ADACS hackastron challenge

Challenge description:

Decomposing complex spectra made up of multiple, potentially overlapping Gaussian components is a challenge for Galactic and extragalactic astrophysics, but is critical for understanding the properties of gas within galaxies. Our team is currently using a code called GuassPy (Lindner et al. 2015) that autonomously decomposes spectra into Gaussian components. which improves upon previous methodologies (namely by-hand decomposition) by removing the need for human input (and subjectivity!), increasing the reproducibility, and allowing for the decomposition of large samples of data. GaussPy identifies different Gaussian components by finding inversions in the second derivative of the data, but since the data includes noise that mimics inversions, it uses a machine learning algorithm to determine the optimal parameters for first smoothing the data. For lower signal-to-noise data, the maximum confidence level of the decomposition produced by GaussPy is low (at best ~70%, Murray et al. 2017) and primarily due to the difficulty in separating the noise from the spectral signal using only smoothing. We would like to improve the performance of the Gaussian decomposition algorithm by first devising a way to isolate the noise from the spectral features or ''de-noise'' the spectra (e.g. using wavelet transforms) in order to improve the confidence level of the decomposition. Such an application will be extremely valuable to use on the large number of spectra produced by the upcoming large all sky surveys such as GASKAP and WALLABY. Additionally, an easy to use ''de-noising'' software would be useful for other astronomical data.

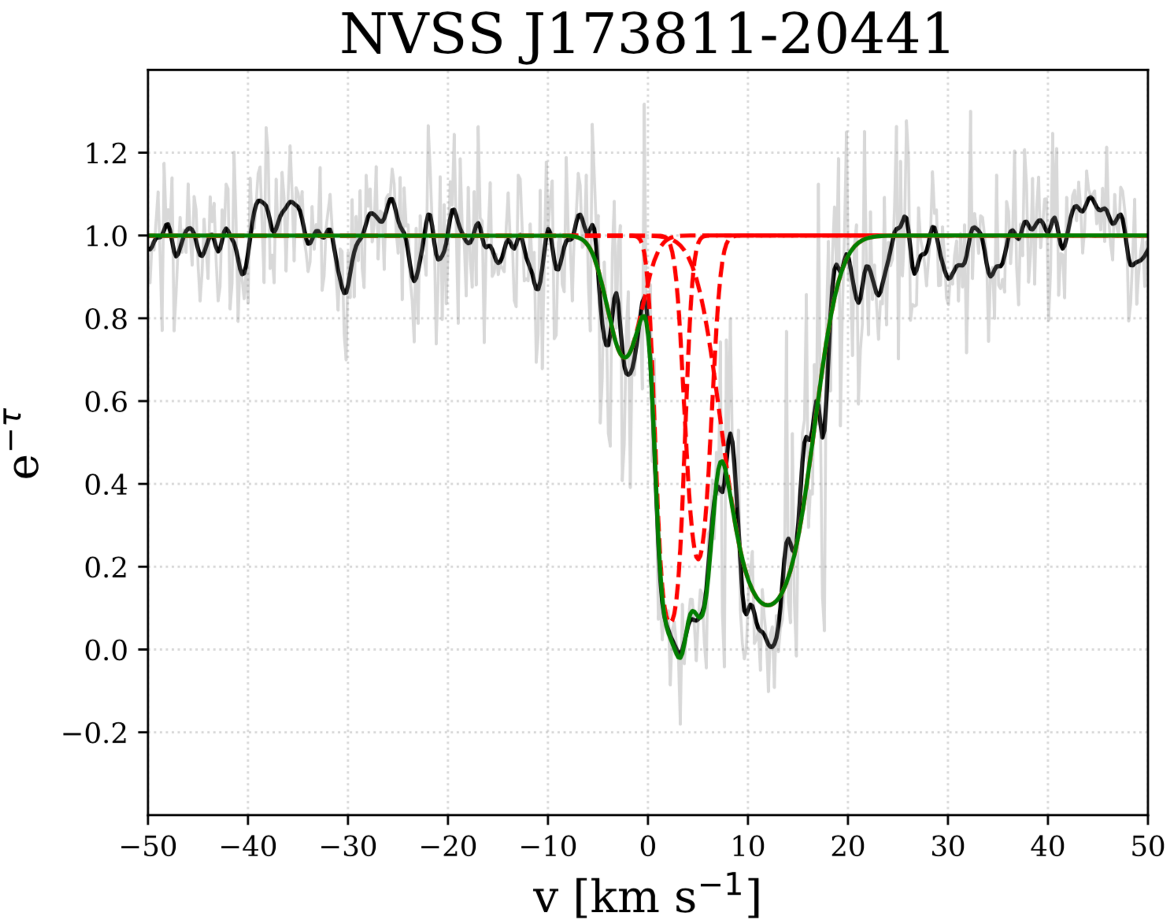
What we know about the signal:
The spectra or signal is composed of several Gaussian components. The components can vary between (i) deep, narrow Gaussians to (ii) shallow, broad Gaussians. Where the deep components represent cold gas and the shallow components represent warm gas. The spectra was sampled from a region of the sky where we know that there is a significant amount of cold gas present. This means that the spectra should have several deep and narrow components. We also know that these narrow components can be close to each other and blend together which will produce a signal that looks like a deep and wide Gaussian.

The goal:
To build a program that can denoise the spectra without loosing features of the signal.

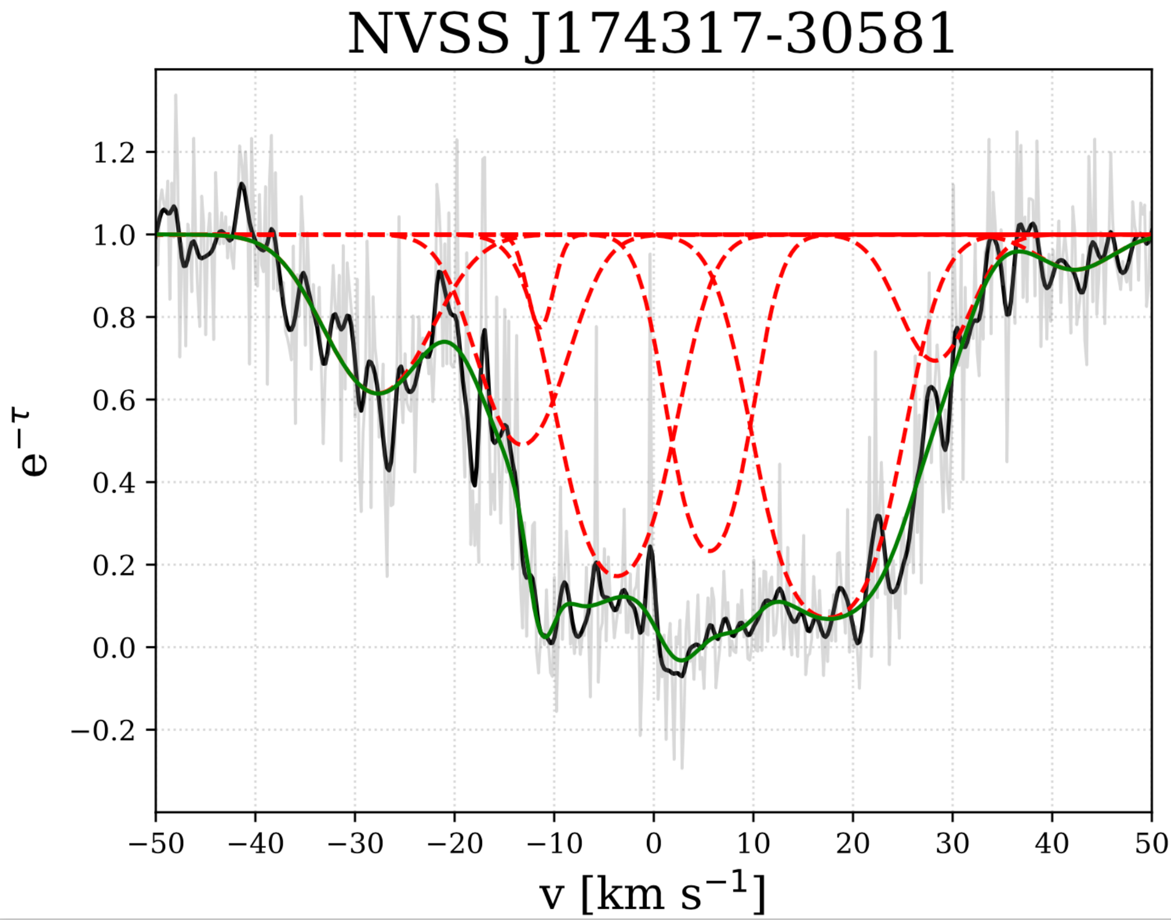
Our spectra have a varying level of signal to noise, some spectra will need less denoising compared to others. The resulting solutions should also incorporate this aspect.

Examples:
This figure illustrates the problem. We smoothed the spectra (grey) with a Gaussian kernel (black) and decomposed the smoothed spectra with AGD (red - individual components, green - overall fit). AGD identified 4 Gaussian components, but there are probably more than 4 components in the spectra. The “deep” absorption feature on the left (at ~5 km/s) is decomposed into two narrow components, but the “deep” feature on the right is only decomposed into one broad component. Typical line widths for the deep (cold gas)

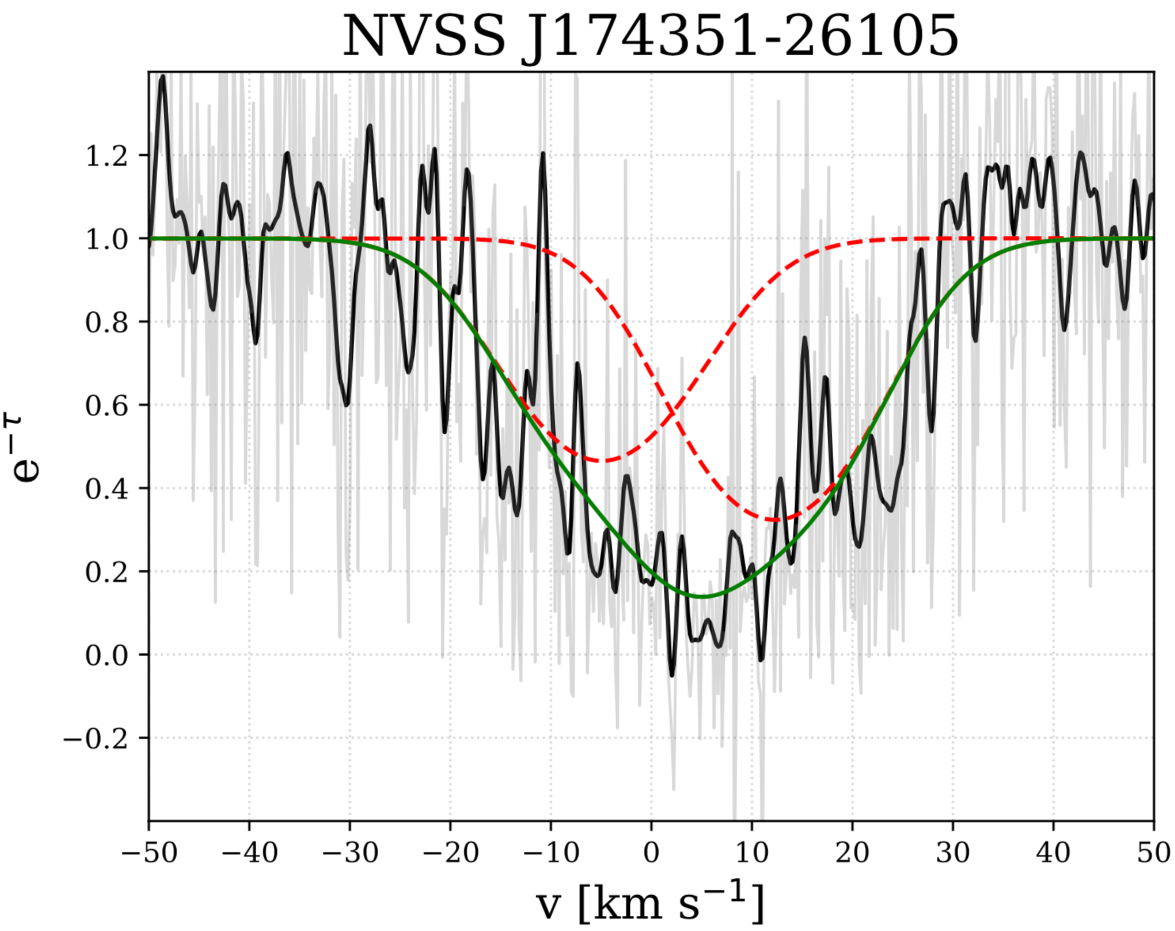
components should be a few km/s. Based on the physical properties of the interstellar medium, we expect that the feature on the right should also be composed of more narrow components, similar to the feature on the left. We think that with a better denoising method we could archive a decomposition that would give us more physical results. A better decomposition would ultimately aid us to better understand the structure and temperature distribution of the hydrogen gas in the interstellar medium (ISM).



This spectra is another example that illustrates the difficulty of distinguishing multiple components. Again, the deep (amplitude ~ 1) components are a bit too broad to be physically realistic (10-15 km/s FWHM instead of ~5 km/s).



In this case the signal to noise is so low that the decomposition gives entirely unusable results.



Additional information:
To decompose the spectra into individual components we use a Python software called GaussPy. More information about GaussPy can be found in this paper: Lindner et al. 2015, ApJ, 149, 138 (http://iopscience.iop.org/article/10.1088/0004-6256/149/4/138/meta, or https://arxiv.org/abs/1409.2840). GaussPy is available on Github: https://github.com/gausspy/gausspy.