**Automating NEMESIS**

**Brief layman-ish description of what I do**: I take spectral data, i.e. how many photons hit a telescope detector for a given wavelength of light, and I attempt to reproduce that spectral data using a model atmosphere. If I “get the atmosphere right”, i.e. choose the ‘correct’ temperature profile, haze profile, and abundance profiles of the various gases, then the resulting model atmosphere produces a spectrum which closely matches the observations. That is what NEMESIS does, it allows me to model an atmosphere, which in turn models the spectral observations.

**The Automation:** I would like to automate the process of generating forward models from previously calculated models, which entails copying of directories, copying/renaming certain files within the copied directories, adjusting values contained in the files of those directories, running a script called Profile in the new models’ directories, and then running NEMESIS calculation on the newly created directories. I need this automated process to repeat many times (with the max running at a given time specified by the user) right after another, each in different terminal window or tabs, so that I can instantaneously get multiple NEMESIS calculations running at once without having to do all the copying, renaming, adjusting of parameters, etc. by hand. Instructions for executing this process will be entirely contained in a .txt file, and driven by a very small number of user inputs to the script. So the automation script just needs to read in that .txt file, and then execute the procedure.

**Relevant Background:** I store my atmospheric models in a directory called Nemesis (see figure below). The models themselves are called “run###’, where ### is a three digit number. For a given set of up to 100 models, Run00, run30, and run60 are special in that they are ‘full models’, where many parameters in the model are permitted to vary, achieving a good fit to the data. Run00 is the ‘full model’ for one data set, run30 the full model for a different data set, and likewise for run60. All the other models for that set of up to 100 are ‘forward’ models, which means that I just add the target gas that I’m looking for, in an abundance of my choosing, and see how that tiny change affects the fit to the data. So, run01 through run29 are the forward models associated with run00, run31 through run59 are the forward models associated with run30, and so on.

A screenshot of a computer

Description automatically generated with medium confidence

The idea is that for some optimal’ abundance of the target gas, we will see an optimal improvement to the fit of the data. Quantifying the statistical significance of that improvement to the fit allows us to derive upper limits on the abundances of the target gases, e.g. a given result might show that at most, there can only be 100 parts per billion of a given target gas, which means on average, 100 out of every 1 billion molecules in the atmosphere are the target gas.

The trend described above (run00, run30, run60 being full models, all others, 1-90, being forward models) continues for each set of 100, which I will refer to as a century. So, run100, run130, run160 are full models for the 100s ‘century’. In each different century, I am testing a different type of atmospheric profile, e.g. is the target gas’s abundance constant through the whole atmosphere? Or does it increase or decrease rapidly in some given region? All other runs in the century are forward models for that particular profile. The same goes for runs 200, 230,260, ……, 900, 930, 960.

All of my ‘full models’ are already generated and ready to use. I just need to copy them to new directories, give those directories a new name, make a few adjustments to the files in those directories, and then start NEMESIS in those new directories.

**Bonus task:** It would also be really nice to automate the process of running multiple calculations, and then having the *next* set of calculations commence as soon as the first set is done, and then have the next set begin, and so on. I have no idea of how to do this, or even if this is possible, and it’s really secondary to the main task; it would just be an additional luxury.

Text

Description automatically generated**The .txt file:** This should be the main input to the automation script. I call it a ‘run\_dictionary’, because my plotting scripts read this information in as a dictionary/map and use it for labeling the models in my plots so I can distinguish between different results. The run\_dictionary looks like this:

Each line is one of only two things. It’s either a comment, or a model. If it begins with a 2 or 3 digit number, or four digits in the case of something like 300.0 (don’t worry about the .0 runs, it’s not important), then this is a model in my NEMESIS directory. For example, if that 2 or 3 digit number is ‘###’, then this corresponds to ‘run###’ in the NEMESIS directory. What follows the model number is the description that I have stored to identify the model. In the case of full models, i.e. run00, run30, run60, …… , run930, run960, etc., the description is usually the name of the data set, which for me is either ‘EQ’, ‘T3’, or ‘T35’. In the case of the forward models, the description will always be a single float ‘scalar’. It means that for that forward model, I’m going to take my target gas’s abundance profile, and scale it by that number, effectively adding or removing more of the target gas into that model. By repeating this process for a set of differing scalars, I am able to pin down that optimal target gas abundance.

**The Steps:** Alright, so at this point, I think the process is clear, and you should understand why I’m doing what I’m doing. I should be able to just list the steps that need to be done for each forward model, and I think you’ll understand what to do. Let’s consider one data set’s models (run00 through run29) for one particular experiment (i.e. run00 through run29 as opposed to run400 through run 429). Here’s what the automation script needs to do:

1. Allow the user to store a couple of different floats / strings / containers in the script itself, easily identifiable, so that the user can change them as necessary, before running the script, based on their particular project. The variables (names in bold) will be as follows:
   1. **molecule** (this will be a two digit integer, and will always be either 50 or 78 for my own work at least, referring to methyl cyanide and n-butane, respectively, my two target gases)
   2. **models\_at\_a\_time** (number that will limit the script to running this many NEMESIS calculations at one time)
   3. **run\_name** (this will be a string and will let the user define the main name used for most of the files for a single project)
   4. **profile\_names (**this is a container which needs to contain about 6 separate strings, which will look like “C4H10\_krasnopolsky”, i.e. a target gas’s molecular formula, followed by a last name of the author that developed this particular abundance profile. Associated with each string will be a three digit number, 100, 200,…..900, to tell the script which profile needs to be added in step. You could accomplish this one easily with a dictionary, or an array, whatever you like).
2. Read in the run\_dictionary file, which I will prepare myself, and in this case would look like the first 30 or so lines of the screenshot I showed you on the previous page.
3. Store each model number and its associated scalar.
4. For each of the model numbers contained in the .txt file (up to a maximum of ‘**models\_at\_a\_time’**), copy the associated full model (run00 in this case) in my NEMESIS directory to a new directory there, and name it appropriately, i.e. for model number 24, name the copied directory run24.
5. In each of the new forward models’ directories, execute the following steps:
   1. Open up the file called run\_name.apr. Change the first value in the fourth row to the appropriate scalar for the model.
   2. In the new model’s directory, start NEMESIS for that forward model with the command: **Nemesis<run\_name.nam>run\_log.log**
   3. Once this model calculation finishes, give me a print statement saying that “run### has completed” or something like that.

And that’s it! So again, the above needs to happen for each of the forward models in the run\_dictionary, ten at a time, and if possible, start the next ten when the previous ten finish. If the latter part proves possible, the script will need some way to execute Step 4 above understanding that forward models 1-29 are associated with full model run00, and 31-59 are associated with full model run30, etc. all the way on up to runs961-989 being associated with full model run960. In this way, I could set up one single long run\_dictionary file, for all of the calculations I want to do for this upcoming publication, or for future work, and just hit run and let it go for a couple of days (each set of 10 forward models takes 1-2 hours, and I need to do about 450 of them). Also, if we can go this route, we would modify Step 1A and just let model numbers <500 be associated with methyl cyanide (identified by the number 50), and model numbers >=500 are associated with n-butane (identified by the number 78).