**Setting up the models on NYU's High Performance Computing System**

I recommend that you start with the exact same structure for running the models that I use. You can customize things as you begin to understand how it all works. This is set up for work on prince, as this is the main machine for parallel computing.

1) An overview of the file system: where to put code, run the model, and store the output.

There are three main file systems on the HPC. (Wherever you see "nyuID" that means your nyuID, i.e. epg2 for me.)

`/home/nyuID`
This is a place for codes, scripts, etc. -- anything that you write -- but not for model output or data. This file system is backed up, but it is small -- your quota is on the order of a gigabyte or so. When you log in to the HPC, you begin here. (Note that you have a separate home system on bowery and union square.)

`/scratch/nyuID`
This is a place to run the models and analyze output, but is not backed up. Your quota here is a few terabytes. Data that is left here, un-accessed for more than 60 days, is subject to be deleted without notice; this is to make sure that the system doesn't fill up with left over data. This won't be a problem for our class, but could pose a problem in your research. In addition to the archive discussed below, you can also use your NYU google drive to save data indefinitely.

`/archive/[first initial]/nyuID`
(where [first initial] is the first letter of your login, e for me). This is a place to store important data -- it's backed up -- but it cannot be accessed from the "compute nodes," the part of the system that crunches the numbers. You must deliberately move any data that you need backed up here while

logged into a login node. What is important data? Anything that would be terrible to lose (i.e. hard to re-compute, or be hard to replace — i.e. observation data that you have collected!) Note that the model output that you produce for this class is not important — and could be easily re-computed if needed!

When you first log in, you are on a "login node." From here, you can see all of the file systems (/home, /scratch, and /archive), but you don't have access to parallel computing. For running a parallel job, you will have to use a compute node. These nodes can see /home and /scratch, but not /archive. More on this later...

2) Get the codes, scripts, etc. Here we copy the key files from directory.

There are four sets of files that you will need for your model. We will put all of them in a directory called "student_models." i) you need the codes and executables, which we place in: student_models/codes/. ii) you will need the input parameters, which inform the model what to do: we put these in student_models/run_parameters/ iii) you'll need to get the scripts that run the model, which we put in models/jobs. Later on, you might want some postprocessing scripts which will help you organize and analyze the model output: I would put these in student_models/scripts.

Start in your home directory. You can be sure to be here by running this command:

```
> cd
```

Note that all commands will denoted by the > symbol. To make sure you are in the right place, you can type:

```
> pwd
```

(Path to Working Directory) It should now say:

```
/home/nyuID
```

where nyuID is your ID!

I've set it up so that you can get everything with one command. (If you'd like to do a "manual install," you can copy over everything individually, as detailed at the end of this document.)

```
> cp -r /home/epg2/student_models .
```

What do you have now? Let's take a look:

```
> ls
student_models
```

You now have a directory called models. Let's take a look inside it. You can move into the directory with the command cd, for change directory. To see where you are, you can always type pwd.

```
> cd student_models
> pwd
/home/nyuID/student_models
```

Now take a look at what you have in this directory, using the command ls.

```
> ls
code  jobs  run_parameters  scripts
```

These are the four directories we discussed at the top of this section. Take a look inside each of them to see what files you have now. For example, what is in jobs?

```
> ls jobs
create_runscript.csh template.iterate.nrelax.nyu
template.iterate.gray.nyu  template.iterate.rrtm.nyu
```

You have two kinds of files. One, create_runscript.csh, will be used to create run scripts for the model. It uses the "template" files to run three different models. Note that you can look in directories from higher up in

the file system. If you were in your home directory (get there by typing cd), you could look in the same place like this.

```
> cd
> pwd
/home/nyuId
> ls student_models/jobs
create_runscript.csh template.iterate.nrelax.nyu
template.iterate.gray.nyu  template.iterate.rrtm.nyu
```

You will learn more about all the files you just obtained as we move on.

3) Set up your environment on the hpc.

You'll need to set up a few things to compile and run the core on the hpc. I have it load the necessary paths by default by putting them in my shell start up file. I use bash, and think it is the default -- with hope you chose this option when you initially set up your HPC account. If you've already customized your .bashrc file, then you'll just want to copy the module load commands from my sample.bashrc. Otherwise, start with the sample.bashrc file. Remember that the cd command at first ensures that you are in your home directory. You need to be here for the following commands.

```
> cd
> cp /home/epg2/for_students/sample.bashrc .bashrc
> emacs .bashrc
```

This is to edit the file. You will need to change "epg2" to your login to make the short cut qs work. I'll try to explain to you what these commands are in class. Lastly, you need to activate the changes.

```
> source .bashrc
```

These commands set up your paths so that the system knows that you'll be using fortran, parallel processing, matlab, etc.. Note that the last command:

```
umask u=rwx,g=rx,o=rx
```

makes it so that files that you create can be viewed by other people on the hpc. In general you should not consider anything on a public system like this to be private, but this does allow anyone to look at your files. This can be useful when you want to share data or want me to help you debug; without it, I can't read any of your files! That said, if this option makes you feel uncomfortable, please delete this line!!!

I've also included some useful aliases in the sample.bashrc file. I'll refer to them later. (An alias is a short cut for typing long commands.)

Next, create space to run your model, remembering to use your nyuID. All the output will be put here when you run the model. Also create a place to store the output text files when the model runs. Normally you don't need to look at these output files, but they can be useful when the model crashes.

```
> mkdir /scratch/nyuID/model_output
> mkdir /scratch/nyuID/job_output
```

It wouldn't hurt to create a place to put the model output on the archive, too. (That is, if you have archive space. It's not necessary for our class, but you will need it if you want to complete research on the hpc.) You can move data here after running the model.

I've also written scripts to make it easier to analyze model output in matlab, and will give you this, too. Set up a directory to put these scripts, too.

```
> mkdir matlab
```

```
> cp /home/student_models/scripts/ncget.m matlab
```

Okay, now you should be ready to start running the code.  Go to the next file, “running_the_model_interactively.pdf”