Draft version April 12, 2024
Typeset using LaTeX **twocolumn** style in AASTeX631

Using Hybrid Machine Learning for Fundamental Stellar Parameter Inference

Sujay Shankar,[1] Michael Gully-Santiago,[1] and Caroline V. Morley[1]

[1] *Department of Astronomy, The University of Texas at Austin, Austin, TX 78712, USA*

ABSTRACT

Text

Keywords: Text

1. INTRODUCTION

Stellar spectra are exceedingly rich sources of information about the stars which create them. They contain information about the fundamental properties of the star such as its temperature and surface gravity, as well as countless absorption lines reflecting the star's chemical composition. On top of that, extrinsic factors such as the radial velocity of the star or the rotation of the star itself further transform spectra. When these spectra are observed, even the location of the observing instrument and the capabilities of the instrument itself introduce further modifications. In short, stellar spectra represent extremely complex, high-dimensional data that has gone through multiple transformations before reaching our eyes as a deceptively simple graph of flux versus wavelength.

Naturally, scientists want to extract as much information as possible from a star's spectra to get back the properties that lie at the root of it all. However, this has proven to be no easy task. Spectroscopic surveys such as APOGEE develop their own in-house pipelines to extract stellar parameters from spectra (ASPCAP in the case of APOGEE), and these pipelines are fairly effective. However, they all work on the core assumption that the data to be analyzed is a list of pixels. In addition, the pipelines that surveys develop are closed-source and usually limited to the scope of the survey itself, causing a sort of fragmentation. This motivates the development of a more universal, open-source tool that can do what spectral analysis pipelines do, but in an alternative way.

Multiple efforts have been made in this direction, treating spectra in different ways. Some take the wavelength and flux values and construct neural networks to create this mapping between stellar parameters and the observed spectra, such as `IN-SYNC`. Others take a more physics-informed approach by decomposing spectra into some eigenbasis and treating it as a regression problem, such as `starfish`. However, `blase` takes a different approach. It treats spectra not as a set of pixels or a set of eigenbasis coefficients but as a set of spectral lines, specifically Voigt profiles. There is no metric of this approach being better or worse than the others mentioned so far, but it is a different paradigm and is worth exploring as a semi-empirical tool for spectral inference. This study aims to take advantage of `blase`'s ability to change a spectrum's basis from flux versus wavelength into spectral line versus spectral line properties, and use that to develop a robust tool for stellar parameter inference.

However, this is only a proof of concept. Here we develop a tool that represents a very basic realization of the idea of line-by-line inference. We make no claims as to the performance of this tool, but rather aim to show that the idea is sound and able to be implemented, even if rudimentary, and that with further research and development, this tool has the potential to become useful for spectral inference.

We chose the PHOENIX synthetic spectral model grid as the basis for this study, because it is fairly well-known. While we plan to extend this tool to leverage information across multiple model grids in the future, both for accuracy and to increase the scope to include substellar objects with grids such as Sonora, for now we limit our scope to a subset of the PHOENIX grid.

2. CLONING THE PHOENIX MODEL GRID

2.1. *The PHOENIX Subset*

For the purposes of this study, we did not consider the full range of the PHOENIX synthetic spectral model grid. Instead, we focused on a subset of the grid, whose parameter ranges are given in Table 1.

2.2. *Preprocessing with `gollum`*

PHOENIX Grid Subset		
Parameter	Symbol	Range
Alpha Element Abundance	α	[0] dex
Iron Abundance	[Fe/H]	[-0.5, 0] dex
Effective Temperature	T_{eff}	[2300, 12000] K
Surface Gravity	$\log(g)$	[2, 6]
Wavelength	λ	[8038, 12849] Å

Table 1. The subset of the PHOENIX grid used in this study. These limits were imposed to reduce the computational cost of the algorithms and to ensure a rectilinear parameter space in order to work with the chosen interpolator backend. The wavelength limits in particular roughly line up with that of the Habitable Zone Planet Finder (HPF) spectrograph.

First, the PHOENIX subset was accessed directly from the PHOENIX website using `gollum`'s download option, which uses the website's FTP server to download spectra without using any disk space. This came at the cost of longer load times due to internet connection being significantly more of a bottleneck than disk I/O, but was acceptable as this was a one-time operation.

The spectra were then put through a three-step preprocessing pipeline.

1. Blackbody Division: Since the T_{eff} of each spectrum is known, the according blackbody spectrum was divided out.

2. Percentile Normalization: The spectra were normalized by dividing them by their 99th percentile.

3. Continuum Normalization: The spectra were further normalized by dividing them by a 5$^{\text{th}}$ order polynomial continuum.

2.3. *Line Identification with `blase`*

The next step was to convert the PHOENIX subset into an interpretable format. We wanted to represent the spectra as a list of spectral lines rather than a list of fluxes. This was done using `blase`, which detects spectral lines as Voigt profiles and tunes the profiles to mimic the original PHOENIX spectrum with back propagation. Four parameters were optimized: the line center μ, the log-amplitude $\ln(a)$, the Gaussian width σ, and the Lorentzian width γ. The optimization used the Adam optimizer with a learning rate of 0.05 over 100 epochs. In addition, we limited two custom parameters: wing cut to 6000 and prominence to 0.005. Wing cut is a parameter that determines the extent of the Voigt profile to evaluate, saving computational resources by not evaluating small numbers. Prominence sets a lower limit for the amplitude of detected lines, which also saves resources by disregarding shallow lines. In short, larger values for wing cut and smaller values for prominence both increase the accuracy of using `blase`'s line detection and optimization at the expense of an increased computational cost.

Once optimization was complete, the list of identified lines, present in the state dictionary of `blase`'s model, was saved to a `.pt` file for each PHOENIX subset grid point.

3. INTERPOLATING MANIFOLDS

3.1. *Cross-Model Line Identification*

Earlier we mentioned that `blase` tuned the line centers of detected lines. This means that from one PHOENIX spectrum to the next, the same line could have a slightly different line center. Since the goal of this study is to interpolate the properties of each line, we needed to identify the presence of a particular line across the PHOENIX subset. We decided to do this by using the line centers of the detected lines pre-optimization. This introduced a new variable called the jitter $\delta\mu$, which measures the shift of the line center between pre-optimization and post-optimization. Now with each spectral line indexed by μ, we had four parameters to interpolate: $\delta\mu$, $\ln(a)$, σ, and γ.

There was also a second issue, and that is that spectral lines were often only detected in some spectra from the PHOENIX subset. This means that in theory, interpolation would be inaccurate in regions where a line does not appear, and also breaks in practice because `scipy`'s regular grid interpolation relies on a rectilinear grid, which we would not have if regions of grid points were missing. To solve this, we artificially populated missing grid points with log-amplitudes of -1000, which retained interpolator stability by not being an infinity, but also essentially nullified the line when evaluated.

3.2. *Continuously Evaluable Manifolds*

For each line, the inputs to the interpolator were the three input parameters T_{eff}, $\log(g)$, and [Fe/H], and the output was a list of four parameters, $\delta\mu$, $\ln(a)$, σ, and γ. For each line, one of these interpolator objects was created using linear interpolation, and these interpolators were aggregated into a single list, which was when pickled and written to a `.pkl` file. These interpolators represent multiple manifolds mapping fundamental stellar parameters to spectral line properties. These interpolators could now be evaluated at any point lying within the domain of the PHOENIX subset, turning a discretely sampled PHOENIX subset into a continuous one. With the given size of the PHOENIX subset, the interpolator list takes up 13.2 GB of disk space. Since this is now evaluable, we call this the PHOENIX generator.

4. BAYESIAN INFERENCE AND TESTING

4.1. *Inference Algorithm*

With the PHOENIX generator complete, we used Bayesian optimization as the inference algorithm, specifically the `gp_minimize` function from the `scikit-optimize` library. This algorithm uses a Gaussian Process to model the objective function, which in this case was the RMS (Root-Mean-Square) loss between the interpolated spectrum and the true spectrum. The optimizer was configured to first run 30 random evaluations to seed the surrogate model, then run 20 more evaluations now guided by the surrogate model. This totals to 50 evaluations, which was deemed sufficient for this study. We then decided to use ensemble methods, running 32 parallel optimizers to get a more statistically sound result. One inference run takes an average of [TIME] to complete on Triton.

4.2. *Test Data*

To test the performance of the inference algorithm, we used the PHOENIX subset itself. At first glance, this may seem like a useless endeavor, as the PHOENIX generator has indeed memorized the PHOENIX subset, being able to reconstruct a PHOENIX spectrum when evaluated at that grid point. However, that is precisely what allows us to use the PHOENIX subset as test data. The Bayesian optimizer's surrogate model is seeded by random continuously sampled generator evaluations within the search space, *not* grid points of the PHOENIX subset, meaning the surrogate model has no 'memorization' to speak of. If the optimizer had been an iterative grid refinement search, this would not have been possible, because grid methods rely on just that, the grid.

4.3. *Performance Results*

Text

5. DISCUSSION

The computer used for this study, Triton, has the following specifications:

Triton Specs	
CPU	AMD EPYC 7513
Memory	256 GB
GPU	Nvidia A100 40GB (×2)

Table 2. This computer is owned by Caroline Morley and is available to members of her research group. It was used for all computations, but not for generating visualizations. The EPYC 7513 is a 32c/64t CPU with a boost clock of 3.65 GHz.

It should be noted that this study represents a proof of concept, and that there are numerous design considerations that could be improved upon with future work. These considerations include but are not limited to the following:

- *Limited PHOENIX Subset*: The PHOENIX subset used in this study was just that, a subset. The full PHOENIX grid not only expands the [Fe/H] range to [-4.0, 1.0] dex and the log(g) range to [0, 6], but also includes the alpha element abundance parameter, which we elected to fix at 0 for this study. In addition to the actual fundamental stellar parameters, we also took a subset of the PHOENIX wavelength range, with the full [500, 55000] Å wavelength range also being left to future work.

- *Strict Wavelength Range*: Currently, the generator only supports inference on spectra whose wavelength limits are either equal to it, or encompass that of the generator and have been truncated to match. However, when the spectrum in question has a smaller wavelength range than the generator, currently there is no functionality to truncate the generator. This would require externally indexing the generator's individual interpolators by line center position and selectively evaluating those to eliminate wasteful computation.

- *Low-Order Interpolation*: The current design of the algorithm uses linear interpolation to construct manifolds, however this could be made more sophisticated with more complex interpolation schemes. However, the drawback of this is that higher orders of interpolation become increasingly expensive both in terms of disk utilization, and in terms of the computational cost to generate and evaluate them. Future studies should sidestep this issue by using more generalizable methods that are much more efficient.

- *Single Model Grid*: The PHOENIX grid is not the only model grid of synthetic spectra available, and it does not apply to all types of stars. Future work would extend the reach of this study's algorithm to encompass other model grids such as the Sonora series for substellar models. `blase` should be able to have an option for the user to input which model grid they would like to base the inference on, and to get even more advanced, perhaps even have the ability to intelligently determine which model grid to use automatically.

- *Memorization vs. Generalization*: The current design of the algorithm constructs manifolds using interpolation. This means that performance is good at points close to PHOENIX subset grid points, but is highly dependent on the type of interpolation used. As interpolators require memorization of the data, advanced interpolation becomes extremely expensive in terms of disk utilization. Future work would involve constructing manifolds using regression, which would allow for much better generalization and lower disk utilization at the expense of some accuracy.

- *Extrinsic Absence*: The current design of our algorithm does not account for extrinsic parameters that modify the appearance of spectra such as rotational broadening and doppler shifting. Future work would need to develop ways to tune these extrinsic parameters alongside the fundamental stellar parameters.

- *Framework Overhead*: As this algorithm is currently more proof of concept than practical, it uses convenience functions from various libraries, which naturally introduces some level of overhead and leaves performance on the table. Future work would involve writing custom functions expressly designed for `blase`, most likely a complete rewrite of the library from the ground up.

- *Pseudo-Interpretability*: Our algorithm boasts interpretability by considering spectral lines as the objects of interest as opposed to the rather uninterpretable flux values of other approaches. However, this is only a step in the direction of interpretability. True interpretability would decompose a spectrum not into a set of spectral lines, but into a set of species component spectra, which requires a much more advanced understanding of different species and their behavior, as well as direct access to a radiative transfer code as opposed to an off-the-shelf model grid. This approach would also extend the inference from just fundamental stellar parameters defined by a grid to any set of parameters accounted for in the radiative transfer model, down to specific species abundances.

- *The Continuum Black Box*: Continuum normalization is a process that is not yet completely understood, and is currently done as a preprocessing step with a fairly simple algorithm. Future work would dive deeper into the science of continuums and develop more advanced methods that can discern continuums with greater accuracy and less modeling restrictions.

- *One Voigt Fits All*: The current assumption of `blase` is that every spectral line is a Voigt profile. This assumption is largely true, but there are situations where that is simply not enough. Future studies need to account for more advanced spectral line profiles and procedures to deal with phenomena such as ro-vibrational bands.

In short, it is quite clear that `blase` is a long way from being a powerhouse in spectral inference, however it represents a step down a road not yet traveled, and the potential for future growth is immense. At the end of the day, we aim to develop a tool that can analyze spectra with the advent of physics-informed machine learning.

ACKNOWLEDGEMENTS

Text