**High Performance Computing**

Author: Noelle Kosarek

**What is a High-Performance Computing System and Why Should We Use Them?**

A High-Performance Computing System (HPC) is a powerful computing infrastructure designed to process and analyze large amounts of data at high speeds. HPC systems typically consist of multiple interconnected processors (CPUs and GPUs), large memory capacity, high-speed networking, and specialized storage systems, enabling parallel computing.

You will want to use an HPC system in you work for the following reasons:

1. Faster Processing: HPC systems can perform complex calculations in parallel, reducing the time required for tasks.
2. Handling Large Datasets: Ideal for big data applications in genomics, climate modeling, machine learning, and scientific simulations.
3. Parallel Computing: Many research and engineering applications benefit from running multiple computations simultaneously across different nodes.

**What Will You Need to Log On To The HPC System?**

Most of the folks in this class are not on Dartmouth’s campus. As such, you will need to download Dartmouth VPN before being able to log on to the Discovery Cluster. To download Dartmouth’s VPN please follow [these instructions](https://services.dartmouth.edu/TDClient/1806/Portal/KB/ArticleDet?ID=72204) carefully.

Once you have successfully installed Dartmouth’s VPN, you will need a terminal system. For my Mac people, your terminal is built in to your system and there is nothing you need to do. I’ll show you how to navigate to your terminal in the HPC video for this class. Windows people, you will need to download either one of these terminal systems: [MobaXterm](https://mobaxterm.mobatek.net/) or [Putty](https://www.putty.org/)

Both the Mac and Windows people should also download either [Cyberduck](https://cyberduck.io/) or [FileZilla](https://filezilla-project.org/). Cyberduck and FileZilla are open source SFTP and FTP clients. I like to use Cyberduck. Think of these SFTP and FTP clients as the “Finder”, just for your HPC system. I find it makes the terminal environment a bit easier to navigate.

**Signing In to Dartmouth’s Discovery Cluster**

Without further ado, let’s log on to our HPC system. Navigate to your terminal and enter this line of code (make sure to enter your NetID!):

ssh [your\_netid@discovery8.dartmouth.edu](mailto:your_netid@discovery8.dartmouth.edu)

Enter your password – The password should be the password for all of your Dartmouth related accounts, like your email. NOTE: The password will not appear when you type it! This is simply frustrating in the event that you mistype.

After entering your password, hit Enter. You should now be on Dartmouth’s Discovery cluster!

**Generate a Script You Would Like To Run**

There is a preponderance of tasks we can execute in the HPC system. We can run bash, R, and Python code interactively. We can make directories and files and share data with our friends on the HPC system. Today, we will be learning how to run R code interactively on the HPC system.

Where you are on the HPC system, simply type in “R” and hit enter. R should appear in your terminal with a “>” prompt. Let’s execute some of the r code below:

print(“Hello World”)

Sys.sleep(10)

print(“My Name is Noelle”)

Sys.sleep(10)

print(“Welcome to FoDS”)

Sys.sleep(10)

print(“It is nice to meet you.”)

A screenshot of a computer

AI-generated content may be incorrect.But what if we wish to run all of our R code in a file at one time on our HPC system? We can do this by first creating a new file and putting this code into that file. Open Cyberduck and log in as shown below:

**A screenshot of a computer

AI-generated content may be incorrect.**You should end up in your home directory in Discovery. It should look a little something like this:

Place your cursor in the window, right click, and select “New File…”. Name this new file “R\_Code\_Test.R” and click “Create”. The new empty file should show up in your window. Right click that file and open it with any text editor you like. I like to use Sublime. Copy and paste the R code we worked with interactively above and save the file.

Now go back over to your terminal. You should be in the path for your home directory. Mine looks like this : /dartfs-hpc/rc/home/8/f002yt8. To check if you are in the correct directory, type in “pwd”. This command will return where you are at in the Discovery. If you are in the correct directory, execute the command below:

**chmod +x R\_Code\_Test.R** # Makes your code executable

**srun Rscript R\_Code\_Test.R** # Runs your code

And your code should run!

**The Anatomy of a Job Submission**

One very useful skill is learning how to do on an HPC system is submit what is called a job. A job is simply instructions sent to the HPC system that tell it to execute certain commands or run a certain script or scripts.

A job submission script has a particular anatomy to it. Let’s look at the example below and write a job submission script to run the R code we generated above.

#This line is used to instruct the operating system to use the bash as a command interpreter

**#!/bin/bash**

#This line is the name of the job we will be running

**#SBATCH --job-name=My\_Test\_Run**

#This file will automatically generate and store any error messages we get when we run our script

**#SBATCH --out=My\_Test\_Run.out**

#This file will automatically generate and store any results from our job run

**#SBATCH --error=My\_Test\_Run.err**

#This line indicates the number of nodes we are requesting. You can increase the number of nodes if you wish to run multiple scripts

**#SBATCH --nodes=1**

#This line tells us the amount of memory we will need

**#SBATCH --mem=200G**

#This line tells us the wall time

**#SBATCH --time=36:00:00**

#The partition refers to a particular set of nodes to which we have access

**#SBATCH --partition=standard**

#This is the number of CPUs requested for our task

**#SBATCH --cpus-per-task=16**

#This is the name of the R script we generated above that we would like to run in this job

**Rscript R\_Code\_Test.R**

Like we did with the R script, right click in Cyberduck in your home directory and generate a new file called “job\_submission.bash”. Copy and paste the above code into your new file and save. Go back to the terminal and run the following command:

**sbatch job\_submission.bash**

**Important Commands to Run Your Job and To Check If Your Job Is Running**

Here is a summary of all the commands we have used today:

Runs code, but not as a job:

**srun Rscript R\_Code\_Test.R**

Makes your code executable:

**chmod +x R\_Code\_Test.R**

We can submit jobs with this command

**sbatch your\_bash\_script\_name.bash**

We can check on the progress of just our job with this command:

**squeue -u your\_net\_id**

We can check on the progress of all of the jobs running on Discovery with this command:

**squeue**