Fred Hutch Cluster 101

November 18, 2024

Contents

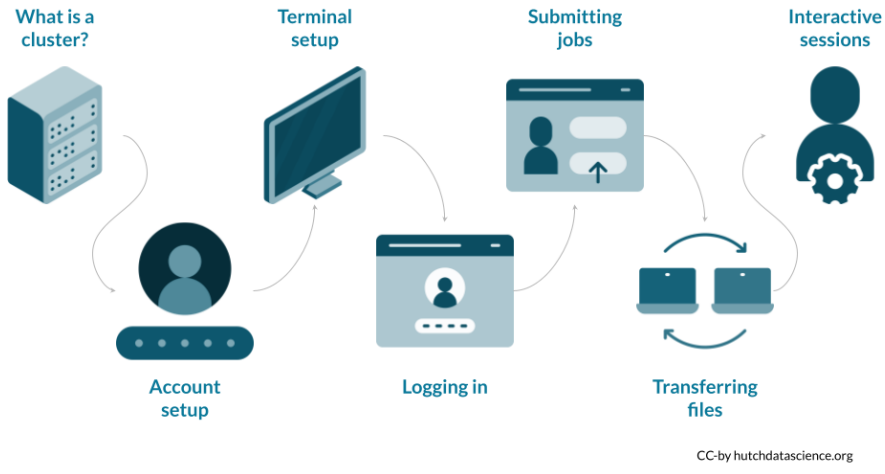
About this Course

Fred Hutch maintains a high performance computing cluster specifically to support work that requires intensive computing. The Fred Hutch cluster (sometimes called "Rhino" and/or "Gizmo") allow you to do more than your average desktop computer can handle.

Our goal for this course is to get you running on the Fred Hutch cluster **quickly and efficiently**. It is intended for everyone from brand new cluster users to researchers who have used a cluster at another institution but are new to Fred Hutch. We hope that the following modules will help you take advantage of the powerful resources the Fred Hutch has to offer!

The Fred Hutch cluster is supported by a group in IT called Scientific Computing, and this course was developed by the Fred Hutch Data Science Lab in collaboration with them. Please see the author credits for more information.

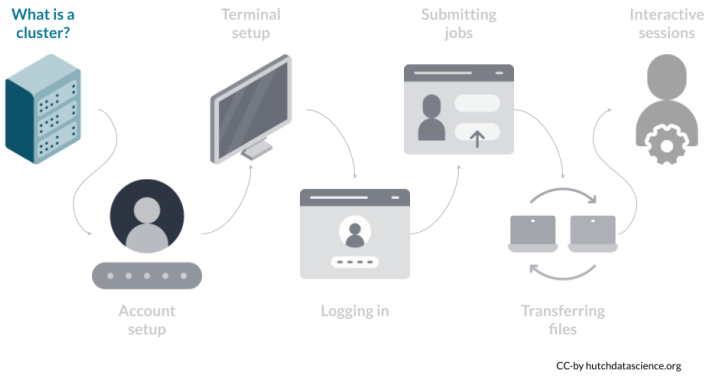
This course is available in Bookdown and Leanpub formats. If you want a certificate, you need to take the Leanpub version of the course. **The Leanpub course can be taken for free, but you still have to put the course in your cart and check out**.



Introduction

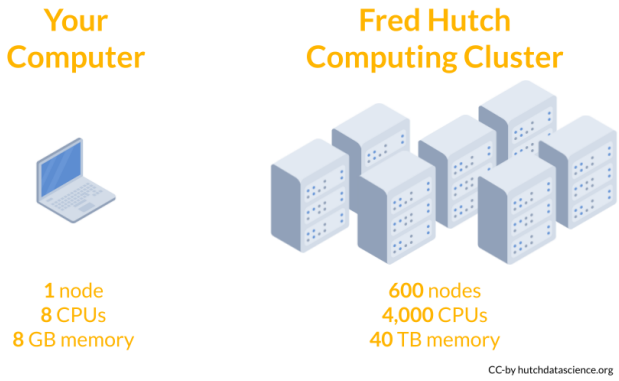
Chapter 1

What is a Cluster



A computing cluster is a set of many computers networked together. Because there are many computers working together, the network is able to handle computationally expensive tasks, like genome assemblies or advanced algorithms. Imagine you're building a house. It would take a long time by yourself! It's much better to have many builders working together.

Now that we have a team of workers, the next challenge is task management. A home construction team will need a manager to help delegate tasks. Similarly, the computing cluster uses management software to prioritize tasks, delegate workers (resources), and check on progress. The Fred Hutch cluster uses a common management and scheduling tool called Slurm.



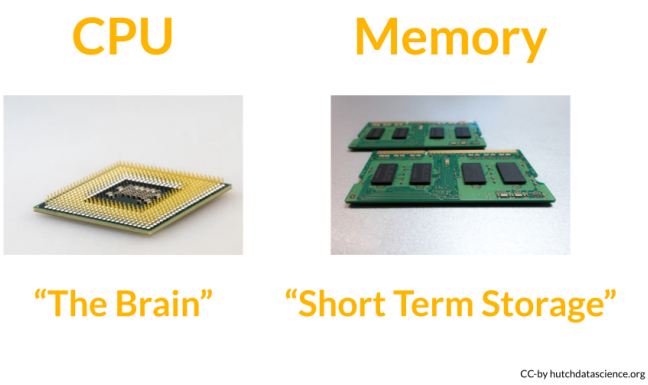
How is the cluster different from a laptop or desktop? First, on your laptop you most likely interact with it using an operating system like Windows or MacOS. The Fred Hutch cluster uses a Linux operating system. Second, because many people use the cluster for many tasks, there isn't a central screen and keyboard. You access the cluster remotely from your computer! We will talk more about how to connect to the cluster in a following chapter. Third, because the cluster is a set of many computers networked together, you will usually access only a few of the many computers for your work.

1.1 Parts of the Cluster

Each of the networked computers within a cluster is called a **Node**. Each node has two main components:

The **CPU**, or Central Processing Unit, is the brain of the computer that performs and orchestrates computational tasks. Modern computers often have multiple CPUs (or also known as "cores") to perform multiple tasks at once, ranging from 4 tasks on a typical laptop to 48 tasks or more on higher end servers. Examples of tasks can be running a simulation many times, in which each CPU performs a simulation in parallel.

The **RAM**, or Random Access Memory, is often simply referred to as memory. This short term memory holds the information that the CPU needs to perform calculations. One distinctive feature of memory is that it is short term. In other words, when the electricity is shut off, the data stored in memory disappears. To save the CPU's work, you usually save files to your computer. Running highly complicated analyses or algorithms can often require additional memory resources.



On your own personal computer, you may have 2-4 CPUs, and 8-16 Gigabytes of working memory. On the Fred Hutch cluster, there are around 600 nodes, and each node has a huge amount of CPUs and RAM compared to your personal computer: for instance, each "K Node" have 36 CPUs and 700GB of RAM, and each "J Node" have 24 CPUs and 350 of RAM! Therefore, when you access any of these K or J nodes, you will be sharing its computing power with other users also.

Computing cluster

A set of computers networked together to perform large tasks.

Node

One of the networked computers in the cluster.

CPU

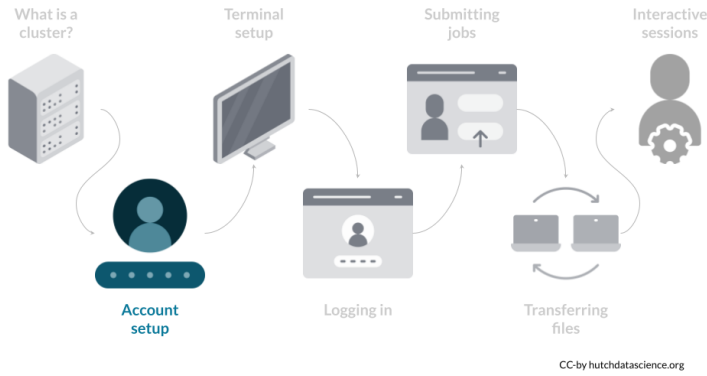
A computer component that performs and orchestrates computational tasks.

Memory

A computer component that stores calculations and information in the short term.

Chapter 2

Account Setup



You will need an account to log in to the cluster. This ensures that data stays protected and that computing resources are shared fairly.

2.1 Check your HutchNet ID

Your HutchNet ID is the standard login you receive when you start working at the Hutch or are an official affiliate. You can use it to login to most resources at Fred Hutch (Desktop Computer, Employee Self Service, VPN, Webmail) and our Scientific Computing systems.

For example:

- my email is `jsmith3@fredhutch.org`.

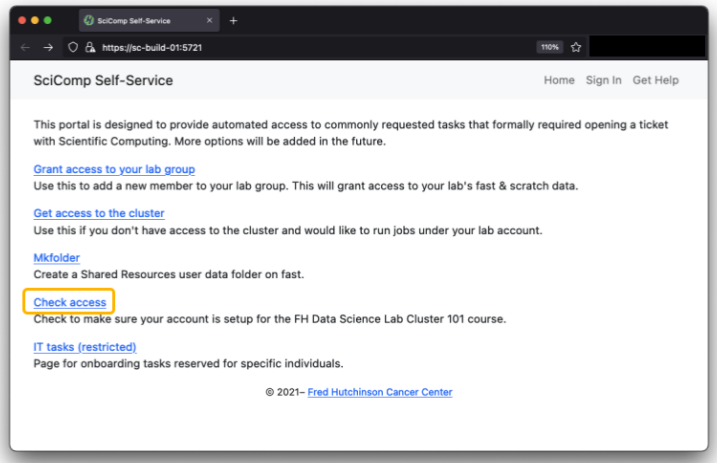
- my HutchNet ID is `jsmith3`.

If one of your collaborators requires access to the Fred Hutch network you can submit a non-employee action form. "Non-employee" is a generic administrative term for affiliates, students, contractors, etc.

2.2 Connect to a PI Account

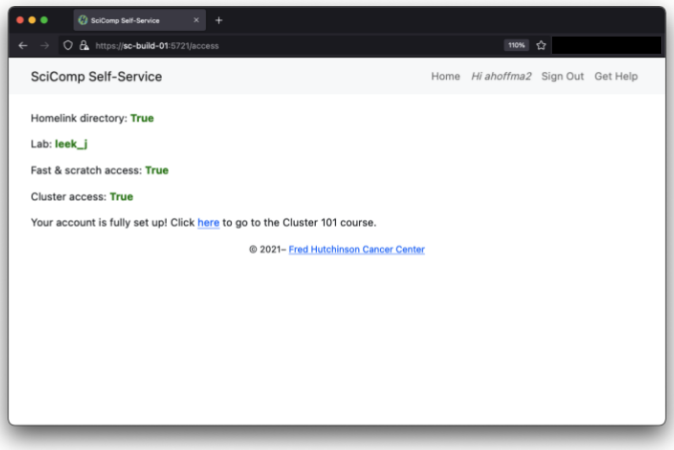
Your HutchNet ID must be associated with a PI cluster "account". The Scientific Computing Team (SciComp) tries to set users up with a connection to a PI account before they need it, but this is not automatic! To ensure that you have been set up to use the cluster, please follow the following steps. You must be connected to the campus wifi network, plugged into a networked ethernet jack, or connected to the Fred Hutch VPN.

1. Go to the SciComp Self-Service Portal
2. Click on "Check Access"
3. Log in by entering your HutchNet ID (don't use `@fredhutch.org`, just the ID) and password



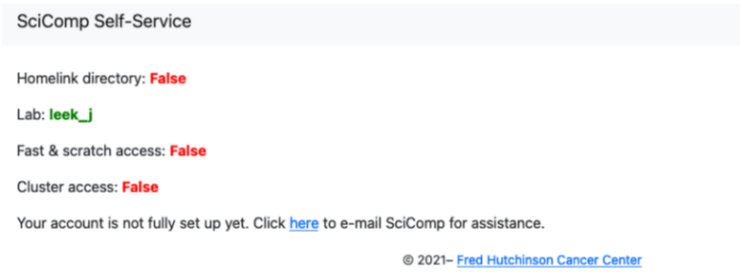
Screenshot of the SciComp Self-Service Portal homepage.

If everything is green as shown below, you are ready to proceed. You can proceed with the course or skip to certification.



Screenshot of the SciComp Self-Service Portal verified access check. CC-by hutchdatascience.org

If you see anything in red as shown below, click the link to e-mail SciComp to finish setting up your account. Be sure to include your HutchNetID and PI name in the email.



Screenshot of the SciComp Self-Service Portal access check. Account is not yet set up. CC-by hutchdatascience.org

Note that it may take just a bit of time for SciComp to see your email request, but it is usually fairly quick! Once approved, you will receive an email back from the SciComp team that you now have cluster access. The Self-Service Portal will also show that you have cluster access if you refresh the page.

Now, let's connect to the cluster!

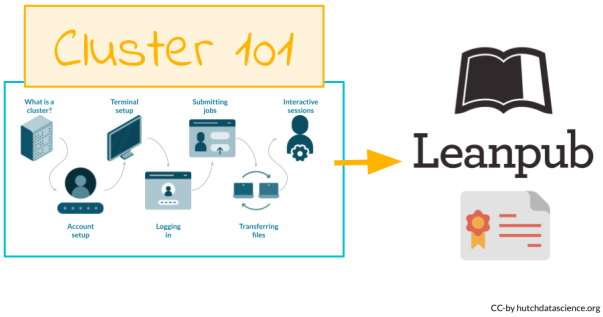
Skip to Certification

Are you an experienced user? Do you need certification? You can jump straight to the 10-question quiz by clicking the link below. **This Leanpub quiz can be taken for free, but you still have to put the course in your cart and check out**.

Self-Test: Cluster 101

An experienced user:

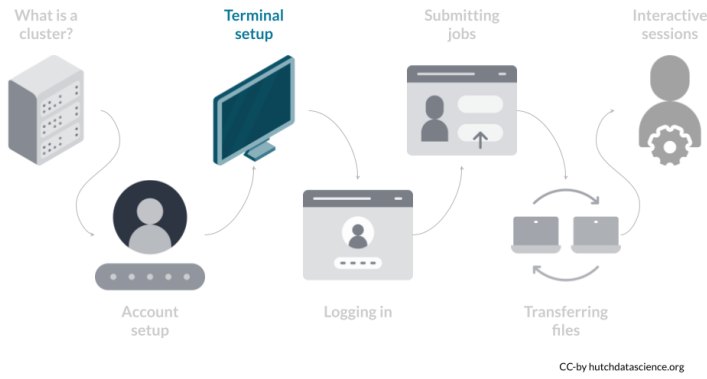
- Has used SSH to connect to a computing cluster
- Is familiar with the methods used to connect to the Fred Hutch cluster
- Has used Slurm to submit a computing job
- Is familiar with the Cyberduck application for transferring files



Cluster 101

Chapter 3

Terminal Setup



The next step is getting familiar with your Terminal. This is your portal to the cluster.

3.1 What is a terminal?

The Terminal is a command line interface. In other words, the Terminal is a software application that allows you to issue commands directly to your laptop or desktop computer. The Terminal is very useful because it allows you to run commands that don't have a point-and-click equivalent. It can also connect you to computer networks, such as the Fred Hutch cluster! The Terminal setup is different depending on your operating system. Jump to the Windows, MacOS, or Linux sections below.

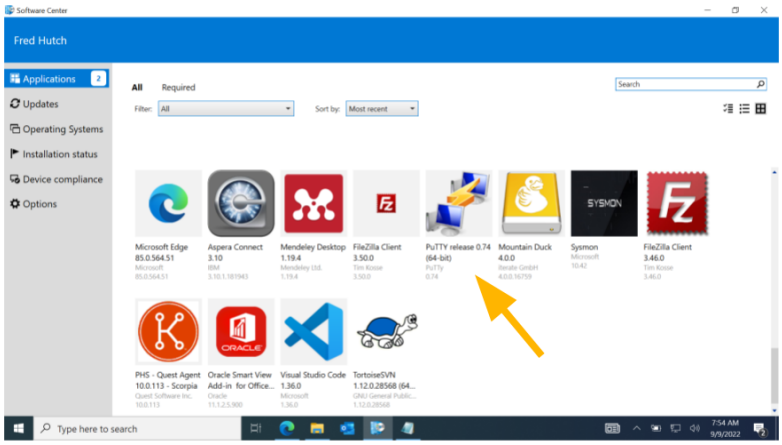
"Terminal" used to be synonymous with "computer". With the creation of operating systems like Windows and MacOS, computers became much easier to use and exploded in popularity! Your colleagues are almost always referring to the command line application when they say "Terminal".

3.2 Windows Setup

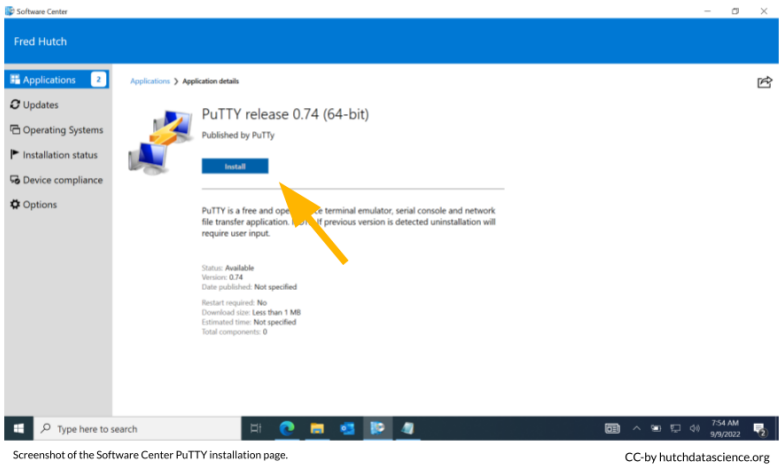
Click to view steps

You will need to install a Terminal application called PuTTY to connect to the Fred Hutch Cluster.

1. You should then see PuTTY available in the Software Center. Click "Install" and go through the Setup Wizard.



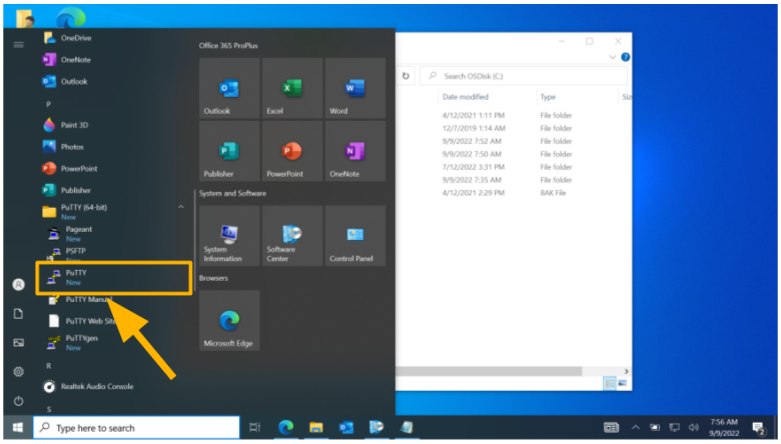
Screenshot of the Software Center application. CC-by hutchdatascience.org



Screenshot of the Software Center PuTTY installation page. CC-by hutchdatascience.org

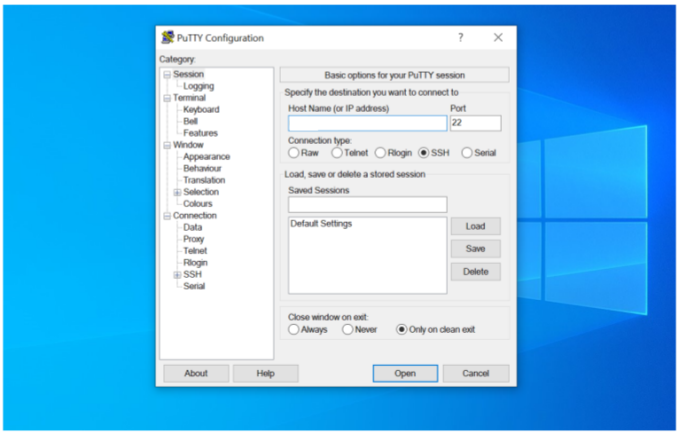
You can also install PuTTY manually if you don't see it in the Software Center.

2. PuTTY should now be available in your applications. Click on PuTTY to open.



Screenshot of the Windows application menu. CC-by hutchdatascience.org

3. You should now see the PuTTY Configuration menu.



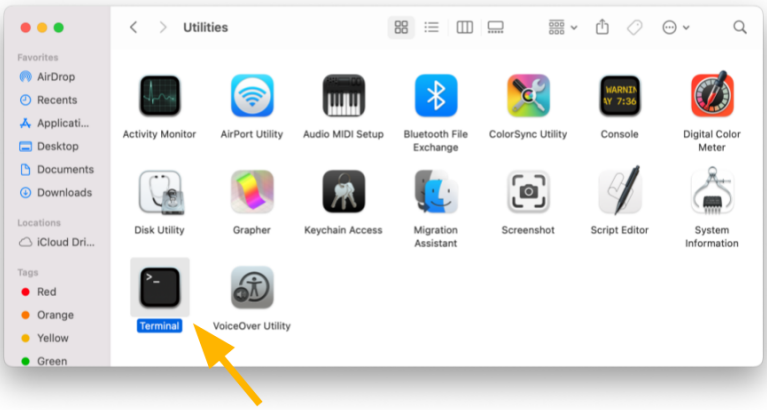
Screenshot of the PuTTY Configuration menu. CC-by hutchdatascience.org

3.3 Mac Setup

Click to view steps

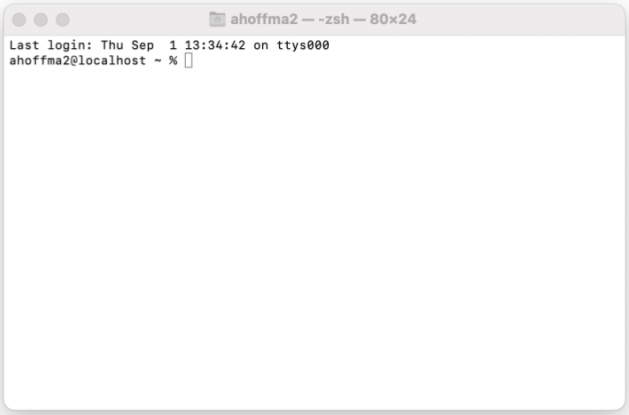
Mac machines come with a Terminal installed.

1. Go to Finder > Applications > Utilities > Terminal and double-click.



Screenshot of the Mac Utilities directory. CC-by hutchdatascience.org

2. Your Terminal should look like this:



Screenshot of the Mac Terminal. CC-by hutchdatascience.org

3.4 Linux Setup

Click to view steps

The commonly used Linux distribution, Ubuntu, already comes with a Terminal installed.

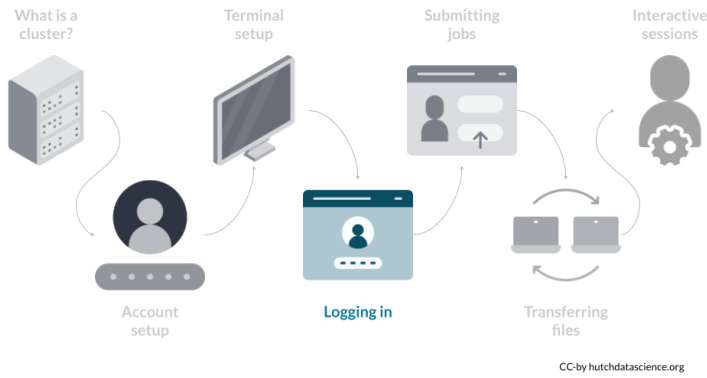
1. Press ctrl + alt + T. This should open a Terminal window.
2. Update the Terminal and prepare it for connecting to the cluster by running:

```
sudo apt install openssh-client
```

Enter your password and enter `Y` when prompted.

Chapter 4

Logging In



Now that you have your Terminal application ready, you want to connect to the cluster. You will do this using a method called SSH, which stands for "Secure SHell".

4.1 What is `SSH`?

SSH is a secure way to remotely connect to another computer or network of computers. In other words, SSH helps us protect your data and the data on the Fred Hutch cluster through authentication.

Hostname

The hostname is the name, or label, assigned to a computer in a network. We are connecting to hostname `rhino.fhcrc.org` or `rhino` for short.

4.2 Connect Securely

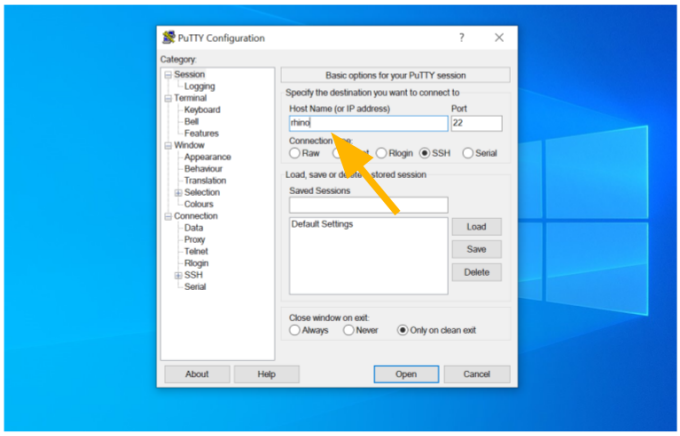
Before moving on, you will need to connect to the Fred Hutch wifi network, a networked ethernet jack, or the Fred Hutch VPN. This is the first layer of security.

The next set of steps are specific to your operating system.

4.3 Windows Login

Click to view steps

1. Go to the PuTTY Configuration menu. Under "Host Name" type **rhino** and click "Open".

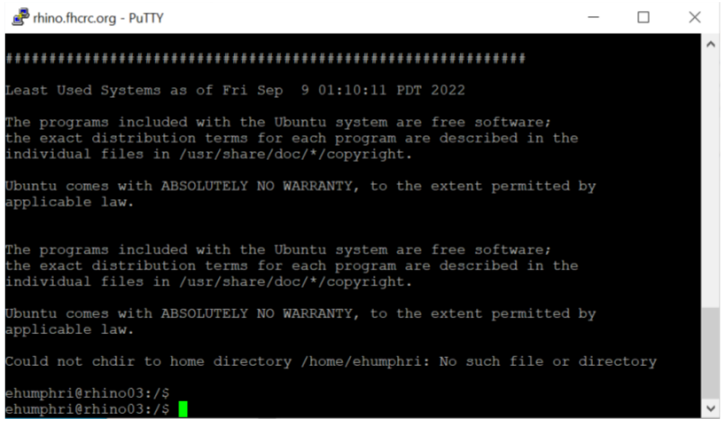


Screenshot of the PuTTY Configuration menu.

2. You will be prompted to login. Type in your HutchNetID (e.g., `jsmith3`).

Screenshot of the PuTTY login prompt for `rhino`. CC-by hutchdatascience.org

3. Enter your password. No∗ or symbols will show up, so type it in carefully!
4. You are now logged in! There should be a login message, with your name at the bottom.



Screenshot of the Fred Hutch cluster login message. CC-by hutchdatascience.org

Congratulations! You are now logged in to the Fred Hutch cluster!

4.4 Mac Login

Click to view steps

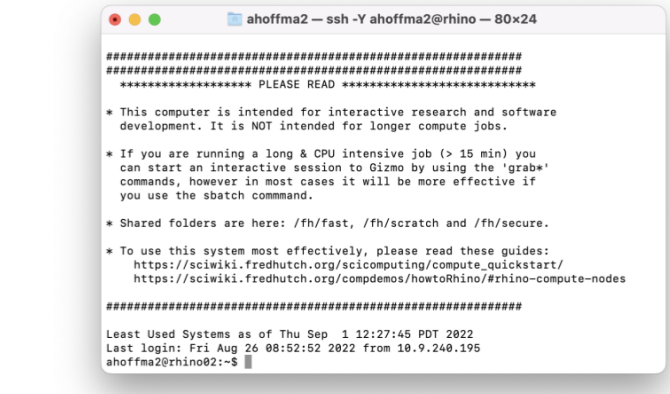
1. Type the following commands, substituting in your HutchNet ID (no brackets):

```
ssh [HutchID]@rhino
```

2. You will see a message that looks like `The authenticity of host 'rhino (XXX.XXX.XX.XX)' can't be established.` Type in `yes` and hit enter.

3. Enter your password. No* or symbols will show up, so type it in carefully!

4. You are now logged in! There should be a login message, with your name at the bottom.



Screenshot of the Fred Hutch cluster login message. CC-by hutchdatascience.org

Congratulations! You are now logged in to the Fred Hutch cluster!

4.5 Linux Login

Click to view steps

1. Type the following commands, substituting in your HutchNet ID (no brackets):

```
ssh [HutchID]@rhino
```

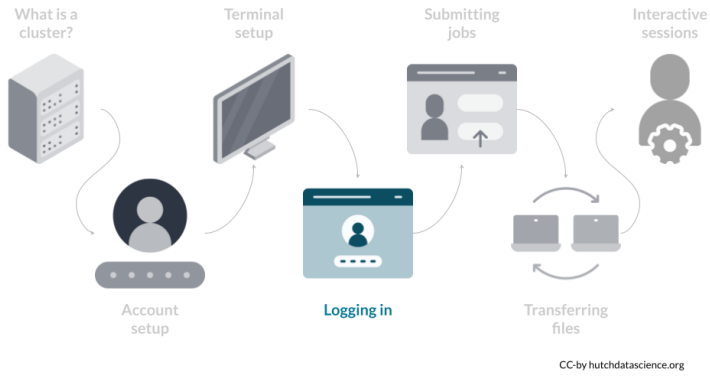
2. Enter your password. No* or symbols will show up, so type it in carefully!

3. You are now logged in! There should be a login message, with your name at the bottom.

Congratulations! You are now logged in to the Fred Hutch cluster!

Chapter 5

Look around



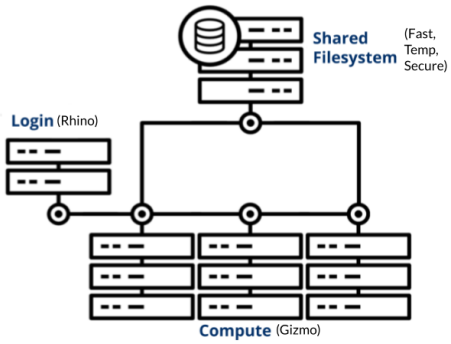
Now that you have successfully logged in to the Fred Hutch cluster, it's time to look around to see what compute node you have connected to, and what file storage systems are available.

5.1 Head and Compute Nodes

After logging in, you are connected to the **Head Node** (alternative names include the Login Node, or the Submit Node). At Fred Hutch, head node is also called **Rhino**. The purpose of the head node is to serve as a launching pad for the user: you are encouraged to look for files you want to use for your analysis, and once you know what data and software you want to use, you perform the computation on the **Compute Nodes** (alternative names include the Worker Node). At Fred Hutch, the compute nodes are called **Gizmo**. The compute

notes have high CPU and RAM capacity for running demanding jobs, whereas the head node has very limited CPUs and RAM.

The diagram below illustrates the relationship between the head node, compute nodes, and the shared file storage systems: you start at the login (head) node, and you can look at the files on the shared file storage system. When you know what data and software you want to use for compute, you let the compute nodes do the job. The compute nodes are also connected to the shared file storage system.



Overview of Fred Hutch cluster, including the Login node, Compute node, and the Shared File System. CC-by hutchdatascience.org

5.2 File Storage Systems

Your current working directory (which can be printed by the `pwd` command) is the Home filesystem. You are given ~100GB of personal space, and this is usually for personal configuration of the cluster as a user.

The other storage resources available to researchers include:

- **Fast** for shared data, including the majority of large scale research data. You can find it at `/fh/fast`.
- **Temp** for temporary storage of files when using the cluster. You can find it at `/hpc/temp`.
- **Secure** for data with higher-level security needs (PHI/encryption/auditing). You can find it at `/fh/secure`.

The Fred Hutch Biomedical Data Science Wiki keeps up to date information about the file storage system.

Head Node

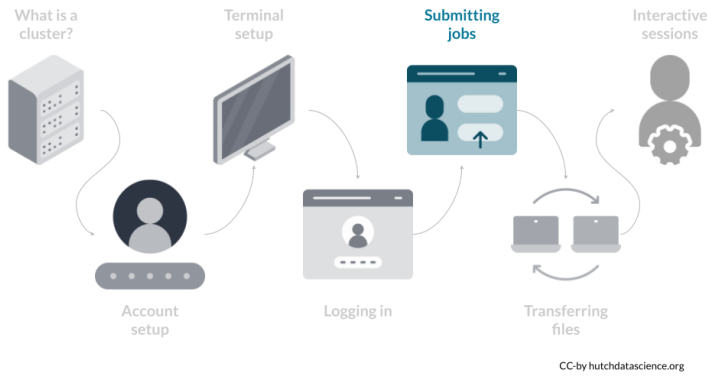
A node that is the default computer you log into when you connect to the Cluster. It has limited CPU and memory, and is used to explore the needed data and software to be used on the Compute Node. Also known as **Login Node** or **Rhino** at Fred Hutch.

Compute Node

A node with extensive CPU and memory used for high performance computing. Also known as the **Worker Node** or **Gizmo** at Fred Hutch.

Chapter 6

Submit Your First Job



The strength of a computing cluster is the ability to do many jobs in parallel or on computers with more computing power than you have on your local computer. The best way to use the cluster is to create short snippets of instructions (a script) that a computer can perform without human input. Your script tells the computers to execute the instructions as individual jobs.

Now that you've logged into rhino you will be able to send scripts to the networked computers in the cluster. The Fred Hutch cluster uses Slurm to organize and prioritize jobs. Based on the instructions in your script, Slurm will find computing resources within the cluster to run your job along with all the other requests from other users.

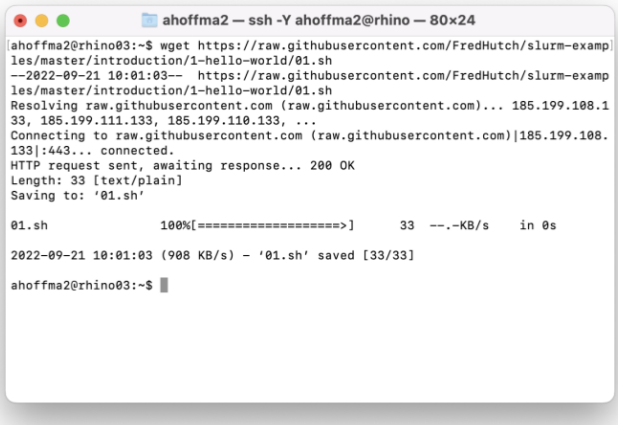
In the next steps, we will go through a simple example where we download a single file. More complicated examples will use multiple files. We will discuss how to transfer files from your computer to the cluster in the following chapter.

The part of the cluster where you **log in** is called `rhino`. The part of the cluster where **jobs are run** is called `gizmo`.

6.1 Download the Script

We can use the `wget` command to download a script from GitHub. This means we don't have to write the script from scratch. Copy and paste the following into the terminal, and hit return:

```
wget https://raw.githubusercontent.com/FredHutch/slurm-examples/main/01-introduction/1
```



Screenshot of wget command output.

6.2 Confirm the Download

Let's confirm that we can see the file we just downloaded. We can use the `ls` (list files) command for this. Type `ls` and hit return. You should see the file `01.sh` in your home directory. The `.sh` ending means this is a script meant to run from the command line.

```
ls
```

Screenshot of ls command output.

6.3 Inspect the Script

Let's next inspect the script. The `cat` command, followed by a file name, lists the entire contents of a specific file.

```
cat 01.sh
```

Screenshot of cat command output, the contents of the script.

- The first line of the script, `#!/bin/bash`, indicates that this is a command line or "bash" script.

- The second line is empty, and the third line, `echo "Hello, World"` means that the computer will "echo", or print out, "Hello, World".

6.4 Submit the Script

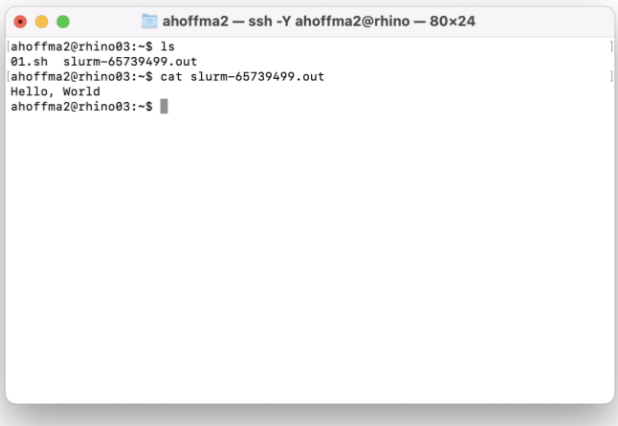
We use the `sbatch` command to submit a script and start running a job on the cluster. Copy the following and hit return. You should see a message like "`Submitted batch job 12345678`". Your number will vary because this is a unique job identifier.

```
sbatch 01.sh
```

6.5 Check the Output

Type `ls` again. You should now see a log file like `slurm-12345678.out` listed alongside your script `01.sh`. Let's use `cat` to inspect the output in the log file (the new file starting with `slurm` and ending with `.out`). Make sure you replace `[your-number-here]` with the number in your actual file. We should see our message has been printed!

```
cat slurm-[your-number-here].out
```



Screenshot of cat command output, the contents of the log file.


```
ls
```

This command lists the files in the current directory.

`cat filename`

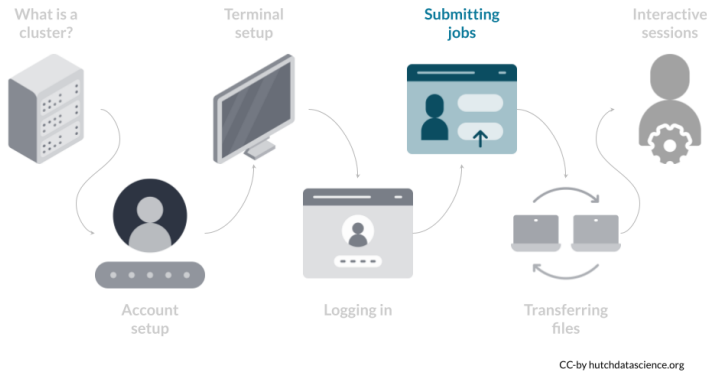
This command prints the contents of a specific file (*filename*).

`sbatch filename.sh`

This command submits a job to the cluster with instructions specified in a `.sh` file.

Chapter 7

Custom jobs



Once you are comfortable submitting simple jobs, we point out ways customize your jobs to make best use of the computing infrastructure: perhaps you want to change the number of CPUs or RAM to request, or you want to monitor the status of a job running. This section highlights some common requests people use on the Fred Hutch Slurm system.

7.1 Requesting Specific Computing Resources

By default, when you ran `sbatch 01.sh` in the previous chapter, `sbatch` requested for one CPU on a computing node for your task. Let's change the number of CPUs requested. You can change the number of requested CPUs based on the `-c` or `--cpus-per-task` option: if you want to request 4 CPUs for your job, you would enter the following:

```
sbatch -c 4 01.sh
```

or

```
sbatch --cpus-per-task 4 01.sh
```

How about changing the amount of RAM (memory) requested? Currently, the Fred Hutch Slurm does not have a memory option built in to guarantee the amount of memory that will be allocated to your job. Instead, the current recommendation is to request CPUs as a proxy for RAM allocation, based on this rule-of-thumb:

> "If you think your job will need more than 4GB of memory, request one CPU for every 4GB required."

The technical reasoning behind the rule-of-thumb can be expanded here.

FH Gizmo has class J nodes which each have 24 cores and 384GB of memory, and class K nodes which each have 36 cores and 768GB of memory. So, if you think you will need 100GB of memory for your job, by this rule of thumb you would request 25 cores. You would be assigned to a class K node, and you would occupy 25/36 cores on that node. On this node, other users can use the remaining 11 cores. You would share the 768GB of memory all together and hope that the other users don't take up more memory than you need: the more cores you occupy on a node, less users will compete for memory. It's an imprecise system and SciComp has interest to make memory allocation more precise in the future.

You can learn more about other use cases of `sbatch` at the Biomedical Data Science Wiki's page on computing jobs.

7.1.1 Saving your `sbatch` options

The options you pick for the `sbatch` command, especially around CPU and RAM requests, can have enormous impact on your computing speed and quality. Therefore, it is often a good idea to save the options you have picked into the shell script itself. Suppose that you launched your job like this:

```
sbatch -c 4 01.sh
```

To save the `-c 4` option in your script `01.sh`, you can add the following header `#SBATCH -c 4` to your script:

```
#!/bin/bash

#SBATCH -c 4

echo "Hello, World!"
```

Now, you will automatically request 4 CPUs when you run:

```
sbatch 01.sh
```

7.2 Monitoring and Ending Jobs

Often, our jobs take time to run, and we want to see the status of our jobs. You can examine the status of your job via the `squeue` command. The default option will show *all* jobs running on the cluster, including other users, so it's common to use the `-u` option to specify the user. The example user here is `clo2`:

```
squeue -u clo2
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          36277207 campus-ne HelloWor     clo2 PD       0:00      1 (Priority)
```

Alternatively, you can also use `--me` to specify the user as yourself:

```
squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          36277207 campus-ne HelloWor     clo2 PD       0:00      1 (Priority)
```

Here, the status ("ST") shows `PD`, which means that the job hasn't started to run yet. When it is running the status will be `R`.

Some common reasons for `PD` include: there are no available resources left for the requested resources for your job, or that your account has already maxed out its allocated resources. The decision of which user's jobs are running is based on the Slurm scheduler system. You can learn more about `squeue`'s outputs at the Biomedical Data Science wiki's page here.

If you want to end a job that is either waiting (`PD` status) or running, you can enter:

```
scancel [job id]
```

where `[job id]` is a numerical ID attached to your unique job. It can be found under `JOBID` when you run `squeue`.

```
sbatch -c 4 filename.sh
```

This command submits a job to the cluster with instructions specified in a `.sh` file with request of 4 cores.

```
squeue --me
```

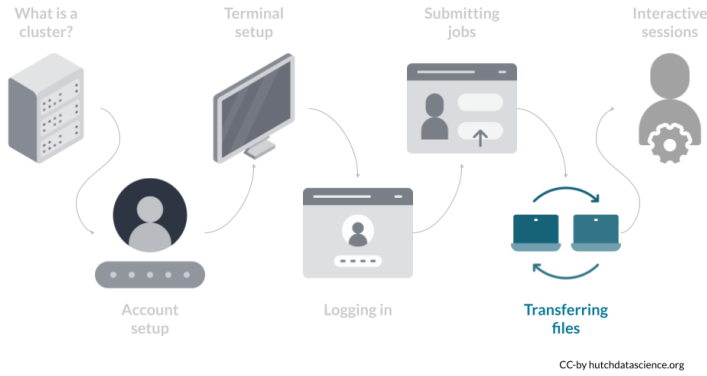
This command prints the job submission status for your username.

```
scancel 1234
```

This command cancels job ID 1234. The job ID can be found in the `squeue` command.

Chapter 8

File Upload and Download



Exchanging files with the cluster is very important. You can imagine scenarios where:

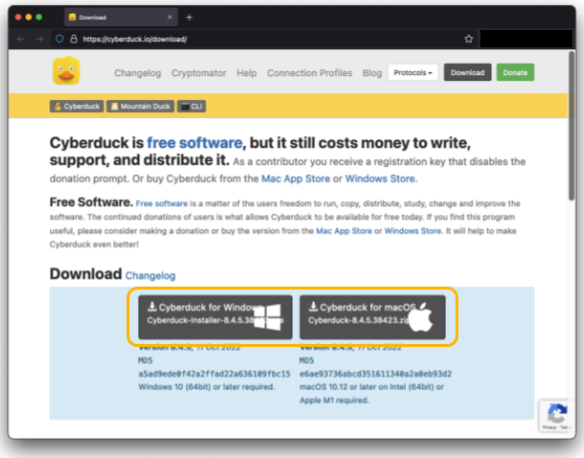
- You want to download log files or output files
- You want to upload a custom `.sh` script file that you wrote on your laptop
- You want to upload other files

In this course, upload and download of files is performed using Cyberduck. Cyberduck is a tool that lets us connect to the cluster securely, browse files, and transfer files securely.

If you are working with sensitive data (such as data with PHI that requires HIPAA compliance, you need to be extra cautious about transferring your data to the cluster. Your home directory is not an appropriate storage option for such data. Make sure you consider any stipulations in your data use agreements.

8.1 Download Cyberduck

Download the latest version of Cyberduck here.



Screenshot of the Cyberduck downloads page.

CC-by hutchdatascience.org

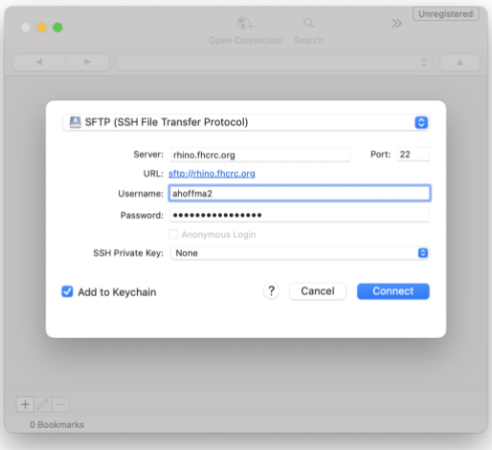
Note that the version of Cyberduck in the Software Center or Self Service might not be current, causing compatibility issues with some operating systems.

8.2 Create Connection

Launch Cyberduck and click on “Open Connection”.

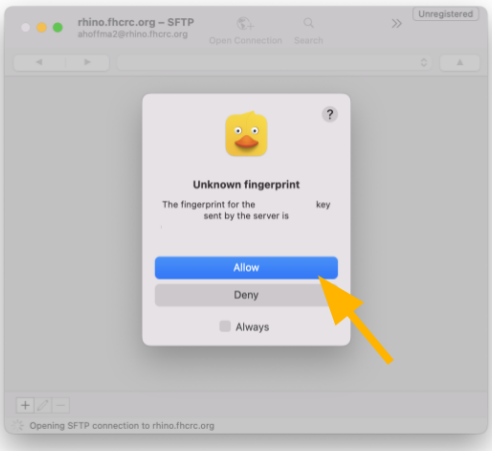
- From the dropdown menu, select “SFTP (SSH File Transfer Protocol)”
- For Server, type “rhino.fhcrc.org”
- Fill in your HutchNetID for Username and fill in your password

Click “Connect”



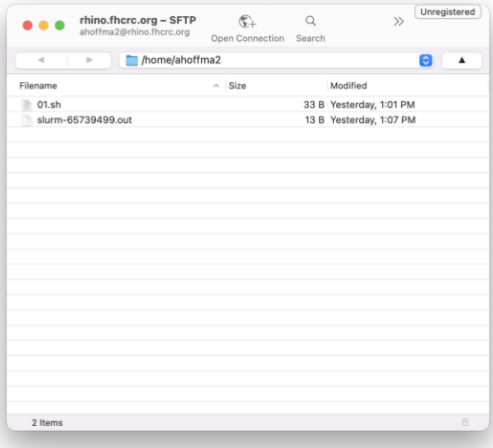
Screenshot of the Cyberduck "Open Connection" configuration.

Click "Allow". You can also check the box to indicate "Always".



Screenshot of server fingerprint prompt.

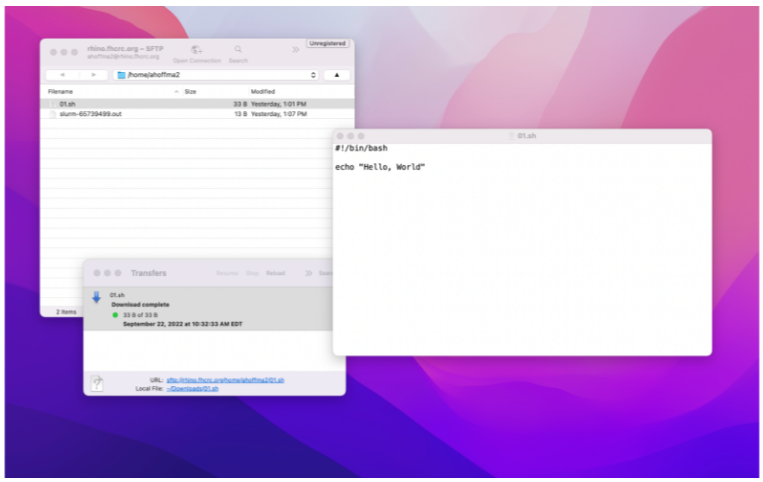
You should see your script file "`01.sh`" and the log file.



Screenshot of files on the cluster, viewed through Cyberduck.

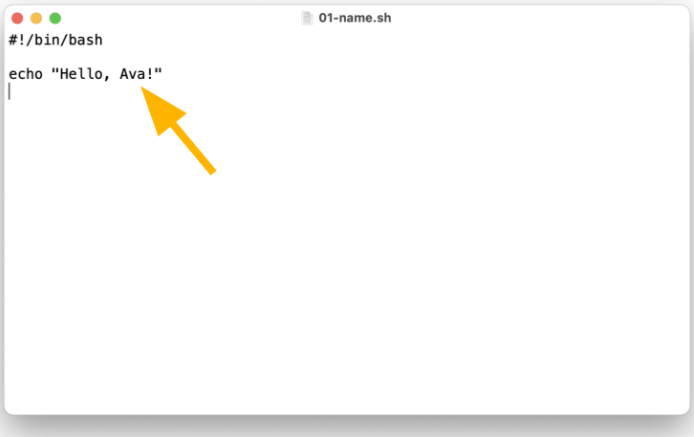
8.3 Download and Edit the Script

- Right click on "`01.sh`" and select "Download"
- You will see a "Transfers" prompt open, and the `01.sh` file should now appear in your Downloads folder
- Open the `01.sh` in your Downloads folder



Screenshot of files, Transfers window, and the downloaded script.

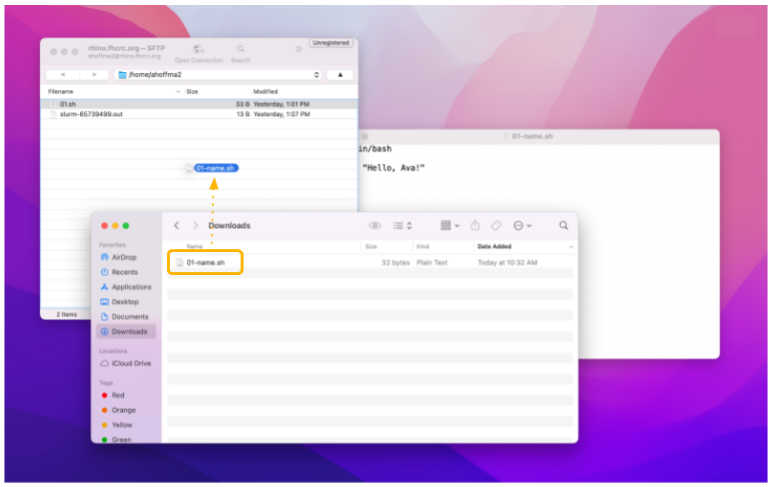
Edit the message to include your name and save the file. Rename the file `01-name.sh`.



Screenshot of the script, edited with a name.

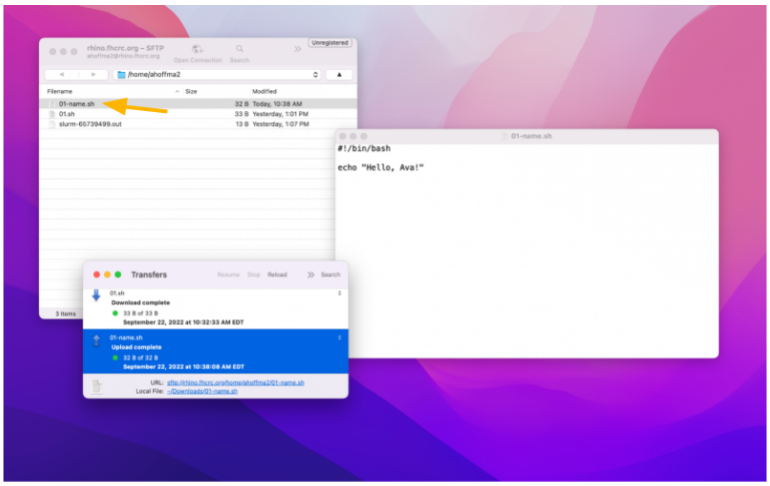
8.4 Upload the New Script

From your Downloads folder, simply drag the file to Cyberduck.



Screenshot of Downloads folder with edited script being dragged over to files via Cyberduck.

You should now see the new script among your cluster files.

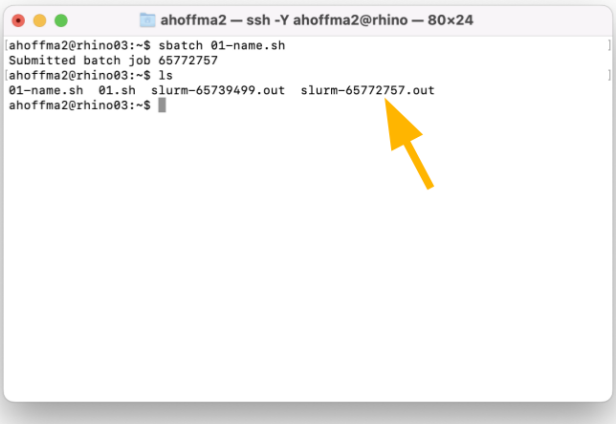


Screenshot of edited script among cluster files visible on Cyberduck.

8.5 Run the New Script

Return to your Terminal. Submit a job with your new script by running the following. When you type `ls` you should see a new log file!

```
sbatch 01-name.sh
```

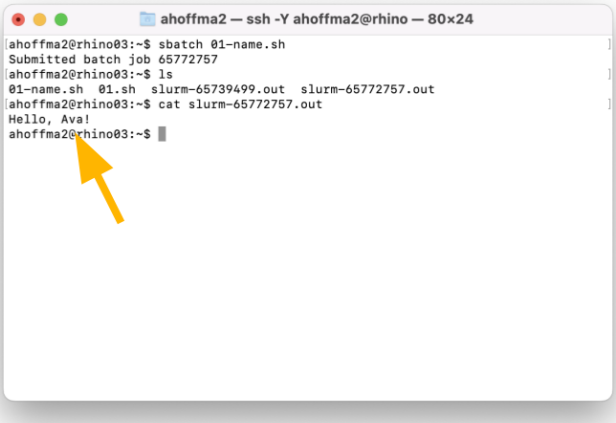


Screenshot showing new log file present on the cluster.

The job numbers included in log file names generally increase in number. The greater the number, the more recently the job was run.

Use the `cat` command to inspect the log. Make sure you replace `[your-number-here]` to match your file. The message should show the new text that you added!

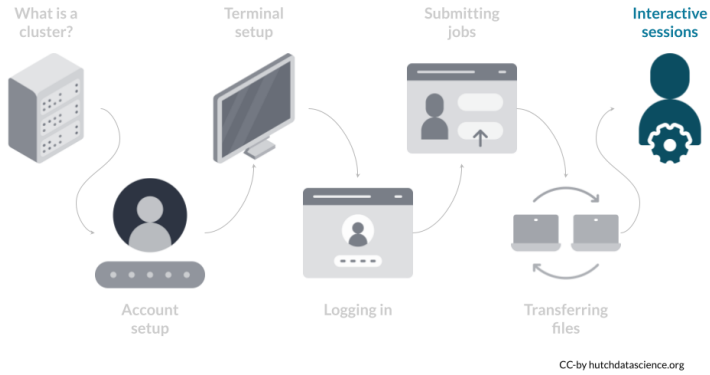
```
cat slurm-[your-number-here].out
```



Screenshot of edited script message output using `cat` command.

Chapter 9

Interactive Session



While using the cluster, you might need to build and test scripts interactively before running them. Luckily, you can work directly on the cluster by creating an interactive session. When you launch an interactive session, the cluster assigns you a portion of the networked computers called a "node". This node (or part of one) is dedicated to you for a period of time rather than using the Slurm job submission system.

Because an interactive session takes up resources directly on the cluster whether you're actively using it or not, it's best to use interactive sessions only when a task cannot be done by submitting a script.

9.1 Starting the session

When starting an interactive session, you're going to need to think about what you are testing and what resources you might need on the node you are request-

ing to use. You can always start an interactive session using the default values if you aren't sure what you need yet.

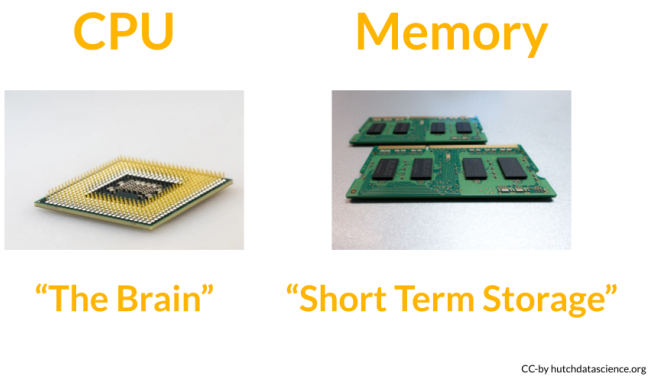
Start an interactive session on a node by running the command:

```
grabnode
```

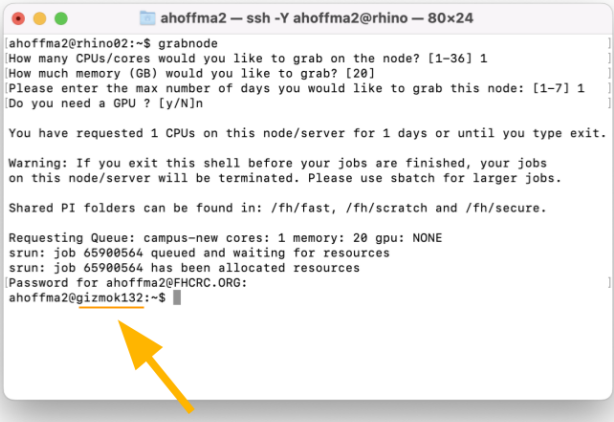
You will be prompted with several questions about the type of resources on the node you want. We don't need anything fancy, so we will set up the session to use minimal resources. You can enter the following:

- *How many CPUs/cores would you like to grab on the node?* **1**
- *How much memory (GB) would you like to grab?* **20**
- *Please enter the max number of days you would like to grab this node:* **1**
- *Do you need a GPU?* **N**
- When prompted, enter your password

You requested 1 CPU, 20GB of RAM, and don't need a GPU. The GPU, or Graphics Processing Unit, is similar to the CPU. The GPU was originally designed to quickly render graphics (such as for video games), but today can be used to run complex artificial intelligence applications or computationally intensive jobs.



You will see that you are now logged on to the compute node "gizmo" instead of the login node "rhino". Remember that the part of the cluster where you log in is called `rhino`. The part of the cluster where jobs are run is called `gizmo`.



Screenshot of configuration prompts for an interactive node.

9.2 Running Interactive Commands

You can start working on the node by running a similar command as we used in the job we submitted via script. Echo a message by running:

```
echo "Hello, again!"
```

Screenshot of an echo message on an interactive node.

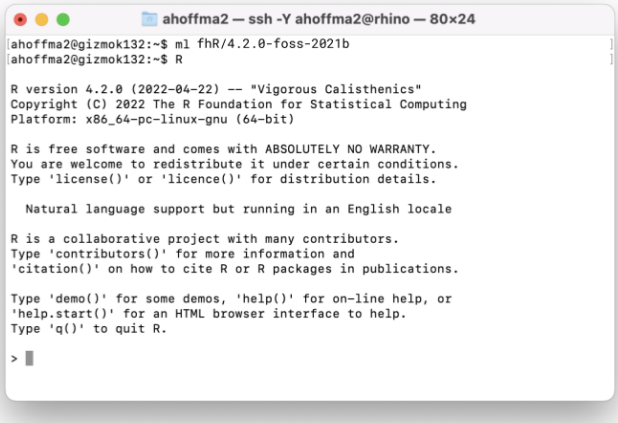
9.3 Using Pre-installed Software Modules

Let's get a bit more advanced. We can load a preconfigured software bundle called a module. This is very convenient because it means we don't need to install anything manually! In this example, we will load a module containing R version 4.2.0. You can learn more about what modules are available and how to request new ones for the Fred Hutch cluster here.

```
ml fhR/4.2.0-foss-2021b
```

Next, launch R:

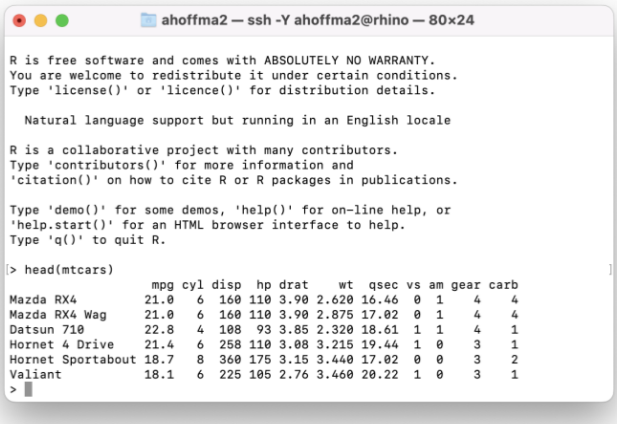
```
R
```



Screenshot of R session launched on an interactive node.

You can play around with R here. For example, you might run:

```
head(mtcars)
```



Screenshot of R commands on an interactive node. CC-by hutchdatascience.org

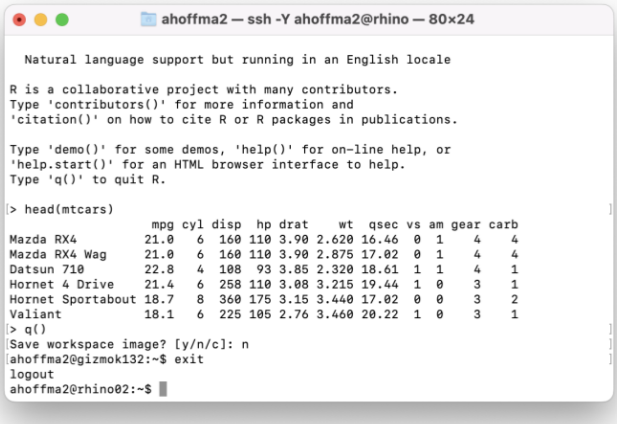
Close the R session by typing:

```
q()
```

R will ask if you want to save your workspace. Type `n` for no and hit return.

Close the interactive node by typing:

```
exit
```



Screenshot exiting the interactive node. CC-by hutchdatascience.org

9.3.1 Loading Modules in non-interactive mode

You have just seen how to load in software modules in the interactive mode. When you launch a job via `sbatch` to start computing on the compute nodes, you may also need to load in software modules. You can add the `ml` command within your shell script, such as the following:

```
#!/bin/bash

ml fhR/4.2.0-foss-2021b
Rscript my_r_script.R
```

```
grabnode
```

This command starts an interactive session on the cluster.

Chapter 10

Getting Help

The Scientific Computing group in IT manages the cluster, provides support with software, advises on data storage, and holds office hours specifically to help users. Here are some ways you can get help for your work on the cluster.

Check out the FAQ page

See our FAQ and Troubleshooting Page to see common errors and what they mean. If you encounter a problem that isn't listed, let us know!

Find Community Support on Slack

Peer-to-peer support can be very valuable in learning and troubleshooting your work. The Fred Hutch Data Slack workspace is open to all with a fredhutch.org, uw.edu, seattlechildrens.org, or related institution email addresses (whi.org, scharp.org, etc). You can ask questions, find out about office hours, and discover other live support and training events that can help you learn more about how to leverage resources at Fred Hutch to advance your science.

Visit the SciWiki

The SciWiki Scientific Computing page is full of useful tips and guides. Remember when using the search that the login "nodes" to the Fred Hutch cluster are called rhino and the cluster "nodes" are called gizmo.

Send an Email

The primary way you can request help for a problem is to send SciComp an email, so a ticket will be created in their tracking system. This allows the details of the problem you're having to be sent to them so they can better help you. Submitting a good email ticket helps the SciComp Team address your needs quickly and efficiently. We suggest you submit the following information:

1. A brief overview of what the problem is.
2. Some specifics about the problem, such as the full text (it's ok if it's long) of any error message or terminal command, or a screen shot of the interface you were using when you had the problem.
3. A description of what you wanted to have happen or what your overall goal is (in case perhaps there is another strategy that might work better).

Chapter 11

Summary

Let's review the key information from the previous sections.

11.1 Overview

A computing cluster is a set of computers networked together to perform large tasks. We usually connect to the cluster using a command-line interface called a Terminal. The Terminal application varies based on your operating system. We connect to the cluster using a secure method called SSH. Exact SSH commands vary depending on your operating system.

Before using the Fred Hutch cluster, it's a good idea to contact the SciComp team to ensure your account is set up correctly. You must be connected to the campus wifi network, plugged into a networked ethernet jack, or connected to the Fred Hutch VPN to connect to the Fred Hutch cluster.

Computer tasks are typically performed via job submission, where you tell the cluster what to do inside a script file. The Fred Hutch cluster uses Slurm to organize jobs submitted by different users. When building, testing, and debugging jobs, you might want to launch an interactive session on the cluster. You will be asked several questions about resources before your session starts.

You can use the application Cyberduck to transfer smaller files between your local computer and the cluster via SFTP (SSH File Transfer Protocol). This course does not cover transferring sensitive data, which requires extra precautions.

Don't be afraid to ask for help! Check out the SciWiki, join the Slack workspace, or contact the SciComp team if you get stuck.

11.2 Glossary

- **computing cluster** - A set of computers networked together to perform large tasks.
- **CPU** - A computer component that performs and orchestrates computational tasks.
- **Cyberduck** - Third-party software which transfers files between your local machine and the cluster.
- **hostname** - The hostname is the name, or label, assigned to a computer in a network. We connect to hostname `rhino.fhcrc.org` or `rhino` for short.
- **memory** - A computer component that stores calculations and information in the short term.
- **node** - A part of the cluster; a computer in the network.
- **SSH** - A secure method for remotely connecting to another computer or network of computers; stands for "Secure SHell".
- **terminal** - A command line interface; a software application that allows you to issue commands directly to a computer.

11.3 Commands (command line interface)

Command	Description	Usage
`cat`	displays all contents of a file	`cat [filename]`
`exit`	terminates an interactive session	`exit`
`grabnode`	launches an interactive session	`grabnode`
`ls`	lists files in the current folder	`ls`
`sbatch`	submits a script containing job instructions requesting custom cores	`sbatch -c [cores] [scriptname.sh]`
`squeue`	lists job submission status for the current user	`squeue --me`
`scancel`	cancel job submission by job ID	`scancel [job ID]`
`wget`	downloads a file from the internet	`wget [url]`

Appendix

Chapter 12

Provide Feedback

We'd love to hear from you! Please fill out the anonymous Google Form with your feedback.

You can submit an issue about this course at our GitHub repository. You can also click the edit button on the top of the page in question.

Chapter 13

FAQ and Troubleshooting

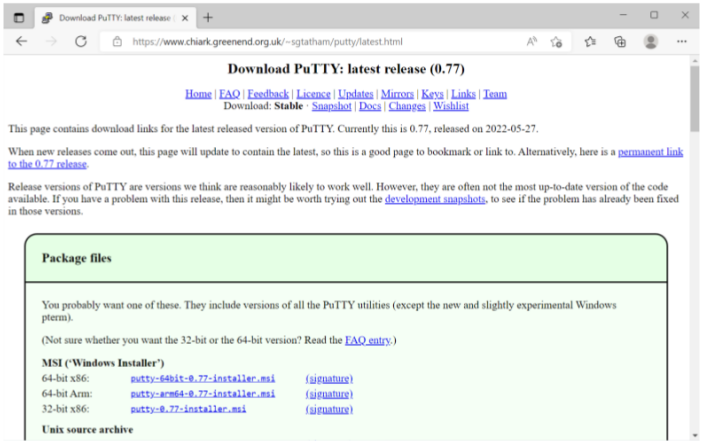
13.1 FAQ

Here are some questions you might have.

13.1.1 How can I manually install PuTTY?

Click to view steps

1. Click here to install the latest version of PuTTY. You will choose the 64-bit x86 installation with few exceptions.



Screenshot of the PuTTY download page. CC-by hutchdatascience.org

2. Click through to install via the Setup Wizard.

13.2 Troubleshooting

Here are some issues you might encounter.

`ssh: Could not resolve hostname rhino: nodename nor servname provided, or not known`

Click to view steps

This error means that your computer is having trouble connecting to rhino. Ensure one of the following is true:

1. You are connected to the Fred Hutch wifi network on campus.
2. You are connected to the Fred Hutch VPN
3. You are plugged into an ethernet cable on campus that taps into the Fred Hutch network. Note that not all ethernet wall jacks have this capability, so try another jack if you are having trouble. Please email the IT helpdesk and include your office number and the number on the jack if you find a jack that isn't working.

`ssh: connect to host rhino port 22: Undefined error: 0`

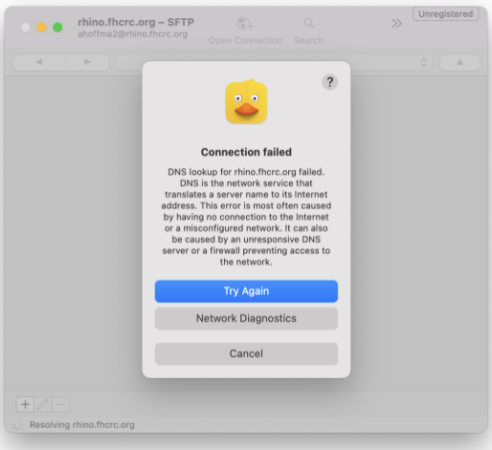
Click to view steps

This likely indicates a disruption to your internet connection and/or VPN. Ensure you are connected to the internet and connected to the Fred Hutch network on campus or the VPN.

`Connection failed` message in Cyberduck

Click to view steps

This likely indicates a disruption to your internet connection and/or VPN. Ensure you are connected to the internet and connected to the Fred Hutch network on campus or the VPN.



Screenshot of the Cyberduck "Connection failed" message.

`Invalid account or account/partition` when logging in

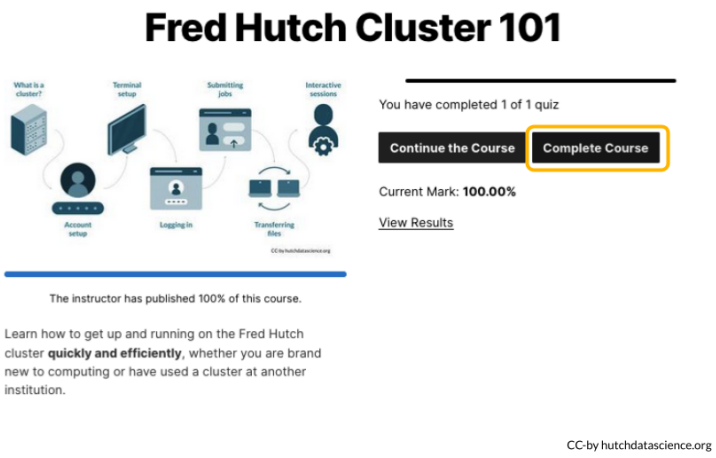
Errors similar to this typically indicate that the account hasn't been set up by SciComp. This is a quick fix if you use the form mentioned in the course.

Get Your Certificate

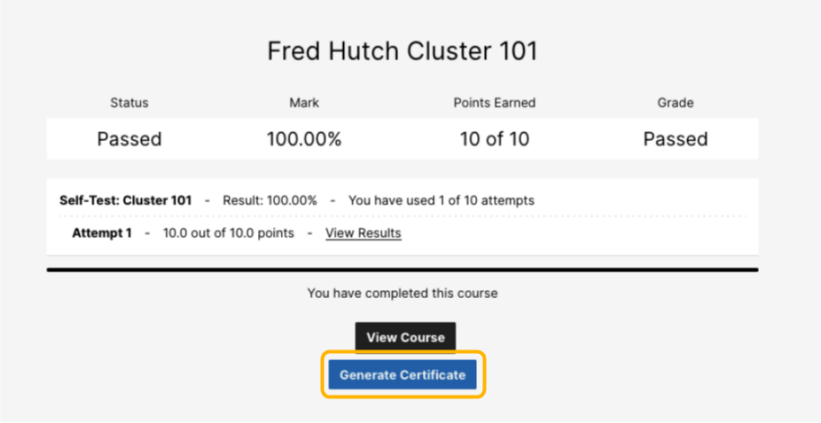
You must complete the quiz - Self-Test: Cluster 101 - with all questions correctly answered to earn your certificate for this course. Once complete:

1. Go to the course homepage

2. Click on “Complete Course”



3. Click on “Generate Certificate”



Contact the DaSL Team to submit your Cluster 101 certificate for an awesome Hex Sticker!

About the Authors

These credits are based on our course contributors table guidelines.

Please note that this course is under development and these credits are subject to change!

Credits	Names
Pedagogy	
Lead Content Instructor	Ava Hoffman
Content Authors	The Fred Hutch SciComp Team
Content Editors	Amy Paguirigan
Content Reviewers	Elizabeth Humphries, Candace Savonen, Carrie Wright
Content Consultants	The Fred Hutch SciComp Team
Technical	
Course Publishing Engineer	Ava Hoffman
Template Publishing Engineers	Candace Savonen, Carrie Wright
Publishing Maintenance Engineer	Candace Savonen
Technical Publishing Stylists	Carrie Wright, Candace Savonen
Package Developers (ottrpal) Candace Savonen, John Muschelli, Carrie Wright	

```
## - Session info ---------------------------------------------------------------
##  setting  value
##  version  R version 4.3.2 (2023-10-31)
##  os       Ubuntu 22.04.4 LTS
##  system   x86_64, linux-gnu
##  ui       X11
```

```
##  language (EN)
##  collate  en_US.UTF-8
##  ctype    en_US.UTF-8
##  tz       Etc/UTC
##  date     2024-11-18
##  pandoc   3.1.1 @ /usr/local/bin/ (via rmarkdown)
##
## - Packages -------------------------------------------------------------------
##  package     * version date (UTC) lib source
##  askpass       1.2.0   2023-09-03 [1] RSPM (R 4.3.0)
##  bookdown      0.41    2024-10-16 [1] CRAN (R 4.3.2)
##  cachem        1.0.8   2023-05-01 [1] RSPM (R 4.3.0)
##  chromote      0.3.1   2024-08-30 [1] CRAN (R 4.3.2)
##  cli           3.6.2   2023-12-11 [1] RSPM (R 4.3.0)
##  devtools      2.4.5   2022-10-11 [1] RSPM (R 4.3.0)
##  digest        0.6.34  2024-01-11 [1] RSPM (R 4.3.0)
##  dplyr         1.1.4   2023-11-17 [1] RSPM (R 4.3.0)
##  ellipsis      0.3.2   2021-04-29 [1] RSPM (R 4.3.0)
##  evaluate      0.23    2023-11-01 [1] RSPM (R 4.3.0)
##  fansi         1.0.6   2023-12-08 [1] RSPM (R 4.3.0)
##  fastmap       1.1.1   2023-02-24 [1] RSPM (R 4.3.0)
##  fs            1.6.3   2023-07-20 [1] RSPM (R 4.3.0)
##  generics      0.1.3   2022-07-05 [1] RSPM (R 4.3.0)
##  glue          1.7.0   2024-01-09 [1] RSPM (R 4.3.0)
##  hms           1.1.3   2023-03-21 [1] RSPM (R 4.3.0)
##  htmltools     0.5.7   2023-11-03 [1] RSPM (R 4.3.0)
##  htmlwidgets   1.6.4   2023-12-06 [1] RSPM (R 4.3.0)
##  httpuv        1.6.14  2024-01-26 [1] RSPM (R 4.3.0)
##  httr          1.4.7   2023-08-15 [1] RSPM (R 4.3.0)
##  janitor       2.2.0   2023-02-02 [1] RSPM (R 4.3.0)
##  jsonlite      1.8.8   2023-12-04 [1] RSPM (R 4.3.0)
##  knitr         1.48    2024-07-07 [1] CRAN (R 4.3.2)
##  later         1.3.2   2023-12-06 [1] RSPM (R 4.3.0)
##  lifecycle     1.0.4   2023-11-07 [1] RSPM (R 4.3.0)
##  lubridate     1.9.3   2023-09-27 [1] RSPM (R 4.3.0)
##  magrittr      2.0.3   2022-03-30 [1] RSPM (R 4.3.0)
##  memoise       2.0.1   2021-11-26 [1] RSPM (R 4.3.0)
##  mime          0.12    2021-09-28 [1] RSPM (R 4.3.0)
##  miniUI        0.1.1.1 2018-05-18 [1] RSPM (R 4.3.0)
##  openssl       2.1.1   2023-09-25 [1] RSPM (R 4.3.0)
##  ottrpal       1.3.0   2024-10-23 [1] Github (jhudsl/ottrpal@2e19782)
##  pillar        1.9.0   2023-03-22 [1] RSPM (R 4.3.0)
##  pkgbuild      1.4.3   2023-12-10 [1] RSPM (R 4.3.0)
##  pkgconfig     2.0.3   2019-09-22 [1] RSPM (R 4.3.0)
##  pkgload       1.3.4   2024-01-16 [1] RSPM (R 4.3.0)
##  processx      3.8.3   2023-12-10 [1] RSPM (R 4.3.0)
```

```
##  profvis       0.3.8   2023-05-02 [1] RSPM (R 4.3.0)
##  promises      1.2.1   2023-08-10 [1] RSPM (R 4.3.0)
##  ps            1.7.6   2024-01-18 [1] RSPM (R 4.3.0)
##  purrr         1.0.2   2023-08-10 [1] RSPM (R 4.3.0)
##  R6            2.5.1   2021-08-19 [1] RSPM (R 4.3.0)
##  Rcpp          1.0.12  2024-01-09 [1] RSPM (R 4.3.0)
##  readr         2.1.5   2024-01-10 [1] RSPM (R 4.3.0)
##  remotes       2.4.2.1 2023-07-18 [1] RSPM (R 4.3.0)
##  rlang         1.1.4   2024-06-04 [1] CRAN (R 4.3.2)
##  rmarkdown     2.25    2023-09-18 [1] RSPM (R 4.3.0)
##  rprojroot     2.0.4   2023-11-05 [1] CRAN (R 4.3.2)
##  sessioninfo   1.2.2   2021-12-06 [1] RSPM (R 4.3.0)
##  shiny         1.8.0   2023-11-17 [1] RSPM (R 4.3.0)
##  snakecase     0.11.1  2023-08-27 [1] RSPM (R 4.3.0)
##  stringi       1.8.3   2023-12-11 [1] RSPM (R 4.3.0)
##  stringr       1.5.1   2023-11-14 [1] RSPM (R 4.3.0)
##  tibble        3.2.1   2023-03-20 [1] CRAN (R 4.3.2)
##  tidyselect    1.2.0   2022-10-10 [1] RSPM (R 4.3.0)
##  timechange    0.3.0   2024-01-18 [1] RSPM (R 4.3.0)
##  tzdb          0.4.0   2023-05-12 [1] RSPM (R 4.3.0)
##  urlchecker    1.0.1   2021-11-30 [1] RSPM (R 4.3.0)
##  usethis       2.2.3   2024-02-19 [1] RSPM (R 4.3.0)
##  utf8          1.2.4   2023-10-22 [1] RSPM (R 4.3.0)
##  vctrs         0.6.5   2023-12-01 [1] RSPM (R 4.3.0)
##  webshot2      0.1.1   2023-08-11 [1] CRAN (R 4.3.2)
##  websocket     1.4.2   2024-07-22 [1] CRAN (R 4.3.2)
##  xfun          0.48    2024-10-03 [1] CRAN (R 4.3.2)
##  xml2          1.3.6   2023-12-04 [1] RSPM (R 4.3.0)
##  xtable        1.8-4   2019-04-21 [1] RSPM (R 4.3.0)
##  yaml          2.3.8   2023-12-11 [1] RSPM (R 4.3.0)
##
##  [1] /usr/local/lib/R/site-library
##  [2] /usr/local/lib/R/library
##
## ------------------------------------------------------------------------------
```