0. Title: your title should start with the package name, feel free to use a Combination Title, e.g. NEMO: where did the fish go? Author: don't forget to author your work, a date is always nice too.

PySpecKit: Discovering the Secrets of Stars
Jasmin Mohammadi, 05/09/25

1. **Name of the package, describe what is the basic aim of what the package does or solve?**

The PySpecKit package is a toolkit based in Python used to complete spectroscopic analysis, typically for optical, infrared, and radio spectra. It can take in a variety of data files, ranging from FITs to CLASS files, and computes fits using gaussian, voigt profile, or continuum fitting. The fits this toolkit creates are plotted using matplotlib as its underlying basis, which can also be used to compute the error bars on this model. PySpecKit uses the Levenberg-Marquardt optimization method to get the most accurate fit possible. The goal of this package is to make spectral fitting more user-friendly and easy for the user to understand and was created to address the lack of spectral fitting tools for various wavelengths.

2. **Why/how did you select this package?**

As a physics and astronomy major, I have always been very interested in stellar composition and evolution, as astronomers learn almost everything they know from light. Understanding this light through spectroscopy is vital to understanding deeper questions about our Universe. I also have a bit of experience in analyzing spectral data from stars through the research I have been working on. I am working to use a similar Python package called SEASAMME to fit the age, mass, metallicity, and extinction of star clusters in a handful of galaxies that have a high redshift using spectral analysis. I am interested in seeing how another package does a similar, but slightly different, process.

3. **How old is the package? does it have a genealogy, i.e. what related codes came before or after. Are there other codes you can find that solve the same problem?**

The development of PySpecKit began in 2009 and it was released in September of 2011, making it about 14 years old. Before this package, the Image Reduction and Analysis Facility (IRAF) was often used to compute line-fitting, but in recent years it has stopped receiving updates. PySpecKit uses a similar GUI to IRAF's plotting tool and fits similarly to Xspec: A X-ray Spectral Fitting Package but for a wider variety of wavelengths than just X-ray. A similar code to this is Specutils, which is a Python package that can be used to analyze astronomical spectroscopic data. It can be used to plot spectral data from stars and galaxies.

4. **Is it still maintained, and by the original author(s)? Are there instructions on how to contribute to this project?**

PySpecKit is still maintained and is done so by Adam Ginsburg, the main (and original) author of this code. You can contribute to this project through their Github issues page (https://github.com/pyspeckit/pyspeckit/issues), which allows user to submit comments about errors and issues that they have run into while also creating a space for people to comment on this package all together. Users are also encouraged to contact the creators at pyspeckit@gmail.com with any issues.

5. **Evaluate how easy it was to install and use. What commands did you use to install?**

This package was easy to install through a simple "!pip install pyspeckit". This package was also pretty easy to use, allowing users to create their own spectra to analyze or import data and compute a multitude of fits depending on the spectrum they were dealing with. The Github provides many examples of fits people have done using pyspeckit, giving many places for users to turn to for help.

6. **does it install via the "standard" pip/conda, or is it more complex?**

PySpecKit is installed via a pip install.
(!pip install pyspeckit)
Import pyspeckit

7. **Is the source code available? For example, "pip install galpy" may get it to you, but where can you inspect the code?**

Yes, the source code is available [https://github.com/pyspeckit/pyspeckit].

8. **is the code used by other packages (if so, give one or two examples). ASCL codes have citations via their ADS link. See also 22.**

The code is used in other packages, such as Astroclover and Pyspecnest. These two utilise the fitting tools in Pyspeckit. Astroclover identifies spectra using the line-of-sight velocity components and Pyspecnest uses the nested sampling components of Pyspeckit. SCOUSEPY also uses Pyspeckit's cube fitting module to compute their own spectral analysis. There are also several different packages out there that do not use pyspeckit but are extremely similar to it.

9. **How is the code used? Is it command line, python script, or a jupyter notebook, or even a web interface?**

The code is given in python scripts but can be used in a jupyter notebook. It can be catered to the different types of spectra the user wants to fit. It will take in whatever data you give it, assuming it is in a file form that it accepts (FITS, plain text, and hdf5), and will compute the model you want to apply to it based on what you specify in the fittype = command.

10. provide examples using the code. if you prefer to use a jupyter notebook instead of a python script, that's ok. See also 12.

```
import numpy as np
import pyspeckit
import matplotlib.pyplot as plt
from astropy import units as u
```

```
data = np.genfromtxt('ngc253_hb.tab')
```

```
wavelength = data[:,0]
flux = data[:,1]
```

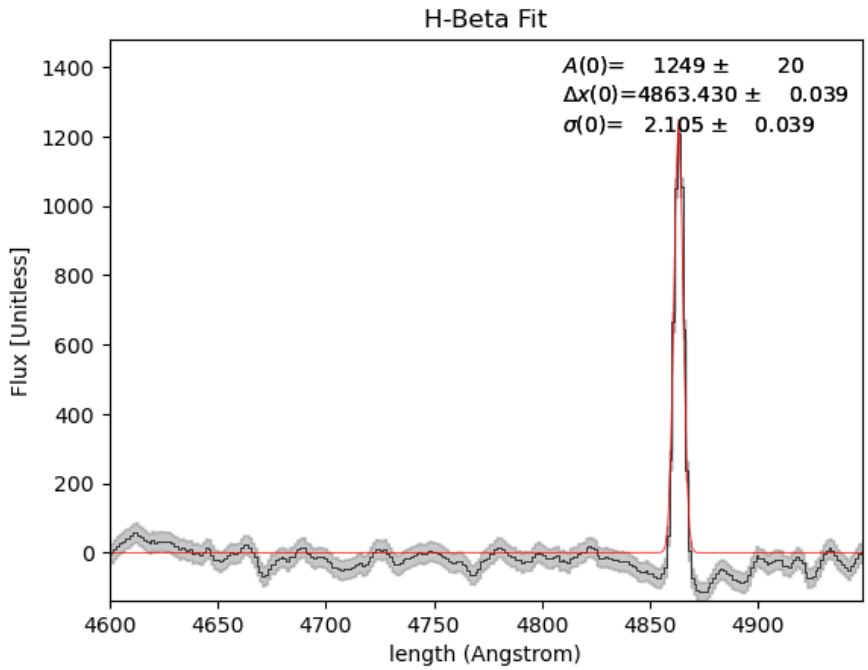
```
wavelength = data[:,0]
sigma = 10.
center = 50.
flux = data[:,1]

# Add noise
stddev = 0.1
noise_1 = flux[:200]
noise_2 = flux[217:]
noise = np.concatenate((noise_1,noise_2))
error = np.std(noise)
print(error)

error_array = error*np.ones(len(flux))

# this will give a "blank header" warning, which is fine
sp = spec.Spectrum(data=flux, error=error_array, xarr=wavelength,
                       xarrkwargs={'unit':'AA'},
                        unit='Flux [Unitless]')

sp.plotter()

sp.baseline()

# Fit with automatic guesses
sp.specfit(fittype='gaussian')

# Fit with input guesses
# The guesses initialize the fitter
# This approach uses the 0th, 1st, and 2nd moments
amplitude_guess = flux.max()
center_guess = 4863.45849609375
width_guess = 5
guesses = [amplitude_guess, center_guess, width_guess]
sp.specfit(fittype='gaussian', guesses=guesses)

sp.plotter(errstyle='fill')
sp.specfit.plot_fit()
```

With this assignment I have also attached my entire jupyter notebook of the code I used.

11. Does the package produce figures, or are you on your own? Is matplotlib used?

Pyspeckit does produce figures. Matplotlib is used behind the scenes, as it was used to build the GUI, but does not need to be imported directly to work correctly. Plotting with this package can be done with sp.plotter(), assuming sp is a function where you define the data, the errors, and the x array you want this data to be plotted on. The axes of the plot are labeled using LATEX.

12. **your code and report should show at least one figure, and create a nice figure caption explaining what it shows. Your notebook should show how the figure was made (i.e. be reproducible). Second figure is optional, but only use it when you need to illustrate something extra.**



This is a plot showing our fit for the H-Beta line from NGC253, the Sculptor Galaxy. Our gaussian fit is shown in red, the data is the black line, and the grey is the error. On the y-axis we are plotting flux in some unknown units, as noted by the MUSE database, and on the x-axis is wavelength in angstroms. The peak in the plot shows the h-beta line. The error in this data is computed by taking the standard deviation of the continuum and applying that to each point.

13. **Is the package pure python? or does it need accompanying C/C++/Fortran code?**

This package is purely python and does not require any accompanying coding languages.

14. **What is the input to the package? Just parameters, or dataset(s), or can they be generated from scratch?**

The input in this package is the data file you provide that contains the spectrum. It should have data containing flux and wavelength. However, this package will also compute fits on datasets created from scratch, so long as it appears how normal spectral data would (i.e. including noise).

15. **What is the output of the package? Just parameters, or dataset(s)?, or just a screen output you would need to capture**

This package outputs a plot with the fit applied as well as the values the fit finds. It gives the amplitude, position, and width of the h-beta line, all with their corresponding uncertainties.

16. Does the code provide any unit tests, regression or benchmarking?

On the pyspeckit github there are some actions set up under the Actions tab that automatically tests the code to ensure it is still working for different versions of Python. There is also a part of the pyspeckit code on github called main.yml, which describes different tests that it runs. Every Tuesday at 5 am UTC, it tests the code and checks that it works for the latest Python versions. I have attached a link to this page here. (https://github.com/pyspeckit/pyspeckit/blob/master/.github/workflows/main.yml)

17. How can you feel confident the code produces a reliable result? (see also previous question)

This code is kept up to date and tested often, ensuring it is working properly. Pyspeckit also provides the errors from the fit on the values being calculated, which can tell you the quality of the fit if the uncertainties seem reasonable. It can also be helpful to follow the methodology the code is using and seeing if it makes sense logically. If the process seems convoluted and incorrect, there is a good chance the result is not reliable.

18. what (main) python package(s) does it use or depend on (e.g. numpy, curve_fit, solve_ivp) - how did you find this out?

This package mainly relies on numpy, astropy, and matplotlib. Much of this information about the package can be found on the paper they published on ADS (https://iopscience.iop.org/article/10.3847/1538-3881/ac695a/pdf). In this paper it states that pyspeckit is a part of the "astropy affiliated package ecosystem". Astropy is also the basis for the unit system in this package. The paper also details how the GUI was built using matplotlib's tools. It is also listed on their Github in the readme file that plotting is done in matplotlib-based codes and states that astropy, matplotlib, and numpy are requirements to this package. Numpy can also be useful with this package as it can be used to create a spectrum from scratch. Many of the errors from this type of fitting are also calculated using numpy in the background.

19. What kind of documentation does the package provide? Was it sufficient for you?

This package comes with an entire Github repository with documentation upon documentation of how to use this package. I did struggle a bit at first as this was my first time ever looking at a package this way, but once I understood where to look, their Github was easy to navigate and had a solution to every problem I faced. The Pyspeckit Github also links you to their personal website which has another git repository where users can list any errors they ran into and converse with others on how to fix it. Within the git repository, there are also dozens of examples to follow of different fits people have computed.

20. if you use this code in a paper, do they give a preferred citation method?

The preferred citation method for this package is to cite the paper they published on ADS: Pyspeckit: A Spectroscopic Analysis and Plotting Package.

21. **provide any other references you used in your report.**

A link to their ASCL website: https://ascl.net/1109.001
A link to their personal website: https://pyspeckit.readthedocs.io/en/latest/
Astroclover: https://github.com/jakeown/astroclover
Pyspecnest: https://github.com/vlas-sokolov/pyspecnest

22. **Can you find two other papers that used this package? E.g. use ADS citations for ASCL based code. See also 8.**

This package is used in two papers: *Galactic H_2CO Densitometry. I. Pilot Survey of Ultracompact H II Regions and Methodology* and *A Large Systematic Search for Close Supermassive Binary and Rapidly Recoiling Black Holes. IV. Ultraviolet Spectroscopy*. Pyspeckit is cited 62 times on ADS (https://ui.adsabs.harvard.edu/abs/2022AJ....163..291G/citations).

23. **Did you have to learn new python methods to use this package? Or was the class good enough to get you through this project.**

The class was enough to get me through this project. None of the code in the package seemed too out of my grasp. While it did require thorough reading of the documentation and the source code, I was able to understand how the code worked and what each component was doing. It works fully in packages that we have explored before, the only new thing was understanding the pyspeckit documentation and how to use the plotter function.

24. **Final Disclaimer: you need to state if you have prior experience in using the package or the data, or this is all new to you. In addition, if you collaborated in a group, as long as this is your work.**

This is my first time working with this package and I collaborated in a group with Debika Biswas, Meg Haswell, and Zya Woodfork.