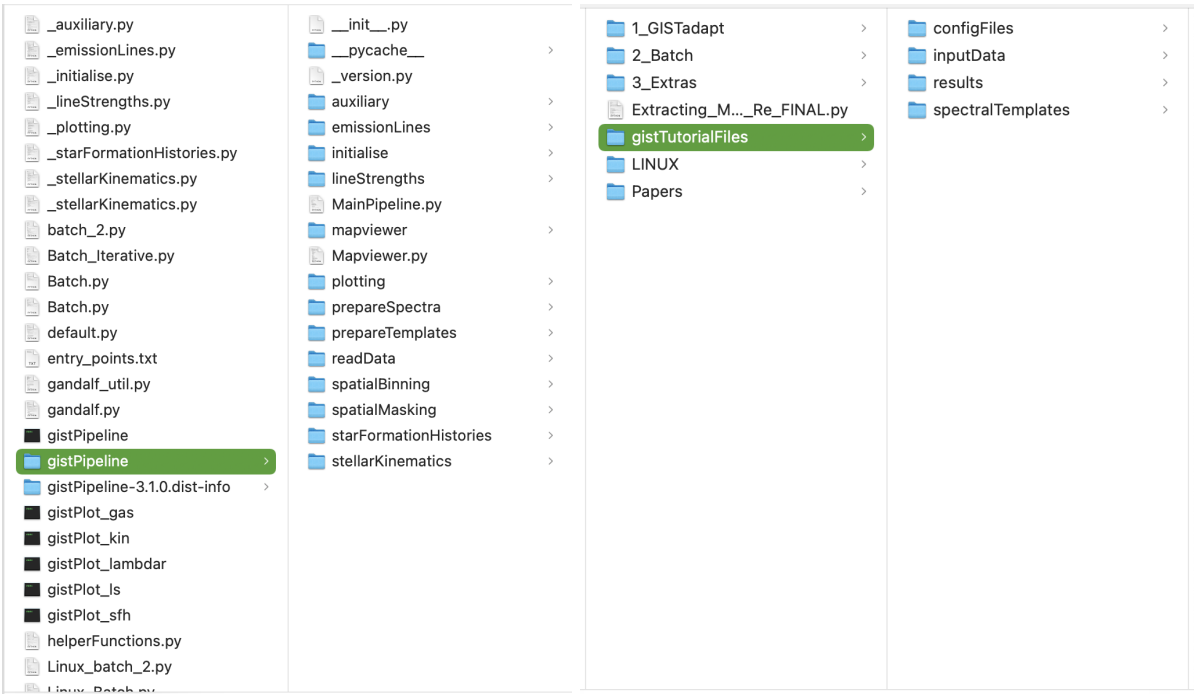
Hello, welcome to my life's work! I hope you treat it as such and keep the judgment very much to yourself, unless, of course, you're here to improve it. As you will see, coding is not where I'm strongest, so that's where I hope the person reading this document decides to start making changes. Granted, I don't really think there is any other place to start except reading my thesis, which I hope you've done by now (here's a link: https://etd.library.emory.edu/concern/etds/1j92g8682?locale=en). I can see how many people download it so I will not be fooled. Anyway… since you've read this far I assume you're committed to the cause of the BRAVEs. NOT baseball, please. We're muuuch more important than that. This document presents to you the structure of the files and will serve as your guide as you navigate… you guessed it! My life's work. Good luck, and most of all, ENJOY. There is no greater joy than learning blah blah blah I'm sure you've heard this before so I'll skip it. But in all seriousness, this is one of the cooler projects happening anywhere so consider yourself lucky.

TIP: PLEASE READ CAREFULLY, not everything important is bulleted, underlined, capitalized, italicized, or in bold. There is a descriptive element to this document to make reading this less of a pain, but there are no wasted paragraphs.

The structure is the following:

1. **Codes for GIST adaptation**
    a. SAURON_LR
2. **Codes for Batch Processing**
    a. Simple Batch
    b. Iterative Batch
3. **Supplemental Files**
    a. MasterConfig + Doc
    b. Spectral Masking
    c. Other Useful Codes

Before we get started, there are two main sets of directories that you'll need to be aware of: one is the GIST folder with all the GIST codes, that you will download inside your environment (more on this in a bit) and the other is the directory with the GIST inputs (i.e. where your galaxy data/MasterConfig would go). You can find much more detailed information about the pipeline in their online documentation (https://abittner.gitlab.io/thegistpipeline/) or their paper (see GIST_Paper.pdf).



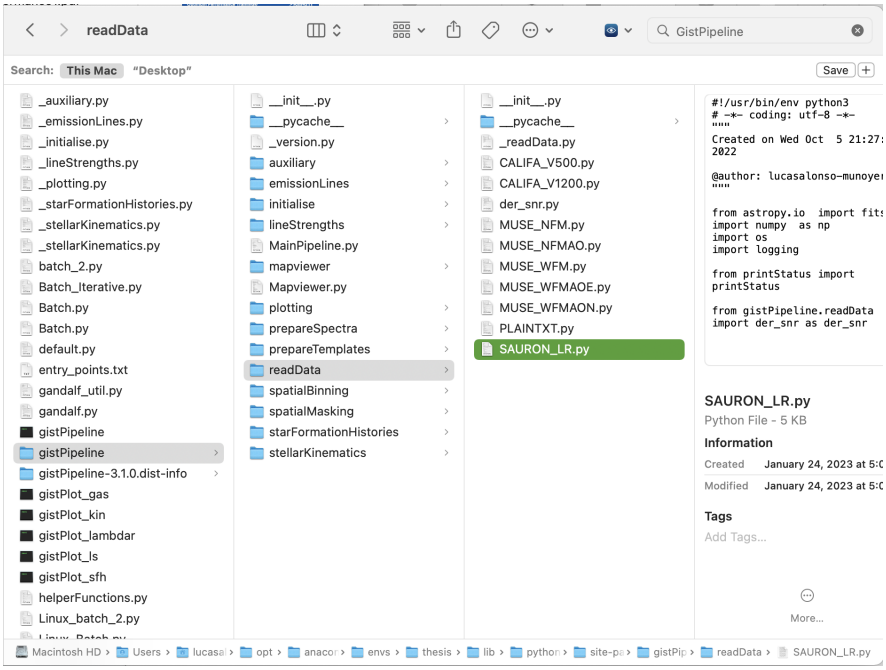
Here, the first image (left) is to show the GIST directory that should be inside your python environment (from now on referred to as /GIST) and the second is a documentation sample directory where the inputs and outputs can be found. The latter is where you would have the data cube (/inputData), the MasterConfig (/configFiles), and spectral masking (/configFiles). There are other files in there too but these are the main ones.

TIP: If you want to make your life easier you can use the /GistTutorial directory that I put in here to get all the files. CAREFUL. You need to replace the MasterConfig and spectral masking files with your own and add your data cube (that you will specify in the MasterConfig). For more information on the MasterConfig, read the supplemental document that tells you what I used to analyze the galaxies.

TIP 2: To edit a MasterConfig file, such as the template file I give you, you need a text editor that preserves spacing, etc… I used Acquamacs, but feel free to use whatever you want. If you're using MacOS, DO NOT use the default TextEdit.

1. Codes for GIST adaptation

Here you will find the code that will literally CARRY your research. Without it, you can't read any of the SAURON data cubes. It is an adaptation of the MUSE_WFM code that comes with the pipeline. To use it, you have to add it to /GIST/readData and you would have to specify it in the MasterConfig.



Either way, the general idea of the code is as follows:

1. Read Cube
2. Extract the spectra + de-redshift the spectra by using your estimate for rs
3. Choose spectra within wavelength range specified
4. Set coordinates, compute signal and noise & SNR per spaxel
5. Store into new cube

First point of improvement here: As I was writing this document I noticed that I had never created a version of the code that would allow you to re-analyze the output files from GIST. This would be especially helpful if you're looking to pPXF it up in your areas of interest as is mentioned in the concluding remarks of the thesis. I would start by looking at my plotting routine that allows you to match the BIN_IDs to the position. You'd need to specify these things in your readData routine as well as change the MasterConfig to read the new data/wavelengths. This is a big project and you'd have to figure out which spectra you would use. It goes beyond the scope of my project so… good luck!

2. Codes for Batch

Here we find general codes that can be used to batch process a set of galaxies. In order to do this you would need a .csv file that contains columns for the data that needs to be changed for each (see supporting .csv file– BatchConfigInfo.csv). The LINUX batch code is the simple code that uses this attached BatchConfigInfo.csv file and runs the code assuming you're in my LINUX

directory, which you can if you want to by accessing the SSH. I highly recommend using FileZilla, a free software that allows you to easily see you computer and the linux directories as well as easily transfer from one to the other. It will make your life 1000000000x easier if you're not a computer GOD and have the ability to connect your brain to the web.

Important headers for .csv file columns in Batch.py and Batch_Iterative.py as shown in the codes. The variable names are made this way to easily find and replace them on the MasterConfig temp file.

'AAAA' = 'RUN_ID'
'BBBB' ='INPUT'
'CCCC' = 'REDSHIFT'
'DDDD' = 'SIGMA'
'EEEE' = 'MC'

Make sure that this .csv file only contains rows with galaxy information and no blank rows. This small problem made me go crazy for a few weeks, for the code ended up failing without much explanation. If you want to just run a code in the background, I recommend using "nohup".

Example:

nohup python3 batch_2.py &

In gray is whatever your command may be and in black the "nohup" command terms. Feel free to look up more ways to use it, I just found this one particular command especially helpful.

The code is then called from your environment. **DO NOT** attempt to run the code from anywhere else. This is because the batch code will try to run GIST without first activating any environment and so **it will fail** if you don't do so beforehand. The iterative code is still a work in progress. I believe it should work so feel free to give it a shot. My last problem was with the final iteration of GIST. I was unable to see that _final_iter folder. With these results, the pipeline plots will not work but do not be alarmed. This is because for the sake of testing and obtaining easy to maneuver results I used a lot of underscores in naming things. If you get to the point where you use the batch processing code for a large sample and do not want to manually create each plot, make sure the naming for the final iteration does not contain any underscores (and previous iterations if you want to see plots for those). It should be a straightforward change in the code. Hence:

MAIN POINTS FOR PLOTTING:

- Choose a RUN_ID that does not include underscores. The plotting routine uses LaTeX, meaning that it will not work if you have any LaTeX commands.
- If you're plotting independently see documentation for the command. Do so through your terminal and give the initial guesses that they recommend to see the full range of values.

This brings me to the effective radii analysis.

3. Bin Matching, Centering and Effective Radii Plotting

This section focuses on the last bit of my thesis. It deals with obtaining, from the fully processed spectra, the kinematics associated with each bin.

How GIST works is that it doesn't. Kidding. It does, but in a very weird way. The kinematics are in one file with certain BIN_ID's and the position of these bins in a separate one (_kin.fits, _table.fits respectively). However, in the latter file the bins are not in their final form. Because of this, you have to go through a reduction and matching process which you can find in two places. The original one I found was in their plotting.py routine found within the /GIST directory and the second one is the one I made, inspired by the plotting.py routine (mine can be found in the ExtractingAndMapping_Re_FINAL.py under the /3_Extras/OtherCodes directory. I have no idea why they made it so complicated, but they did.

Once you have both matched, it is possible to extract the kinematics for certain positions. The way I have it is that I create a table with the necessary data from both .fits files. Depending on what you're interested in you can add or remove columns. From here you can choose a certain radius starting at a certain x,y position and obtain the kinematics for those points. This is a good code to look at when you're figuring out how to analyze GIST output data for if you have the bins matched, you can theoretically also use it to extract spectra.

TO DO: A problem I had that I was trying to correct was with the plotting of the kinematics. My plots filled in the blank pixels, meaning I would get the galaxy plus something more. I believe I was able to obtain the correct bins and positions and that this is only a problem when plotting so any help getting this up and running would be great. It would also give us a better view of the kinematics inside the bulge.

I have also added some useful codes that helped me understand and visualize the data. These are under /3_Extras/SanityCodes directory. The first (binspec_bestfit_plotoverlay.py) is to find an overlay of the pPXF best fit with the original spectrum and the second (spectrum_extraction.py) to extract and plot the original spectrum at any position.

4. A little extra advice

I want to give one final piece of advice that helped me maintain mental sanity and relatively good deadlines throughout my thesis. This is a document to *guide* you. It isn't perfect, I'm sure, but it does contain bits and pieces of my notes that I found to be especially useful. However, I'm not an expert in the field and much less have a PhD. So if there's something I recommend it's to ASK FOR HELP. Dr. Batiste is a wonderful mentor who cares about what you're doing and how you're doing too. She will give you a lot of liberty to go about your way and let you blunder for your own good but ultimately she's there to help. Take advantage of that! You're here to learn, not prove that you're the shit. Off you go!

Oh and also you can always reach out to me and I'd be happy to help.