Hackman Summer Research Summry in AGN with Professor Ryan Trainor

Youssef El Gharably

July 11, 2024

Contents

Foreword

I have had the honor and privellege of working alongside Professor Ryan Trainor, and wonderful colleagues: Omar Khattab, Delaney Adair, and Hongbo Chen. I have always wanted to work in Professor Trainor's research since my freshmen year in college, so having the opportunity to do so in my last summer as an undergraduate student was beyond any expectation I had. In this document, I will be describing the work we've done in the project over the course of the 5-week period of work we had, a description of the code I wrote, and at the end of the document you'll find my personal reflection and a bunch of fun pictures. I would also like to point out that non of this could have been possible without Professor Trainor's endless support and encouragement throughout the 5-week period, as well as his brilliance and passion he has for his work.
This document will also serve as documentation for this work so that future people who work on this project could know their way around it.

1 Project Description

In this summer, the four of us (Me, Omar, Delaney, and Hongbo) worked on different but similar projects. The big picture was understanding how galaxies form, and how they evolve throughout their lives. My project was based off of *Charles Reisner 24'* undergraduate thesis on *Modeling High-Redshift Nebular Emission Lines to Characterize Observed Parameters.* (Reisner, 2024) Omar, Deleany, and Hongbo were attempting to understand more details about the galaxy emissions obtained from the Keck Observatory and MOSFIRE. I, on the other hand, was attempting to build a statistical model using data obtained from Cloudy (a spectral synthesis designed to model a wide range of interstellar clouds) and compare it to flux data points obtained by Delaney from the CECILIA survey.

1.1 Project Online Directory

To access the files in this project, you can click on this link that takes you to a google drive containing all the project's files. THE DIRECTORY

1.2 Cloudy Software

I didn't get a chance to operate the cloudy software myself, but I have worked with data obtained using cloudy. From my current understanding of how cloudy works, the user varies certain parameters in which cloudy synthesizes model spectral data based on these variations. These variations include but not limited to: Hydrogen Density, Spectral Energy Diagram, Nebular Metalicity Percentage, Stellar Metallicity Percentage, and Ionization Parameters. The data structure of the cloudy outputs were an array of data points with `.in`, `.grd` and `.linelistoutput` file extensions, which can be found in the directories called *sm_ variations*, *HDEN_ variations*, and *SED_ variations* in the project directory. If provided enough time, I would have liked to learn more about how cloudy operates and works.

1.3 Reisner's Thesis

Charles Reisner was a student at Franklin and Marshall College who graduated in 2024. He worked with Professor Trainor on building a statistical model using cloudy data and comparing it to CECILIA data by using Monte Carlo - Markov Chain (MCMC) algorithim, which is a very lengthy and takes between 30-40 minutes to work. I will write more about MCMC in detail in other sections. If the reader is not familiar with how statistical models work, they serve as tools that help us predict and estimate points outside the data scope depending on how accurate the model is. So, by creating a statistical model using Cloudy data, we should be able to predict any emission lines based on our variations and they would

be closely related to the actual data points if CECILIA had them. For the sake of writing his thesis, he also wrote a number of other scripts visualizing the data points produced by the MCMC algorithim compared to CECILIA data points, creating histograms, scatter plots, line flux grids, and more.
Additionally, Charles worked on how variations of stellar metalicities (`Zst`) varied the line fluxes measured. So, his code mainly worked on varying `Zst` only.

1.4 Monte Carlo Markov Chain Algorithim

Assuming the reader is already familiar with Monte Carlo, and Markov Chain individually.
Think of the MCMC as a maze runner, if a maze runner goes through one path and runs into a dead end, it just turns around and tries another path. It keeps doing that until the it reaches the exit of the maze. However, unlike a maze runner, the MCMC doesn't run into dead ends, and it doesn't have an exit. It uses *walkers*, which are essentially our maze runners that walk through the maze. These walkers don't really know what path they're going into, which is where Monte Carlo comes in place. Monte Carlo generates random pseudo-points for the walkers to go into. Then by employing the usage of statistical parameters like *Likelihood*, the walker creates a markov chain and runs it aganist the actual data points we provide it with (in this case CECILIA points) which then puts it as a threshold. Then it runs into its second point, by using markov chain calculations, the walker is able to determine which one of the two points has the best likelihood. It keeps doing this for a finite element of iterations that the user defines in the script. So, instead of running into dead ends and going through the exit, it runs into probabilities. Its exit becomes the best probability, which isn't really an exit because statistically speaking none of these probabilities could be fully perfect.
More on MCMC here.

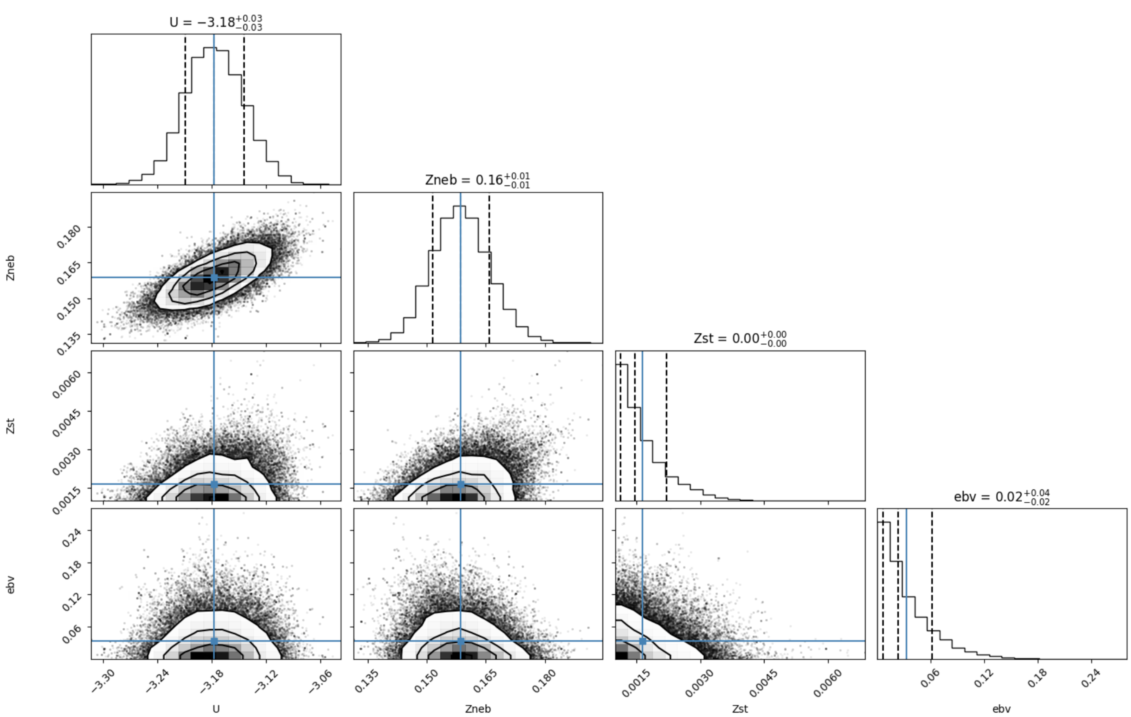

Figure 1: Corner plot created by MCMC using `Zst` variations. (Reisner, 2024)

2 My Efforts in Project

Throughout my 5-week period working in the project, I have worked on a couple of things simultaneously. I will talk about them here in chronological order.

2.1 Computational Optimization of Reisner's Scripts

Charles' code took about 35 minutes to run and show a corner plot output. I took this as a learning opportunity to learn how to optimize the code for efficiency without jeoprdizing the data analysis process. By pre-defining multiple functions and going through the code line by line, I was able to get it down to about 10 minutes. Later in the project, I also employ the usage of using `multiprocessing` which is a python library that allows multiple computations to run simulteneously.

2.2 Varying SED values in the MCMC algorithim

Ben Tubiello 25' ran cloudy variations on spectral energy distributions (SED's) and Hydrogen Density percentages. (HDEN) Spectral energy distribution are given as a function of the Eddington ratio, allowing emission correlations to be investigated on a fundamental basis. (Ferland, Done, Jin, Landt, & Ward, 2020)
By varying between low, intermediate, high, and highest spectral distributions, I was able to obtain a bunch of results based on CECILIA data obtained from Delaney Adair with her work on different MOSFIRE apretures. Part of varying these distributions was reading through the paper to acquire quantitative values for different SEDs. These values are shown in Table 1. These files were `levelSED_script`, where level was either low, inter, high, or highest. **Note:** the `levelSED_script` files only have a small variation interval of metalicity percentages. The files starting with `biggrid` have larger intervals of metalicity variations. The `biggrid` files will be talked about more in 2.3. A work in progress modification was extracting SED variations alone from the `biggrid` files, which will be part of our future work.

SED	$\log(L/L_{Edd})$
Low	-1.15
Intermediate/Mid	-0.55
High	-0.03
Highest	0.65

Table 1: Table showing values of SED Eddington ratios on the logarithmic scale.

2.3 Variations of HDEN alone, or combined with SED. (W.I.P.)

Ben also generated cloudy outputs which varied hydrogen density percentages WITH variations in spectral distributions. The files containing these data generations start with `biggrid`, they should contain an extra column of data called `HDEN`. Note that this file also varies metalicity percentages, so for whoever is working on this in the future, DO NOT omit the metalicity column. This is still a work in progress part of the project.

2.4 Automation

I was able to modify some aspects of the code that were mostly hard codded. The code can now extract file paths automatically based on a directory that you specify. I ran into some issues where python didn't like how windows machines write their file paths, but the code also correct for that. So, the files extraction works for both Mac and Windows machines. Additionally, the code is now able to extract the `Zst_values` from files as well as the `SED_values` from the file name. These automations assume a number of things:

1. The cloudy files with `Zst` variations are names `sm_cc` where the two `c`'s are numbers. I.e. `sm_01`, `sm_02`, etc. Charles named these cloudy output files such that the last two digits represent the `Zst` values of this file, so for a file called `sm_14` that would mean it has a `Zst_value = 0.014`.
2. The cloudy files state the value of `log(Zst)` at their first line, so the code actually extracts that value and assigns each file the value of $10^{\log(\mathtt{Zst})}$, which is the same as `Zst,` so I made sure that the code checks that as well.
3. The script assumes that files' names containing `HDEN` variations start with the term `biggrid`. This is easily modifiable, but it was worth noting that this is what the code assumes.

These automations open up oppurtunities for code accesibility and flexibility to be used by other astronomers who are working in the same field of research. However, there is a fine line between efficiency and effictiveness. It's important to make sure that the program's efficiency modifications don't affect its effictivness in achieving the goal we aim it to achieve.

2.5 Data Structure in the script (W.I.P.)

I attempted to rewrite the scripts to use pandas data frames rather than astropy tables for accessibility purposes, but I wasn't able to finish working on it in time before the end of the research period. However, I have made substantial progress in making this better data structure method, so with more work put into it in the future, I'm sure it will work!
I've only had time to work on the `Zst` version of the MCMC algorithim for dataframes, so you'll find them in a directory called `df_mcmc` or something along those lines.

3 Code descriptions and functions

In this section, I will be discussing how the code functions work, and how it eventually gets you to the corner plot we're looking for. Charles also wrote multiple other scripts for the purpose of displaying the data, I think it'll also be helpful to talk about these files here.

`mcmc_code_modified.py`

This is the MCMC program that was initially written by Charles then modified by me. The code contains all the functions you need, except a couple which will be pulled from modules `k_lambda` and `makeCloudyTable` There should be sufficient comments in the script to help you navigate your way through it. The script runs a burn in with a small number of iterations that you define then runs the production itself to ultimately end up with a simillar plot to the one in figure 1. This file also exports a document called `mcmc_output` which is used later by other scripts.

`mcmc_code_modified_multi.py`

Basically the same as the other MCMC code, only this one utilizes multiprocessing and has a seperate module called `mcmc_functions.py` that it pulls functions from.

`make_hist_modified.py`

This creates a histogram for each line individually based on the `mcmc_output` document, as well as some error lines and such.

`line_flux_grid.py`

This creates a scatter plot showing the errors of the CECILIA flux points compared to the MCMC points.

`makeCloudyTable/NEWmakeCloudyTable`

These are the scripts that create the table used for `LLT` in the `mcmc_code_modified.py` (This was written by Professor Trainor)

4 Results

Based on the fluxes provided by Delaney using different MOSFIRE apretures, I was able to get the following SED variations:

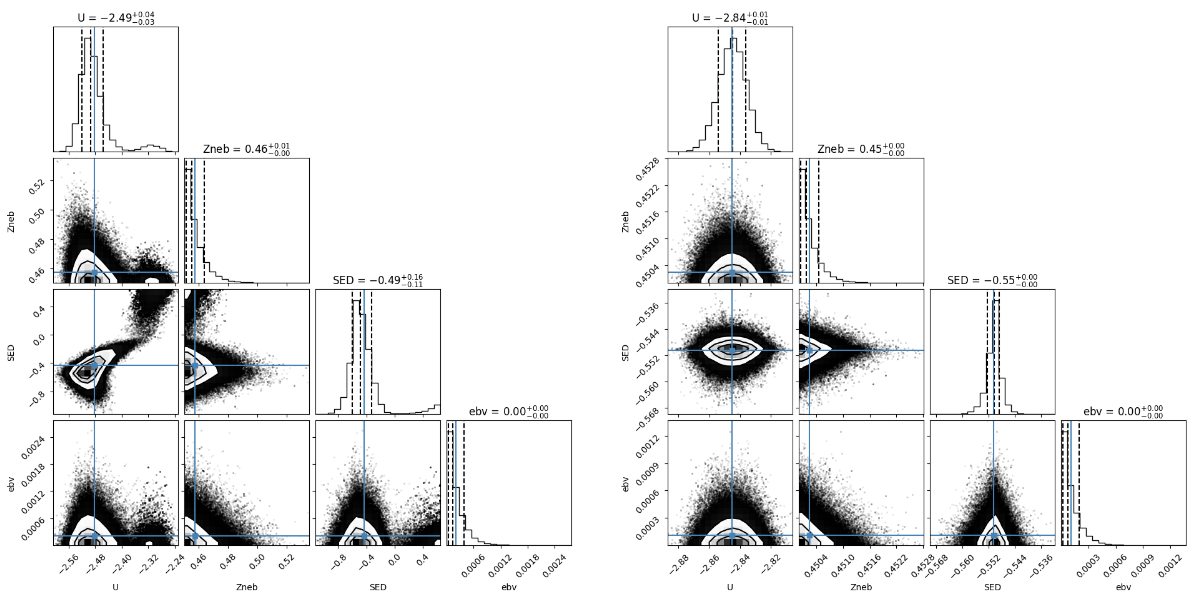

Figure 2: Left: Apreture A with only Apreture A lines. Right: Apreture A lines with K and H slit line.

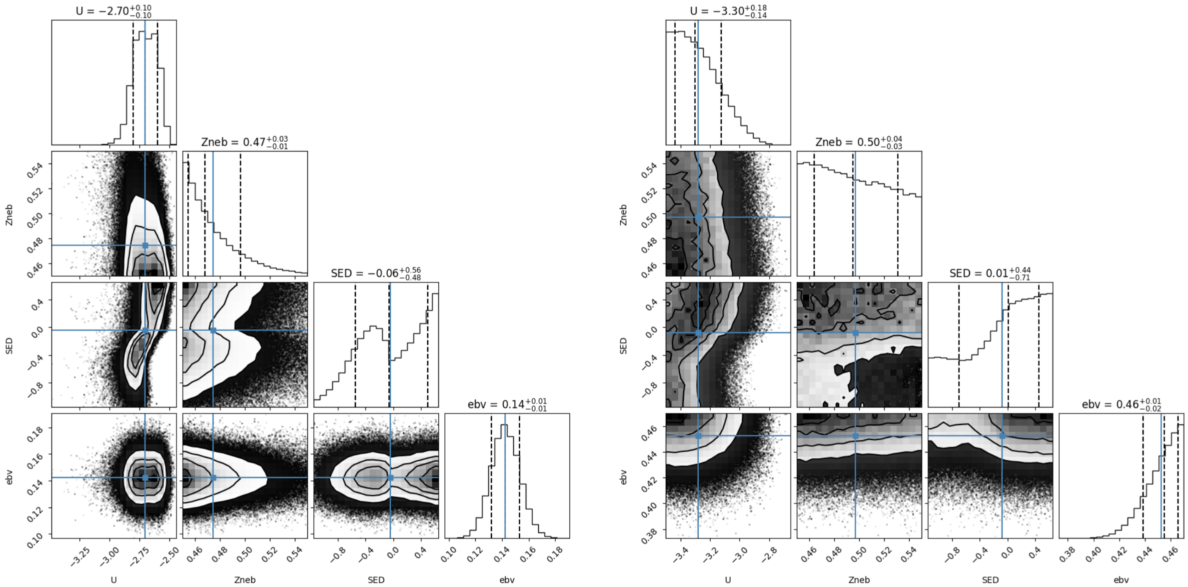

Figure 3: Left: Apreture B lines. Right: Apreture C lines.

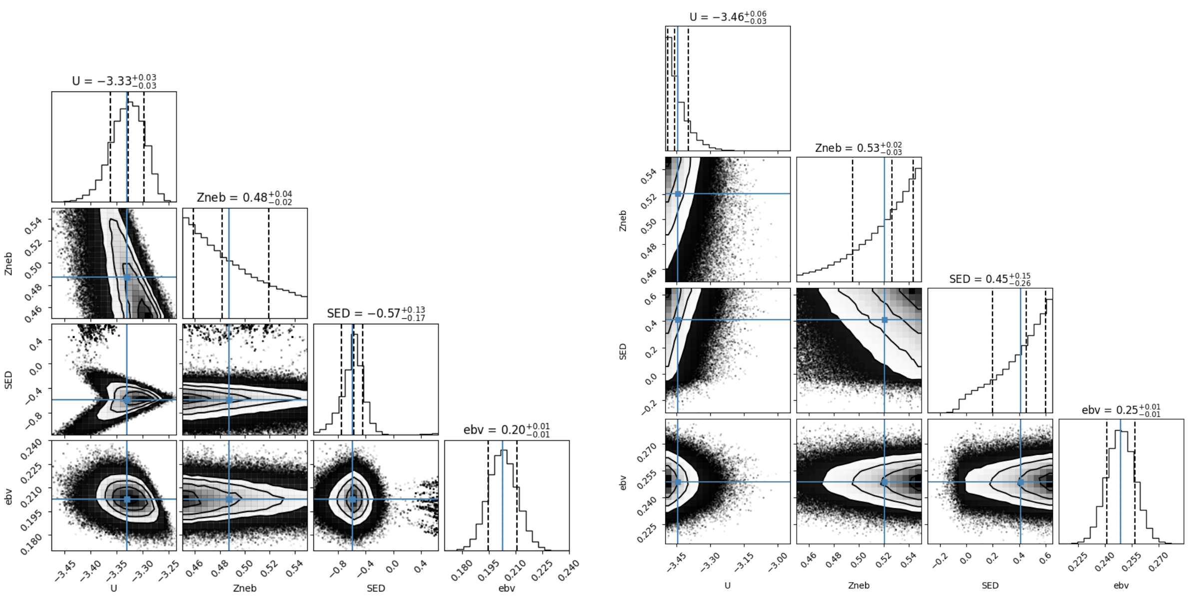

Figure 4: Left: H slit lines. Right: K slit lines.

We didn't really have many lines to go with, so the results are kind of weird. For reference, Charles' corner plot in figure 1 had 22 lines. These ones have between 4 and 9 lines.

5 Personal Experience and Reflection

Throughout these 5 weeks, I have learned endless amounts of information. Not only did I learn a lot about galaxies and AGN's from Professor Trainor's lectures, but I also learned so much about computations and models, and just generally knowing my way around python even more than I do. I have always loved working with data analysis in astronomy, but working on this was even more enjoyable because I got to learn more about a complex mathematical method to get best fits of complex data with. (MCMC)
Even though I'm planning on working in astronomical instrumentation in the future, I believe learning more about how the data looks like and what we could learn from it is very useful in my career as a physicist. I'm very excited to see where life takes me after I graduate with these skills!

6 Fun picture!

Figure 5: Group picture of the team after presenting our research in July to a bunch of summer students and faculty. Left to right: Professor Ryan Trainor, Delaney Adair, Me! (Youssef El Gharably), Omar Khattab, Hongbo Chen.

References

[Ferland et al.(2020)] Ferland, G. J., Done, C., Jin, C., Landt, H., & Ward, M. J. 2020, *Monthly Notices of the Royal Astronomical Society*, 000, 1–6, arXiv:2004.11873v1