6-2022

Statistical Effects of Synthetic Template Mismatch with Model DEIMOS Observations

Christos Kakogiannis
Union College - Schenectady, NY

Statistical Effects of Synthetic Template Mismatch with Model DEIMOS Observations

By

Christos Kakogiannis

****

Submitted in partial fulfillment

of the requirements for

Honors in the Department of Physics and Astronomy

UNION COLLEGE

June, 2022

Abstract

KAKOGIANNIS, CHRISTOS Statistical Effects of Synthetic Template Mismatch with Model DEIMOS Observations. Department of Physics and Astronomy, June 2022

ADVISOR: Jonathan Marr

High-precision kinematics of individual stars in Milky Way satellite galaxies and globular clusters is crucial for studies of galaxy formation, cosmology, alternative gravity models, and dark matter. We tested a rigorous method that matches model DEIMOS observations of such targets with synthetic template spectra that deduces radial velocities and [Fe/H] abundances. We added noise to create a model spectrum with SNR = 30, fitted templates with different parameters to it, and calculated the best-match velocity and the goodness of the fit. The best fits occur for templates that are close in parameter space to the model spectrum, and the fits get worse the further away the templates are from the model spectrum. Within a window of $\sim \pm 1500\,\mathrm{K}$ from the model spectrum, the bias in inferred velocity increases at a rate of roughly $\lesssim \frac{0.5\,\mathrm{km\,s^{-1}}}{1000\,\mathrm{K}}$, but stays below the DEIMOS resolution. We conclude that the method does not introduce a bias to the inferred velocity larger than $\sim$ 30% of the instrumental resolution and that the uncertainty is dominated by the resolution, suggesting that the step size in our parameter space was well chosen.

Chapter 1

Introduction

1 Milky Way Satellite Galaxies and Globular Clusters

Milky Way (MW) satellite galaxies and globular clusters are of particular importance as they give insights on galaxy formation, cosmological models, gravity modification models, and indirect detection of dark matter. Being among the faintest and lowest mass stellar systems in the Universe and because of their small distances from us, they are excellent candidates to test models formulated based on observations of the Universe beyond the Local Group (LG).

LG environmental effects, such as hydrogen re-ionization, are known to bear significant impacts on the intergalactic medium, and hence on the star formation rates and history of the MW, M31, and their satellites. These effects are more pronounced in low-mass systems, such as in MW dwarf satellite galaxies, due to their shallow gravitational potentials. Several studies (Iliev et. al. (2011); Ocvirk et. al. (2014)) have concluded that LG re-ionization has occurred inside-out from the LG's own sources. However, the assumptions about the efficiency of this process could alter the conclusion. Understanding of re-ionization processes and therefore galaxy formation in the Local Group would be enhanced by dark matter simulations modeling of the mean LG re-ionization history and radiative feedback in satellite low-mass galaxies, as suggested by Dixon et. al. (2018).

Additionally, observations of MW and M31 satellites disagree with Cold Dark Matter (CDM)[1] predictions on the LG length scale, as Sawala et. al. (2016) suggest. While CDM models have been successful in predicting the anisotropy of the Microwave Background Radiation and the large scale distribution of galaxies in the Universe, they conflict with LG observations. In particular, the problems include the scarcity of luminous satellites compared to the number of dark matter (DM) substructures predicted by ΛCDM[1] models, and the confined orbits of these satellites to a thin plane as opposed to the predicted anisotropy by ΛCDM models.

Dwarf Spheroidal MW satellites provide excellent test-beds for gravity modification models, as suggested by Haghi et. al. (2016). The observed velocity dispersions in such systems requires particularly large Newtonian mass-to-light ratios beyond the limits predicted by stellar population synthesis models. Accurate knowledge of internal dynamics of MW globular clusters and satellite galaxies can help discriminate between alternative gravity and dark matter models.

Moreover, satellite galaxies of the MW and M31 can provide indirect evidence for dark matter interactions. Analysis of the total γ-ray emission of the MW Galactic Center (GC) suggests the presence of excess signal consistent with WIMP[2] annihilation models. However, point γ-ray sources, millisecond pulsars, and other compact sources in the Galactic Center are likely to contribute to this residual γ-ray signal. Scientists, therefore, wish to search for similar evidence of dark matter annihilation in "inner Galaxy" dwarf satellites (within 20° of the Galactic Center) as Abazajian et. al. (2016) point out. These satellites are particularly interesting as they typically lack any γ-ray signal. This, in turn, can constrain the parameter space of GC γ-ray emission.

[1] CDM refers to **C**old **D**ark **M**atter, whereas ΛCDM refers to Cold Dark Matter models with Dark Energy (hence Λ) considerations. The latter models are primarily constrained by observations of the Universe outside the Local Group.

[2] WIMP stands for **W**eakly **I**nteracting **M**assive **P**article. These are dark matter candidate particles.

In order to understand how these satellite systems were formed and distinguish between internal or external processes, orbital histories are needed. For that purpose, full orbital solutions from proper motion measurements from the *Gaia* Data Release 2 (Gaia Collaboration (2018)), and radial velocity measurements for a large fraction of MW satellites by Simon (2018) and Fritz et. al. (2018) have been used. In particular, accurate kinematics of individual stars are crucial for the tests described above.

Proper-motion-only observations have been previously used in determining dynamical masses of the nearest MW globular clusters. However, observational errors limit the feasibility of this method to within $\sim 20\,\mathrm{kpc}$. Currently, radial velocities provide the only way to determine dynamical masses in the lowest luminosity stellar systems. Consequently, small improvements in radial velocity measurements can result in significant improvements of the constraints for these systems.

To obtain accurate radial velocity measurements for several MW satellite galaxies and globular clusters, several researchers (Helmi et. al. (2006); Walker et. al. (2007)) have employed *multi-slit* and *multi-fiber spectroscopy*. These techniques have increased the number of available radial velocity measurements in MW satellites. At the same time, the Sloan Digital Sky Survey (SDSS) has doubled the number of known MW dwarf galaxies. Therefore, a rigorous and efficient method is needed to determine accurately radial velocities of individual stars through deep spectroscopy in an increasing number of MW satellites.

2 The DEIMOS Spectrograph

DEIMOS[3] (**D**eep **E**xtragalactic **I**maging **M**ulti-**O**bject **S**pectrograph) on the Keck II-10 m telescope has confirmed that ultra-faint dwarf galaxies of the MW are indeed dark-matter dominated systems (Simon et. al. (2007), Martin et. al. (2007)). Furthermore, it

[3]More information about DEIMOS characteristics can be found at: https://www2.keck.hawaii.edu/inst/deimos/specs.html.

has provided high-accuracy radial velocity measurements for both known and newly discovered MW satellite systems since its commission. However, these data have been processed differently by different groups. The work by Geha et. al. (2022), to which we aim to contribute, wishes to provide a *homogeneous* analysis of radial velocity measurements in MW satellite systems.

Geha et. al. (2022) have complied a catalogue of stellar spectra for $>$ 13,000 stars in 35 nearby MW satellite galaxies and 26 MW globular clusters. The data were collected with the DEIMOS spectrograph in a total of 379 unique Keck/DEIMOS pointings. The observations spanned a period exceeding 15 years, from 2003 to 2020.

For these observations, the multi-slit function of DEIMOS was used with $0.7''$ slitwidths usually and $> 4''$ slitlengths, along with an order-blocking filter. In this function, the spectra of nearly two hundred objects in a $4' \times 16'$ field of view can simultaneously be observed. The exposure time for all observations was $> 60\,\mathrm{s}$, and spectra of all objects within two effective radii of the galactic target were analyzed by Geha et. al. (2022). The uncertainties in the inferred velocities in these observations, when in the greatest resolution mode, used here (1200 l/mm), is $\sim 1 - 2\,\mathrm{km\,s^{-1}}$. In this mode, the spectral resolution is $R \sim 7000$ at 8500 Å, the spectral dispersion is 0.33 Å and the FWHM spectral resolution is 1.37 Å. The plate scale is $0.12''$ per pixel.

3 Our Method: Modeling DEIMOS Spectra

With appropriate data reduction, individual stellar radial velocities, [Fe/H] abundances, and membership probabilities for these stars can be deduced from the original catalogue. In this way, dynamical masses for stellar systems in the neighborhood of Milky Way can be estimated.

Due to the large number of DEIMOS observations to be analyzed, an automated method

needs to be created. However, one must make sure that such a method works as expected and does not introduce any bias, and test it for accuracy. The method we present here determines the radial velocities and [Fe/H] abundances by fitting the observed spectra to a collection of synthetic spectra. The velocity is found by shifting each synthetic template spectrum over a range of velocities and then fitting to the observed spectra. Our work tries to test this algorithm for efficiency, and accuracy.

However, any collection of synthetic spectra can only contain a finite number of templates generated by discrete steps in parameters. Therefore, we can classify observed spectra as different if their parameters differ by more than our chosen parameter step size. Inevitably, most observed stars will have parameters that fall between those of the synthetic spectra. These observed spectra, therefore, will be *mismatched* with different, but fairly similar, synthetic spectra (depending on the step sizes we use).

With this work, we wish to contribute to (a) determining the *maximum* step size in synthetic stellar parameters that we can use in order to probe the parameters of real observations well enough, and (b) *quantify* how well we can probe real observations with this method. In other words, we want to determine how coarse we can allow our parameter grid to be in order to maintain a desired precision. This would also optimize the computation time needed to match synthetic templates with DEIMOS observations.

For this purpose, we present here an algorithm that *models* real observations made with DEIMOS using synthetic spectra from a given library of spectra. This algorithm fits a model observation to other spectra from the library with similar parameters. Our algorithm (a) investigates whether the method used by Geha et. al. (2022) introduces a bias, (b) quantifies the precision in the velocity determination for DEIMOS observations, and (c) determines the *goodness* of the method we described above.

The outline of this work is as follows. In Chapter 2, we present how we chose and

prepared our synthetic spectra to model DEIMOS observations, and introduce the mathematical tools we used to quantify the mismatch between spectra. In Chapter 3, we present the results we obtained in our method, while we discuss their significance in Chapter 4. Lastly in Chapter 5, we draw our conclusions on the efficiency of this method, and suggest several other uses of it.

Chapter 2

Procedure

1 Selection of Library of Synthetic Spectra

We used a collection of 43 synthetic stellar spectra templates from a library generated with the high-resolution (0.01 Å) atmosphere code PHOENIX developed by Husser et. al. (2013)[1]. We created several Python programs for all spectral processing.[2] In our discussion here, as a demonstration of the process, we present the processing and results for a single synthetic observed spectrum.

Each spectrum in our collection is specified uniquely by its effective *surface temperature*, the strength of its *gravitational field* on the surface, and the ratio of the abundances of Fe to H compared to the corresponding solar abundance. We shall refer to the latter quantity as the *metallicity* of the spectrum.

These spectra span a three dimensional grid with the following ranges in parameters: $3000\,\mathrm{K} \leq T_{\mathrm{eff}} \leq 6000\,\mathrm{K}$, with steps of 500 K usually (although some templates differ by as little as 100 K); $+1.0 \leq \log\left(\frac{\mathrm{g}}{\mathrm{cm\,s^{-2}}}\right) \leq +5.0$, with steps of 0.5; and $-3.0\,\mathrm{dex} \leq [\mathrm{Fe/H}] \leq +0.0\,\mathrm{dex}$, with steps of 0.5 dex in the range $[\mathrm{Fe/H}] \in [-1.0, +0.0]$ and by 1.0 dex outside this range. Figure 2.1 shows one of the spectra in our collection.

[1] Synthetic spectra generated by the PHOENIX code can be downloaded from: https://phoenix.astro.physik.uni-goettingen.de/?page_id=15.

[2] The code is available upon request.

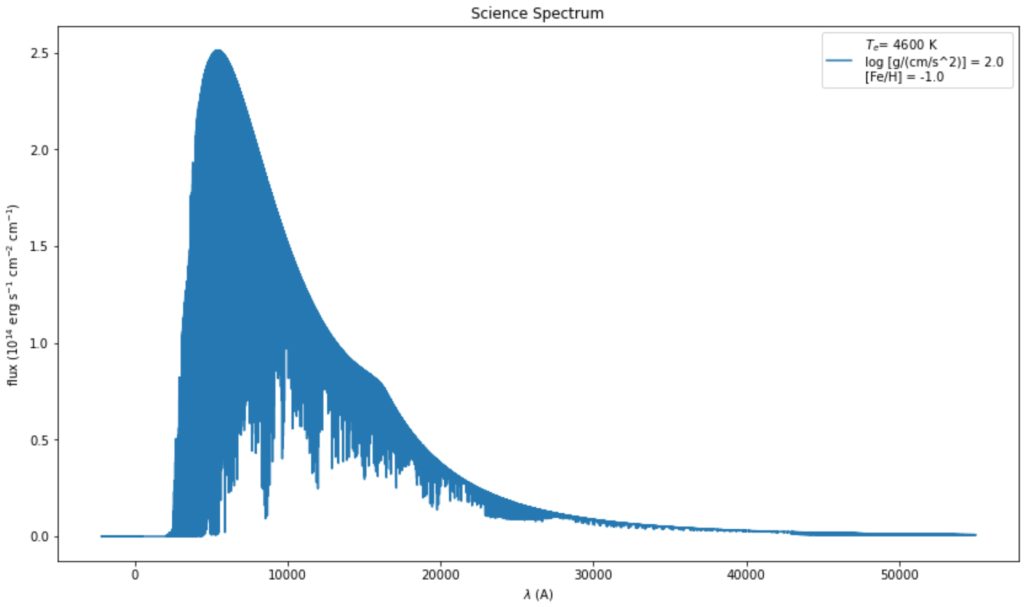


Figure 2.1: A synthetic spectrum in the library created by Husser et. al. (2013). The spectrum is from a star with $T_{\rm eff} = 4600\,{\rm K}$, $\log\left(\frac{\rm g}{\rm cm\,s^{-2}}\right) = +2.0$, and $[{\rm Fe/H}] = -1.0$ dex. Notice how the spectrum resembles that of a blackbody, and how it contains a great deal of narrow absorption lines. Other spectra in this library look similar to this one.

To simulate a DEIMOS observation, we used a synthetic spectrum from our collection and added noise to it. We shall refer to this spectrum as the *science spectrum*.

In our work, we performed two tests:

- We fitted the science spectrum to its own template spectrum by redshifting or blueshifting the template. In this way, we are able to make sure that our method works as expected. We refer to this test as the **same-same analysis**.
- The science spectrum was fitted to templates with different parameters, by shifting the template in different velocities. First, we mismatched the science spectrum with all templates in the library. This allows us to quantify the amount by which the mismatch gets worse as we move away from the science spectrum. Then, we chose the templates

closest to the science spectrum in our parameter grid in each direction:

$$\Big\{ T_{\rm eff}, \log(g), [{\rm Fe/H}] \Big\}_{\rm template} \in \Big[T_s \pm \delta T, \log(g_s) \pm \delta g, [{\rm Fe/H}]_s \pm \delta [{\rm Fe/H}] \Big]$$

This allows us to quantify how much a mismatch template disagrees with the science spectrum. We refer to this test as **template-mismatch analysis**.

For both tests, we focused on the wavelength region to which DEIMOS is sensitive, that is $\lambda \in [6200, 9500]$ Å. We then convolved all the spectra with the DEIMOS resolution, 0.5 Å, using a Gaussian smoothing kernel with relative width of $\frac{\delta\lambda_{\rm Deimos}}{\delta\lambda_{\rm Phoenix}} = \frac{0.5\,\text{Å}}{0.01\,\text{Å}} = 50$. Figure 2.2 shows how a certain absorption line in the spectrum of one of one of the synthetic spectra gets "broadened" to match the DEIMOS resolution.

2 Science and Template Spectra Preparation

When matching a real observation from DEIMOS with templates in our collection, we are interested in fitting only the velocity and so, we do not want the apparent brightness of the star, which depends on its distance, to affect the fitting of the template. We thus normalized each spectrum in our collection and so removing the flux level as a free parameter in the fitting. We did so by fitting a $5^{\rm th}$-order polynomial to the trimmed spectrum, and dividing the flux by the best-fit curve at each wavelength. Figure 2.3 shows the result of the normalization of one of the raw spectra in our collection.

We then re-binned the science spectrum to a new wavelength grid to match the wavelength binning of DEIMOS. This new grid contained a smaller number of wavelength values, which reduced the computation time. Thus, we *interpolated* the normalized flux onto an array of new wavelength values, all within the range $[6200, 9500]$ Å. We will refer to this as the *science wavelength grid*.

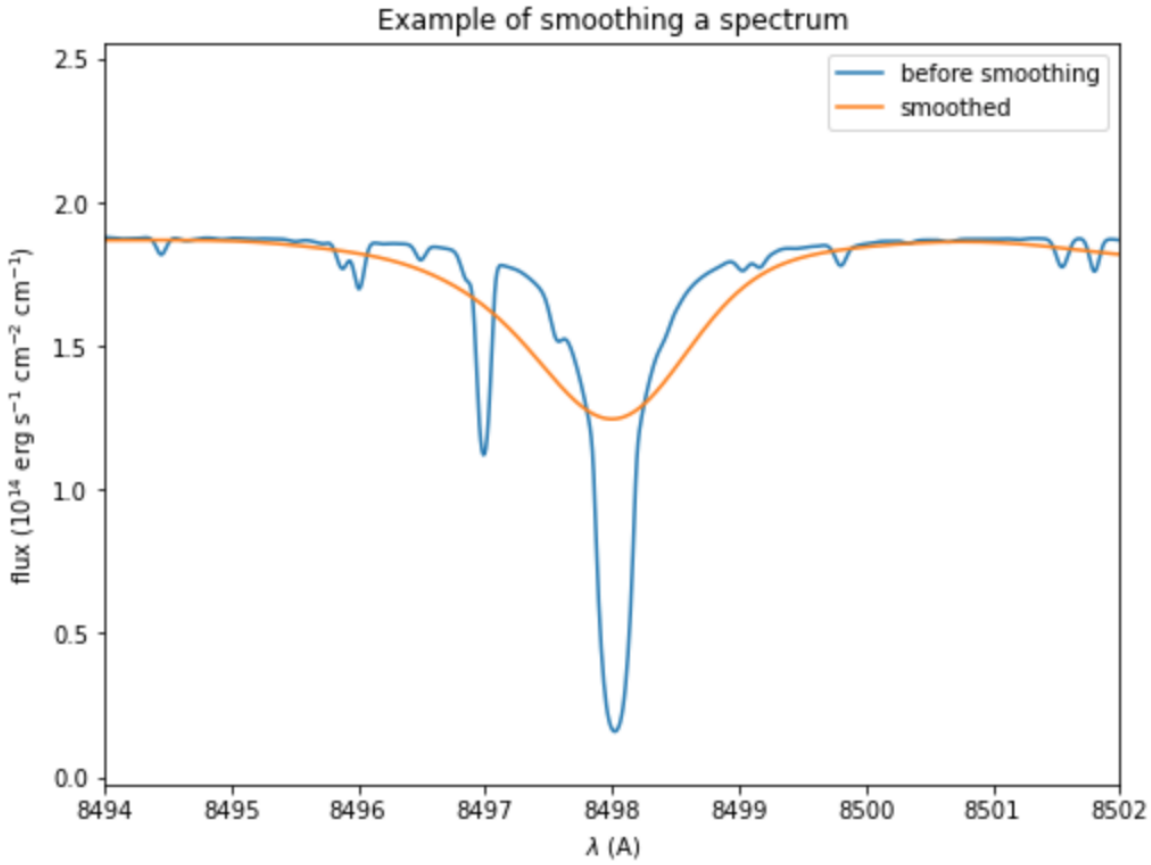


Figure 2.2: Example of smoothing a line in the range of $[8494, 8502]$ Å in the science spectrum using a Gaussian filtering function with a width of 50 times the initial resolution. This spectrum belongs to the template presented in Figure 2.1.

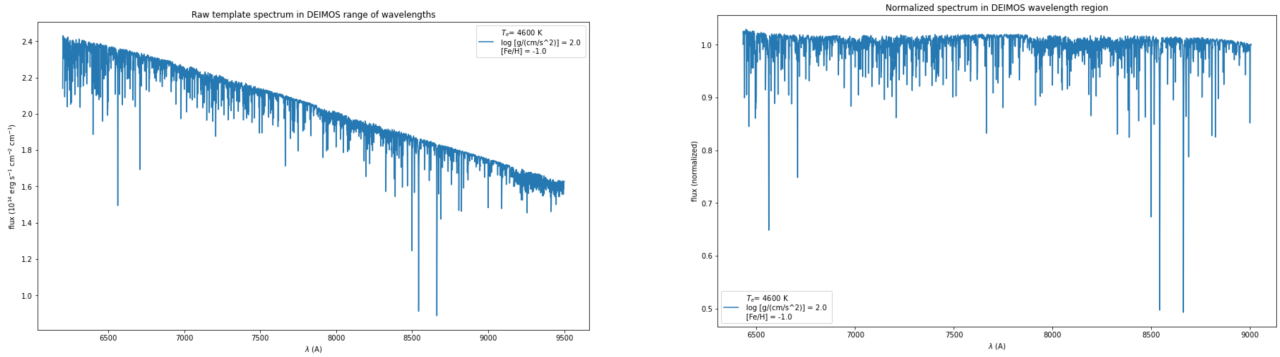


Figure 2.3: **Left:** The flux of the template spectrum in the DEIMOS range of wavelengths. **Right:** The same normalized template. Both correspond to the same science star as in previous figures.

To simulate a real observation, we added noise to the science spectrum by using a particular **S**ignal-to-**N**oise-**R**atio, SNR, value. Since our templates are normalized, the average noise, σ_{noise}, equals the inverse of the SNR value. We used an SNR value of 30 in our testing which corresponds to an average-quality signal. The noise added to the normalized flux of the science template was generated as a list of random numbers drawn from a Gaussian distribution centered at 0, with $\sigma_g = \text{SNR}^{-1}$.

No noise was added to the comparison templates.

3 Spectral χ^2_ν Analysis

We shifted the templates relative to the science spectrum by many different velocities. After each velocity shift, we re-binned the template in wavelength space to match the science spectrum wavelength grid. This process was performed for both the same-same and template mismatch analyses.

For each velocity shift, we calculated the *reduced chi squared*, χ^2_ν, for the fit. This quantity allows us to quantify the level of mismatch between the two spectra, and is defined in Equation 2.1:

$$\chi^2_\nu = \chi^2_\nu(v) = \frac{1}{N}\sum^{N}\left(\frac{F_s - F_t(v)}{1/\text{SNR}}\right)^2, \tag{2.1}$$

where N is the number of points in the science wavelength grid (number of pixels), which is ≈ 8000, F_s represents the flux of the science spectrum, and F_t the flux of the template spectrum. Note that in a real DEIMOS observation, the uncertainty in each measurement (here $1/\text{SNR}$) would be replaced by the RMS of the noise.

For both the same-same and template-mismatch analyses, we used a velocity grid spanning from $-10.0\,\text{km s}^{-1}$ to $+10.0\,\text{km s}^{-1}$, with steps of $0.2\,\text{km s}^{-1}$. We used each velocity in

this grid to redshift or blueshift the template relative to the model spectrum. A large range of values was used to make sure that our analysis captures a large enough portion of the $\chi^2_\nu(v)$ curves, and is not limited in a small interval around the minimum of each curve. Our step size was arbitrarily chosen to yield a reasonable computation time, but it is within the velocity resolution of DEIMOS ($1 - 2$ km s^{-1}).

While our method allows us to find the velocity where the two spectra fit the best (by determining the location where χ^2_ν is minimized), this inferred velocity would depend on the random noise distribution in the science spectrum. For this reason, we added noise randomly 1000 times to our science spectrum, in both analyses. We then ran the chi square evaluation 1000 times for each template and averaged the resulting velocities. We thus generated a *distribution* of χ^2_ν curves for each template at a given SNR for both the same-same and template-mismatch analyses.

The location of the minimum in each χ^2_ν curve indicates the velocity at which the template fits the science spectrum best. For each template, we found the velocities at which all the 1000 χ^2_ν curves have their minimum and averaged the velocities. Therefore, for each mismatch, we found the mean velocity at which the fit is the best and the standard deviation of this distribution.

For the same-same analysis, we expect the best fit velocity to be very small, $v_{\rm min} \approx 0.0\,{\rm km\,s^{-1}}$, as our method fits the science spectrum with a noisier version of itself. Additionally, the standard deviation of the best-fit velocities in the same-same analysis should be the smallest. In the same spirit, we also expect $\langle\chi^2_\nu\rangle \sim 1$; there should be an almost ideal fit between the science spectrum and the same template. We use the results of the same-same analysis, therefore, to confirm that our algorithm finds the best fit velocity and calculates the reduced chi square correctly.

For the template-mismatch analysis, we can deduce any potential bias of this method

from the average of the best-fit velocities distribution (a large non-zero average velocity would indicate there is a bias when mismatching templates). Moreover, we can deduce the precision of the method from the standard deviation of the same distribution. These can be used jointly to quantify how "bad" the mismatch is between the model science spectrum and a nearby template in our parameter space. We expect the mismatch to be worse as the parameters of the template get further away from that of the model spectrum. Lastly, we calculated the average χ^2_ν for each science-template pair from the distributions of the 1000 χ^2_ν curves for each SNR value. This will allow us to determine how likely it is for the method we are trying to establish to misidentify a real DEIMOS observation with the wrong template in our collection.

Chapter 3

Results

1 χ^2_ν-Distributions

For each template (including the same-same and all the template-mismatches), we plotted the distribution of the χ^2_ν values at the minimum of each curve. For example, in Figure 3.1 we present the distribution of χ^2_ν curves for the same-same analysis. Notice how all the curves exhibit a clear minimum at velocities $\approx 0.0\,\mathrm{km\,s^{-1}}$. We expect the fit in the same-same analysis to be the best among all template fits. In Figure 3.2, we can see how the minimum chi square values, $\chi^2_\nu(v_{\rm min})$, are distributed canonically around the mean χ^2_ν value. There are only a few outlying curves, at relatively high or low χ^2_ν, as expected in a canonical distribution. This can be attributed to the randomness of the noise added. With the mismatch templates, as expected, the minima of the χ^2_ν-curves occur at a $\chi^2_\nu \gtrsim 1.00$.

The *average* of the reduced chi square values is a good measure of the fit between the template and the science spectrum. For the same-same analysis, with all 1000 versions of the science spectrum, we observe that $\langle\chi^2_\nu\rangle \gtrapprox 1.00$. We note here that $\chi^2_\nu = 1$ would indicate that the two spectra fit as good as possible. In general, we expect our model to yield $\langle\chi^2_\nu\rangle_{\text{same-same}} \leq \langle\chi^2_\nu\rangle_{\text{mismatch}}$. In section 3 we present color diagrams that illustrate this idea. In Figure **??** we present χ^2_ν distributions, similar to the one in Figure 3.1, but for all template mismatch analyses of the science spectrum.

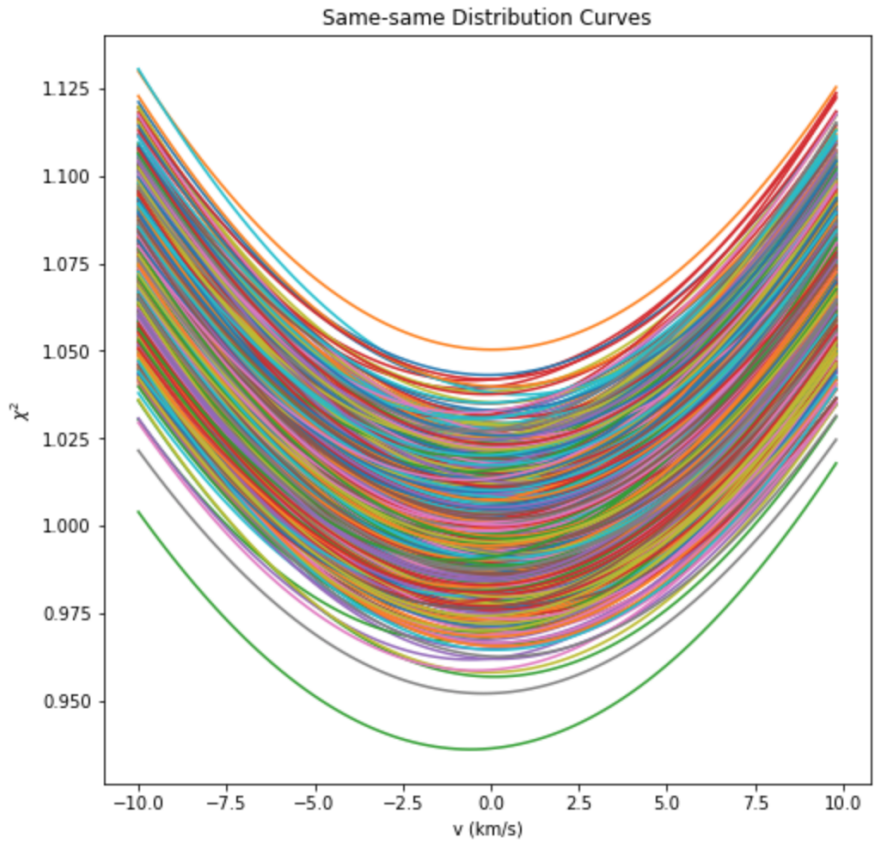


Figure 3.1: Distribution of $\chi^2_\nu(v)$ curves for the same-same analysis of the same template spectrum used in the previous figures. The model spectrum had SNR = 30. Each χ^2_ν curve corresponds to one of the one thousand different model spectra for SNR = 30. Notice how all curves feature a clear minimum at $\sim 0\,\mathrm{km\,s^{-1}}$.

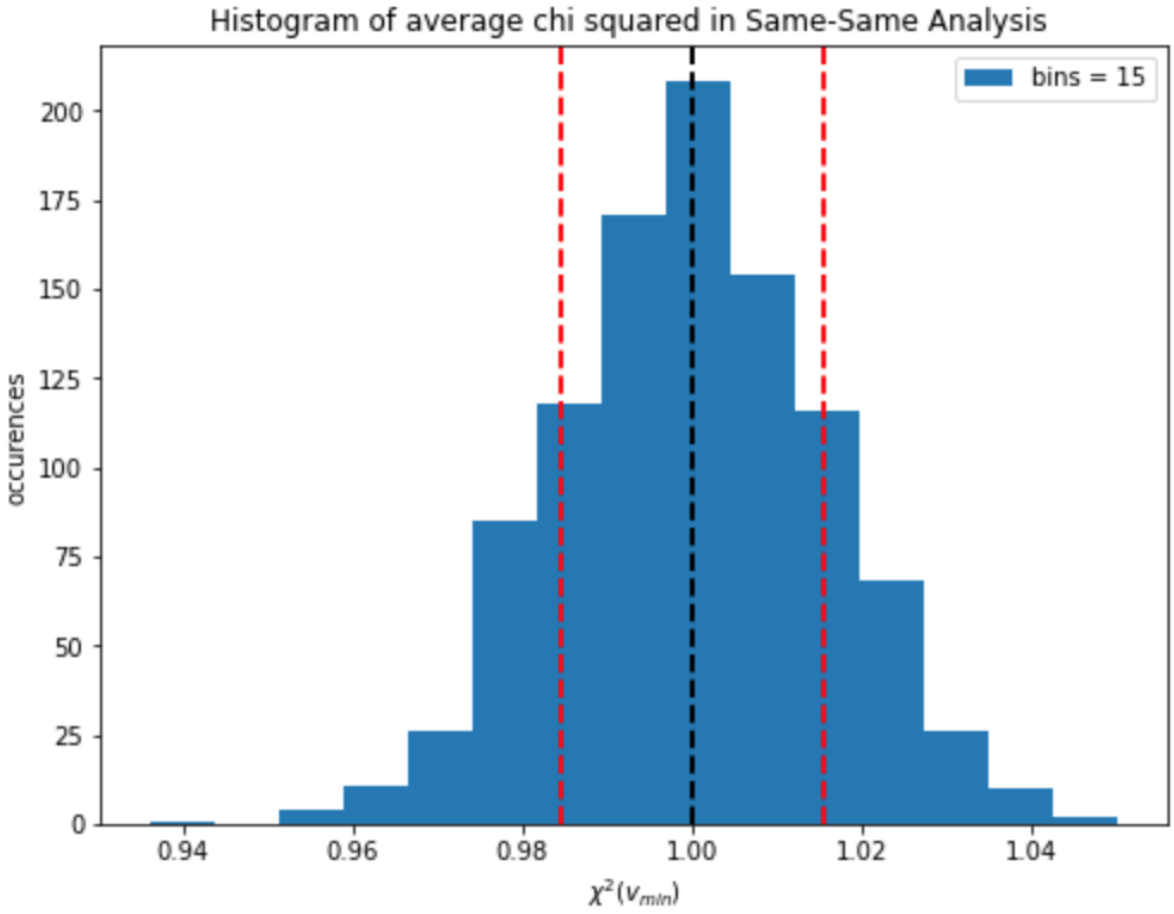


Figure 3.2: Distribution of the minimum χ^2_ν values from the same-same analysis of the template used in the previous figures.

2 Velocity Histograms

For both the same-same and template mismatch analyses, we located the velocity v at which each one of the 1000 χ^2_ν-curves has its minimum. An algorithm trying to match a real observation with the best template in our collection would determine the best fit by finding the template that would yield the minimum of all χ^2_ν. The velocity at the minimum χ^2_ν value would be the inferred recessional velocity of the observed star.

For our science spectrum and for the resulting χ^2_ν distributions that arise from the same-same and template-mismatch analyses, we calculated the average of the velocities at the minimum, $\langle v_{\rm min}\rangle$, and the standard deviation associated with these distributions, σ_v. Figure 3.3 shows an example of a histogram for the same-same analysis.

Table 1 shows the $\langle\chi^2_\nu\rangle$, $\langle v_{\rm min}\rangle$, and σ_v for the same-same and template-mismatch analyses of a particular science spectrum in our collection. Clearly,

$$\left\{\langle\chi^2_\nu\rangle, |\langle v_{\rm min}\rangle|\right\}_{\text{same-same}} \leq \left\{\langle\chi^2_\nu\rangle, |\langle v_{\rm min}\rangle|\right\}_{\text{mismatch}}$$

Note that σ_v (the width of the velocity distributions) is roughly the same for nearby template mismatches, as it depends only on the number of trials.

3 Color Diagrams

In Figure 3.4 we present color diagrams for our chosen science spectrum using all other spectra in the library as comparison templates. The figures show how the average velocity at the χ^2_ν minimum, $\langle v_{\rm min}\rangle$, the standard deviation of the velocities, σ_v, and the average chi squared, $\langle\chi^2_\nu\rangle$ depend on two of the parameters of the template used. In Figure 3.5 we present a closeup of Figure 3.4 in the parameter space around the science spectrum. In both

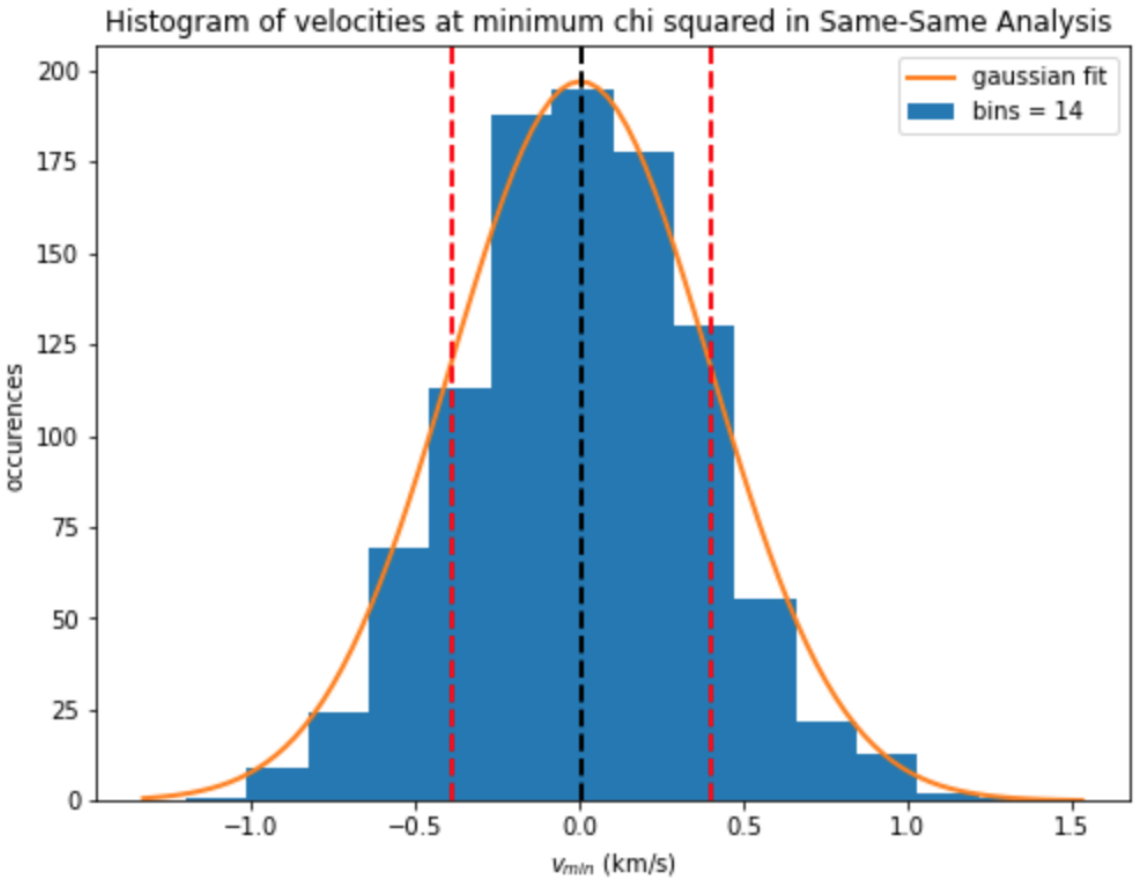


Figure 3.3: Histogram of the velocities at the minimum of each χ^2_ν curve for the same-same analysis of the model spectrum from the previous figures with SNR = 30. A Gaussian fit was added to the histogram to illustrate that the distribution is canonical. Note that the non-zero values of $v_{\min}$ result from the addition of random noise to the science spectrum.

figures, the value of [Fe/H] varies along the y-axis and $T_{\rm eff}$ varies along the x-axis.

From Figure 3.4, we see that around the science spectrum $\langle v_{\rm min} \rangle \sim 0\,{\rm km\,s^{-1}}$, $\sigma_v \lesssim 0.5\,{\rm km\,s^{-1}}$, and $\langle \chi^2_\nu \rangle \sim 1$. Moreover, for templates far from the science spectrum the same quantities get significantly large. For example, mismatching our science spectrum ($T_{\rm eff} = 4600\,{\rm K}$, [Fe/H] $= -1.0$) with a template with $T_{\rm eff} = 3000\,{\rm K}$ and the same metallicity we get $\langle \chi^2_\nu \rangle$ as large as ~ 26. This indicates that the fit gets quite poor for significantly different temperatures. However, because of the large scale of the color bar we cannot distinguish the level of mismatch for templates nearby the science spectrum. We thus chose to focus in a small region of the parameter space around the science spectrum shown in Figure 3.5.

Again, the same-same analysis yields the best fit, as suggested by Figure 3.5. It is clear that reasonable mismatches occur for templates with parameters that are different only by a single grid step size to the parameters of the science spectrum. The further away the parameters get from those of the science spectrum, the worse the fit becomes, as indicated by the $\langle \chi^2_\nu \rangle$ and σ_v values associated with these spectra. However, as can be seen from the last plot in Figure 3.5, the mismatch template with the smallest χ^2_ν is not one of the closest templates to the science spectrum. In particular, the mismatch template with the smallest $\langle \chi^2_\nu \rangle$ is the one in the upper right, which is a step away in both temperature and metallicity. Therefore, it is possible that templates that are more than a single grid step away from the science spectrum may also yield sufficiently low $\langle \chi^2_\nu \rangle$ values to be selected as the matched spectrum.

Notice, however, that for very low metallicity stars compared to our chosen science template ([Fe/H] $\leq -2.0\,{\rm dex}$ for the case we present in Figure 3.5), the match gets significantly worse as indicated by the χ^2_ν values, regardless of whether the templates have temperatures close to the science temperature. These templates have fewer or shallower absorption lines, so matching them with more metallic spectra typically does not yield good results.

Regardless of the $\langle\chi^2_\nu\rangle$ values for the mismatches we present here, the average velocity at the χ^2_ν minimum is fairly low for all templates within $\sim 500\,\mathrm{K}$ of the science spectrum. Again, there is a strong dependence of $\langle v_{\mathrm{min}}\rangle$ on [Fe/H]. When the metallicity of the template spectra is much smaller than that of the science, $\langle v_{\mathrm{min}}\rangle$ is significantly worse than for other templates.

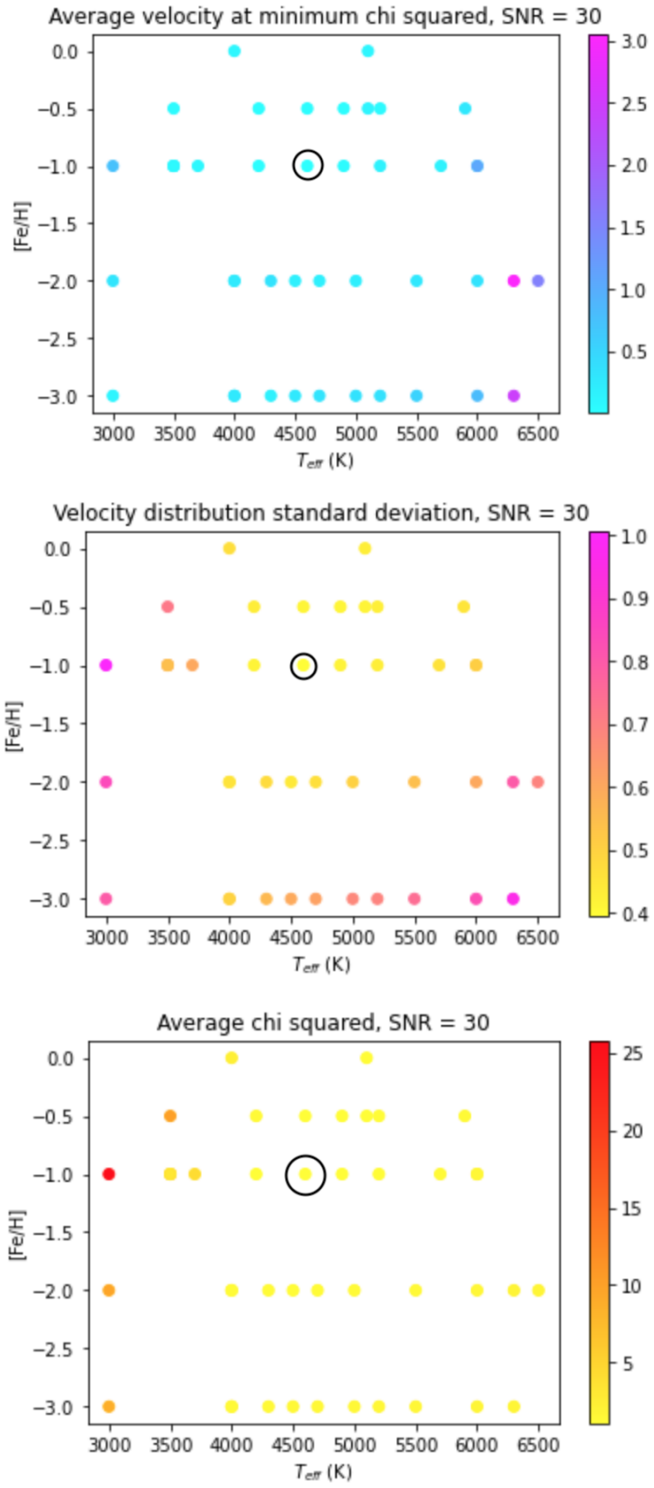


Figure 3.4: Color diagrams showing $\langle v_{\min} \rangle$ (top), σ_v (middle), and $\langle \chi^2_\nu \rangle$ (bottom) as functions of [Fe/H] and $T_{\rm eff}$, using a science spectrum of $T_{\rm eff} = 4600\,{\rm K}$, $\log\left(\frac{g}{\rm cm\,s^{-2}}\right) = +2.0$, and [Fe/H] $= -1.0$ for SNR $= 30$. All other templates in the library were used as mismatch templates. The same-same analysis is circled in the middle of each graph. The velocities are in units of km s^{-1}.

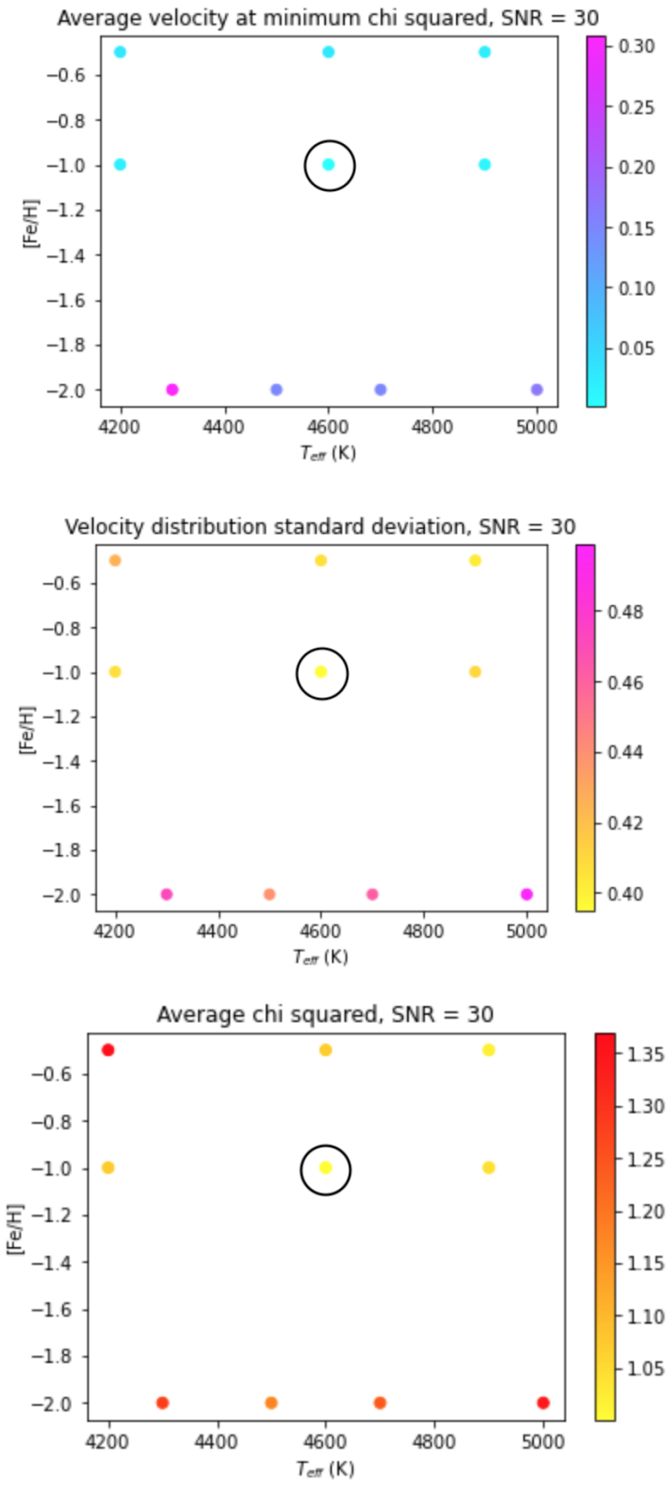


Figure 3.5: Close-up of the color diagrams from Figure 3.4 in the parameter region around the science spectrum, showing $\langle v_{\rm min} \rangle$ (top), σ_v (middle), and $\langle \chi^2_\nu \rangle$ (bottom) as functions of [Fe/H] and $T_{\rm eff}$, using the same science spectrum. The mismatch templates are one step away in parameters from the science spectrum. The same-same analysis is again circled in the middle of the graph. The velocities are in units of km s^{-1}. Note how the color scale range is smaller here, thus the coloring of corresponding spectra in Figure 3.4 and this figure does not correspond to the same numerical values.

Chapter 4

Discussion

1 Same-Same Analysis

For the same-same analysis, we obtained $\langle v_{\rm min} \rangle \approx 0.001\,{\rm km\,s^{-1}}$, and $\langle \chi^2_\nu \rangle \approx 1.000$, as expected. Refer to Table 1 for the specific values. We are thus able to verify that the algorithm works as intended. We also observed that the quantities $\langle v_{\rm min} \rangle$, σ_v, and $\langle \chi^2_\nu \rangle$ are the smallest for the same-same analysis of the science spectrum. This suggests that our algorithm can properly match an observed spectrum to the correct template.

2 Template-Mismatch Analysis

From Figure 3.4 we observe that the average inferred velocities are relatively small for almost all template mismatches and well below the resolution of DEIMOS ($1 - 2\,{\rm km\,s^{-1}}$). They only get comparable to or greater than the resolution for hot low-metallicity stars. In this case, the temperature difference between the science and the mismatch spectra is $\gtrsim 1500\,{\rm K}$ which is more than 3 steps away in the temperature parameter space.

To examine how the quantities $\langle v_{\rm min} \rangle$, and σ_v, vary with increasing shift in temperature, we plot them against the temperature difference between the mismatch template and the science spectrum, $\Delta T \equiv T_{\rm template} - T_{\rm science}$ in different panels in Figure 4.1. This figure also shows—as we noted before—that for templates within a window of $\sim \pm 1500\,{\rm K}$ of the science

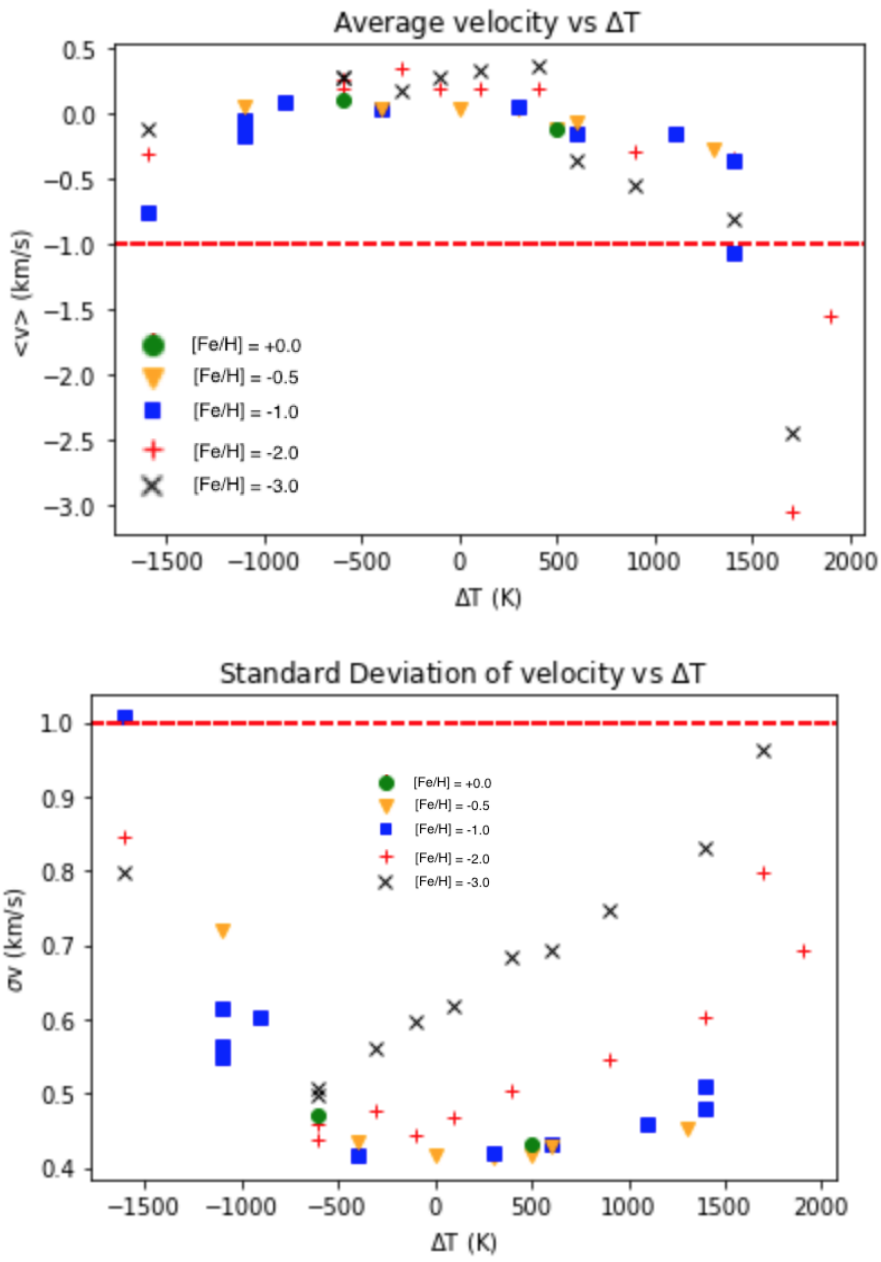


Figure 4.1: Plot of $\langle v_{\min} \rangle$ (top) and σ_v (bottom) as a function of the temperature difference between each mismatch template and the science spectrum. Each marker shape and color represents a different metallicity. The instrumental resolution ($1\,\mathrm{km\,s^{-1}}$) is marked in both figures with the red dashed line. In the top figure, the dotted line represents $v_{\min} = 0.0\,\mathrm{km\,s^{-1}}$. Note that we have not included the same-same matching here.

spectrum, the average inferred velocity stays below the resolution limit of the spectrograph and that for templates with $\Delta T \gtrsim 1500\,\mathrm{K}$, the velocity discrepancies are larger. Since we have not probed the corresponding temperature region in the cool end, we cannot conclude whether large velocities would similarly occur.

The top panel in Figure 4.1 shows that there exist small biases in different temperatures regimes. In particular, for nearby templates with $|\Delta T| \lesssim 500\,\mathrm{K}$, the inferred velocities are slightly positive, whereas outside this regime, they are negative. For nearby mismatch spectra, the average minimum velocity is on the order of a few tenths or hundredths of a kilometer per second ($\lesssim 0.05\,\mathrm{km\,s^{-1}}$ to $\sim 0.3\,\mathrm{km\,s^{-1}}$), as Table 1 also suggests. Within this regime, the inferred velocity changes at a rate of $\lesssim \frac{0.5\,\mathrm{km\,s^{-1}}}{1000\,\mathrm{K}}$. In contrast, the DEIMOS resolution ($\sim 1\,\mathrm{km\,s^{-1}}$) is one to two orders of magnitude larger. Therefore, the bias our method introduces is no larger than $\frac{\max(v_{\mathrm{min}})}{\langle v_{\mathrm{D}} \rangle} = \frac{0.3\,\mathrm{km\,s^{-1}}}{1.0\,\mathrm{km\,s^{-1}}} = 30\%$ of the instrumental resolution for nearby templates.

Notice in Figure 3.5 that for templates in the neighborhood of the science spectrum the largest velocity discrepancies occur for stars with low metallicities (in this case [Fe/H] $= -2.0$ dex). This is to be expected as these have very few absorption lines which makes the matching with a more metallic spectrum more difficult.

With regards to the standard deviation of the velocity distributions, plotted in the lower panel of Figure 4.1, we see that the lowest metallicity templates have significantly higher σ_v values at all temperatures. Moreover, for a given metallicity, the σ_v values increase for increasing $|\Delta T|$. In general, the slope of σ_v versus ΔT is steeper in the low temperature end than in the high one for constant metallicity.

Also shown in Figure 3.5, for neighboring comparison spectra, the standard deviations are fairly low (the largest being $\sim 0.4\,\mathrm{km\,s^{-1}}$). This implies that the uncertainty introduced for neighboring templates is, at most, $\frac{\max(\sigma_v)}{\langle v_d \rangle} = \frac{0.4\,\mathrm{km\,s^{-1}}}{1.0\,\mathrm{km\,s^{-1}}} = 40\%$ of the DEIMOS resolution.

Thus, the uncertainty of this method is limited by the instrumental resolution.

Considering the quality of the fits, the algorithm is able to yield particularly low $\langle\chi^2_\nu\rangle$ values for all mismatches near the science spectrum, as shown in the bottom panel of Figure 3.5. However, these mismatches have $\langle\chi^2_\nu\rangle$ values larger than for the same-same analysis, as Table 1 also suggests. Furthermore, $\langle\chi^2_\nu\rangle$ values increase as templates get further away from the science spectrum in parameter space, and that for all mismatch templates $\langle\chi^2_\nu\rangle - \sigma_{\chi^2_\nu} > 1.0 \equiv [\langle\chi^2_\nu\rangle]_{\text{same-same}}$. This indicates that the algorithm is likely to match an observed spectrum to the most similar template with good confidence.

The color diagrams in section 3 suggest that our algorithm is able to distinguish between matching the science spectrum to itself and mismatching it with neighboring spectra. The smallest $\langle\chi^2_\nu\rangle$ occurs when the comparison and science spectra are the same. Even if an observed spectrum does not correspond directly to any of the spectra in the collection, the best fit will be to a neighboring template with fairly similar parameters. At the same time, it can yield an average minimum velocity comparable to the same-same velocity (we would consider the latter as the "true" stellar velocity of a DEIMOS observation). We have confidence, therefore, that this algorithm will yield a velocity not too different from the true velocity. Because a mismatch does not affect significantly the inferred velocity of a star, we can conclude that the step size of our grid is sufficient for the algorithm to produce good quality results.

We have thus strong evidence from this preliminary analysis that a future automated method searching through the templates in our collection to match an observation made with DEIMOS will yield accurately enough (for the purposes of the large-scale work we presented in the beginning) the velocity and stellar properties of the observation.

Chapter 5

Conclusions

To obtain the stellar velocities of a large number of observations made with the DEIMOS spectrograph of stars in nearby Milky Way satellite dwarf galaxies and globular clusters, an automated process that would match an observation to synthetic stellar templates in a library of spectra is to be used. With this work, we wished to assess the overall effectiveness of this method, identify potential biases introduced by it, and ensure reasonable and accurate results.

We used a collection of synthetic spectra using the effective stellar temperature, the surface gravity, and metallicity to identify each star. We modeled DEIMOS observations by matching these synthetic spectra to the instrumental resolution, normalizing, re-binning to the instrumental pixel size, and adding noise. We then ran two analyses for the DEIMOS model spectrum that we chose: first, we matched the science spectrum to itself, and second, we mismatched all other templates to the science spectrum.

For both cases, we quantified the goodness of the match, by locating the minimum χ^2_ν, and recorded the velocity where the match is the best. The average reduced chi square is related to the overall level of mismatch between spectra, while from the average minimum velocity we could deduce any potential bias of this method. The standard deviation of the average velocity is related to the velocity uncertainty.

From this work, we observed that inside a window of $\sim \pm 1500\,\mathrm{K}$ from the science

spectrum, the bias in the inferred velocity for this method is at most 20% of the DEIMOS resolution, and that the uncertainty in matching a DEIMOS observation with a template from our collection is dominated by the resolution. Moreover, we found that the lowest $\langle\chi^2_\nu\rangle$ occurs in the same-same analysis, and that it increases as the templates get further from the science spectrum away in parameter space. Even in the case of a mismatch of a real observation with a nearby template, the resulting bias and uncertainty are both well below the DEIMOS resolution. Lastly, we deduced that when the template and the science temperatures differ by more that $\sim 1500\,\mathrm{K}$, the bias and the uncertainties become comparable to the resolution, however significantly high $\langle\chi^2_\nu\rangle$ values prevent the algorithm from matching real DEIMOS observations with such templates. We thus concluded that this method is viable, the bias is insignificant, and the inferred stellar velocities are reasonable, with relatively low uncertainties.

Our results suggest that this method can be used further to match DEIMOS observations with templates from our collection. The algorithm would iterate over all spectra in our collection, and determine the best possible match, by finding the template that would yield the smallest χ^2_ν value. The stellar velocity would then be taken as the velocity at which the smallest χ^2_ν value is achieved. The stellar type of the observation would be close in parameter space to that of the template that yielded the smallest χ^2_ν value. In this way, a large number of DEIMOS observations would be classified, accurate stellar kinematics would be obtained, and membership probabilities of the targets would be estimated.

Due to its ease of use and reliability, this process can be used in contexts other than classifying DEIMOS observations, e.g. in homogeneously analyzing spectra from multiple spectrographs. Lastly, to enhance its result quality (potentially when being used for spectrographs with higher resolution), this method could be modified to perform cross-checking of stellar kinematics and classification of known stellar sources.

Chapter 6

Acknowledgments

I would like to thank the Yale University Astronomy Department for partially funding this project through the 2021 Dorrit Hoffleit Scholarship. I would like to thank Dr. Marla Geha, Professor of Physics and Astronomy at Yale University, for her willingness to take me on to be involved in this project, and for providing significant resources, guidance, and feedback throughout the researching and writing period of this work. I would like to thank Dr. Jonathan Marr, Senior Lecturer in Physics and Astronomy at Union College, for his invaluable guidance, suggestions, and overall contribution in creating this written report. Lastly, I would like to thank Mr. Imad Pasha, NSF GRFP Fellow at Yale University, for helping with the Python code created for this project.

Chapter 7

References

1. Iliev T. I., Moore B., et. al. *"Reionization of the Local Group of galaxies"*, MNRAS, **413**, pp. 2093-2102, 2011.

2. Ocvirk P., Gillet N., *"The Reionization of Galactic Satellite Population"*, ApJ, **794**, 2014.

3. Dixon K. L., Iliev T. I., et. al. *"Reionization of the Milky Way, M31, and their satellites – I. Reionization history and star formation"*, MNRAS, **477**, pp. 867-881, 2018.

4. Sawala T., Frenk C. S., et. al. *"The APOSTLE simulations: solutions to the Local Group's cosmic puzzles"*, MNRAS, **457**, pp. 1931-1943, 2016.

5. Haghi H., Amiri V., *"Testing modified gravity with dwarf spheroidal galaxies"*, MNRAS, **463**, pp. 1944-1951, 2016.

6. Abazajian K. N., Keeley R. E., *"Bright gamma-ray Galactic Center excess and dark dwarfs: Strong tension for dark matter annihilation despite Milky Way halo profile and diffuse emission uncertainties"*, PhRvD, **93**, 083514, 2016.

7. Gaia Collaboration, Brown A. G., Vallenari et. al., *"Gaia Data Release 2: Summary of the contents and survey properties"*, AA, **616**, A1, 2018.

8. Simon J. D., *"Gaia Proper Motions and Orbits of the Ultra-faint Milky Way Satellites"*,

ApJ, **863**, 89, 2018.

9. Fritz T. K., Battaglia G., et. al. *"Gaia DR2 Proper Motions of Dwarf Galaxies within 420 kpc: Orbits, Milky Way Mass, Tidal Influences, Planar Alignments, and Group Infall"*, AA, **619**, A103, 2018.

10. Helmi A. Irwin M. J., et. al. ,*"A New View of the Dwarf Spheroidal Satellites of the Milky Way from VLT FLAMES: Where Are the Very Metal-poor Stars?"*, ApJ, **651**, L121, 2006.

11. Walker M. G., Mateo M., et. al. ,*"Velocity Dispersion Profiles of Seven Dwarf Spheroidal Galaxies"*, ApJ, **667**, L53, 2007.

12. Geha M, Simon J. D., *"The Kinematics of the Ultra-faint Milky Way Satellites: Solving the Missing Satellite Problem"*, ApJ, **670**, 2007.

13. Martin N. F., Ibata R. A., et. al. *"A Keck/DEIMOS spectroscopic survey of faint Galactic satellites: searching for the least massive dwarf galaxies"*, MNRAS, **380**, pp. 281-300, 2007.

14. Geha M., et. al. *"A Homogeneous Catalog of Keck/DEIMOS Radial Velocity Measurements: 68 Milky Way Dwarf Galaxies and Globular Clusters"*, 2022 [unpublished]

15. Husser T. O., Wende-von Berg S., et. al. *"A new extensive library of PHOENIX stellar atmospheres and synthetic spectra"*, AA, **553**, A6, 2013.

Appendices

Appendix A

Template-Mismatch Analysis: Numerical Results

We present a collective table with the values of $\langle v_{\min} \rangle$, σ_v, and $\langle \chi^2_\nu |_{\min} \rangle$ we obtained for the same-same analysis for the science spectrum, and the template-mismatch analysis for all other spectra in the library.

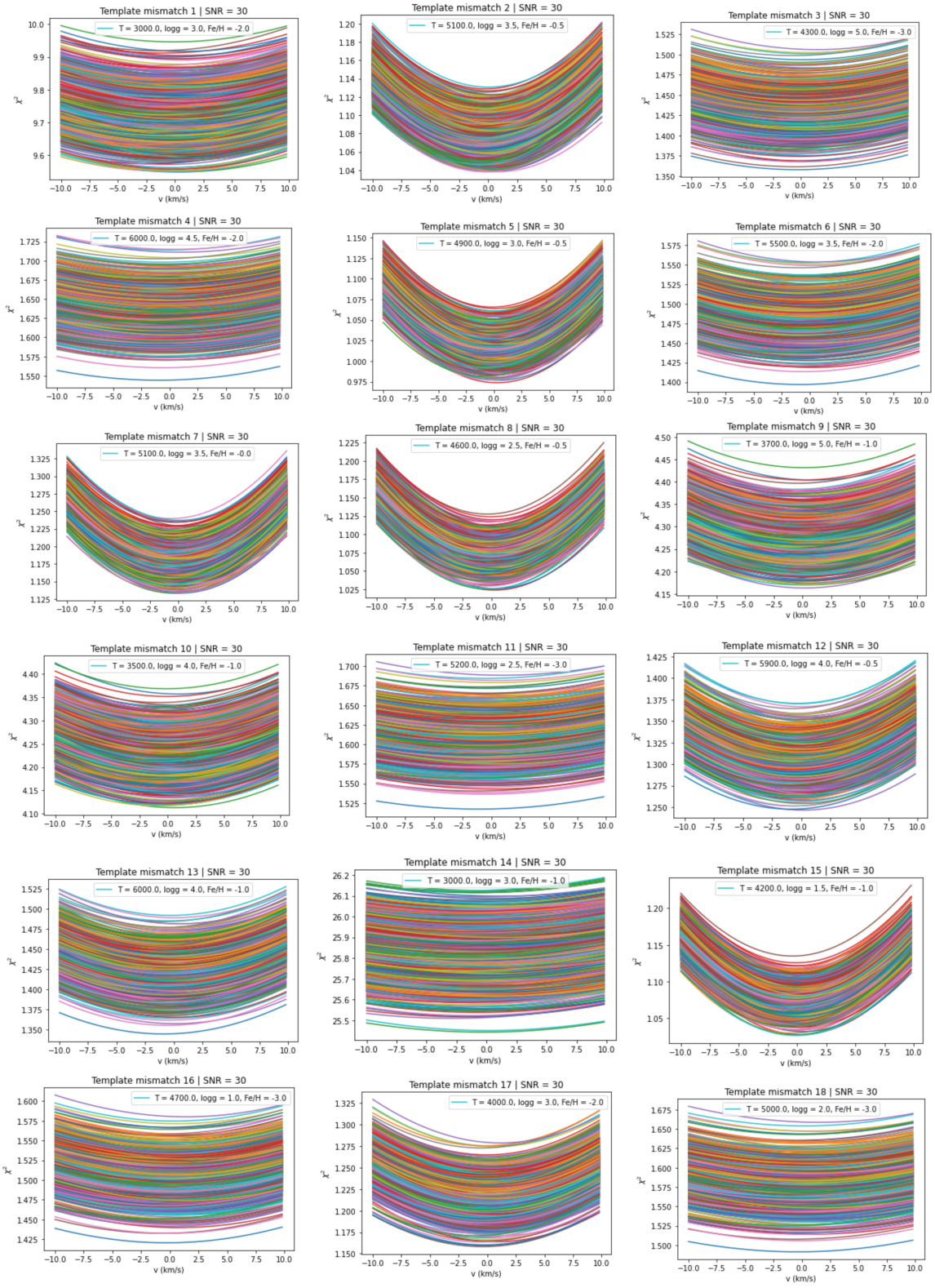
Template	$T_{\rm eff}(K)$	[Fe/H]	log(g)	$\langle v_{\rm min}\rangle$ (km s^{-1})	σ_v (km s^{-1})	$\langle\chi^2_\nu\vert_{\rm min}\rangle$
T1	4000	+0.0	3.0	+0.100	0.470	2.446
T2	5100	+0.0	3.5	-0.118	0.432	1.183
T3	3500	-0.5	5.0	+0.057	0.718	10.199
T4	4200	-0.5	1.5	+0.029	0.435	1.369
T5	4600	-0.5	2.5	+0.027	0.415	1.072
T6	4900	-0.5	3.0	+0.033	0.414	1.018
T7	5100	-0.5	3.5	-0.122	0.418	1.080
T8	5200	-0.5	4.5	-0.064	0.429	1.140
T9	5900	-0.5	4.0	-0.281	0.454	1.305
T10	3000	-1.0	3.0	-0.761	1.007	25.823
T11	3500	-1.0	4.0	-0.049	0.616	4.229
T12	3500	-1.0	2.0	-0.165	0.563	6.845
T13	3500	-1.0	3.0	-0.077	0.547	3.207
T14	3700	-1.0	5.0	+0.086	0.602	4.277
T15	4200	-1.0	1.5	+0.034	0.418	1.075
Self	**4200**	**-1.0**	**2.0**	**+0.001**	**0.395**	**1.000**
T16	4900	-1.0	2.5	+0.060	0.419	1.042
T17	5200	-1.0	3.5	-0.159	0.433	1.160
T18	5700	-1.0	4.5	-0.162	0.460	1.310
T19	6000	-1.0	4.0	-0.357	0.480	1.419
T20	6000	-1.0	2.0	-1.075	0.511	1.602
T21	3000	-2.0	3.0	-0.314	0.845	9.725
T22	4000	-2.0	3.0	+0.196	0.437	1.215
T23	4000	-2.0	4.0	+0.253	0.458	1.281
T24	4300	-2.0	5.0	+0.344	0.478	1.284
T25	4500	-2.0	1.5	+0.188	0.443	1.178
T26	4700	-2.0	1.5	+0.188	0.468	1.241
T27	5000	-2.0	2.0	+0.195	0.503	1.346
T28	5500	-2.0	3.5	- 0.287	0.547	1.479
T29	6000	-2.0	4.5	-0.337	0.602	1.633
T30	6300	-2.0	2.5	-3.056	0.798	1.938
T31	6500	-2.0	4.0	-1.557	0.692	1.865
T32	3000	-3.0	3.0	-0.126	0.797	8.814
T33	4000	-3.0	2.5	+0.274	0.507	1.432
T34	4000	-3.0	2.0	+0.275	0.499	1.400
T35	4300	-3.0	5.0	+0.176	0.562	1.4290
T36	4500	-3.0	4.0	+0.269	0.596	1.517
T37	4700	-3.0	1.0	+0.325	0.619	1.501
T38	5000	-3.0	2.0	+0.359	0.684	1.576
T39	5200	-3.0	2.5	-0.365	0.693	1.605
T40	5500	-3.0	3.5	-0.546	0.746	1.677
T41	6000	-3.0	4.5	-0.818	0.829	1.789
T42	6300	-3.0	4.0	-2.455	0.961	1.866

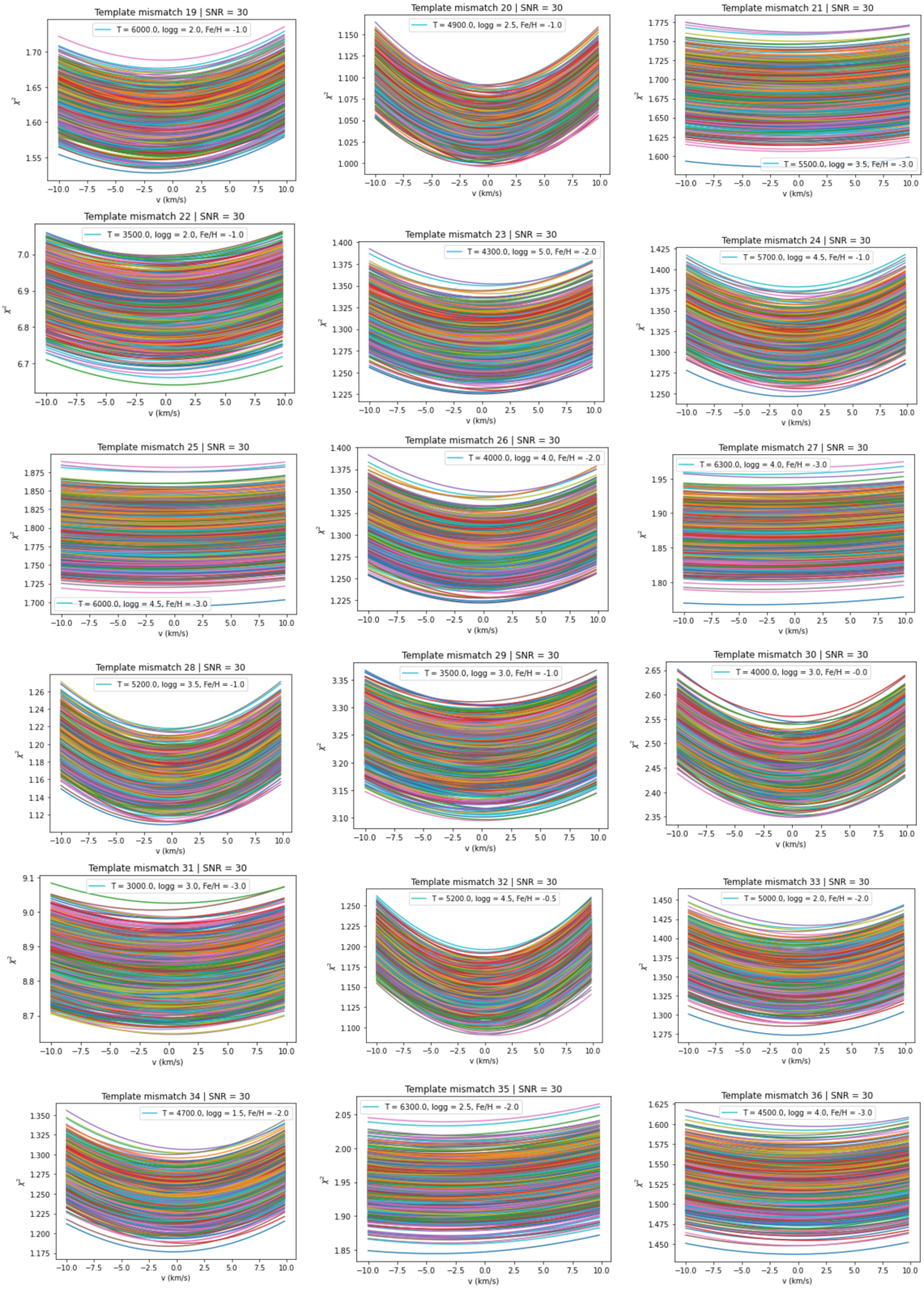
Table 1. The values of $\langle v_{\rm min}\rangle$, σ_v, and $\langle\chi^2_\nu\vert_{\rm min}\rangle$ from the same-same and template-mismatch analysis of the science spectrum with $T_{\rm eff} = 4600\,{\rm K}$, $\log\left(\frac{g}{\rm cm\,s^{-2}}\right) = +2.0$, and [Fe/H] $= -1.0$ dex. The "T" rows refer to different template mismatches, whereas the 'Self' row refers to the same-same analysis, but the numbering of templates here does not match the numbering of the figures in Appendix B. The properties of each mismatch spectrum ($T_{\rm eff}$, [Fe/H], and $\log\left(\frac{g}{\rm cm\,s^{-2}}\right)$) are shown. Note how the same-same analysis yields the best match between the template and the science spectra as expected.

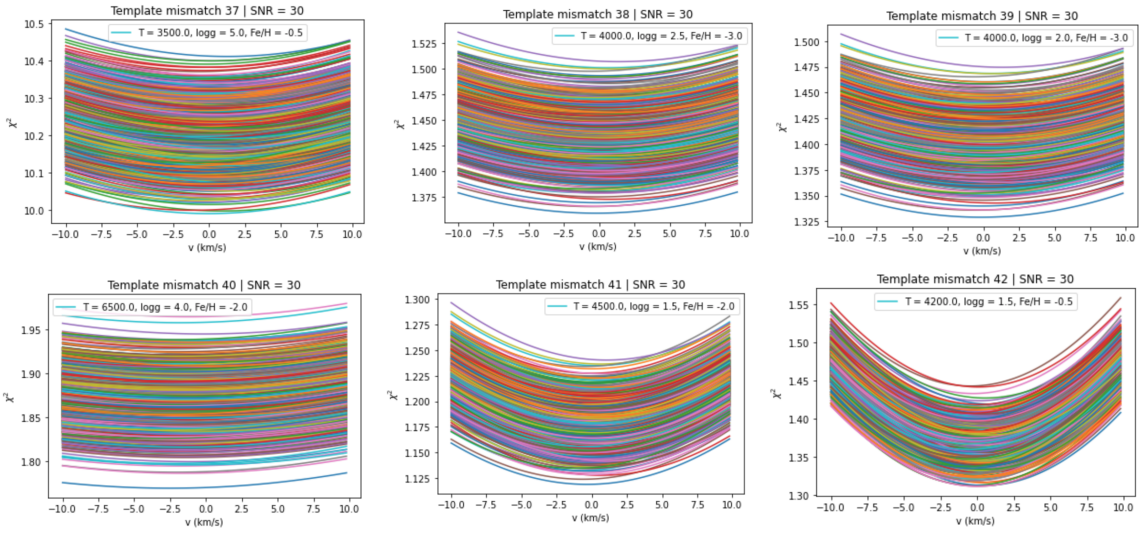
Appendix B

Distributions of χ^2_ν

We present the distributions of $\chi^2_\nu(v)$ curves for the template-mismatch analysis of our chosen science spectrum using all other templates in the library. The science spectrum has SNR = 30. Each χ^2_ν figure corresponds to one of the 1000 mismatch trials. Notice how all distributions feature a clear minimum at $\sim 0.0\,\mathrm{km\,s^{-1}}$, but have different widths, depending on the chosen template spectrum. The parameters of the mismatch templates are also presented in Table 1.


Template mismatch 1 | SNR = 30
T = 3000.0, logg = 3.0, Fe/H = -2.0
Template mismatch 2 | SNR = 30
T = 5100.0, logg = 3.5, Fe/H = -0.5
Template mismatch 3 | SNR = 30
T = 4300.0, logg = 5.0, Fe/H = -3.0
Template mismatch 4 | SNR = 30
T = 6000.0, logg = 4.5, Fe/H = -2.0
Template mismatch 5 | SNR = 30
T = 4900.0, logg = 3.0, Fe/H = -0.5
Template mismatch 6 | SNR = 30
T = 5500.0, logg = 3.5, Fe/H = -2.0
Template mismatch 7 | SNR = 30
T = 5100.0, logg = 3.5, Fe/H = -0.0
Template mismatch 8 | SNR = 30
T = 4600.0, logg = 2.5, Fe/H = -0.5
Template mismatch 9 | SNR = 30
T = 3700.0, logg = 5.0, Fe/H = -1.0
Template mismatch 10 | SNR = 30
T = 3500.0, logg = 4.0, Fe/H = -1.0
Template mismatch 11 | SNR = 30
T = 5200.0, logg = 2.5, Fe/H = -3.0
Template mismatch 12 | SNR = 30
T = 5900.0, logg = 4.0, Fe/H = -0.5
Template mismatch 13 | SNR = 30
T = 6000.0, logg = 4.0, Fe/H = -1.0
Template mismatch 14 | SNR = 30
T = 3000.0, logg = 3.0, Fe/H = -1.0
Template mismatch 15 | SNR = 30
T = 4200.0, logg = 1.5, Fe/H = -1.0
Template mismatch 16 | SNR = 30
T = 4700.0, logg = 1.0, Fe/H = -3.0
Template mismatch 17 | SNR = 30
T = 4000.0, logg = 3.0, Fe/H = -2.0
Template mismatch 18 | SNR = 30
T = 5000.0, logg = 2.0, Fe/H = -3.0
v (km/s)
χ^2


Template mismatch 19 | SNR = 30
T = 6000.0, logg = 2.0, Fe/H = -1.0
Template mismatch 20 | SNR = 30
T = 4900.0, logg = 2.5, Fe/H = -1.0
Template mismatch 21 | SNR = 30
T = 5500.0, logg = 3.5, Fe/H = -3.0
Template mismatch 22 | SNR = 30
T = 3500.0, logg = 2.0, Fe/H = -1.0
Template mismatch 23 | SNR = 30
T = 4300.0, logg = 5.0, Fe/H = -2.0
Template mismatch 24 | SNR = 30
T = 5700.0, logg = 4.5, Fe/H = -1.0
Template mismatch 25 | SNR = 30
T = 6000.0, logg = 4.5, Fe/H = -3.0
Template mismatch 26 | SNR = 30
T = 4000.0, logg = 4.0, Fe/H = -2.0
Template mismatch 27 | SNR = 30
T = 6300.0, logg = 4.0, Fe/H = -3.0
Template mismatch 28 | SNR = 30
T = 5200.0, logg = 3.5, Fe/H = -1.0
Template mismatch 29 | SNR = 30
T = 3500.0, logg = 3.0, Fe/H = -1.0
Template mismatch 30 | SNR = 30
T = 4000.0, logg = 3.0, Fe/H = -0.0
Template mismatch 31 | SNR = 30
T = 3000.0, logg = 3.0, Fe/H = -3.0
Template mismatch 32 | SNR = 30
T = 5200.0, logg = 4.5, Fe/H = -0.5
Template mismatch 33 | SNR = 30
T = 5000.0, logg = 2.0, Fe/H = -2.0
Template mismatch 34 | SNR = 30
T = 4700.0, logg = 1.5, Fe/H = -2.0
Template mismatch 35 | SNR = 30
T = 6300.0, logg = 2.5, Fe/H = -2.0
Template mismatch 36 | SNR = 30
T = 4500.0, logg = 4.0, Fe/H = -3.0
v (km/s)
χ^2


Template mismatch 37 | SNR = 30
T = 3500.0, logg = 5.0, Fe/H = -0.5
Template mismatch 38 | SNR = 30
T = 4000.0, logg = 2.5, Fe/H = -3.0
Template mismatch 39 | SNR = 30
T = 4000.0, logg = 2.0, Fe/H = -3.0
Template mismatch 40 | SNR = 30
T = 6500.0, logg = 4.0, Fe/H = -2.0
Template mismatch 41 | SNR = 30
T = 4500.0, logg = 1.5, Fe/H = -2.0
Template mismatch 42 | SNR = 30
T = 4200.0, logg = 1.5, Fe/H = -0.5
v (km/s)
χ^2

Appendix C

Same-Same Analysis Code

We present the Python program we created for the purpose of the same-same analysis of the spectra.

```
#This program will open all synthetic spectra in the library,
#will use every spectrum as a template and will match it to itself one thousand times,
#using SNR = 30, and reduced chi squared

#Preamble

import numpy as np
from numpy import exp, loadtxt, pi, sqrt
import os, sys
import matplotlib.pyplot as plt
import matplotlib.backends.backend_pdf
from mpl_toolkits import mplot3d
%matplotlib inline

from astropy.table import Table,Column
from astropy import units as u
import astropy.constants as const
from astropy.io import ascii,fits
from astropy.modeling import models

from scipy.optimize import curve_fit

import math
import glob
import scipy.ndimage as scipynd

from scipy.stats import norm
import matplotlib.mlab as mlab


deimos_dropbox = '/Users/xristoskakogiannis/Dropbox/Yale Research/'

import time

#Constants

h = const.h.value
c = const.c.value * 1e-3
k = const.k_B.value

#Fit a fifth order polynomial to continuum and Normalize
```

```
#########################################################################
def normalize (wavelengths, fluxes):
    '''Returns an array of normalized fluxes in a given wavelength range.
    Fits a fifth order polynomial between the 15th and 99th percentile of the fluxes.
    Then divides the given flux by the best-fit curve evaluated at the same wavelengths'''

    #returns only these flux values that are between 15% and 99% of all values
    for_cont = (fluxes > np.percentile (fluxes,15)) & (fluxes < np.percentile (fluxes,99))

    p = np.polyfit (wavelengths[for_cont], fluxes[for_cont], 5)
    fit = np.poly1d(p)
    continuum = fit (wavelengths)
    normalized = fluxes / continuum

    return normalized

#Add gaussian noise
#########################################################################
def add_noise (SNR, array_size):
    '''Returns an array of random numbers drawn from a gaussian distribution'''
    stdev = 1/SNR
    noise = np.random.normal (0, stdev, array_size)
    return noise

#Find nearest index to a value in an array
#########################################################################
def find_nearest(array, value):
    array = np.asarray(array)
    idx = (np.abs(array - value)).argmin()
    return idx

#Get wavelenghts
with fits.open (deimos_dropbox + 'wavelength_arrays/tot_wavelength_array') as hduw:
    wavelengths = hduw[0].data #all wavelengths

#Get wavelength grid to interpolate
with fits.open (deimos_dropbox + 'wavelength_arrays/interpolation_wavelength_grid') as hdug:
    wave_array = hdug[0].data #interpolation grid

#Get all fluxes
#create raw flux list
raw_flux_list = []

spectra = glob.glob (deimos_dropbox + 'lte_and_dmost_files/lte_files/lte*.fits')

for s in spectra:
    with fits.open (s) as hdu:
        raw_flux = hdu[0].data #unsmoothed flux (raw flux)
    raw_flux_list.append (raw_flux)

#Get spectra parameters
spectra_dict = []

for s in spectra:
    s = s.replace (deimos_dropbox + 'lte_and_dmost_files/lte_files/lte0', '')
    s = s.replace ('.PHOENIX-ACES-AGSS-COND-2011-HiRes.fits', '')

    teff, logg, feh = s.split('-')

    parms = {'teff': float (teff), 'logg': float (logg), 'feh': -1 * float (feh)}
    spectra_dict.append (parms)

#Choose raw science
science_idx = spectra_dict.index ({'teff':4600., 'logg': 2.0, 'feh':-1.0})
raw_science = raw_flux_list [science_idx]
print (science_idx)
```

```
#Trim wavelengths
lower = find_nearest (wavelengths, 6200)
upper = find_nearest (wavelengths, 9500) + 1

waves_deimos = wavelengths [lower:upper] #waves in deimos region

# Plot raw cience
plt.figure (figsize = (14,8))
plt.plot (wavelengths, raw_science*1e-14, label =
          ' $ T_e $= {} K \n log [g/(cm/s^2)] = {} \n [Fe/H] = {}'.format (
          int(spectra_dict [science_idx]['teff']), spectra_dict [science_idx]['logg'], spectra_dict [science_idx]['feh'],
          ))
plt.title ('Science Spectrum')
plt.xlabel ('$\lambda$ (A)')
plt.ylabel ('flux ($10^{14}$ erg s$^{-1}$ cm$^{-2}$ cm$^{-1}$)')
plt.legend ()

# Process science
raw_science_deimos = raw_science [lower:upper] #trim
smoothed_raw_science_deimos = scipynd.gaussian_filter1d (raw_science_deimos, 50) #smooth to DEIMOS resolution
normed_science = normalize (waves_deimos, smoothed_raw_science_deimos) #normalize flux in deimos region
science  = np.interp (wave_array, waves_deimos, normed_science) #interpolate smoothed flux

# Plot entire science spectrum
plt.figure (figsize = (14,8))
plt.plot (waves_deimos, smoothed_raw_science_deimos*1e-14,label =
          ' $ T_e $= {} K \n log [g/(cm/s^2)] = {} \n [Fe/H] = {}'.format (
          int(spectra_dict [science_idx]['teff']), spectra_dict [science_idx]['logg'], spectra_dict [science_idx]['feh'],
          ))
plt.title ('Raw template spectrum in DEIMOS range of wavelengths')
plt.xlabel ('$\lambda$ (A)')
plt.ylabel ('flux ($10^{14}$ erg s$^{-1}$ cm$^{-2}$ cm$^{-1}$)')
plt.legend ()


# Plot an example of smoothing an absorption line
plt.figure (figsize = (8,6))
plt.title ('Example of smoothing a spectrum')
plt.plot (waves_deimos, raw_science_deimos*1e-14, label = 'before smoothing')
plt.plot (waves_deimos, smoothed_raw_science_deimos*1e-14, label = 'smoothed')
plt.xlim (8494, 8502)
plt.xlabel ('$\lambda$ (A)')
plt.ylabel ('flux ($10^{14}$ erg s$^{-1}$ cm$^{-2}$ cm$^{-1}$)')
plt.legend ()

# Plot normalized science
plt.figure (figsize = (14,8))
plt.plot (wave_array, science, label =
          ' $ T_e $= {} K \n log [g/(cm/s^2)] = {} \n [Fe/H] = {}'.format (
          int(spectra_dict [science_idx]['teff']), spectra_dict [science_idx]['logg'], spectra_dict [science_idx]['feh'],
          ))
plt.title ('Normalized spectrum in DEIMOS wavelength region')
plt.xlabel ('$\lambda$ (A)')
plt.ylabel ('flux (normalized)')
plt.legend ()


# Add noise to science

nscience_array = [] #nscience array for 1000 iterations

#idx : trial num

tbeg = time.time ()
for j in range (1000):
```

```
    nscience_array.append (
        np.array (
            np.add (
                science, add_noise (30, len (wave_array))
            )
        )
    )

tend = time.time ()
print ('Science Spectrum took {:.3f} s'.format (tend - tbeg))


velocities = np.arange (-10., 10., 0.2) #km/s

def calculate_chi2 (nscience, template, SNR, size_of_wave_array):
    chi2 = chi_squared = (1/size_of_wave_array) * np.sum ((nscience - template)**2/ (1/SNR)**2 )
    return chi2

#Calculate chi2

chisq_list = []


t0 = time.time ()

normed_template = normed_science [:] #copy of smoothed + normed science for same-same

for j in range (1000):
    t0j = time.time ()

    chisq_in_trial = []

    for v in velocities:
        waves_shifted = waves_deimos * (1+v/c)

        template = np.interp (wave_array, waves_shifted, normed_template)

        nscience = nscience_array [j]

        chi2 = calculate_chi2 (nscience, template, 30, len (wave_array))

        chisq_in_trial.append (chi2)
    chisq_list.append (chisq_in_trial)
    t1j = time.time ()
    print ('   Loop {} took {:.3f} s'.format (j+1, t1j-t0j))

t1 = time.time ()
print ('To generate the entire list took {:.3f} s'.format (t1-t0))

# Plot distribution of reduced chi squared
plt.figure (figsize = (8,8))
for j in range (1000):
    plt.plot (velocities, chisq_list [j])
plt.title ('Same-same Distribution Curves')
plt.ylabel ('$\chi^2$')
plt.xlabel ('v (km/s)')

# Find average and standard deviations of velocities at minimum, and minimum chi squared

velmin_array = []
offsets = []
chisqmin_array = []

for j in range (1000):
    idx = np.argmin (chisq_list [j])
    chisq_min = np.amin (chisq_list [j])
```

```
    vel_min = velocities [idx]

    velmin_array.append (vel_min)
    chisqmin_array.append (chisq_min)

avg_chisq = np.mean (chisqmin_array)
stdev_chisq = np.std (chisqmin_array)

avg = np.mean (velmin_array) #km/s
stdev = np.std (velmin_array) #km/s
offsets.append ([avg, stdev, avg_chisq, stdev_chisq])

# Make chi-squared histogram

plt.figure (figsize = (8,6))
plt.hist (chisqmin_array, bins = 15, label = 'bins = 15')
#plt.plot (bins, 1/sqrt(2*pi*stdev**2)*exp(-(bins-avg)**2/(2*stdev**2))*np.amax (count), '-', linewidth = 3)
plt.ylabel ('occurences')
plt.xlabel ('$\chi^2 (v_{min})$' )
plt.legend ()
plt.axvline (avg_chisq, linestyle = 'dashed',linewidth = 2, color = 'black')
plt.axvline (avg_chisq + stdev_chisq, linestyle = 'dashed',linewidth = 2, color = 'red' )
plt.axvline (avg_chisq - stdev_chisq, linestyle = 'dashed',linewidth = 2, color = 'red' )
plt.title ('Histogram of average chi squared in Same-Same Analysis ')


# Make velocity histogram

avg, stdev = offsets [0][0], offsets [0][1]

plt.figure (figsize = (8,6))
count, bins, ignored = plt.hist (velmin_array, bins = 14, label = 'bins = 14')

xmin, xmax = plt.xlim()
vels = np.linspace(xmin, xmax, 100)
gaussian = norm.pdf(vels, avg, stdev)*np.amax (count)

plt.plot (vels, gaussian, linewidth = 2, label = 'gaussian fit')
plt.ylabel ('occurences')
plt.xlabel ('$v_{min}$ (km/s)' )
plt.legend ()
plt.axvline (avg, linestyle = 'dashed',linewidth = 2, color = 'black')
plt.axvline (avg + stdev, linestyle = 'dashed',linewidth = 2, color = 'red' )
plt.axvline (avg - stdev, linestyle = 'dashed',linewidth = 2, color = 'red' )
plt.title ('Histogram of velocities at minimum chi squared in Same-Same Analysis ')
```

Appendix D

Template-Mismatch Analysis Program

We present the Python program we created for the purpose of the template-mismatch analysis of the spectra.

```
#1) Use a single spectrum as the science spectrum and all other spectra in library as templates
#2) Match the template spectra to the science spectrum in order to determine how good the match is
#3) Use SNR = 30, and repeat the process 1000 times


#Preamble

import numpy as np
from numpy import exp, loadtxt, pi, sqrt
import os, sys
import matplotlib.pyplot as plt
import matplotlib.backends.backend_pdf
%matplotlib inline

from astropy.table import Table,Column
from astropy import units as u
import astropy.constants as const
from astropy.io import ascii,fits
from astropy.modeling import models

from scipy.optimize import curve_fit

import math
import glob
import scipy.ndimage as scipynd

from scipy.stats import norm
import matplotlib.mlab as mlab


deimos_dropbox = '/Users/xristoskakogiannis/Dropbox/Yale Research/'

import time
import copy

#Constants

h = const.h.value
c = const.c.value*1e-3
k = const.k_B.value
```

```
######### FUNCTIONS

#Fit a fifth order polynomial to continuum and Normalize
#############################################################################
def normalize (wavelengths, fluxes):

    #returns only these flux values that are between 15% and 99% of all values
    for_cont = (fluxes > np.percentile (fluxes,15)) & (fluxes < np.percentile (fluxes,99))

    p = np.polyfit (wavelengths[for_cont], fluxes[for_cont], 5)
    fit = np.poly1d(p)
    continuum = fit (wavelengths)
    normalized = fluxes / continuum

    return normalized

#Add gaussian noise
#############################################################################
def add_noise (SNR, array_size):
    stdev = 1/SNR
    noise = np.random.normal (0, stdev, array_size)
    return noise

#Find nearest index to a value in an array
#############################################################################
def find_nearest(array, value):
    array = np.asarray(array)
    idx = (np.abs(array - value)).argmin()
    return idx

#Return parameters of a given spectrum
#############################################################################
def give_parms_of (Teff, feh, spectra_list):
    for idx in range (len(spectra_list)):
        if spectra_list [idx][0] == float(Teff) and spectra_list [idx][2] == float (feh):
            return spectra_list [idx], idx+1
    print ('No such spectrum in the library')

################## CODE

#Get wavelengths
with fits.open (deimos_dropbox + 'wavelength_arrays/tot_wavelength_array') as hduw:
    wavelengths = hduw[0].data #all wavelengths

#Get wavelength grid to interpolate
with fits.open (deimos_dropbox + 'wavelength_arrays/interpolation_wavelength_grid') as hdug:
    wave_array = hdug[0].data #interpolation grid

#Create raw flux list to write all spectra
raw_flux_list = []

#Get all fluxes
spectra = glob.glob (deimos_dropbox + 'lte_and_dmost_files/lte_files/lte*.fits')

for s in spectra:
    with fits.open (s) as hdu:
        raw_flux = hdu[0].data #unsmoothed flux
    raw_flux_list.append (raw_flux)

#Get spectra parameters
spectra_dict = []

for s in spectra:
    s = s.replace (deimos_dropbox + 'lte_and_dmost_files/lte_files/lte0', '')
    s = s.replace ('.PHOENIX-ACES-AGSS-COND-2011-HiRes.fits', '')
```

```
    teff, logg, feh = s.split('-')

    parms = {'teff': float(teff), 'logg': float(logg), 'feh': float(feh)*-1}
    spectra_dict.append (parms)

#Create a list with fluxes and parameters
fluxes_dictionary = []

for s in range (43):
    fluxes_dictionary.append ([raw_flux_list [s], spectra_dict [s]])

#Trim wavelengths to DEIMOS range

lower = find_nearest (wavelengths, 6200)
upper = find_nearest (wavelengths, 9500) + 1

waves_deimos = wavelengths [lower:upper] #waves in deimos region

#Smooth all spectra

smoothed_flux_list = []

for t in range (43):
    trimmed_spec = raw_flux_list [t][lower:upper]
    smoothed_spec = scipynd.gaussian_filter1d (trimmed_spec, 50)
    smoothed_flux_list.append (smoothed_spec)

#Process science

smoothed_science = smoothed_flux_list [science_idx]
normed_science = normalize (waves_deimos, smoothed_science)
science = np.interp (wave_array, waves_deimos, normed_science) #interpolated smoothed flux


#Add noise to science to create the nscience array
#(1000 iterations for a single SNR value)

nscience_array = [] #nscience array for 1000 iterations

#1nd idx: SNR value
#2rd idx: iteration


tbeg = time.time()

for j in range (1000):
    nscience_array.append (np.array
                                (np.add (science, add_noise (30, len(wave_array))
                                )
                            )
                        )
tend = time.time ()
print ('Science spectrum took {:.3f} s'.format (tend - tbeg))



nscience_array = np.array (nscience_array) #ndarray


all_templates = []
for t in range (43):
    all_templates.append ([smoothed_flux_list [t], spectra_dict [t],  t])

del all_templates [science_idx]

all_normed_templates = []
```

```
for t in range (len(all_templates)):
    normed_template = normalize (waves_deimos, all_templates [t][0])
    all_normed_templates.append ([normed_template, all_templates [t][1], all_templates [t][2]])


velocities = np.arange (-10, 10, 0.2) #km/s

def calculate_chi2 (nscience, template, SNR, size_of_wave_array):
    chi2 = chi_squared = (1/size_of_wave_array) * np.sum ((nscience - template)**2/ (1/SNR)**2 )
    return chi2

#Calculate chi2

all_chisq_list = []

t0 = time.time ()


for j in range (1000):
    t0j = time.time ()

    chisq_in_trial = []

    nscience = nscience_array [j]

    for t in range (42): #for the number of templates

        chisq_in_template = []

        normed_template = all_normed_templates [t][0][:]
        #copy of smoothed + normed template for mismatch

        for v in velocities:
            waves_shifted = waves_deimos * (1+v/c)

            template = np.interp (wave_array, waves_shifted, normed_template)

            chi2 = calculate_chi2 (nscience, template, 30, len (wave_array))
            chisq_in_template.append (chi2)
        chisq_in_trial.append (chisq_in_template)
    all_chisq_list.append (chisq_in_trial)
    t1j = time.time ()
    print ('   Loop {} took {:.3f} s'.format (j+1, t1j-t0j))

t1 = time.time ()
print ('To generate the entire list took {:.3f} s'.format (t1-t0))


#Generate chi2 distributions for the template-mismatch analysis
for t in range (42):
    plt.title ('Template mismatch {} | SNR = 30'.format (t+1))
    for j in range (1000):
        if j != 999:
            plt.plot (velocities, all_chisq_list [j][t])
        else:
            plt.plot (velocities, all_chisq_list [j][t], label = 'T = {}, logg = {}, Fe/H = {}'.format (
                all_normed_templates [t][1]['teff'], all_normed_templates [t][1]['logg'],
                all_normed_templates [t][1]['feh']
            ))
    plt.xlabel ('v (km/s)')
    plt.ylabel ('$\chi^2$')
    plt.legend ()
    plt.show ()

# Find average and standard deviation of the velocities at minimum and of minimum chi squared
```

```
all_velmin_array = []
all_chisqmin_array = []
all_offsets = []

for t in range (42):
    vels_in_template = []
    chisqs_in_template = []

    for j in range (1000):
        idx = np.argmin (all_chisq_list [j][t])
        chisqmin = np.amin (all_chisq_list [j][t])
        vel_min = velocities [idx]

        vels_in_template.append (vel_min)
        chisqs_in_template.append (chisqmin)


    all_velmin_array.append (vels_in_template)
    all_chisqmin_array.append (chisqs_in_template)

    avg_chisq = np.mean (chisqs_in_template) #average chi^2
    stdv_chisq = np.std (chisqs_in_template)
    avg = np.mean (vels_in_template) #km/s
    stdev = np.std (vels_in_template) #km/s
    all_offsets.append ([avg, stdev, avg_chisq, stdv_chisq])

smallest_sigma_chisq = np.amin (np.transpose (all_offsets) [3])



# For color bar chart

all_combined_offsets = [[0.0013999999999644679, 0.394813930858574, 1.0000673673312885, smallest_sigma_chisq ]] + all_offsets
#imported from the other document

### First entry is same-same
### The standard deviation of chi squared for the first entry does not correspond to the same-same analysis

cb_idx = [science_idx]
for t in range (42):
    for templ_idx in [all_templates [t][2]]:
        cb_idx += [templ_idx]

temps = []
fehs = []

for c in cb_idx:
    temps.append (spectra_dict [c]['teff'])
    fehs.append (spectra_dict [c]['feh'])

plt.scatter (temps, fehs, c = np.absolute (np.transpose (all_combined_offsets)) [0], cmap = 'cool')
plt.colorbar ()
plt.Circle ((4600, -1.0), 100, color = 'black')
plt.title ('Average velocity at minimum chi squared, SNR = 30')
plt.xlabel ('$T_{eff}$ (K)')
plt.ylabel ('[Fe/H]')
plt.show()

plt.scatter (temps, fehs, c = np.absolute (np.transpose (all_combined_offsets)) [1], cmap = 'spring_r')
plt.colorbar ()
plt.title ('Standrard Deviation of Velocity Distribution, SNR = 30')
plt.xlabel ('$T_{eff}$ (K)')
plt.ylabel ('[Fe/H]')
plt.show()

plt.scatter (temps, fehs, c = np.absolute (np.transpose (all_combined_offsets)) [2], cmap = 'autumn_r')
```

```
plt.colorbar ()
plt.title ('Average chi squared, SNR = 30')
plt.xlabel ('$T_{eff}$ (K)')
plt.ylabel ('[Fe/H]')
plt.show()

plt.scatter (temps, fehs, c = np.absolute (np.transpose (all_combined_offsets)) [3], cmap = 'cividis_r')
plt.colorbar ()
plt.title ('Standard Deviation of chi squared, SNR = 30')
plt.xlabel ('$T_{eff}$ (K)')
plt.ylabel ('[Fe/H]')
plt.show()

# Create table with template characteristics and average and standard deviation of velocities and chi2

for t in range (42):
    print ('{} |{} |{} |{} | {:.3f} | {:.3f} | {:.3f} | {:3f}'.format ( t+1,
            all_templates [t][1]['teff'], all_templates [t][1]['logg'], all_templates [t][1]['feh'],
            all_offsets [t][0], all_offsets [t][1], all_offsets [t][2], all_offsets [t][3]
    ))



# Create average velocity vs temperature difference plot

Ts = 4600

for t in range (42):
    plt.title ('Average velocity vs $\Delta$T')
    if all_templates [t][1]['feh'] == -0.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][0], 'o', color = 'green')
    if all_templates [t][1]['feh'] == -0.5:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][0], 'v', color = 'orange')
    if all_templates [t][1]['feh'] == -1.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][0], 's', color = 'blue')
    if all_templates [t][1]['feh'] == -2.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][0], '+', color = 'red')
    if all_templates [t][1]['feh'] == -3.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][0], 'x', color = 'black')
    plt.axhline (-1.0, color = 'red', linestyle = '--')
    plt.axhline (0.0, color = 'black', linestyle ='dotted')
    plt.xlabel ('$\Delta$T (K)')
    plt.ylabel ('<v> (km/s)')

# Create velocity standard deviation vs temperature difference plot

Ts = 4600

for t in range (42):
    plt.title ('Standard Deviation of velocity vs $\Delta$T')
    if all_templates [t][1]['feh'] == -0.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][1], 'o', color = 'green')
    if all_templates [t][1]['feh'] == -0.5:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][1], 'v', color = 'orange')
    if all_templates [t][1]['feh'] == -1.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][1], 's', color = 'blue')
    if all_templates [t][1]['feh'] == -2.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][1], '+', color = 'red')
    if all_templates [t][1]['feh'] == -3.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][1], 'x', color = 'black')
    plt.axhline (1.0, color = 'red', linestyle = '--')
    plt.xlabel ('$\Delta$T (K)')
    plt.ylabel ('$\sigma$v (km/s)')

# Create average chi2 vs temperature difference plot
```

```
Ts = 4600

for t in range (42):
    plt.title ('Average chi2 vs $\Delta$T')
    if all_templates [t][1]['feh'] == -0.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][2], 'o', color = 'green')
    if all_templates [t][1]['feh'] == -0.5:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][2], 'v', color = 'orange')
    if all_templates [t][1]['feh'] == -1.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][2], 's', color = 'blue')
    if all_templates [t][1]['feh'] == -2.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][2], '+', color = 'red')
    if all_templates [t][1]['feh'] == -3.0:
        plt.plot (all_templates [t][1]['teff'] - Ts, all_offsets [t][2], 'x', color = 'black')
    plt.xlabel ('$\Delta$T (K)')
    plt.ylabel ('$\chi^2$')

# Template-mismatch analysis for two steps away

template_characteristics2 = [
    [5200, -1], [5100, -0.5], [5100, 0], [4000, 0], [4000, -2.], [4000, -3], [4300, -3], [4500, -3], [4700, -3],
    [5000, -3], [5200, -3], [5500, -2]
]
```