use in deriving  $T_{eff}$ ,  $[Fe/H]$  and  $\log g$ , then we would expect to see a significant improvement in the results for all 6 parameters. This will be discussed further in Chapter 6.

## 4.4 Analysis of CNO Abundances

In this section, the results of just a CNO abundance analysis are shown, with comparisons to ASPCAP results. The stellar parameters are imported from ASPCAP and then the CNO abundances are derived iteratively (the derivation order is O, then C, then N) using the lines shown in Table 4.5. Firstly, we use the DR12 data release to both import the stellar parameters and to then compare our CNO abundances against, then we repeat the procedure using DR14 stellar parameters and CNO abundances. Given the results shown in the previous section when STARPANDA is run with a full 6 parameter analysis option selection, we address the question of what effect importing ASPCAPs

stellar parameters and running STARPANDA with a CNO abundance analysis option will have on the CNO results obtained.

#### 4.4.1 DR12 Analysis

First, STARPANDA took just 3891 seconds to analyse the 79519 stars present in the input files and obtained CNO abundances for 62586 - this gives a  $\sim 79\%$  success rate with an average of 0.05 sec/star analysis time. This is a significant improvement over the  $\sim 35\%$  return achieved when deriving all 6 parameters. Examining the results of the Analysis Flags, shown in Figure 4.22, we conclude that the biggest point of failure is due to targets being outside of model parameter space when deriving  $[N/M]$  values, though there are still significant numbers of failures during the derivation of both  $[C/M]$  and  $[O/M]$ . Looking at the results of the Iteration Flag, shown in Figure 4.23, it is clear that the bulk of the analysis failures occurred during the first iteration. One way this could be explained is by invoking the cuts that have been made to the model parameter space, thereby excluding any star occupying the more unusual areas of parameter space. Another explanation is that we assume initial values for the CNO abundances and since O is here the first to be analysed, it could fail due to the assumed C & N values being far from their actual values. This would also be true for C (which is next to be analysed) as it is still taking the initial assumed N value.

One solution to this would be to run the code multiple times with a range of initial values for the CNO abundances and then merging the resulting output files to achieve the maximum number of successful results. Given the speed of the code, this could be a reasonable solution. However, it can be hoped that by expanding the range of fully occupied model parameter ranges, this would reduce the number of analysis failures and also allow the pipeline to derive parameters for the more unusual (and to some interesting) stars; see Chapter 6 for more on this.



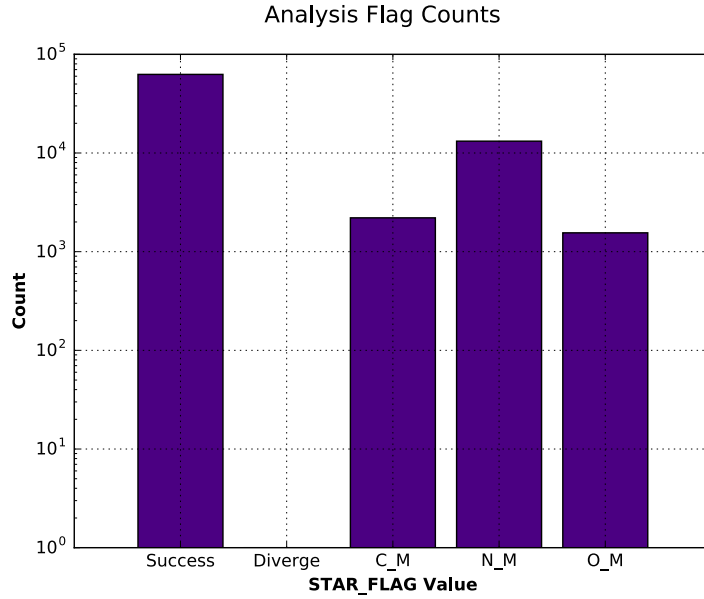


Figure 4.22: Analysis Flags for a 3 parameter run of STARPANDA. The x-axis shows, Successful Analysis, Failure due to Divergence, Failure at [C/M] analysis, Failure at [N/M] analysis or Failure at [O/M] analysis. The y-axis is then the log of the number of occurrences.

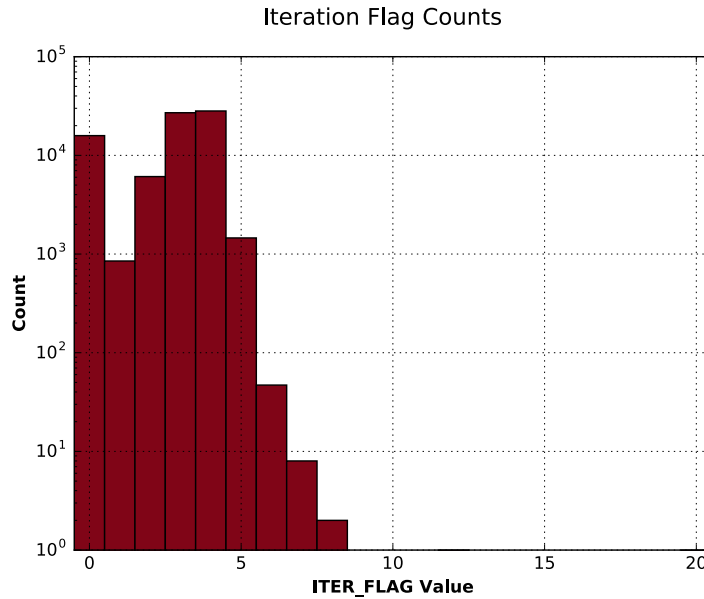


Figure 4.23: Iteration Flag results for a 3 parameter run of STARPANDA. This histogram shows the number of iterations completed during analysis, with the count on the y-axis being logged. There is a large peak at 0 iterations, showing a failure of analysis due to the star being outside of model parameter space; there are no stars showing as reaching 20 iterations (i.e. no stars failed to convergence or move out of model parameter space within the specified convergence limit).

So, considering then the results for each abundance individually, we have once again plotted comparisons between the results obtained by STARPANDA and those obtained

by ASPCAP. Firstly, we have the difference in the STARPANDA and ASPCAP results plotted (on the y-axis) against the ASPCAP result (x-axis) and then we have also plotted the distributions of each pipelines  $[x/M]$  value as a function of the ASPCAP  $[Fe/H]$ .

Starting with O, the first abundance calculated in the results presented in this section, Figure 4.24 (top panel) shows the difference between the  $[O/M]$  values calculated by STARPANDA and ASPCAP as a function of the ASPCAP  $[O/M]$  - this can be compared with the same plot (Figure 4.9) for the 6 parameter analysis run. The mean difference between the pipelines has decreased to 0.080 dex, with the standard deviation also falling to 0.075 dex and while there is still significant scatter, it has been reduced dramatically compared with the 6 parameter analysis results. One obvious feature to note in the top plot is the bi-modal distribution, which increases as the value of ASPCAP  $[O/M]$  increases. This bi-modality is readily explained as reflecting the well-known bi-modal distribution of  $\alpha$  elements at constant  $[Fe/H]$  in disc stars (e.g. Mackereth et al., 2017, and references therein). Figure 4.24 (bottom panels) shows the distributions of  $[O/M]$  as a function of ASPCAP  $[Fe/H]$  for both STARPANDA and ASPCAP (top and bottom respectively) - compare these to the plots in Figure 4.18. The 2 distributions are now much closer and the bimodal distribution of  $[O/M]$  at fixed  $[Fe/H]$  now visible in the STARPANDA results. The main difference appears at high  $[Fe/H]$  values where STARPANDA shows either a more pronounced flattening or even a slight upturn in  $[O/M]$  for stars with super-solar metallicities.

Looking now at C, the second abundance to be derived in this run, Figure 4.25 (top panel) shows the difference between STARPANDA and ASPCAP  $[C/M]$ , as a function of ASPCAP  $[C/M]$  (compare with Figure 4.7). Again, there has been a significant improvement in the results and now we show that, while there is still high scatter, there is no substantial difference between the  $[C/M]$  values output by the two pipelines; the mean difference has fallen to just 0.007 dex, with a standard deviation of only 0.087 dex. The effects of the model parameter cuts are clearly visible in the hard diagonal edges to the results. Figure 4.25 (bottom panels) shows the distributions of  $[C/M]$  for

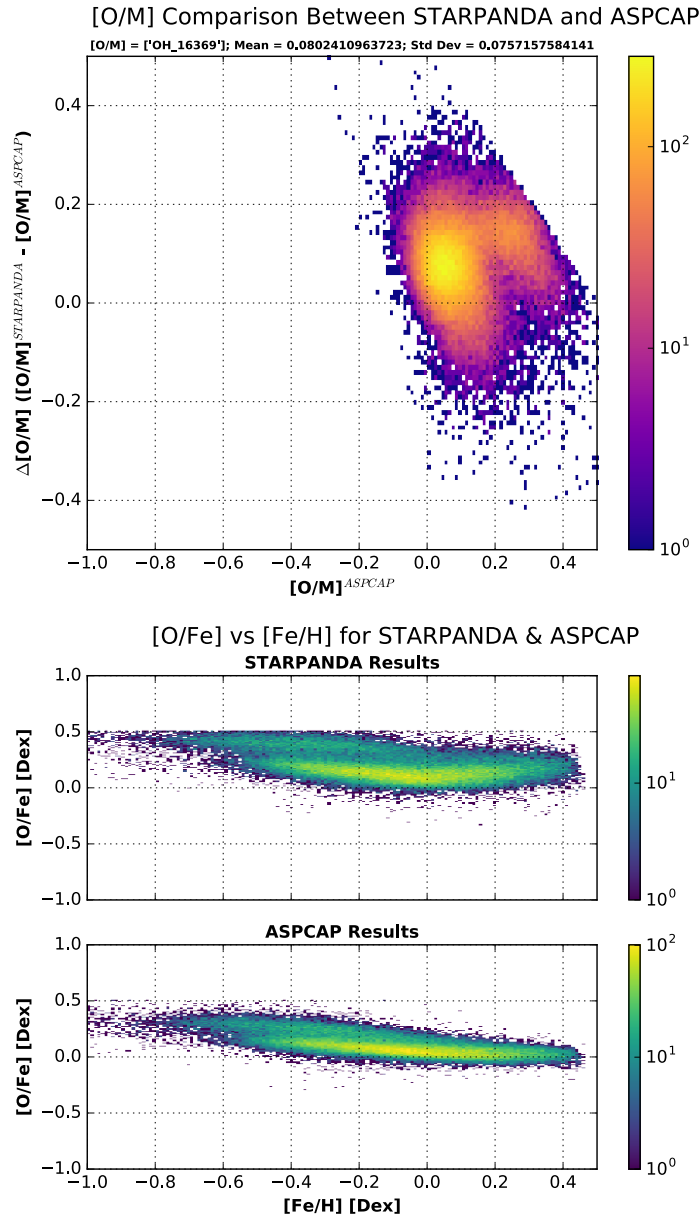


Figure 4.24: The results of a CNO abundance STARPANDA analysis with ASPCAP DR12 results for  $[\text{O}/\text{M}]$ . The top plot shows difference between the abundances derived by STARPANDA and ASPCAP as a function of the ASPCAP abundance. The subtitle shows the lines used in the analysis as well as the mean and standard deviation of the difference between the abundances derived by both pipelines. The middle and bottom plots show the distribution of  $[\text{O}/\text{M}]$  as a function of  $[\text{Fe}/\text{H}]$  (taken from ASPCAP). The bottom two plots share a x-axis and for all three plots the density is represented by the colour, which is shown in the colour bar to the right of the plots (additionally we have shown this on a log scale).

both STARPANDA and ASPCAP as a function of the ASPCAP  $[\text{Fe}/\text{H}]$ . The similarity between the two distributions is clear, with no substantial difference between them.

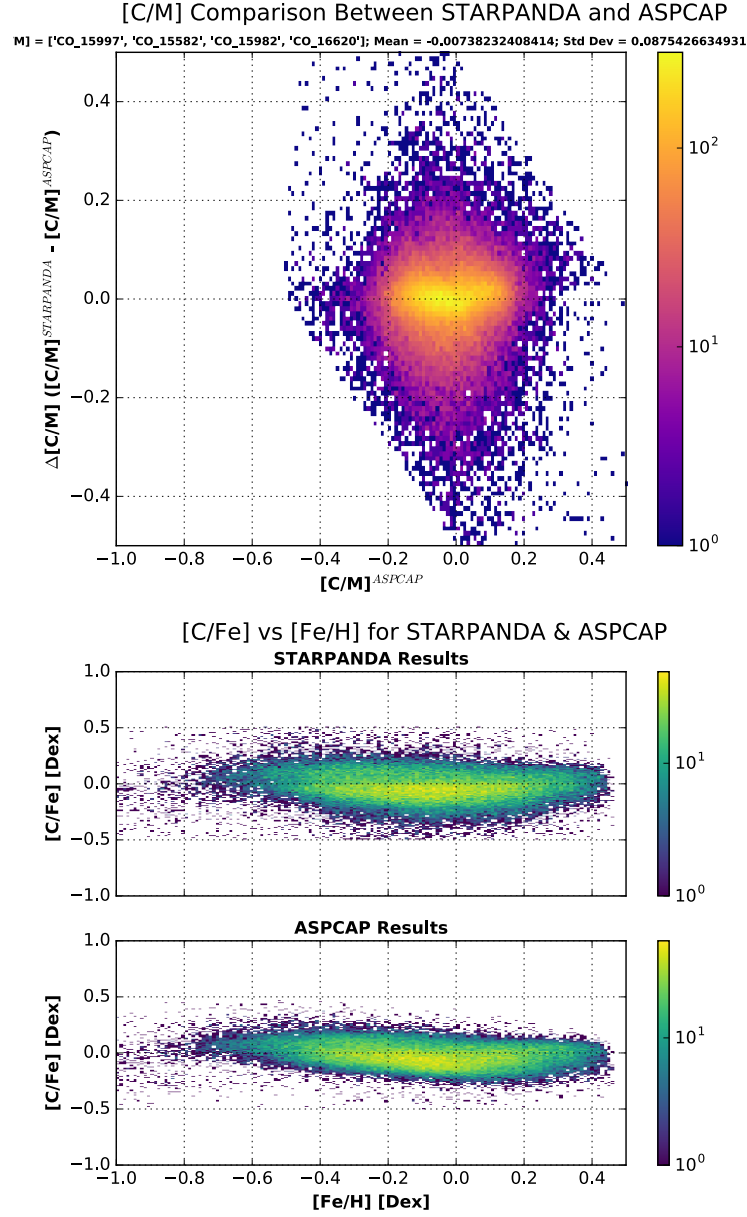


Figure 4.25: The results of a CNO abundance STARPANDA analysis with ASPCAP DR12 results for  $[C/M]$ . The top plot shows difference between the abundances derived by STARPANDA and ASPCAP as a function of the ASPCAP abundance. The subtitle shows the lines used in the analysis as well as the mean and standard deviation of the difference between the abundances derived by both pipelines. The middle and bottom plots show the distribution of  $[C/M]$  as a function of  $[Fe/H]$  (taken from ASPCAP). The bottom two plots share a x-axis and for all three plots the density is represented by the colour, which is shown in the colour bar to the right of the plots (additionally we have shown this on a log scale).

Finally looking at N, which was the last abundance to be derived, Figure 4.26 (top panel) shows the difference between STARPANDA and ASPCAP  $[N/M]$ , as a function of ASPCAP  $[N/M]$  (compare this with Figure 4.8). Now the mean difference has

increased to 0.115 dex with standard deviation reducing to 0.170 dex; however, there is now a clear slope, with the two pipelines agreeing more closely at higher values of ASCAP [N/M]. Figure 4.26 (bottom panels) shows [N/M] for both STARPANDA and ASPCAP as a function of ASPCAP [Fe/H] and the difference between the two pipelines at lower values of ASPCAP [Fe/H] is clear. It will be necessary to compare the results of both pipelines using reference stars in order to understand these differences more fully.

#### 4.4.2 DR14 Analysis

For DR14, STARPANDA took 3300 seconds to analyse 63795 stars (therefore averaging  $\sim 0.05$  sec/star) and it achieved results for 52549 of these (a  $\sim 82\%$  success rate). Looking at Figure 4.27 shows that N was the biggest cause of the analysis failing, while Figure 4.28 shows that the number of stars that failed due to lack of convergence was small (5 stars).

Repeating our previous comparisons again using the DR14 ASPCAP results with those obtained by STARPANDA, we can again re-plot some of our figures and see how the new ASPCAP results compare with ours. So, this time we re-plot Figures 4.24, 4.25 and 4.26 with the new results and produce Figure 4.29, 4.30 and 4.31; in each case the top plot shows the difference between the pipelines as a function of the ASPCAP result, while the bottom two plots shows the results of each pipeline as a function of the ASPCAP [Fe/H] value. Figure 4.29 (top panel) shows the same bi-modal distribution as previously seen in Figure 4.24. This can be explained, once again, by the bi-modality observed in the  $\alpha$  vs Fe plots shown in the lower panels of both figures.

This time, using the DR14 stellar parameters, but calculating CNO abundances using STARPANDA we see an improvement in the match between the values produced by both pipelines.

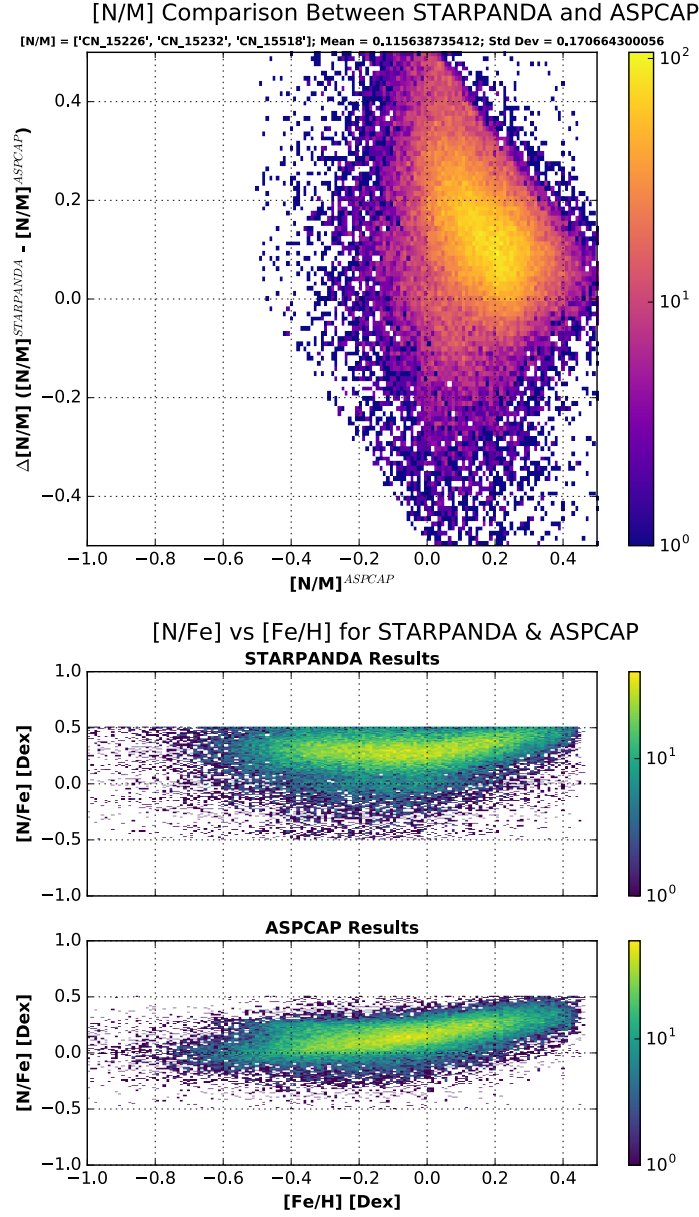


Figure 4.26: The results of a CNO abundance STARPANDA analysis with ASPCAP DR12 results for  $[N/M]$ . The top plot shows difference between the abundances derived by STARPANDA and ASPCAP as a function of the ASPCAP abundance. The subtitle shows the lines used in the analysis as well as the mean and standard deviation of the difference between the abundances derived by both pipelines. The middle and bottom plots show the distribution of  $[N/M]$  as a function of  $[Fe/H]$  (taken from ASPCAP). The bottom two plots share a x-axis and for all three plots the density is represented by the colour, which is shown in the colour bar to the right of the plots (additionally we have shown this on a log scale).

Overall, the results are encouraging. Further work on identifying the best lines to use for CNO could see these results improve, and if better lines for  $T_{\text{eff}}$ ,  $[Fe/H]$  and  $\log g$

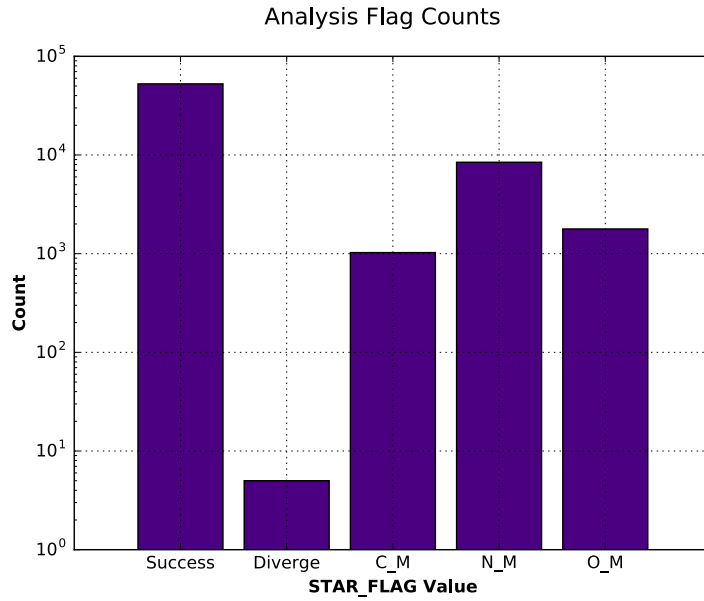


Figure 4.27: Analysis Flags for a 3 parameter run of STARPANDA. The x-axis shows, Successful Analysis, Failure due to Divergence, Failure at [C/M] analysis, Failure at [N/M] analysis or Failure at [O/M] analysis. The y-axis is then the log of the number of occurrences.

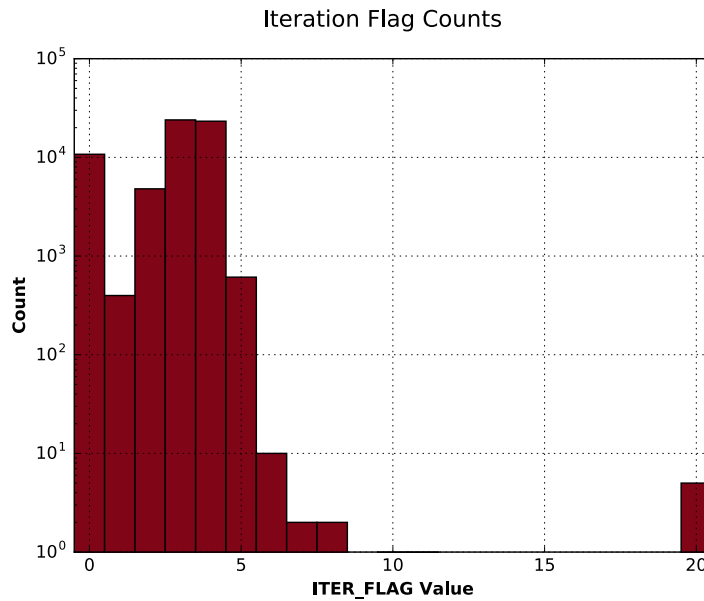


Figure 4.28: Iteration Flag results for a 3 parameter run of STARPANDA. This histogram shows the number of iterations completed during analysis, with the count on the y-axis being logged. There is a large peak at 0 iterations, showing a failure of analysis due to the star being outside of model parameter space and a second, smaller peak at 20 iterations, showing the number of stars for which convergence was not possible.

are found, then it could prove advantageous to go back to a full 6 parameter analysis.

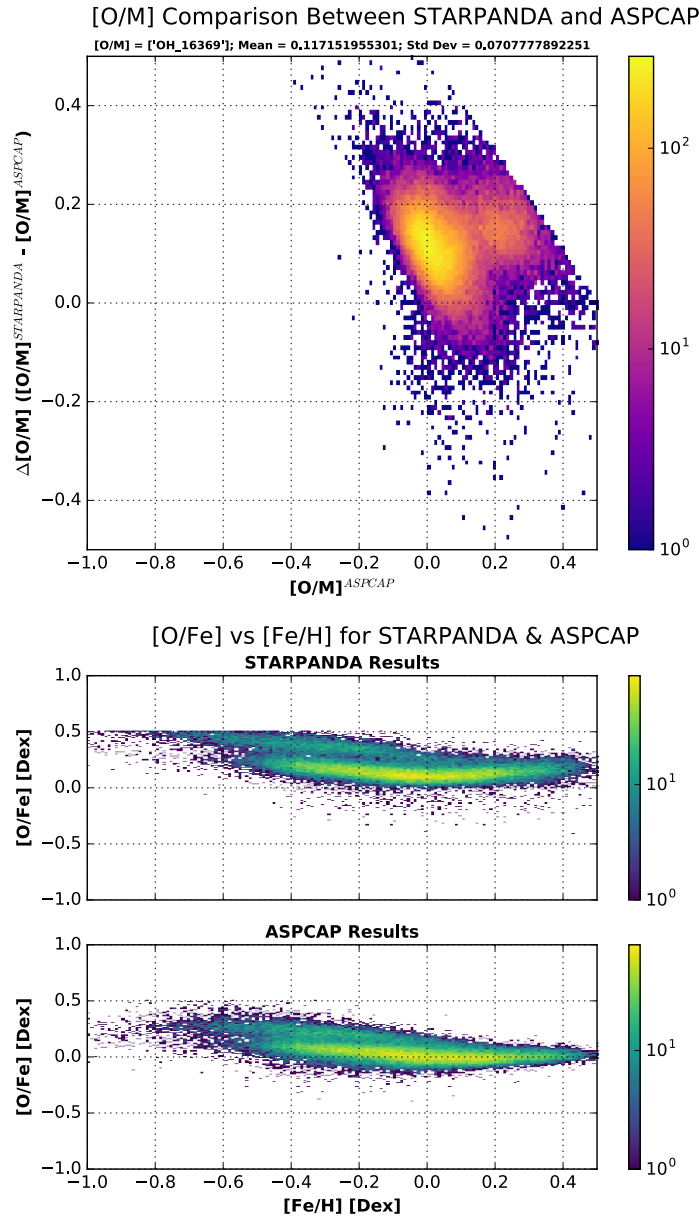


Figure 4.29: The results of a CNO abundance STARPANDA analysis with ASPCAP DR14 results for  $[O/M]$ . The top plot shows difference between the abundances derived by STARPANDA and ASPCAP as a function of the ASPCAP abundance. The subtitle shows the lines used in the analysis as well as the mean and standard deviation of the difference between the abundances derived by both pipelines. The middle and bottom plots show the distribution of  $[O/M]$  as a function of  $[Fe/H]$  (taken from ASPCAP). The bottom two plots share a x-axis and for all three plots the density is represented by the colour, which is shown in the colour bar to the right of the plots (additionally we have shown this on a log scale).



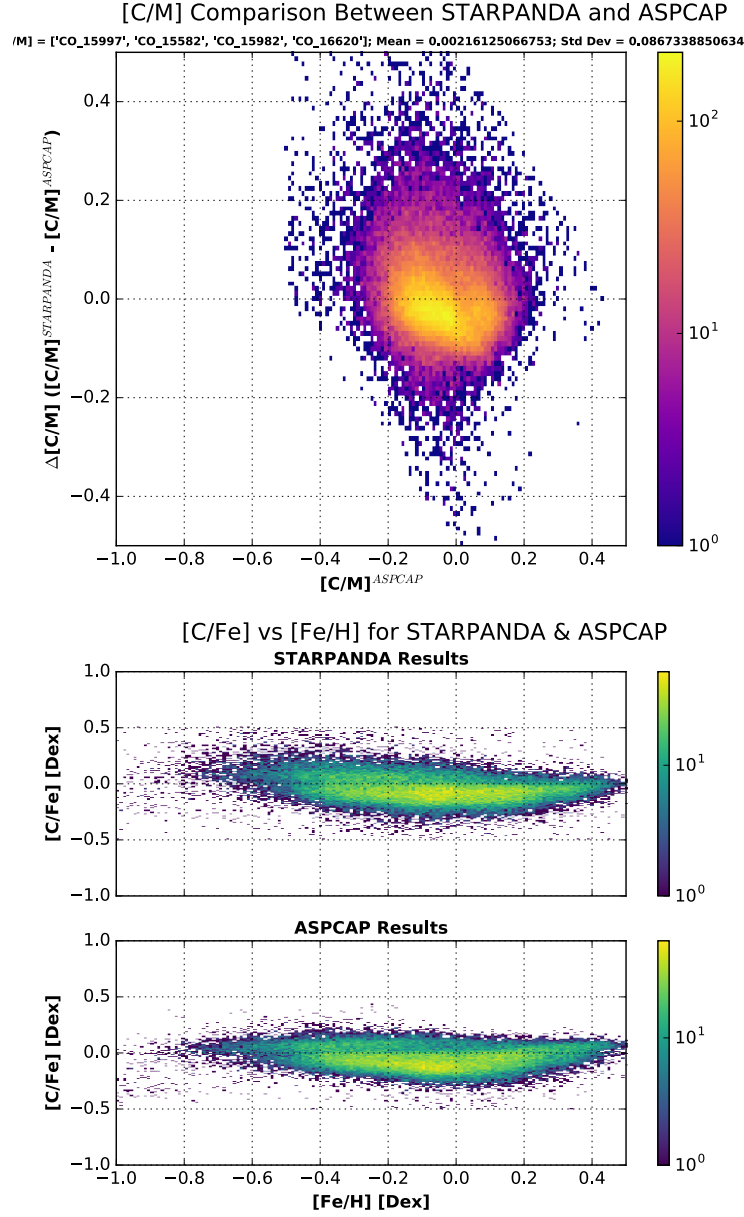


Figure 4.30: The results of a CNO abundance STARPANDA analysis with ASPCAP DR14 results for  $[C/M]$ . The top plot shows difference between the abundances derived by STARPANDA and ASPCAP as a function of the ASPCAP abundance. The subtitle shows the lines used in the analysis as well as the mean and standard deviation of the difference between the abundances derived by both pipelines. The middle and bottom plots show the distribution of  $[C/M]$  as a function of  $[Fe/H]$  (taken from ASPCAP). The bottom two plots share a x-axis and for all three plots the density is represented by the colour, which is shown in the colour bar to the right of the plots (additionally we have shown this on a log scale).

#### 4.4.3 Comments on CNO Abundances

While the stellar parameter results obtained by STARPANDA still require some further work, we are able to obtain good CNO abundances when using the ASPCAP stellar

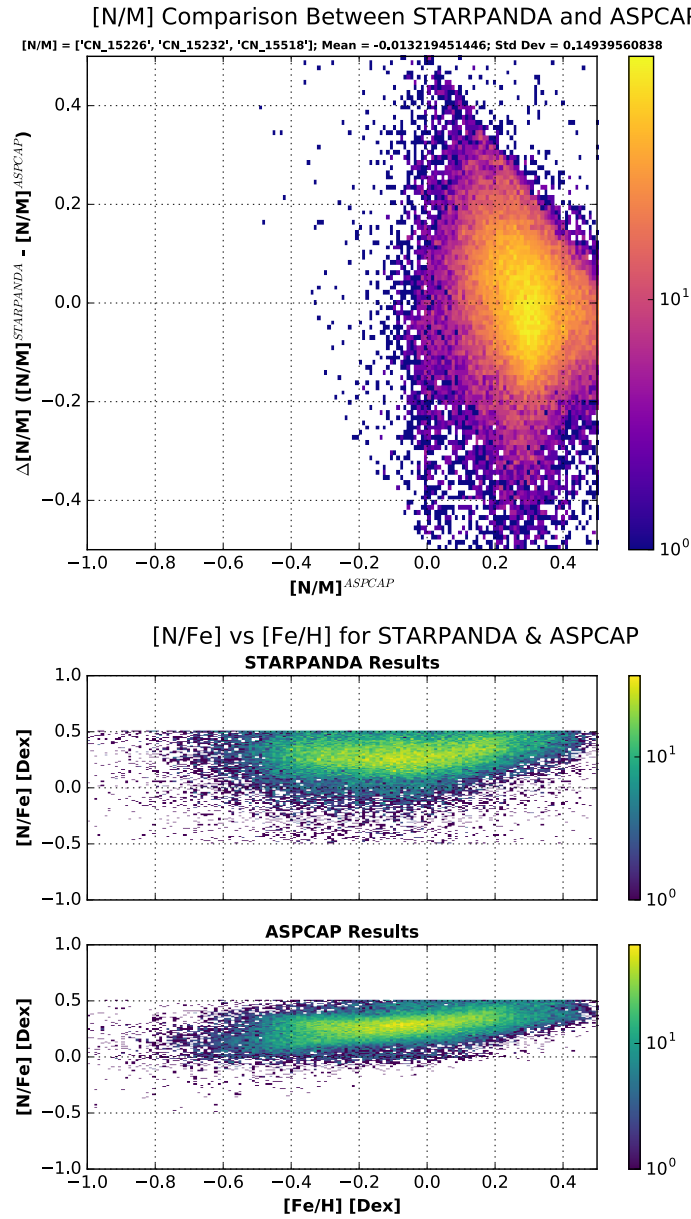


Figure 4.31: The results of a CNO abundance STARPANDA analysis with ASPCAP DR14 results for [N/M]. The top plot shows difference between the abundances derived by STARPANDA and ASPCAP as a function of the ASPCAP abundance. The subtitle shows the lines used in the analysis as well as the mean and standard deviation of the difference between the abundances derived by both pipelines. The middle and bottom plots show the distribution of [N/M] as a function of [Fe/H] (taken from ASPCAP). The bottom two plots share a x-axis and for all three plots the density is represented by the colour, which is shown in the colour bar to the right of the plots (additionally we have shown this on a log scale).

parameters and our own measurements of a carefully chosen selection of CNO lines. While the figures presented earlier in this chapter provide a sense of how well the two pipelines agree, it would be better to have a more quantitative idea. To that end, Table

Analysis Run	CNO Parameter	STARPANDA		ASPCAP		No. of Stars
		Mean [Dex]	Std [Dex]	Mean [Dex]	Std [Dex]	
DR12 6P	C	0.06574	0.21416	-0.06820	0.08434	2078
	N	0.24081	0.21532	0.16224	0.09342	
	O	0.18837	0.04891	0.04806	0.04604	
DR14 6P	C	0.06155	0.21180	-0.08818	0.0814	1890
	N	0.25402	0.20606	0.28864	0.08872	
	O	0.19089	0.09265	0.00324	0.05170	
DR12 CNO	C	-0.05296	0.12117	-0.06134	0.09236	3527
	N	0.22605	0.16914	0.14341	0.11448	
	O	0.11463	0.08562	0.05377	0.05311	
DR14 CNO	C	-0.07546	0.09883	-0.08731	0.09124	2969
	N	0.24133	0.15752	0.28710	0.10513	
	O	0.11274	0.06830	0.00974	0.06158	

Table 4.8: The mean and standard deviation for CNO abundances between STARPANDA & ASPCAP for DR12 & DR14, using both STARPANDA 6 parameter analysis runs, as well as STARPANDA CNO abundance analysis runs. These figures are for a subset stars (the number of stars in each group is given in the final column) in the range  $-0.02 \leq [\text{Fe}/\text{H}]^{\text{ASPCAP}} \leq +0.02$ .

4.8 presents the mean and standard deviations from a thin slice of stars around a  $[\text{Fe}/\text{H}] = 0.0$  ( $\Delta = 0.02$ ).

The values presented in this table re-iterate the comments made in the sections on the individual sets of results. Namely, that when STARPANDA derives all six parameters, the CNO abundances derived have a higher scatter than the equivalent ASPCAP results and show an offset compared to ASPCAP (though the offset is within  $1-\sigma$ ). When STARPANDA uses the ASPCAP stellar parameters to derive CNO abundances, the result we obtain are of comparable precision to those produced by ASPCAP.

We also used the CNO abundance analysis as an opportunity to see how varying the initial CNO abundances affects the number of stars for which STARPANDA is able to derive CNO abundances. Therefore, in Table 4.9 we show the results of this analysis, taking 79519 input stars (from the DR12 Analysis earlier) and running STARPANDA with 7 different initial values of CNO; we take initial values of  $-0.3$  to  $+0.3$  in  $0.1$

Initial Parameters CNO [Dex]	Analysis Results		
	Successes	Failures	Divergences
-0.3	62456	17063	0
-0.2	62534	16985	1
-0.1	62576	16943	1
0.0	62586	16933	1
0.1	62594	16925	1
0.2	62463	17056	1
0.3	62044	17475	0
Combined	61797	16801	0

Table 4.9: The Analysis Results obtained by STARPANDA from 7 runs of the code using different initial values for CNO abundances. The first column shows the initial values used for [C/M], [N/M] & [O/M] (all 3 values are varied together), while the remaining 3 columns show the number of successful analysis cases, the number of failed cases and the number that failed to converge (divergences) out of the total number of stars that were analysed, 79519. The final row shows the number of stars for which STARPANDA was able to successfully derive results regardless of initial parameter values, the number for which it always failed and the number for which it always diverged.

steps and vary CNO together. The first column shows the initial values for CNO, while the remaining columns then show the number of stars for which CNO abundances are successfully derived, the number for which analysis fails and the number for which no convergence is achieved within 20 iterations.

So, we can see from these results, that the number of successful analyses is relatively insensitive to the initial parameters chosen for the CNO abundances. It is only when starting with CNO abundances at +0.3, that we see a slight drop in the number of stars for which parameters can be successfully derived. For initial values from -0.3 to +0.2, the number of successes varies only by  $\sim 130$ , while changing from +0.2 to +0.3 this number decreases by  $\sim 420$ . So, as long as the initial parameters are chosen sensibly, there need be no concern at how small changes in these values might affect the number of stars for which STARPANDA is able to derive results. Taking this slightly further, if we combine the results from all 7 runs of STARPANDA, we find that 61797 stars successfully complete analysis in all runs and 16801 stars that fail analysis in all runs (i.e. these are the stars for which STARPANDA either always derives results or always fails to derive results regardless of initial parameter values). We see, therefore, that

Abundance	Mean Difference [Dex]	Standard Deviation [Dex]
C	0.00638	0.00574
N	-0.00782	0.00795
O	0.00050	0.00093

Table 4.10: Comparing the CNO abundance results obtained by STARPANDA for the same input data, but with 2 different sets of initial CNO parameters. The runs used CNO values of  $-0.3$  and  $+0.3$ . The difference is found by taking the CNO abundance obtained in the  $-0.3$  run and subtracting those obtained from the  $+0.3$  run. The Mean Difference and Standard Deviation are then found from the whole 61797 set of stars which were successfully analysed by all 7 runs of STARPANDA.

the number of stars for which the choice of starting parameter values matter is small compared with the number of stars being analysed. Finally, we can compare the results obtained from various runs to see how the choice of initial starting parameters affects the CNO abundances obtained. Table 4.10 shows the mean difference between the CNO abundances obtained from the first and last run (taken as the abundance value of the first run minus the value obtained in the last run), with initial CNO parameters of  $-0.3$  and  $0.3$  respectively, together with the standard deviation. We see that for all three parameters, the mean difference and standard deviations are small, further confirming that the choice of initial parameter value has little effect on the final results obtained by STARPANDA.

## 4.5 Analysis of Elemental Abundances

In this section, the results of an abundance analysis using imported stellar parameters and CNO abundances are shown. At this stage, we are just showing the results for Al, but no further changes to the pipeline will be required before other elements can be analysed. The only work that will be required is to identify the lines that can be used in analysis, generate the model mini-grids that we use and then measure the lines in both observed and synthetic spectra. Once again, Table 4.5 shows the lines used to produce these results.

Firstly, we used the stellar parameters and CNO abundances given in the DR12 data release and then compared our Al abundances against the DR12 ASPCAP Al abundances. Then we repeated the procedure using DR14 stellar parameters, CNO abundances and Al abundances instead.

### 4.5.1 DR12 Analysis

STARPANDA took 584 seconds to analyse 79519 stars and successfully derived Al abundances for 78595 stars - achieving a  $\sim 99\%$  success rate in  $\sim 0.007$  sec/star. As before, the stellar parameters and CNO abundances have been taken from the ASPCAP DR12 output. Figure 4.32 shows a comparison between the results of the STARPANDA and ASPCAP pipelines, plotting the difference between the two as a function of the ASPCAP  $[\text{Al}/\text{M}]$ . The mean difference between the 2 pipelines is 0.132 dex (with STARPANDA returning slightly smaller abundances than ASPCAP on average), while the standard deviation is 0.156 dex. While the bulk of the results do seem consistent with the ASPCAP results (allowing for the offset), there is a very large amount of scatter.

So, in order to try to understand these results further we have looked at how the STARPANDA  $[\text{Al}/\text{M}]$  values change as a function of various ASPCAP parameters, which were used to derive both Al abundances. Figure 4.33 shows comparisons of the Al abundances produced by both pipelines plotted as functions of the ASPCAP  $T_{\text{eff}}$ ,  $[\text{Fe}/\text{H}]$ ,  $\log g$ ,  $[\text{C}/\text{Fe}]$ ,  $[\text{N}/\text{Fe}]$  &  $[\text{O}/\text{Fe}]$  - in each case the STARPANDA  $[\text{Al}/\text{Fe}]$  distribution is plotted above that of the ASPCAP  $[\text{Al}/\text{Fe}]$ . In all six cases the precision of both sets of results are comparable, with a slight offset; the  $[\text{Al}/\text{Fe}]-[\text{Fe}/\text{H}]$  relations show a slight anti-correlation in the STARPANDA results, which is not seen in the ASPCAP results. This may explain the anti-correlation apparent in Figure 4.32.

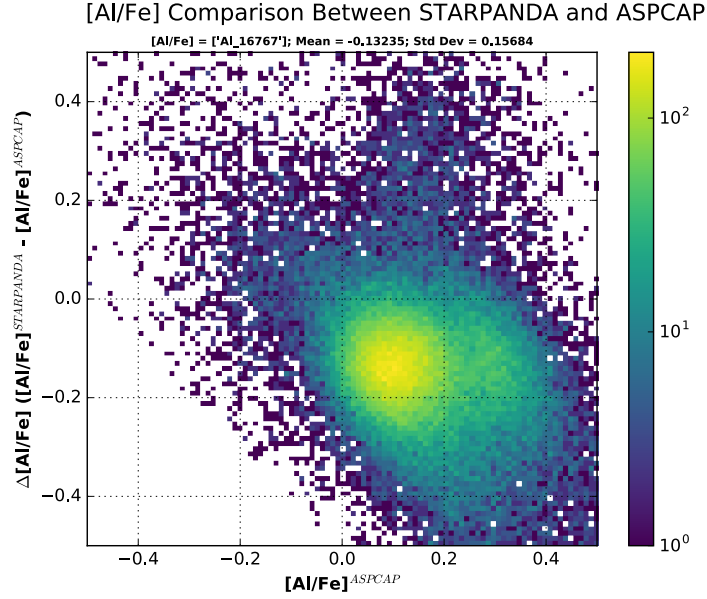


Figure 4.32: The results of an elemental abundance analysis of Al from STARPANDA with the DR12 results from ASPCAP. Plotted on the y-axis is the difference between the STARPANDA and ASPCAP values, while the x-axis shows the ASPCAP results. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

## 4.5.2 DR14 Analysis

STARPANDA took 531 seconds to analyse 63795 stars and successfully derived Al abundances for 63198 stars - achieving a  $\sim 99\%$  success rate in  $\sim 0.008$  sec/star. Once again repeating the analysis with DR14 stellar parameters and CNO abundances produces Figures 4.34 and 4.35. Now the mean difference between the results of the pipelines is 0.122 dex, while the standard deviation is 0.173 dex. These figures are comparable to those achieved using DR12 stellar parameters and CNO abundances, looking at the plots and comparing with the ones above (Figures 4.32 and 4.33 respectively). The main difference between the two sets of plots is that that both pipelines show a decrease in  $[Al/Fe]$  in all six plots using DR14 stellar parameters and CNO abundances.

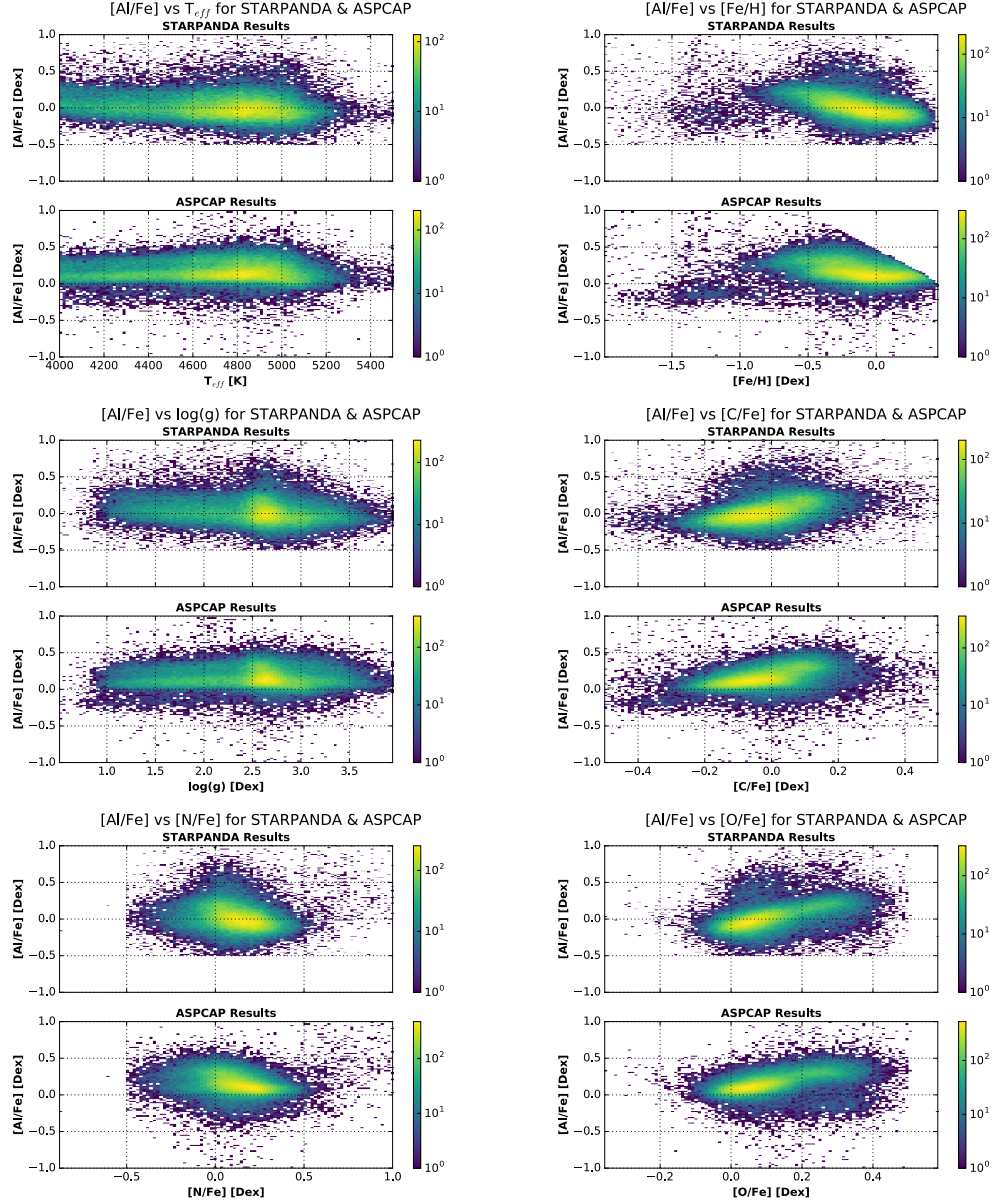


Figure 4.33: The results of an elemental abundance analysis of Al from STARPANDA with the DR12 results from ASPCAP as a function of the 6 parameters. Each of the 6 pairs of plots have the STARPANDA plot above the ASPCAP plot, showing the Al abundance distribution as a function of the ASPCAP  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[C/Fe]$ ,  $[N/Fe]$  &  $[O/Fe]$ . The x-axis is shared between each pair of subplots. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.



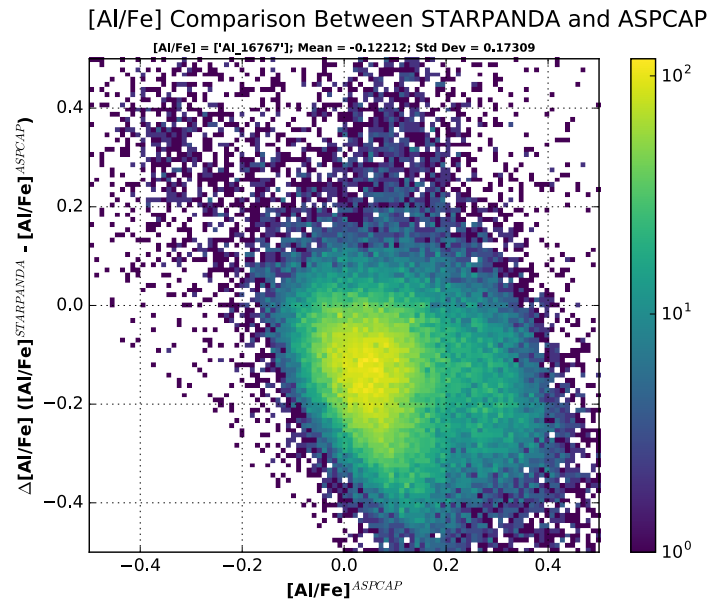


Figure 4.34: The results of an elemental abundance analysis of Al from STARPANDA with the DR14 results from ASPCAP. Plotted on the y-axis is the difference between the STARPANDA and ASPCAP values, while the x-axis shows the ASPCAP results. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

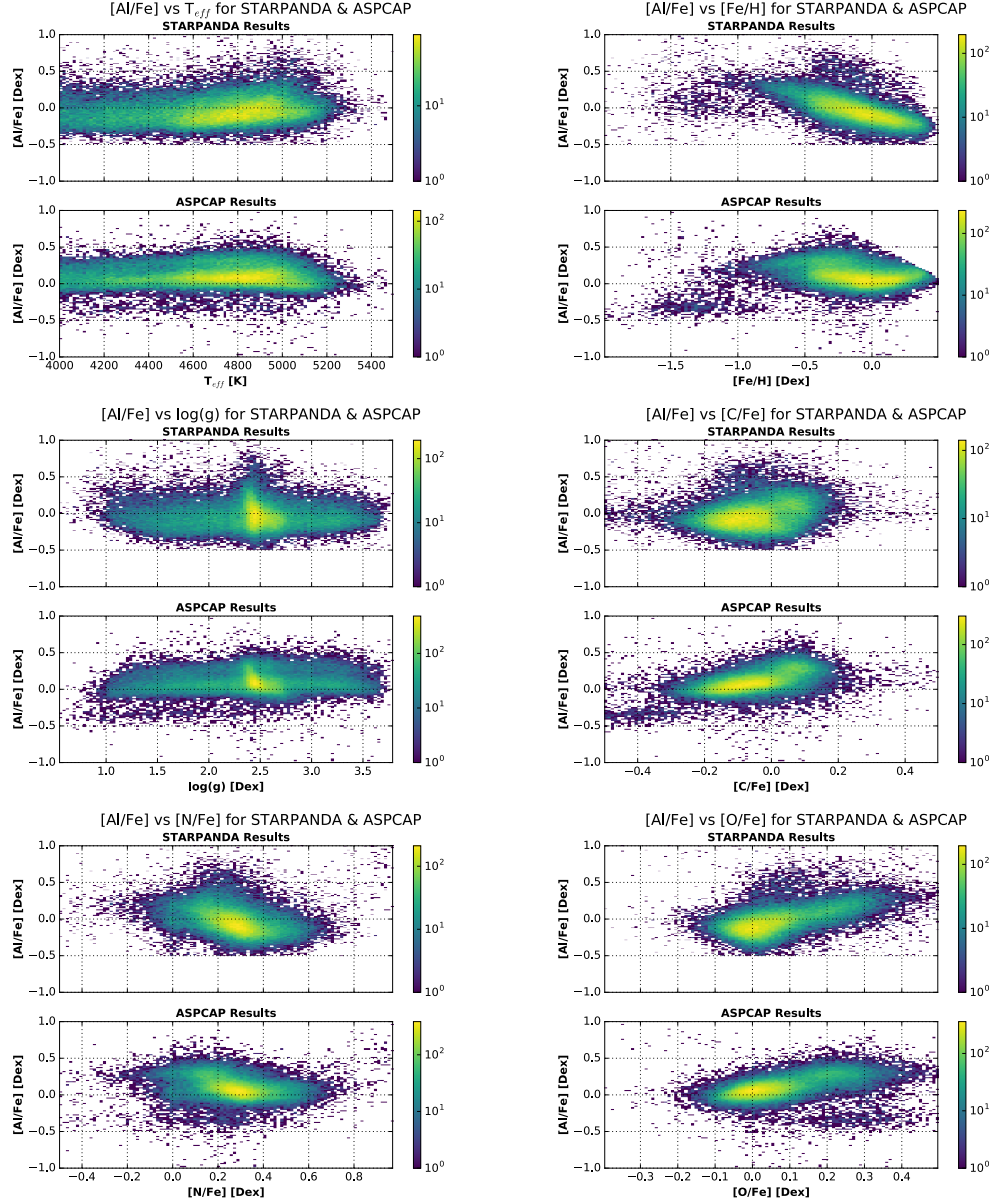


Figure 4.35: The results of an elemental abundance analysis of Al from STARPANDA with the DR14 results from ASPCAP as a function of the 6 parameters. Each of the 6 pairs of plots have the STARPANDA plot above the ASPCAP plot, showing the Al abundance distribution as a function of the ASPCAP  $T_{eff}$ ,  $[Fe/H]$ ,  $\log g$ ,  $[C/Fe]$ ,  $[N/Fe]$  &  $[O/Fe]$ . The x-axis is shared between each pair of subplots. The density of points is represented by the colour, shown in the colour bar to the right of the plot; we show the log of the density.

## Chapter 5

### Conclusions

Our initial goals were to develop a pipeline which would be able to quickly and accurately derive stellar parameters and chemical abundances from the APOGEE spectra. We were aiming to beat the ASPCAP pipeline in terms of speed, while at least matching it in terms of accuracy and precision. On the first requirement we have exceeded our expectations - we had talked about developing a code which would be able to derive stellar parameters and CNO abundances, with errors, in approximately one second per star. After a considerable initial investment of time, we have developed a pipeline which is over ten times faster (see Section 3.4.3 and Chapter 4 for runtime examples). With some further development (see Chapter 6.1), it should be possible to achieve even faster results. All of this has been done on a rather low-powered laptop computer.

On the second requirement we have made clear progress. The best results obtained so far are when STARPANDA imports the stellar parameters obtained by ASPCAP in DR14 and then derives CNO abundances using the lines listed in Table 4.5 - the results can be seen in Figures 4.29 - 4.31 (in Section 4.4). In this case, we obtain results strikingly similar to those obtained by ASPCAP, in a tiny fraction of the time. With more work, we expect to be able to match the accuracy of ASPCAP CNO abundances and may even be able to improve upon them. If we use STARPANDA to obtain all 6 parameters, then the results are not as good as those obtained by ASPCAP (see Section

4.3), This is understandable as our tracers for effective temperature,  $[\text{Fe}/\text{H}]$  and surface gravity may need to be improved; given, then, the nature of our pipeline and the degeneracies between parameters, this leads to less reliable CNO abundances. Once we have been able to identify a good spectroscopic tracer for effective temperature and a better tracers of  $[\text{Fe}/\text{H}]$  and surface gravity we expect that the results will be, at least, comparable to those produced by ASPCAP (see Chapter 6.2 for more detail). Regarding the results obtained for Al abundances, this was not something that we planned for this stage of the pipeline; we had aimed initially for only deriving stellar parameters and CNO abundances, with elemental abundances being added later. However, during the development of the code we devised a method that would allow us to do both CNO and elemental abundances if we wished. Al was chosen as being a relatively simple element to start on and with our decision to calculate synthetic spectra with varying Al abundances (a procedure not adopted by APOGEE), we thought that we would be able to exceed the accuracy of the ASPCAP results. Looking to the future, we are all set to add additional elements and we are actively discussing which are the best prospects and which will be the most beneficial for APOGEE to have a second opinion on.

Finally, thanks to the way the pipeline has been developed, we now have a tool which is completely project, wavelength and model independent. That is to say, there is nothing hard-coded in the software which requires the spectra to be from APOGEE, to be in the infrared or to use the models currently employed. All that is currently required is that you have a set of observed EQWs and a set of model EQWs at given stellar parameters and elemental abundances that you wish to compare. As long as there is a line or combination of lines in the spectra which can be shown to correlate to a specific stellar parameter or elemental abundance, then STARPANDA can derive a value and calculate the errors. There is no need to train the software and there is no ambiguity in where the final parameter values may have come from - there is a clear path from the physics of the stellar atmosphere, to EQW of a line, to the final derived parameter and parameter error values.

This has been an ambitious project and great progress has been made in achieving our initial goals. In some cases we have exceeded what we set out to achieve and we expect that once it is released, that we will have a tool which will be of benefit to both the APOGEE collaboration and the wider astronomical community.

# Chapter 6

## Future Work

Construction of a general spectroscopic analysis pipeline which can be easily utilised and can be run on even low-powered computing hardware is a long-term project which will continue to be developed for some time to come. While the initial goals have largely been met, there have been numerous ideas which have arisen during this work, that were put to one side and marked as future goals.

In this chapter we briefly discuss some of the work that has been started, but has not yet reached a stage where results have been obtained and also some ideas which we would like to see developed in future. So, in Section 6.1, we outline several ways in which we feel the code could be developed further, while in Section 6.2 we present some of the work which can be done with the results obtained by STARPANDA.

### 6.1 Future Pipeline Development

There are two obvious and fairly straightforward ways in which the STARPANDA code could be developed and improved. First off, a graphical user interface (GUI) would make the code much easier to run and would also allow several additional tools

to be built into the overall package. Work has already started on developing a GUI, but this is at a very early stage. The idea would be to have an interface which is able to merge the STARPANDA code together with the code developed by Mackereth (which measures the EQW of lines in the observed and synthetic spectra), and also, adds basic plotting functionality (to allow a quick first check of the derived parameter values for example).

Secondly, parallelisation of the code would offer a dramatic increase in runtime performance. As the code is written, once the observed and model data are read in, a loop is started which goes over all of the input objects and derives the desired values. It is conceptually simple to split the input files into multiple batches and process each on a separate CPU. The difficulty is in dealing with the logfile which is written to continually while STARPANDA is running and in merging the output arrays which store the results of each analysis sequentially. These would need to be split into multiple files and then held separately, before being merged prior to the output files being written to storage; a complicating factor would be in putting a system in place to deal with these files which could handle the unexpected early termination of the code. This requires some thought on how best to structure the code while dealing with parallelisation.

In addition to the development of a GUI and parallelisation of the code, there are several other ideas which should be explored further:

- *Code Optimisation* - The code, while quick, could be made quicker still by utilising the design of the Python language. A simple example is in the use of function references, which are evaluated at each call, and could be replaced by explicit definitions. This would lead to a small time savings in each case, but in the whole code run, could lead to significant runtime improvement.
- *Scripting* - The code currently utilises a config file which holds all the user-defined variables and settings. If the user wished to run the code multiple times

with minor changes between each run, they would need to edit this file between each run. It would be nice to implement a facility to allow the config file to be specified at runtime, so that multiple config files could be used to hold the settings for each unique run, e.g. testing the results from different lines, with different code output filenames for each run.

In addition to these it would be useful to investigate the interpolation methods further both to run additional tests and also to explore other methods to see if there were faster or more accurate options available. One such further test that we have discussed running is using the results plotted in Section 3.4.1 on Testing the **RGI** method. We could take the FWHM from each of the plots and calculate the variation in parameter value in different regions of model parameter space. This would give us a better idea of the errors stemming from the use of **RGI**.

## 6.2 Future Science

In order to further verify and improve on the results already obtained by STARPANDA, it will be necessary to compare the parameter and abundance values we derive to those of more than just ASPCAP. This has already started, but it is still at a very early stage. Starting with work done by Mészáros et al. (2015), we are comparing the *Al* values derived by STARPANDA (utilising both DR12 and DR14 input parameters) against those derived by Mészáros using theoretical isochrones and also against the literature values they assembled (see Tables 2 and 5 of Mészáros et al., 2015). We then narrow in on individual clusters and compare the 4 sets of results directly (cutting the STARPANDA and ASPCAP results to just those stars in the Mészáros isochrone derived set and those in the literature set). The results of this work should then help us in understanding the differences observed between the bulk results of STARPANDA and ASPCAP as shown in Chapter 4.5.



As mentioned previously, STARPANDA utilises mini-grids of synthetic spectra varying individual elemental abundances. This is not something that ASPCAP currently includes, and so we should be in a position to improve on the results obtained by ASPCAP for all 15 elements. This will allow us to not only work on improving the results reported in the APOGEE data releases, but will also give us a better idea of the chemical compositions of all the stars targeted by APOGEE. This, in turn, opens up the possibility of using these improved results for work on galactic archaeology and chemical tagging.

# Appendix A

## How to use STARPANDA

### A.1 Download & Installation

STARPANDA is not currently publicly available, but will be available once work on it has been completed and the code fully tested to ensure that it is reliable. If you wish you use the code prior to its release, please either email the author or contact Dr Riccardo Schiavon (R.P.Schiavon@ljmu.ac.uk).

The STARPANDA code consists of the following files:

- *starpac.py* - Main code - handles data input & output
- *sp\_params.py* - Derives Stellar Parameters & CNO Abundances
- *sp\_abunds.py* - Derives Elemental Abundances
- *sp\_plot.py* - Plotting routines
- *sp\_config.py* - User editable configuration file
- *sp\_readme.txt* - Readme text file

These should be kept in the same directory, while the Input and Output directories specified in the *sp\_config.py* file can be located anywhere. The code is then invoked by running “python starpac.py” from the directory containing the code files. More information on the items contained in the *sp\_config.py* file, can be found in the *sp\_readme.txt* file.

## A.2 STARPANDA Readme File

This is the readme file for the STARPANDA code

```
Created By:      Rob Williams, ARI, Liverpool John Moores University
                  r.a.williams@2013.ljmu.ac.uk
Creation Date:   28 August 2014
Last Modified:   23 January 2017
```

```
#####
### Code Files                                     ###
#####
starpac.py (Main Code)
    Main code. Called by user and responsible for loading and saving data. Reads
    user preferences from sp_config and sp_indices. Calls sp_params (if user
    requires), sp_abunds (if user requires), sp_modparams (if user requires).
sp_config.py
    Contains user specified data locations for input and output, together with
    initial parameter values, index names and tolerance values used while
    analysing the stellar parameters.
sp_params.py
    Code responsible for the derivation of stellar parameters. Called by starpac
    (if the user specifies it) for each object in the input files.
sp_abunds.py
    Code responsible for the derivation of stellar elemental abundances. Called
    by starpac (if the user specifies it) for each object in the input files.
sp_plot.py
    Code called, if specified, either during each iteration or at the end of the
    analysis of each object which plots the grids and curves used in
    interpolation of the parameters & abundances

### Required Directories
Input, Output

### Required Input Files in Input Directory
FITS file containing EQWs of lines from synthetic spectra
FITS file containing EQWs of line from observed spectra

### Python Version
```

## 2.7.10 (Tested)

```
### Modules Required by STARPAC
```

```
numpy, scipy, scipy.interpolate (RegularGridInterpolator, griddata & interp1d),
astrophy (io.fits & table), matplotlib, csv, datetime, time
```

```
#####
### References #####
#####
n/a
```

```
#####
### User Input #####
#####
# sp_config.py
```

Edit the sp\_config.py file to change the input & output file locations to suit. A

```
# Control options for what analysis code runs
```

```
param_analysis = Does the code analyse stellar parameters & CNO abundances
                  for all input objects? (True/False)
```

```
cno_analysis    = Does the code analyse CNO abundances (without the stellar
                  parameters) for all input objects? (True/False)
```

```
abund_analysis  = Does the code analyse elemental abundances for all input
                  objects? (True/False)
```

```
# Control options for what plots the code produces
```

```
full_grid_plot = Plot an initial effective temperature & metallicity grid
                  (assuming starting values of log g & CNO)?
                  Only works when using param_analysis (True/False)
```

```
histo_plot     = Plot histograms of the results of parameter analysis using
                  flag values and iteration count from each object analysis?
                  Only works when using param_analysis (True/False)
```

```
comp_plot      = Plot comparisons between (True/False)
```

```
debug_plots    = Plot grid and curve interpolations for each parameter/CNO
                  abundance at each iteration? (True/False) n.b. this will
                  produce a large number of output files for each target and
                  significantly slow the code
```

```
final_plots    = Plot grid and curve interpolations for each parameter/CNO
                  abundance, but only at the final iteration? (True/False)
```

```
# Input file location & names
```

```
input_dataloc  = Full path of the location of the input datafiles ('String')
obs_datafile   = Name of the FITS file containing the observed data
                  ('String')
```

```
model_datafile = Name of the FITS file containing the synthetic data
                  ('String')
```

```
paramcno_datafile = Name of the FITS file containing stellar parameters & CNO
                    abundances for use in abund_analysis, if not previously
                    deriving these ('String')
```

```
param_datafile  = Name of the FITS file containing stellar parameters for use
                    in CNO_analysis ('String')
```

```
abunds_datafile = Name of the FITS file containing the synthetic abundance
                    data for use in abund_analysis ('String')
```

```

# Output file location & names
output_dataloc = Full path location of the output data ('String')
output_name     = Base name to use in creating output files - this is added to
                  to create the filenames of each output file ('String')
output_plotformat = File type to use for output plots (not all plots use this),
                  either 'pdf', 'eps' or 'jpg' ('String')

# Model Data Parameter, CNO & Elemental Abundance Column Names
model_temp_index = Name of the column in the model_datafile containing the
                  effective temperature EQWs ('String')
model_feh_index  = Name of the column in the model_datafile containing the
                  metallicity EQWs ('String')
model_logg_index = Name of the column in the model_datafile containing the
                  log(g) EQWs ('String')
model_cm_index   = Name of the column in the model_datafile containing the
                  [C/M] abundance EQWs ('String')
model_nm_index   = Name of the column in the model_datafile containing the
                  [N/M] abundance EQWs ('String')
model_om_index   = Name of the column in the model_datafile containing the
                  [Alpha/M] abundance EQWs ('String')
model_abund_labels = Name of the columns in the abunds_datafile containing the
                  elemental abundance EQWs for each element to be analysed
                  during abund_analysis (['String Array'])

# Measured Lines for Stellar Parameters, CNO & Elemental Abundances
temp_label       = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for effective
                  temperature (['String Array'])
feh_label        = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for metallicity
                  (['String Array'])
logg_label       = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for log(g)
                  (['String Array'])
cm_label         = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for [C/M]
                  (['String Array'])
nm_label         = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for [N/M]
                  (['String Array'])
om_label         = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for [Alpha/M]
                  (['String Array'])
abund_labels     = Name(s) of the column(s) in the obs_datafile containing the
                  EQW measurement(s) of the line(s) to be used for elemental
                  abundances (['String Array']) n.b. each entry can itself be
                  an array of strings

# Analysis Order
sp_order         = Order for conducting parameter & CNO abundance analysis
                  effective temperature and metallicity will always be first,
                  so this applies to just log(g) & CNO (['String Array'])
                  e.g. ['logg', 'C', 'O', 'N']
cno_order        = Order for conducting CNO abundance analysis (['String

```

```

        Array')) e.g. ['C', 'O', 'N']

# Input Stellar Parameter & CNO Labels
io_temp_label   = 'TEFF'
io_feh_label    = 'Fe_H'
io_logg_label   = 'LOGG'
io_cm_label     = 'C_Fe'
io_nm_label     = 'N_Fe'
io_om_label     = 'O_Fe'

# Starting parameter & CNO values
model_init_logg = Initial value of log(g), used in first iteration of
                    param_analysis (Float)
model_init_cm   = Initial value of [C/M], used in first iteration of
                    param_analysis (Float)
model_init_nm   = Initial value of [N/M], used in first iteration of
                    param_analysis (Float)
model_init_om   = Initial value of [Alpha/M], used in first iteration of
                    param_analysis (Float)

# Iteration count & tolerances
iter_max        = Maximum number of times the code is allowed to iterate to
                    find parameters (Int)
tol_temp        = Convergence limit for Temperature in iterations (Float)
tol_feh         = Convergence limit for [Fe/H] in iterations (Float)
tol_logg        = Convergence limit for log(g) in iterations (Float)
tol_cm          = Convergence limit for [C/M] in iterations (Float)
tol_nm          = Convergence limit for [N/M] in iterations (Float)
tol_om          = Convergence limit for [Alpha/M] in iterations (Float)

#####
### Flag Values                                     ###
#####
# Parameter Flag Values
0 - Value is Good
1 - Value is Bad

# Error Flag Values
0 - Negative & Positive Error Values are Good
1 - Negative Error Value is Bad; Positive Error Value is Good
2 - Negative Error Value is Good; Positive Error Value is Bad
3 - Negative & Positive Error Values are Bad

# Iteration Flag Values
0 - Analysis converged
1 - Analysis reached iteration limit before convergence
2 - Analysis failed without converging

# Failure Flag Values
0 - Success
1 - Failed due to reaching convergence limit
2 - Failed as unable to find initial Temperature and [Fe/H]
3 - Failed due to divergence whilst trying to find log(g)
4 - Failed due to divergence whilst trying to find [Alpha/M]

```

- 5 - Failed due to divergence whilst trying to find [C/M]  
 6 - Failed due to divergence whilst trying to find [N/M]

```
#####
### Code Breakdown                                     ###
#####
```

- (1) Code Starts (run "python starpac.py" from the command line)
- (2) starpac:
  - (a) Start time is saved and the logfile is created
  - (b) Check if parameter & CNO abundance analysis is required
    - (1) If True, write analysis run details to logfile; if False, make a note to logfile
    - (2) Read in observed data, model data and create output data structure
    - (3) Create interpolated function from the model data for later use (using `scipy.interpolate.RegularGridInterpolator`)
    - (4) Check if full initial grid or comparison plots are required
    - (5) Loop over all objects in the observed datafile and call `sp_params.Find_Params` to derive stellar parameters and CNO abundances
    - (6) Update logfile with runtime and details of successes/failures
    - (7) Write out FITS file with results of stellar parameters & CNO abundances analysis
    - (8) Check if results histograms are required
  - (c) Check if CNO abundance analysis is required and if parameter analysis has not already been run
    - (1) If both are True, then write analysis run details to logfile; if the former is False, make a note to logfile; if the latter is False, make a note to logfile
    - (2) Read in observed data, model data and create output data structure
    - (3) Read in stellar parameter values
    - (4) Create interpolated function from the model data for later use (using `scipy.interpolate.RegularGridInterpolator`)
    - (5) Loop over all objects in the observed datafile and call `sp_params.Find_Params` to derive CNO abundances
    - (6) Update logfile with runtime and details of successes/failures
    - (7) Write out FITS file with stellar parameters (taken from input file) & results of CNO abundances analysis
    - (8) Check if results histograms are required
  - (d) Check if elemental abundance analysis is required
    - (1) If True, write analysis run details to logfile; if False, make a note to logfile
    - (2) Read in observed data
    - (3) Check if parameter analysis has been previously run; if True, use these stellar parameters & CNO abundances, otherwise read in data from file
    - (4) Loop over all elements given in `abund_labels`
      - (a) Read in model data for the selected element
      - (b) Create interpolated function from the model data for later use (using `scipy.interpolate.RegularGridInterpolator`)
      - (c) Loop over all objects in the observed datafile and call

- ```

        sp_abunds.Abund_Analysis to derive elemental abundance
(5)      Update logfile with runtime and details of successes/failures
(6)      Write out FITS file with stellar parameters & CNO abundances
         (taken from input file) plus the results of elemental abundance
         analysis

(3a)     sp_params.Find_Params:
(a)      Save Star ID to output structure
(b)      Save initial values of log(g) (null values for Temperature and
         [Fe/H]) & CNO abundances for checking convergence
(c)      Derive the temperature & [Fe/H]
(1)      Assuming initial values for log(g) & CNO (or taking those
         derived in previous iterations), interpolate a grid from the
         function created by starpac
(2)      Interpolate (using scipy.interpolate.griddata) the Temperature &
         [Fe/H] of the target using the new grid
(3)      Check if debug_plots are required; if so, plot the grid with the
         target point and interpolated Temperature & [Fe/H] values, plus
         current log(g) & CNO values
(4)      Check if the interpolated Temperature & [Fe/H] values are within
         model parameter space; if not, end analysis
(d)      Derive log(g), [C/M], [N/M] & [O/M] in the order specified by
         sp_order
(1)      Taking previously derived values for Temperature & [Fe/H], with
         either initial or previously derived values for remaining
         parameters/abundances, interpolate the variation in model
         parameter/abundance with variation in EQW of line(s) being used
         to trace parameter/abundance using the function created by
         starpac
(2)      Interpolate this curve (using scipy.interpolate.interpld) to
         derive the parameter/abundance value
(3)      Check if debug_plots are required; if so, plot the curve with
         the interpolated curve, nearest node curve and measured target
         line
(4)      Check if the derived value is within model parameter space; if
         not, end analysis
(e)      Check that iteration is required (not set to zero), that the maximum
         number of iterations hasn't been reached and that convergence has
         not been reached; if all are true return to (c) and re-derive new
         values using values from this iteration as starting values
(f)      Check if final_plots are required; if so, plot grid and curves for
         final iteration values
(g)      If analysis was successful, derive errors on all
         parameters/abundances for both positive and negative errors on the
         measured target line EQWs
(h)      Save stellar parameter and CNO abundance values, flags, error
         values & counters to output data structure (using null values if no
         value could be derived)
(i)      Update logfile with details of this analysis, including values of
         parameters/abundances derived at each iteration, counters, flags,
         error values and comments
(j)      Return to starpac

(3b)     sp_params.Find_CNO:
(a)      Save Star ID to output structure

```



- (b) Save initial values of CNO abundances for checking convergence
  - (c) Derive [C/M], [N/M] & [O/M] in the order specified by cno\_order
    - (1) Taking the stellar parameter values supplied, with either initial or previously derived values for abundances, interpolate the variation in model parameter/abundance with variation in EQW of line(s) being used to trace parameter/abundance using the function created by starpac
    - (2) Interpolate this curve (using `scipy.interpolate.interpld`) to derive the parameter/abundance value
    - (3) Check if `debug_plots` are required; if so, plot the curve with the interpolated curve, nearest node curve and measured target line
    - (4) Check if the derived value is within model parameter space; if not, end analysis
  - (d) Check that iteration is required (not set to zero), that the maximum number of iterations hasn't been reached and that convergence has not been reached; if all are true return to (c) and re-derive new values using values from this iteration as starting values
  - (e) Check if `final_plots` are required; if so, plot curves for final iteration values
  - (f) If analysis was successful, derive errors on all abundances for both positive and negative errors on the measured target line EQWs
  - (g) Save stellar parameter and CNO abundance values, flags, error values & counters to output data structure (using null values if no value could be derived)
  - (h) Update logfile with details of this analysis, including values of parameters, abundances derived at each iteration, counters, flags, error values and comments
  - (i) Return to starpac
- (3c) `sp_abunds.Abund_Analysis:`
- (a) Taking the stellar parameter & CNO abundance values supplied, interpolate the variation in model elemental abundance with variation in EQW of line(s) being used to trace abundance using the function created by starpac
  - (b) Interpolate this curve (using `scipy.interpolate.interpld`) to derive the elemental abundance value
  - (c) Check if the derived value is within model parameter space & save values to output data structure
  - (d) Derive errors on the abundance for both positive and negative errors on the measured target line EQWs
  - (e) Check if `debug_plots` are required; if so, plot the curve with the interpolated curve, nearest node curve and measured target line
- (4) `starpac:`
- (a) Calculate the run time of the code
  - (b) Prints a code completion notice containing the runtime of the code and thenumber of objects analysed to the logfile
  - (c) Print code completion notice to screen
- (5) Code Ends

#####

## **Appendix B**

# **Stellar Atmospheres & Line Formation**

STARPANDA works by analysing stellar spectra through comparison with a grid of synthetic spectra which have been calculated using state-of-the-art model stellar atmospheres and a comprehensive line list. Indices are measured in both sets of spectra, and these relate to transition lines from different chemical species in the stellar atmosphere. It is therefore useful to understand some details on the nature of model stellar atmospheres, how lines are formed in spectra and why lines can be used to probe stellar parameters and elemental abundances. This section offers a brief, qualitative description of these topics, starting with Model Stellar Atmospheres in Section B.1, then Line Formation in Section B.2 and finally, Lines as Parameter Traces in Section B.3. There are many papers and books where these subjects are covered in more depth, for example see Allende Prieto (2016), Gray (2005) and references therein.

## **B.1 Model Stellar Atmospheres**

The development of full, detailed models of the atmospheres of stars is an incredibly complex and computational expensive challenge. Thankfully, we have available simpler models which allow us to derive the details we need for our study of the formation and evolution of the Milky Way. The models we make use of through this work are 1-dimensional and plane parallel, in hydrostatic and local thermodynamic equilibrium and ignoring the effects of magnetic fields. Much progress is being made on developing models which incorporate higher dimensions, magnetic fields or do not assume local thermodynamic equilibrium, however those developments are beyond the scope of this appendix.

For our work, we only need to consider the Photosphere, as it is here where the absorption lines that we observed in our spectra are produced. The photosphere sits below the atmospheric layers (the Chromosphere and Corona) and above the layers dominated by convection and radiation. A model photosphere is a tabulation of all the parameters necessary for energy transportation, with each line entry giving these values for a discrete layer. They are calculated by simultaneously solving the equations of radiative transfer, hydrostatic equilibrium and statistical equilibrium (for which continuum and line opacities must be calculated), and assuming a plane-parallel geometry, local thermodynamic equilibrium and consistency for both radiative and convective flux (for the latter this is done using the mixing length approximation). Starting at the bottom of the photosphere, this is done for each layer and finally yields the model photosphere file. As a minimum, the final model photosphere should contain the values for pressure and temperature at each layer, but additional parameters can be included, such as density, electron pressure, etc.

The resulting model is then a table of parameter values describing the state of the photosphere at each layer. The model tells us nothing, however, about the absorption lines that we expect to see in a spectrum for a star with this structure. So, to generate a synthetic spectrum, more information is required.

## B.2 Line Formation

In real stars, lines caused by the absorption of photons coming from hotter, deeper regions of the stellar atmosphere are intrinsically found at very specific wavelengths given by the energy of the transition in question. However, when observing stellar spectra, we do not see such narrow and well-defined absorption lines. Instead what we see is a function of 4 different effects which broaden the line. The first three are caused by processes within the stellar atmosphere, while the latter is due to the instrument used to measure the flux from the target star(s).

The first broadening effect is intrinsic broadening caused by the uncertainty principle. The second broadening effect is caused by the movement of the absorber, which leads to small doppler shifts in the wavelength observed at the detectors. This broadening is temperature related and called Doppler Broadening. The third is caused by collisions and interactions between moving atoms and ions; such interactions can alter the precise distance between energy levels leading to a range of wavelengths given from a single electronic transition. This form of broadening is pressure related and usually called Pressure Broadening. Finally there is the resolution of measuring instrument, which then further broadens the line. Where there are many lines close together, these broadening effects can lead to blending of several different lines from different atomic and molecular species, especially if the resolution of the instrument is low.

The EQW of a line is then defined from a normalised spectrum, as the width of a rectangle, of height equal to the continuum level, which has the same area as that occupied by the absorption line. The measured value is given in the units used by the wavelength scale, e.g. Å or nm. This is what we have then used in this work.

When starting from model atmospheres, the generation of synthetic spectra requires not only the details of the model photosphere, but also information on the lines that may be present in the atmosphere. This line list (both ASPCAP and STARPANDA use the line list described in Shetrone et al., 2015) comprising (in our case) 35 columns, which contains data on the wavelength, the probability of a transition occurring, the chemical species involved, the energy levels of both the start and end point, etc. Combining the model atmosphere and line list data, it is then possible to compute synthetic spectra by solving the equations of radiation transfer and calculating the continuum opacity (which in the H-band is mainly due to  $H^-$  ions). Finally, since the model atmospheres that have been used here are 1D in nature, the broadening that is seen in observed spectra is parameterised by two values, the micro and macro turbulence, which can be varied to match the shape of the observed lines.

### B.3 Lines as Parameter Tracers

While it is easy to see that the strength of a line in a spectrum is related to the abundance of the species that gives rise to that line, with increasing abundance giving rise to increasing EQW values (at least up to the saturation point). However, the strength of most lines are also dependant on the stellar parameters. How, then, can stellar parameters be derived from measurements of spectral features?

Analysis usually starts by attempting to determine the Effective Temperature. Photometric colour can be used to give the  $T_{eff}$ , but it can also be found spectroscopically using pairs of lines. There are many metals which show different sensitivities to changes in temperature for different lines. Thus, taking two lines from the same metal species, preferably lines that are close together in wavelength to minimise any issues with the normalisation of the spectrum, and computing the ratio, should eliminate the dependance on the metal abundance, but leave the dependance on temperature. This is what we attempted to do using two Fe lines (FeI\_16230 & FeI\_16522). Surface Gravity can be found in a similar manner, but using pairs of lines from the same metal, but with different ionisation states, which have different sensitivities to pressure and therefore  $\log g$  (though this is harder as the sensitivity of lines to changes in  $\log g$  values is much smaller). An alternative to using pairs of lines for both  $T_{eff}$  and  $\log g$  is to use the wings of lines and attempt to match observations to model predictions; for  $T_{eff}$ , Balmer lines can be used, while for  $\log g$ , strong metal or molecular lines can be used. Finally, the overall metallicity can be taken from measurements of the Fe abundance.

Given, then, that there is some dependance on all lines to the stellar parameters and CNO abundances, and that we are using molecular lines for the CNO abundances (CO, CN and OH), it is necessary to derive the stellar parameters and CNO abundances iteratively before calculating the elemental abundances using either individual metal lines or a number of lines in combination. This forms the basis for the methodology utilised by STARPANDA.

# Bibliography

Allende Prieto, C., Beers, T. C., Wilhelm, R., et al. 2006, *ApJ*, 636, 804

Allende Prieto, C., Fernández-Alvar, E., Schlesinger, K. J., et al. 2014, *A&A*, 568, A7

Allende Prieto, C. 2016, *Living Reviews in Solar Physics*, 13, 1

Alam, S., Albareti, F. D., Allende Prieto, C., et al. 2015, *ApJS*, 219, 12

Alvarez, R., & Plez, B. 1998, *A&A*, 330, 1109

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, *A&A*, 558, A33

Bensby, T., Feltzing, S., & Oey, M. S. 2014, *A&A*, 562, A71

Benson, A. J. 2010, *Physics Reports*, 495, 33

Blanton, M. R., Bershady, M. A., Abolfathi, B., et al. 2017, *AJ*, 154, 28

Bovy, J., Nidever, D. L., Rix, H.-W., et al. 2014, *ApJ*, 790, 127

Bovy, J. 2016, *ApJ*, 817, 49

Burstein, D., Faber, S. M., Gaskell, C. M., & Krumm, N. 1984, *ApJ*, 287, 586

Casey, A. R., Hogg, D. W., Ness, M., et al. 2016, *arXiv:1603.03040*

Carollo, D., Beers, T. C., Lee, Y. S., et al. 2007, *Nature*, 450, 1020

Carollo, D., Beers, T. C., Chiba, M., et al. 2010, *ApJ*, 712, 692

Cirasuolo, M., Afonso, J., Carollo, M., et al. 2014, , 9147, 91470N

Clough, R., & Tocher, J., 1965, *Proceedings of Conference on Matrix Methods in Structural Analysis*

Dalton, G., Trager, S. C., Abrams, D. C., et al. 2012, , 8446, 84460P

Deng, L.-C., Newberg, H. J., Liu, C., et al. 2012, *Research in Astronomy and Astrophysics*, 12, 735

de Jong, R. S., Barden, S. C., Bellido-Tirado, O., et al. 2016, , 9908, 99081O

de Laverny, P., Recio-Blanco, A., Worley, C. C., & Plez, B. 2012, *A&A*, 544, A126

De Silva, G. M., Freeman, K. C., Bland-Hawthorn, J., et al. 2015, *MNRAS*, 449, 2604

Eggen, O. J., Lynden-Bell, D., & Sandage, A. R. 1962, *ApJ*, 136, 748

Eisenstein, D. J., Weinberg, D. H., Agol, E., et al. 2011, *AJ*, 142, 72

ESO & Brunier, S. 2015, <http://www.eso.org/public/images/eso0932a/>

Feltzing, S., & Chiba, M. 2013, *New Astronomy Reviews*, 57, 80

Freeman, K. C. 1987, *ARA&A*, 25, 603

Freeman, K., Ness, M., Wylie-de-Boer, E., et al. 2013, *MNRAS*, 428, 3660

Fuhrmann, K. 1998, *A&A*, 338, 161

García Pérez, A. E., Allende Prieto, C., Holtzman, J. A., et al. 2016, *AJ*, 151, 144



- Gilmore, G., Wyse, R. F. G., & Kuijken, K. 1989, *ARA&A*, 27, 555
- Gilmore, G., Randich, S., Asplund, M., et al. 2012, *The Messenger*, 147, 25
- González Hernández, J. I., & Bonifacio, P. 2009, *A&A*, 497, 497
- Gorgas, J., Faber, S. M., Burstein, D., et al. 1993, *ApJS*, 86, 153
- Graves, G. J., & Schiavon, R. P. 2008, *ApJS*, 177, 446-464
- Gray, D. F. 2005, “The Observation and Analysis of Stellar Photospheres, 3rd Edition, by D.F. Gray. ISBN 0521851866. Cambridge University Press, 2005.”
- Gustafsson, B., Edvardsson, B., Eriksson, K., et al. 2008, *A&A*, 486, 951
- Heiter, U., Lind, K., Asplund, M., et al. 2015, , 90, 054010
- Holtzman, J. A., Shetrone, M., Johnson, J. A., et al. 2015, *AJ*, 150, 148
- Koesterke, L. 2009, *American Institute of Physics Conference Series*, 1171, 73
- Korn, A. J., Maraston, C., & Thomas, D. 2005, *A&A*, 438, 685
- Kos, J., Lin, J., Zwitter, T., et al. 2017, *MNRAS*, 464, 1259
- Kurucz, R. L. 1993, *Kurucz CD-ROM* (Cambridge, MA; Smithsonian Astrophysical Observatory)
- Mackereth, J. T., Bovy, J., Schiavon, R. P., et al. 2017, *MNRAS*, 471, 3057
- Majewski, S. R. 1993, *ARA&A*, 31, 575
- Majewski, S. R., Schiavon, R. P., Frinchaboy, P. M., et al. 2017, *AJ*, 154, 94
- Mészáros, S., Allende Prieto, C., Edvardsson, B., et al. 2012, *AJ*, 144, 120

- Mészáros, S., & Allende Prieto, C. 2013, MNRAS, 430, 3285
- Mészáros, S., Martell, S. L., Shetrone, M., et al. 2015, AJ, 149, 153
- NASA 2015, [https://solarsystem.nasa.gov/multimedia/display.cfm?IM\\_ID=8083](https://solarsystem.nasa.gov/multimedia/display.cfm?IM_ID=8083)
- Nelder, J. A., & Mead, R. 1965, CompJ, 7, 308
- Ness, M., Freeman, K., Athanassoula, E., et al. 2013, MNRAS, 430, 836
- Ness, M., Freeman, K., Athanassoula, E., et al. 2013, MNRAS, 432, 2092
- Ness, M., Hogg, D. W., Rix, H.-W., Ho, A. Y. Q., & Zasowski, G. 2015, ApJ, 808, 16
- Nidever, D. L., Holtzman, J. A., Allende Prieto, C., et al. 2015, AJ, 150, 173
- Pasquini, L., Avila, G., Blecha, A., et al. 2002, The Messenger, 110, 1
- Plez, B. 2012, Astrophysics Source Code Library, ascl:1205.004
- Randich, S., Gilmore, G., & Gaia-ESO Consortium 2013, The Messenger, 154, 47
- Rix, H.-W., & Bovy, J. 2013, The Astronomy and Astrophysics Review, 21, 61
- Rojas-Arriagada, A., Recio-Blanco, A., Hill, V., et al. 2014, A&A, 569, A103
- Sacco, G. G., Morbidelli, L., Franciosini, E., et al. 2014, A&A, 565, A113
- Schiavon, R. P. 2007, ApJS, 171, 146
- Searle, L., & Zinn, R. 1978, ApJ, 225, 357
- Shetrone, M., Bizyaev, D., Lawler, J. E., et al. 2015, ApJS, 221, 24
- Smiljanic, R., Korn, A. J., Bergemann, M., et al. 2014, A&A, 570, A122

Sofue, Y., Honma, M., & Omodaka, T. 2009, , 61, 227

Tody, D. 1986, , 627, 733

Tripicco, M. J., & Bell, R. A. 1995, AJ, 110, 3035

Venn, K. A., Irwin, M., Shetrone, M. D., et al. 2004, AJ, 128, 1177

Weiser, A., & Zarantonello, S. E., 1988, Mathematics of Computation, 50, 181

Wilson, J. C., Hearty, F., Skrutskie, M. F., et al. 2012, , 8446, 84460H

Worthey, G., Faber, S. M., Gonzalez, J. J., & Burstein, D. 1994, ApJS, 94, 687

Zamora, O., García-Hernández, D. A., Allende Prieto, C., et al. 2015, AJ, 149, 181

Zasowski, G., Johnson, J. A., Frinchaboy, P. M., et al. 2013, AJ, 146, 81