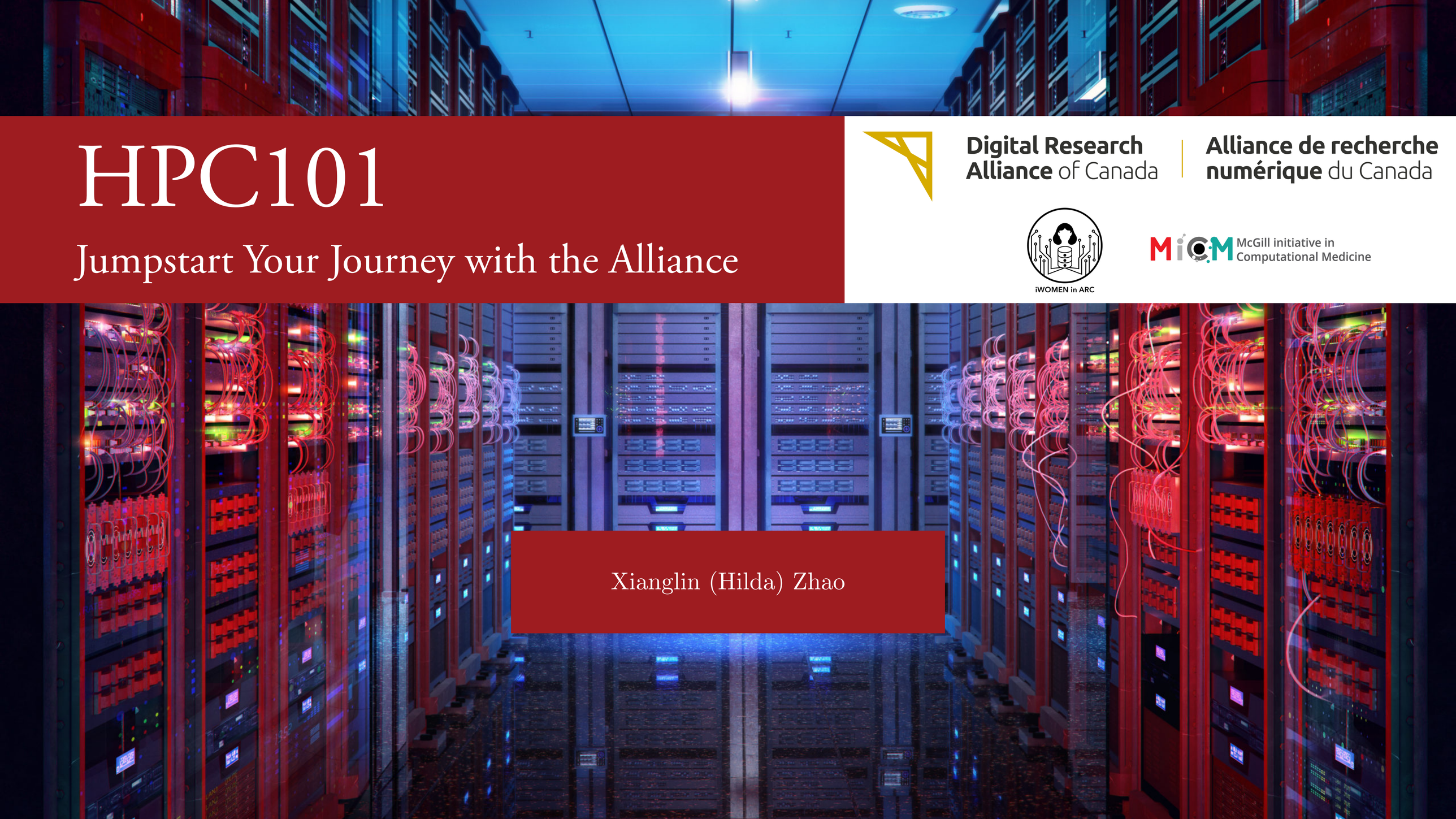


HPC101

Jumpstart Your Journey with the Alliance

Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

iWOMEN in ARC

MiCM McGill initiative in Computational Medicine

Xianglin (Hilda) Zhao

Land Acknowledgement

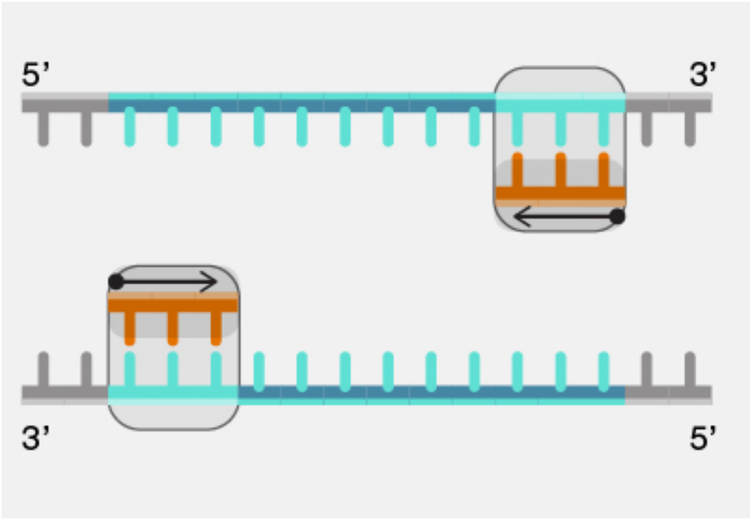
McGill University is on land which has served and continues to serve as a site of meeting and exchange amongst Indigenous peoples, including the Haudenosaunee and Anishinaabeg nations. We acknowledge and thank the diverse Indigenous peoples whose footsteps mark this territory on which peoples of the world now gather.

About mentor…

- I use HPC to..
  - Run simulations that take forever and cause fatal error on my own computer.
  - I mainly use R.
  - I will use MacOS and R as examples for demonstration, but the main ideas are portable.
  - We have TAs with experience with Windows!
- Please do not share the slides outside the workshop

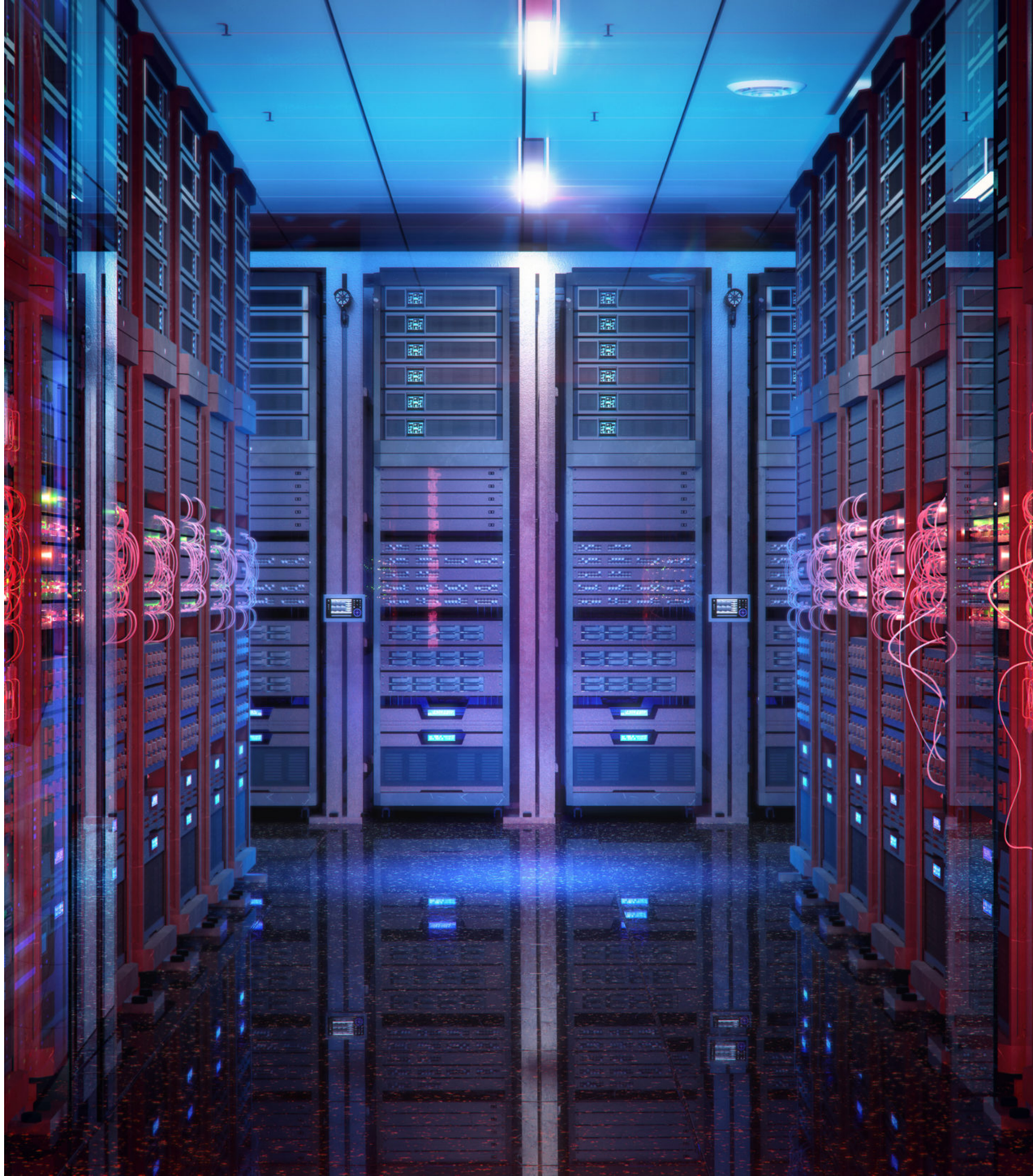
Why the workshop?

Alliance Canada offers a comprehensive wiki, but for newcomers, the information can feel overwhelming. I would like to present the (minimal) knowledge in a user-friendly way and provide some hands-on experience. You will be able to start with your own tasks.



Introduction and General Workflow

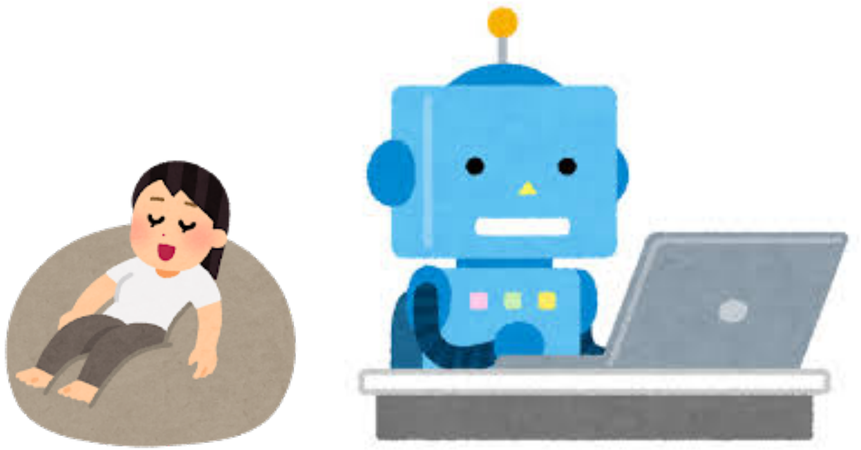
Module 1



Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

Why HPC?

- You have some code that takes forever on your own machine...
  - Problems?
- **Calm down and turn to HPC**



What is HPC?

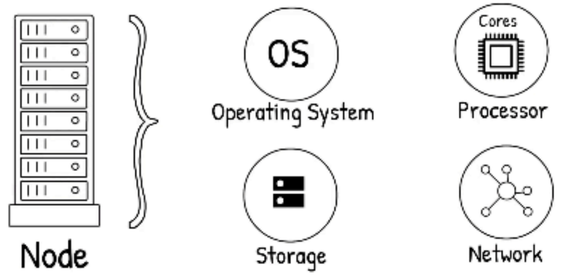
Most consumer laptops and desktops (e.g., personal computers) have one physical CPU with multiple cores.

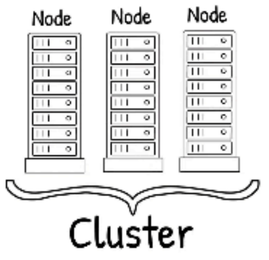
High-Performance Computing (HPC) clusters often use multiple physical CPUs across different nodes. An aggregation of computing power to solve problems that are too complicated for standard computers. e.g. Cedar cluster has 100,400 CPU cores for computation

Clusters of Alliance Canada have cool names:

Béluga, Cedar, Graham, Narval, Niagara, Arbutus

Note: Béluga, Cedar, Graham, Niagara will be replaced by Rorqual, Fir, Nibi, Trillium (also cool names!)

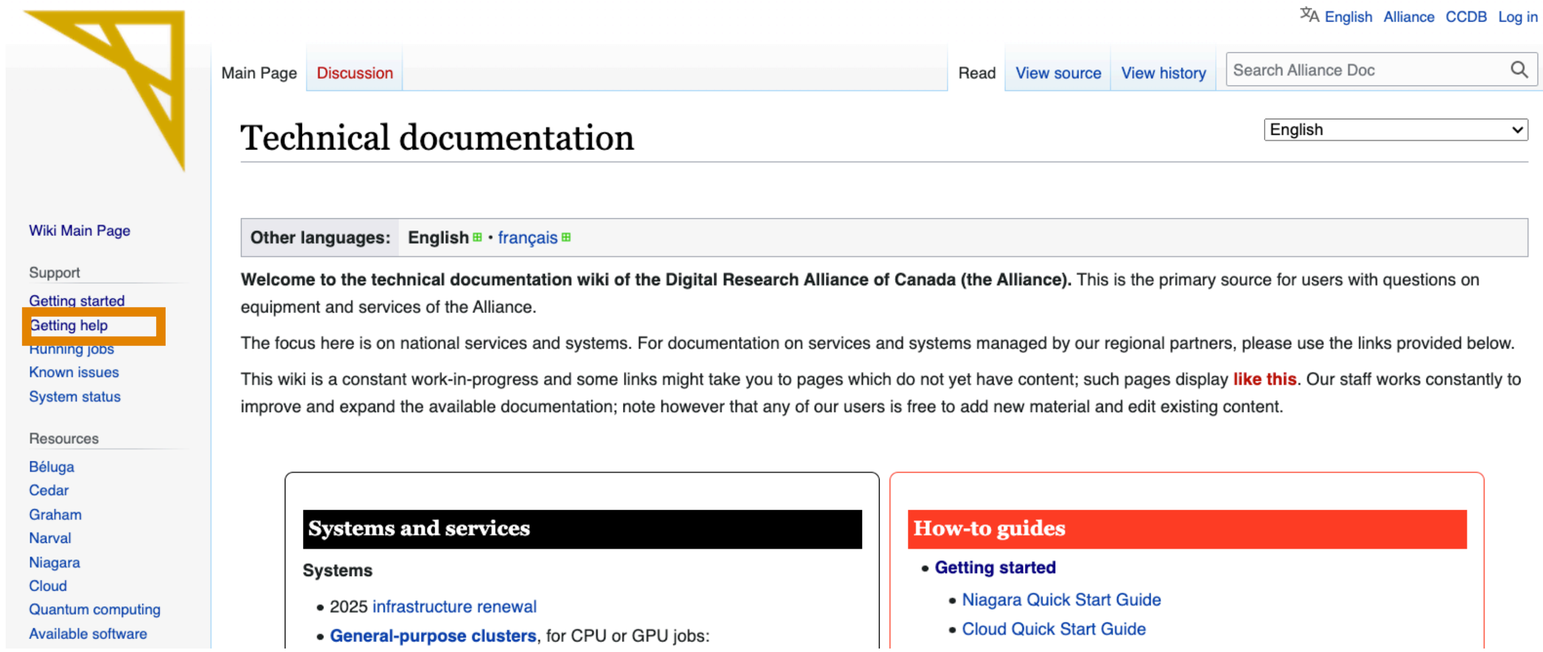




https://www.youtube.com/watch?v=nIBu1EFYmBU

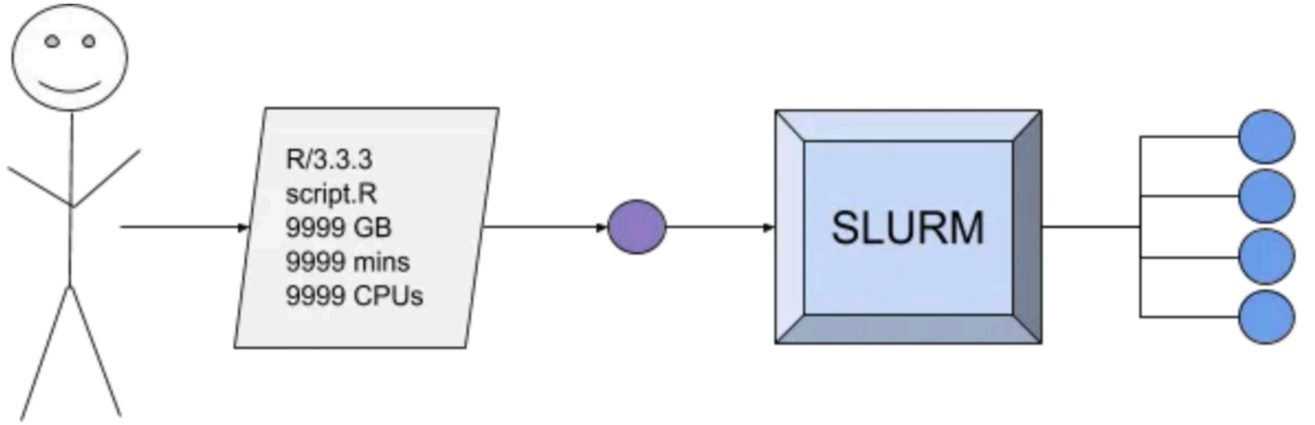
Alliance Canada wiki & Tech Support

- Alliance Canada wiki
  - Step-by-step instructions
  - Workshops that go deeper than this one
- Tech Support ("Getting help")
  - Read their instruction before sending email
  - Respond pretty fast if email is proper



The general workflow

- Apply for an Alliance account
  - If you don't have AllianceCan account we can provide a virtual cluster
- Upload your script to one of these general-purpose clusters: Béluga, Cedar, Graham, Narval
  - Make sure your code runs! Debugging is not straightforward here
  - (Find the bottleneck and improve the efficiency of your code)
  - Make sure all input/output streams use relative paths
- Create connection with login node, the "receptionist"
- Before the first time, prepare the materials to run your code
  - For example `>install.packages( ‘‘Rcpp’’ )`
- Submit job
  - You specify the time, number of CPUs, memory, the command you want to run
  - The "scheduler" will decide when to do it and allocate the resource
  - Nodes "workers" will receive and complete your "order"
- Wait...
- Don't forget to monitor and check if the job is done
- Download the output

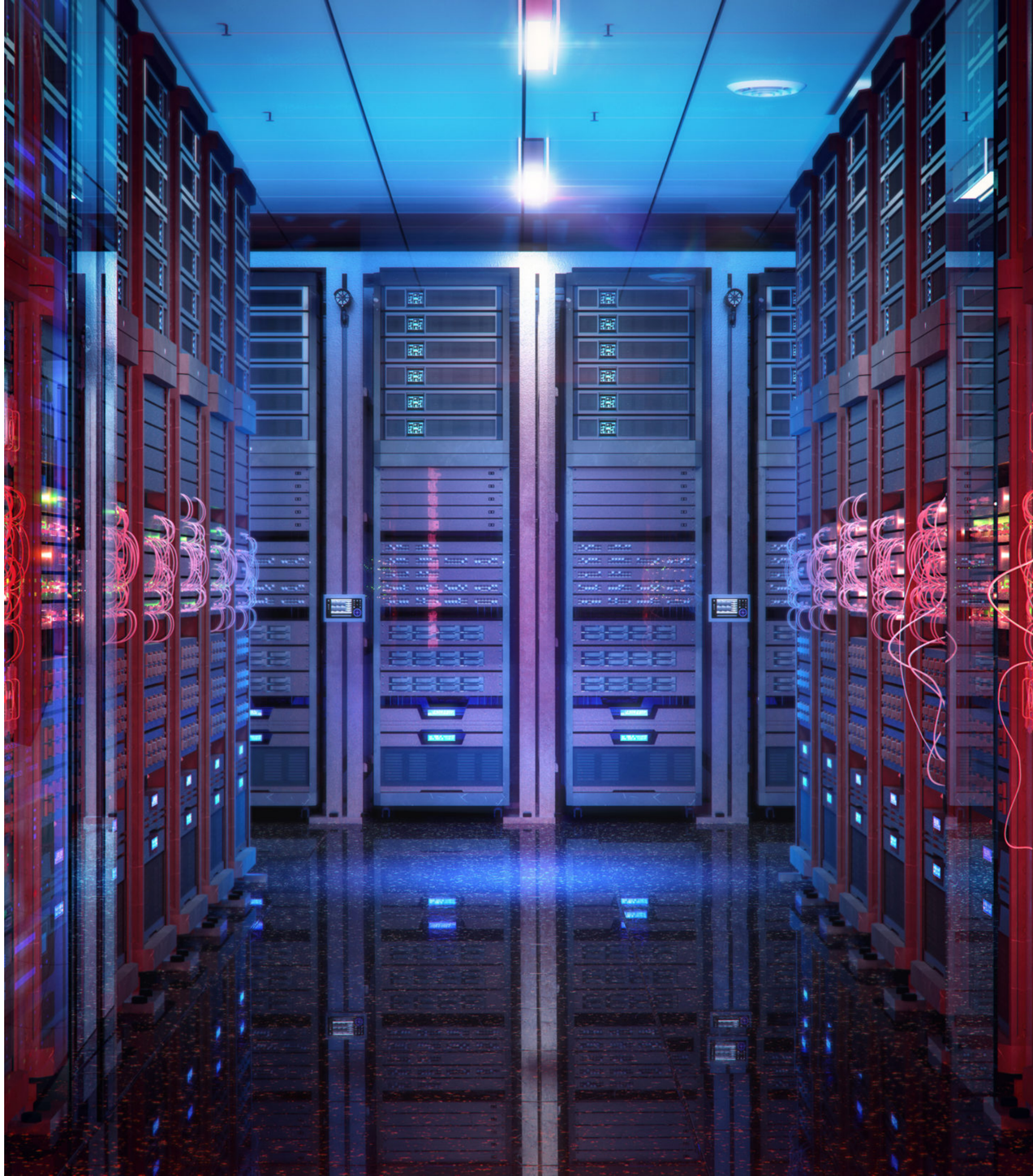


user	job	login node	scheduler	nodes
That's you!	A small file you write that tells the computer what program and script to run and describes the computational power needed for the job. - Memory - Duration - Number of cores - Serial or parallel	A computer that lets you connect to the rest of the Compute Canada computers. You can look at your files on this node, and use it to submit jobs to the scheduler.	An algorithm (called SLURM) that decides when your script will run, based on how long it will take/how many resources it needs.	The scheduler sends your job to one (or many) nodes. These nodes are the computers that will be running your script. Each node has CPUs, RAM and storage.

https://medium.com/the-nature-of-food/how-to-run-your-r-script-with-compute-canada-c325c0ab2973

Transfer Files from/ to Clusters

Module 2



Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

How to transfer file from/to clusters

The easiest way: Globus

- First and last step of the workflow;
- Before running the code you need HPC to have your script
  - say, `VeryVerySlow.R`
- After running you need to download the results
- We show the easiest way to do it: Globus!
  - Other options on wiki

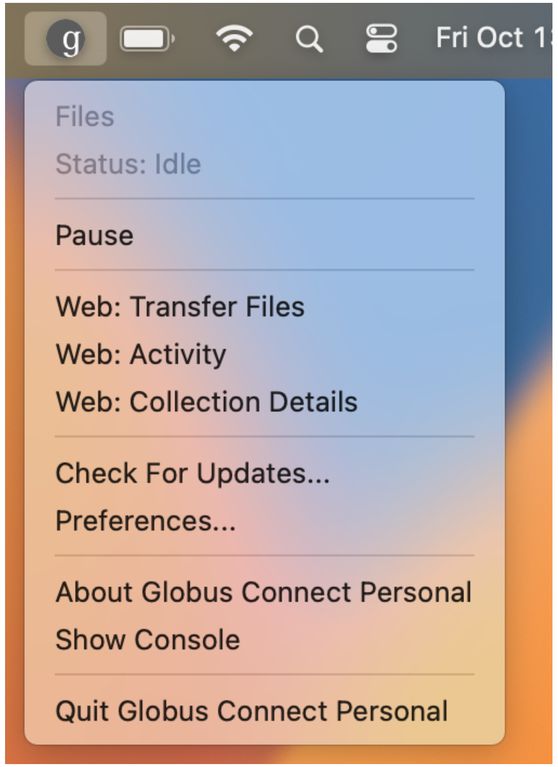
How to transfer file from/to clusters

https://docs.alliancecan.ca/wiki/Globus

- Setup:
  - Download Globus software
  - Before the first time, you need to set up your computer as endpoint
- Before transferring, make sure Globus is running on your machine
- Go to file manager portal: globus.alliancecan.ca/file-manager
  - Left: path on your own machine
  - Right: search for the cluster name; should start with “computecanada#”

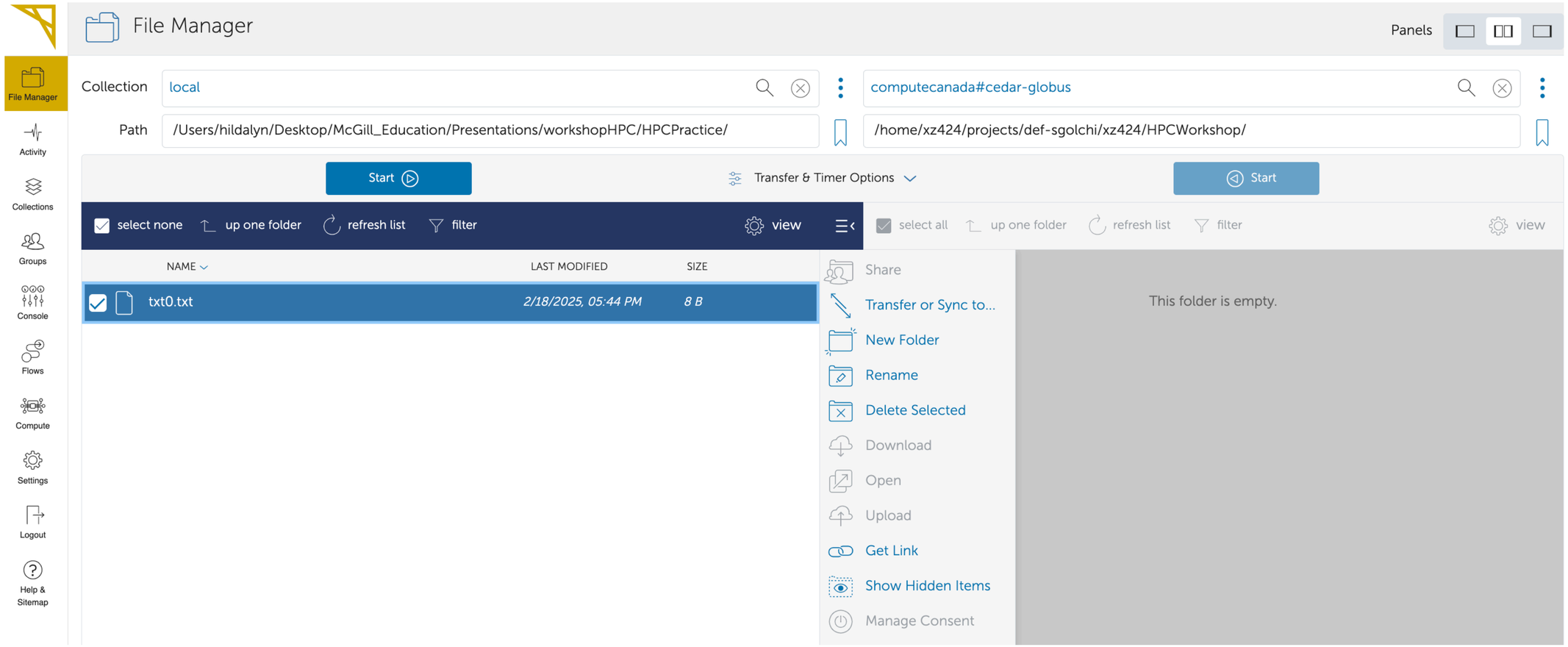
`computecanada#beluga-dtn`, `computecanada#cedar-globus`, `computecanada#graham-globus`...

Note: the Globus endpoints for the new clusters will be different, and won't use "computecanada". Refer to the page of each cluster on the wiki to get the correct Globus endpoints.



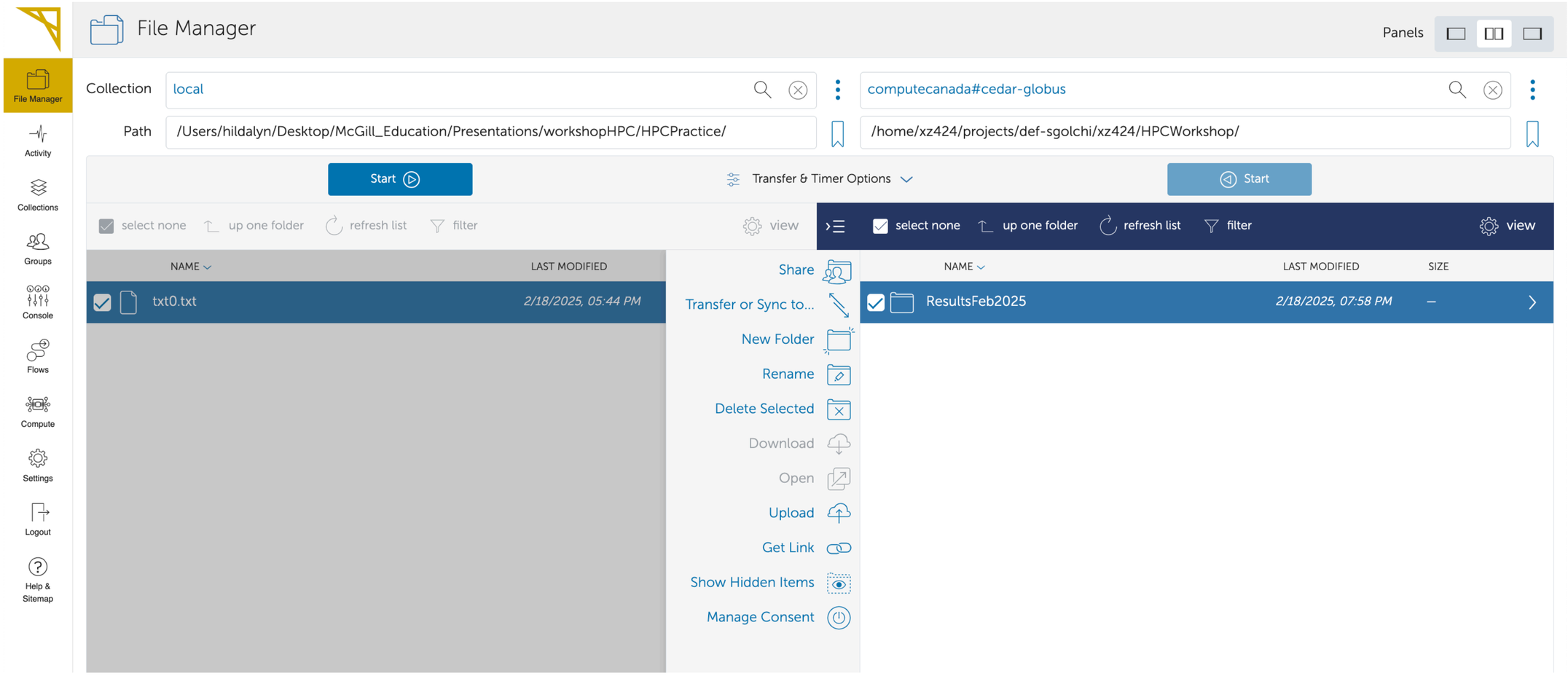
How to transfer file from/to clusters

Globus



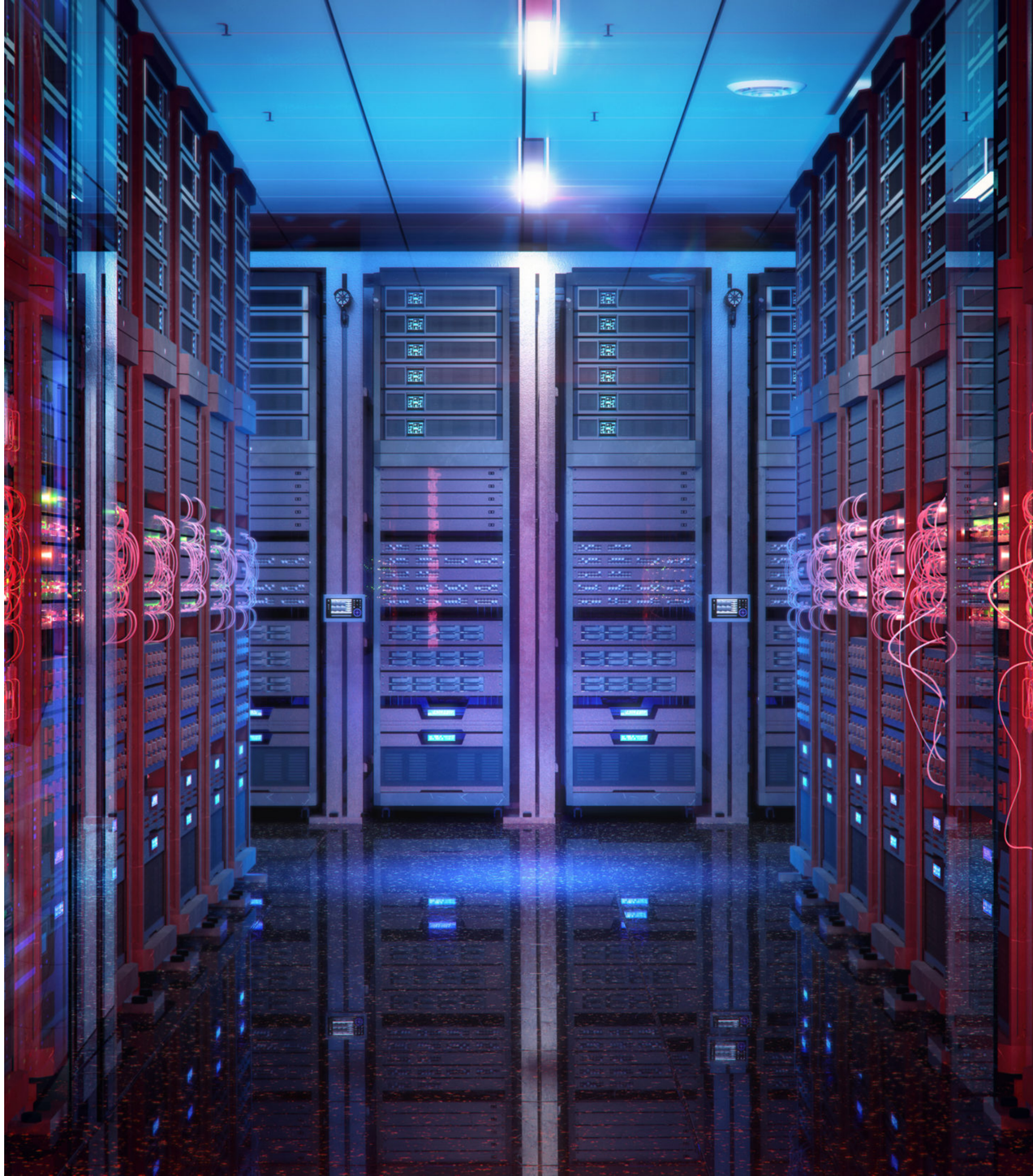
How to transfer file from/to clusters

Globus



Introduction to Shell

Module 3

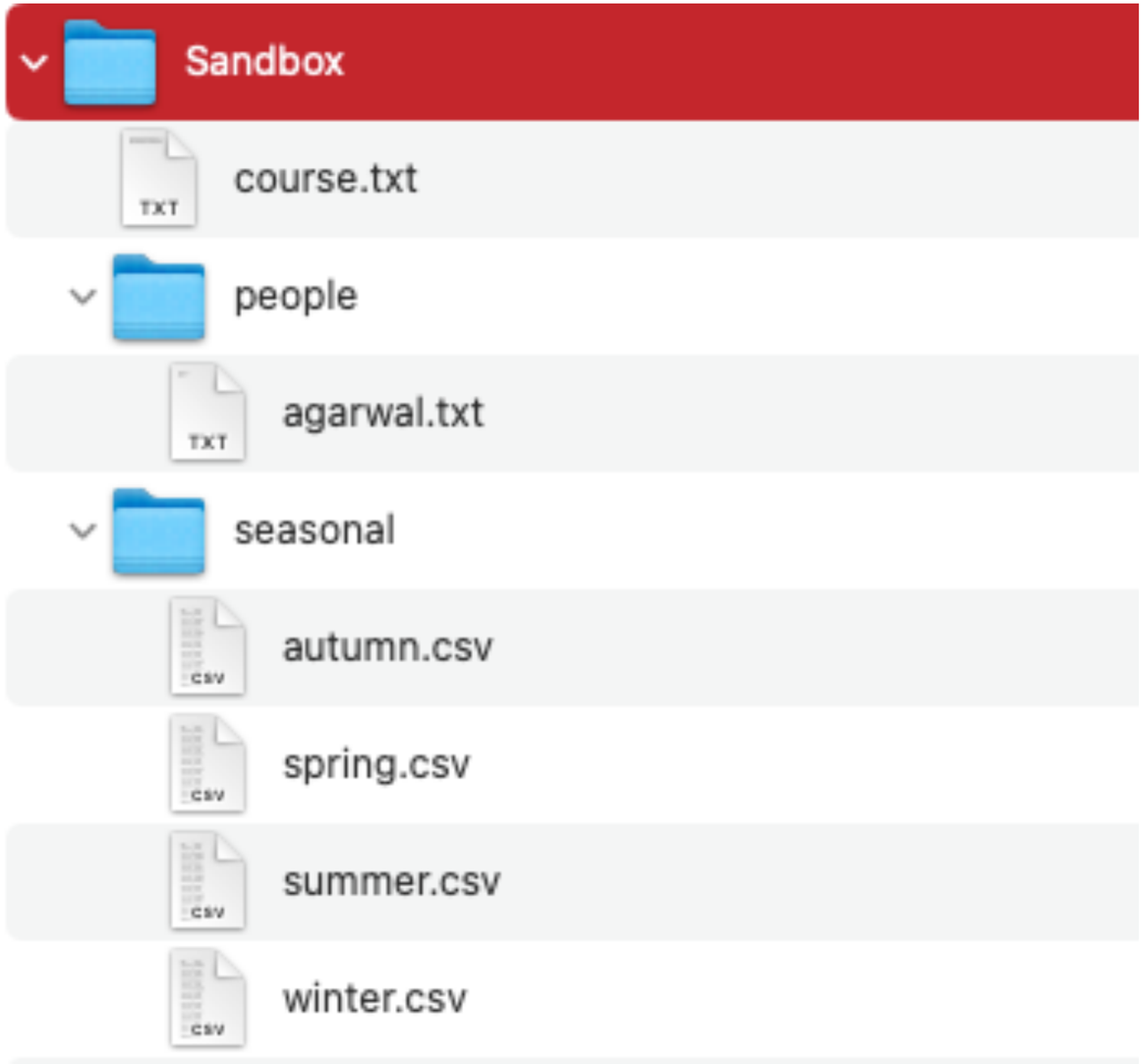


Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

Introduction to shell

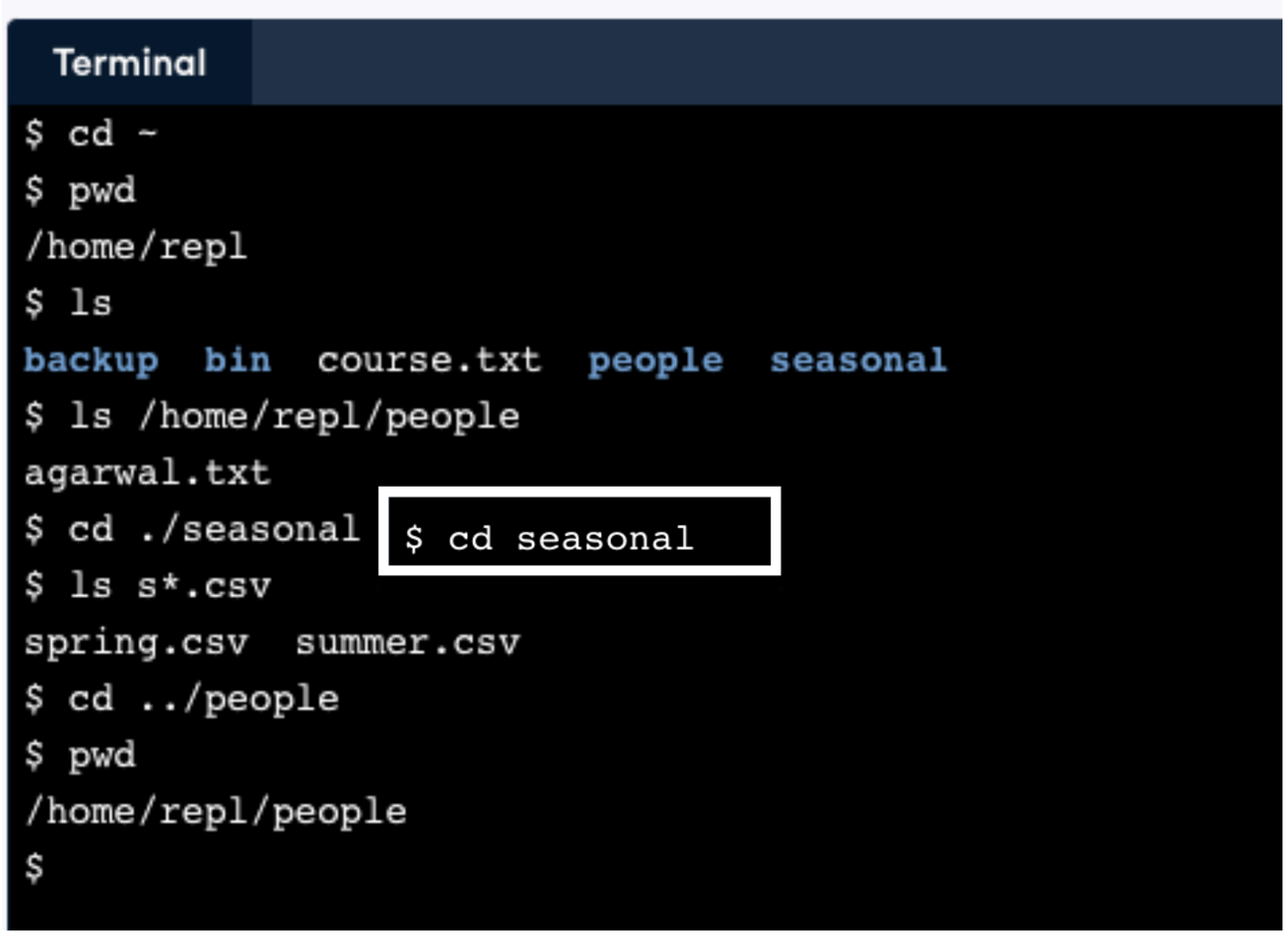
- How do we interact with operating system?
  - Your computer uses Graphical User Interface, translating clicking into commands
  - HPCs are based on Linux: no GUI, we have to use command line shell
  - You enter a command, the shell runs some other programs, prints the output, and lets you know when it is ready to take the next command
- Even if you don't use HPC, it is also a powerful tool on your own machine!
  - Mac OS: "Terminal" Windows: "CMD" ?
- There will be lots of commands on the following slides – no need to remember all of them
- Note: be cautious to run suspicious commands! In this workshop we work in the "Sandbox"

Sandbox in GUI



Use keyboard & Travel along the paths

- Some shortcuts:
  - arrow keys [up/down; across sessions]
  - `ctrl+A` [go to the start of line]
  - `ctrl+E` [go to the end of line]
  - `ctrl+C` [stop the current process]
- Useful tools to save some typing
  - wildcard: `*` [Match any character]
  - `tab` [automatically complete path]
  - `tab*2` [show all options for ambiguous path]
- Some commands:
  - `pwd` [print working directory]
  - `ls` [listing items]
  - `cd` [change to directory]
- Paths
  - Absolute paths begin with `/`
  - Relative paths start from current directory
  - `.` [Current directory]
  - `..` [Parent directory]
  - `~` [Home directory]



Practice

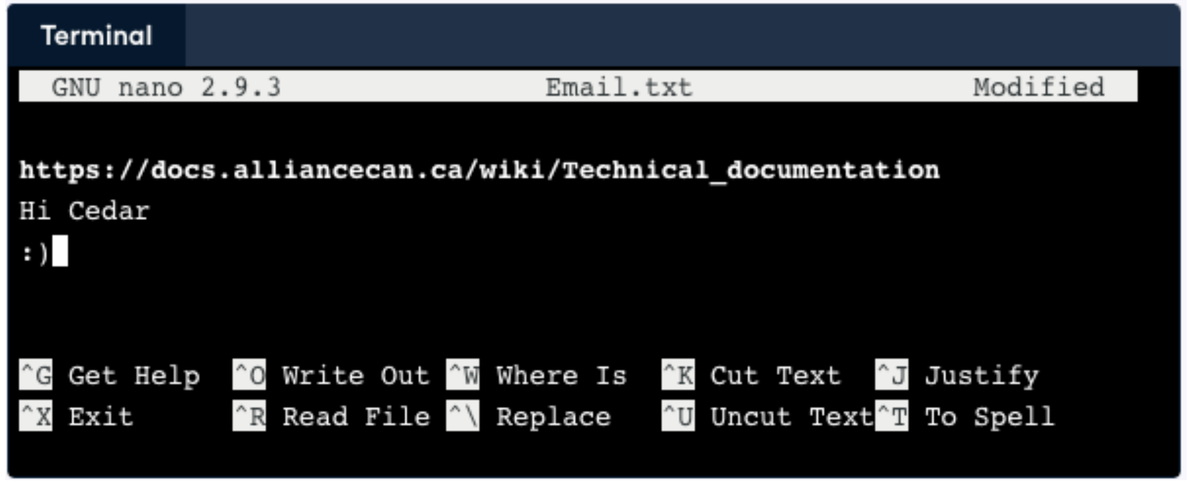
1. (If locally) Download the sandbox folder and record its location; Open your command lines software
2. Change working directory to the sandbox folder (try the shortcuts to navigate in line). If your folder name contains spaces you should use quotes for the path.
3. List all items in sandbox
4. Change directory to “people” and print working directory.
5. Change directory to the folder in its parent folder beginning with “se” (use tab completion)
6. List all csv files that end with “r.csv”
7. Change working directory back to Sandbox without using the absolute path

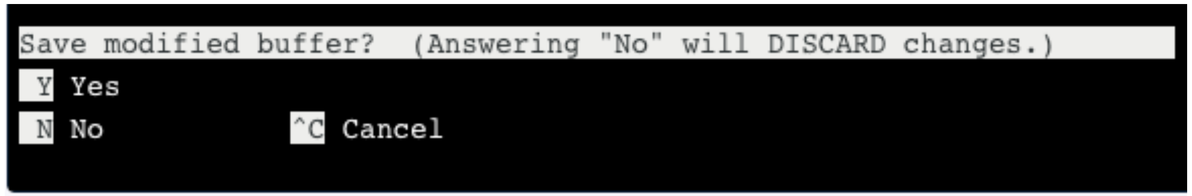
2. Change working directory to the sandbox folder
3. List all items in sandbox
4. Change directory to “people” and print working directory.
5. Change directory to the folder in its parent folder beginning with “se” (use tab completion)
6. List all csv files that end with “r.csv”
7. Change working directory back to Sandbox without using the absolute path

```
Hilda:~ hildalyn$ cd /Users/hildalyn/Desktop/McGill/Presentations/workshopHPC/Sa
ndbox
Hilda:Sandbox hildalyn$ ls
course.txt      people          seasonal
Hilda:Sandbox hildalyn$ cd people
Hilda:people hildalyn$ pwd
/Users/hildalyn/Desktop/McGill/Presentations/workshopHPC/Sandbox/people
Hilda:people hildalyn$ cd ../seasonal/
Hilda:seasonal hildalyn$ ls *r.csv
summer.csv      winter.csv
Hilda:seasonal hildalyn$ cd ..
Hilda:Sandbox hildalyn$
```

Modify the files

- Modify files and directories
  - `cp` [copy]
    - `cp course.txt course_new.txt`
    - `cp spring.csv autumn.csv people`
  - `mv` [move or rename]
    - `mv spring.txt autumn.txt ..`
    - `mv course.txt course-A.txt`
  - `rm` [remove] ⚠️
    - `rm course_new.txt`
  - `rmdir` [remove directory; only delete empty]
  - `mkdir` [?]
    - `mkdir year`
- Create a file
  - `touch Email.txt` [create a file without editing]
  - `nano Email.txt` [text file edit; create if not exist]





Practice

1. Type “`clear`” to clear terminal window
2. In sandbox, create a folder called “backup” with a command
3. Copy the text file in “people” to “backup”.
4. Rename the new copy in “backup” as “people_backup.txt”
5. Use the text editor to create “wiki.txt” in “people”. It contains the address to Alliance Canada wiki.
6. Move “wiki.txt” to backup, overwriting “backup/people_backup.txt”

1. Type “`clear`” to clear terminal window
2. In sandbox, create a folder called “backup” with a command
3. Copy the text file in “people” to “backup”.
4. Rename the new copy in “backup” as “people_backup.txt”
5. Use the text editor to create “wiki.txt” in “people”. It contains the address to Alliance Canada wiki.
6. Move “wiki.txt” to backup, overwriting “backup/people_backup.txt”

```
Hilda:Sandbox hildalyn$ ls
course.txt      people          seasonal
Hilda:Sandbox hildalyn$ mkdir backup
Hilda:Sandbox hildalyn$ ls
backup          course.txt      people          seasonal
Hilda:Sandbox hildalyn$ cp people/*.txt backup/
Hilda:Sandbox hildalyn$ ls backup/
agarwal.txt
Hilda:Sandbox hildalyn$ mv backup/agarwal.txt backup/people_backup.txt
Hilda:Sandbox hildalyn$ ls backup/
people_backup.txt
Hilda:Sandbox hildalyn$ nano people/wiki.txt
Hilda:Sandbox hildalyn$ mv people/wiki.txt backup/people_backup.txt
Hilda:Sandbox hildalyn$ cat backup/people_backup.txt
https://docs.alliancecan.ca/wiki/Technical_documentation
Hilda:Sandbox hildalyn$
```

Print something out & More handy tricks

- Print something out
  - `cat` [look at the content of some file]
  - `echo` [print]
  - `head` [look at the beginning of some file]
  - `tail` [?]
- General structure: <Command> <-options> <arguments>
  - `head -n 3 course.txt`
- `man` [manual of commands] [:q to quit]
  - `man head`
- `>` [store output in a new file]
  - `head -n 3 course.txt > Files.txt`
- `>>` [append output to a file]
  - `ls seasonal >> Files.txt`
- `|` [pipe: pass the output as the input of the next command]

Practice

1. Print “Hello HPC” in the terminal.
2. shell has environment variables, USER is one of them. To get its value it has to follow a $. Try `echo $USER` and `echo USER`
3. Look at the head, tail and content of course.txt
4. `head -n 5 course.txt|tail -c 8` Look at the manual to figure out what does it do. Which word will it print? Run it to check.
5. Collect the last 3 lines of all files in “seasonal” and save in “seasonal/Tails.txt”
6. Challenge: look at the content of “seasonal/Tails.txt”. The repetitive headers are very annoying! Can you find out which flag of tail can remove it ?

1. Print “Hello HPC” in the terminal.
2. shell has environment variables, USER is one of them. To get its value it has to follow a $. Try `echo $USER` and `echo USER`
3. Look at the head, tail and content of course.txt
4. `head -n 5 course.txt|tail -c 8`
   Look at the manual to figure out what does it do. Which word will it print? Run it to check.

```
Hilda:~ hildalyn$ echo Hello HPC
Hello HPC
Hilda:~ hildalyn$ echo $USER
hildalyn
Hilda:~ hildalyn$ echo USER
USER
```

```
-c bytes, --bytes=bytes
       Print bytes of each of the specified files.

-n count, --lines=count
       Print count lines of each of the specified files.
```

1. Collect the last 3 lines of all files in “seasonal” and save in “seasonal/Tails.txt”
2. Challenge: look at the content of “seasonal/Tails.txt”. The repetitive headers are very annoying! Can you find out which flag of tail can remove it ?

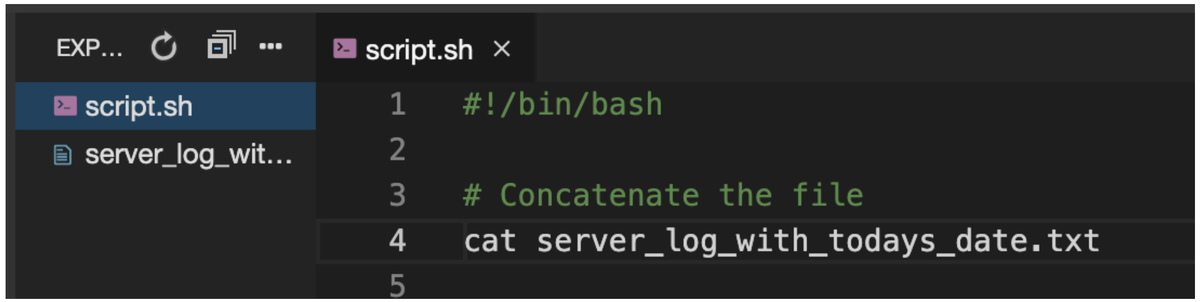
```
-q, --quiet, --silent
        Suppresses printing of headers when multiple files are being
        examined.
```

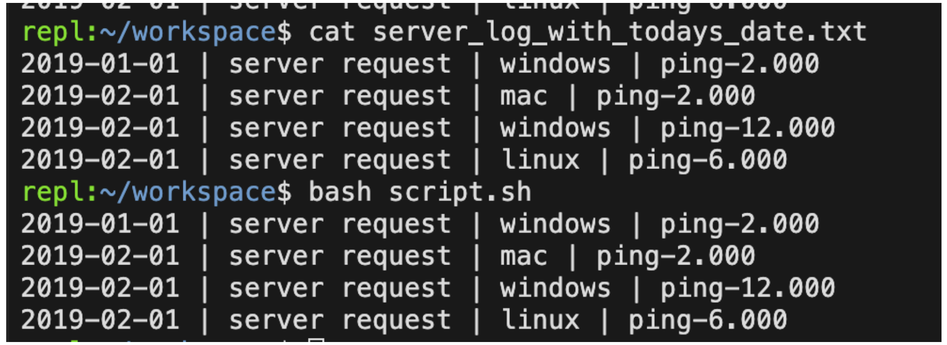
```
$ tail -n 3 seasonal/*
==> seasonal/autumn.csv <==
2017-07-21,bicuspid
2017-08-09,canine
2017-08-16,canine

==> seasonal/spring.csv <==
2017-08-13,incisor
2017-08-13,wisdom
2017-09-07,molar

==> seasonal/summer.csv <==
2017-08-02,canine
2017-08-03,bicuspid
2017-08-04,canine

==> seasonal/winter.csv <==
2017-08-11,bicuspid
2017-08-11,wisdom
2017-08-13,canine
```

```
$ tail -n 3 -q seasonal/* > seasonal/Tails.txt
$ cat seasonal/Tails.txt
2017-07-21,bicuspid
2017-08-09,canine
2017-08-16,canine
2017-08-13,incisor
2017-08-13,wisdom
2017-09-07,molar
2017-08-02,canine
2017-08-03,bicuspid
2017-08-04,canine
2017-08-11,bicuspid
2017-08-11,wisdom
2017-08-13,canine
```

Introduction to Bash scripting

- `nano script.sh` and write some commands in it.
- If you `bash` [run] the `.sh` file, then the shell just run the commands
  - "bin" the location of executable files
  - "bash" Bourne Again Shell, is an app



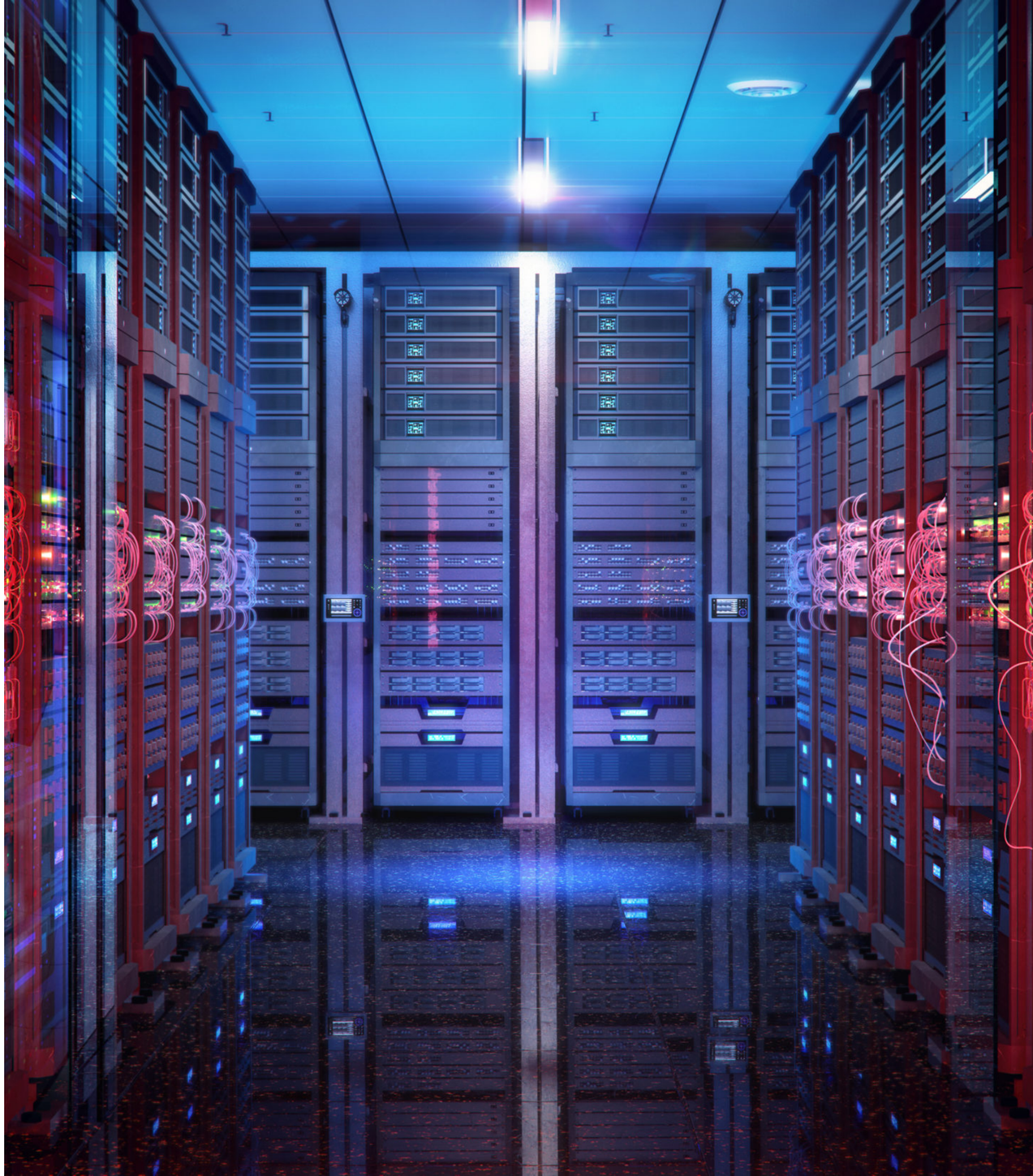


Introduction to Bash scripting

- Powerful tool! Manage files, monitor system performances, automate repetitive tasks...
  - FOR loop, IF statements, WHILE statements, CASE statements, arguments, return values
- Some examples
  - Rename many files by adding a timestamp
  - Extract specific columns from a large dataset without opening it
  - Move all files containing some string to a folder
  - Find all python files modified in the last 7 days

Submitting and Monitoring Jobs

Module 4



Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

How to connect to AllianceCan clusters

- `ssh` [secure shell, a standard to connect to remote machines securely]
  - `ssh <user>@<cluster>.alliancecan.ca`
    - `ssh <user>@micm-edia.calculquebec.cloud`
  - you will be asked to input your password and multifactor authentication

```
Hilda:Sandbox hildalyn$ ssh xz424@cedar.alliancecan.ca
(xz424@cedar.alliancecan.ca) Password:
(xz424@cedar.alliancecan.ca) Duo two-factor login for xz424

Enter a passcode or select one of the following options:

 1. Duo Push to iPhone (iOS)

Passcode or option (1-1): 637448
Success. Logging you in...
Success. Logging you in...
------------------------------------------------------------------------------
==============================================================================
Welcome to Cedar! / Bienvenue sur Cedar!

For information see: https://docs.alliancecan.ca/wiki/Cedar
Email support@tech.alliancecan.ca for assistance and/or to report problems.
```

```
[xz424@cedar1 ~]$ _
```

Submit a job

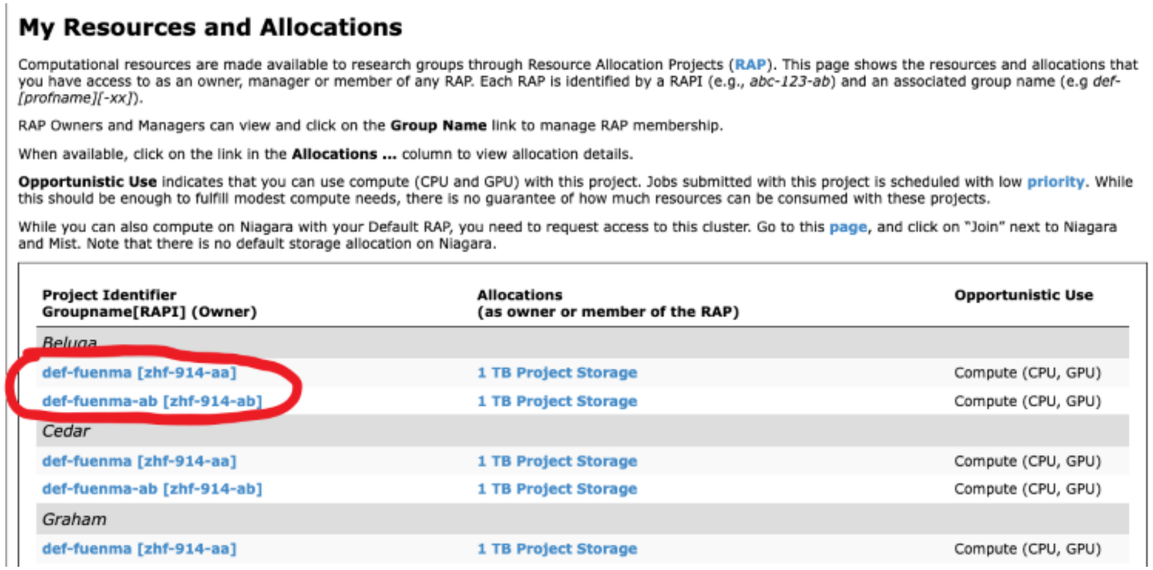
- `sbatch` [submit a job]

```
$ sbatch simple_job.sh
Submitted batch job 123456
```

  - simple_job.sh [a minimal example]

```
#!/bin/bash
#SBATCH --time=00:15:00
#SBATCH --account=def-someuser
echo 'Hello, world!'
sleep 30
```

  - On general-purpose clusters, this job reserves 1 core and 256MB of memory for 15 minutes.
  - account: log in to CCDB; My Account -> My Resources and Allocations.
  - you will receive a jobID such as 123456



My Resources and Allocations

Computational resources are made available to research groups through Resource Allocation Projects (RAP). This page shows the resources and allocations that you have access to as an owner, manager or member of any RAP. Each RAP is identified by a RAPI (e.g., *abc-123-ab*) and an associated group name (e.g *def-[profname][-xx]*).

RAP Owners and Managers can view and click on the **Group Name** link to manage RAP membership.

When available, click on the link in the **Allocations ...** column to view allocation details.

Opportunistic Use indicates that you can use compute (CPU and GPU) with this project. Jobs submitted with this project is scheduled with low priority. While this should be enough to fulfill modest compute needs, there is no guarantee of how much resources can be consumed with these projects.

While you can also compute on Niagara with your Default RAP, you need to request access to this cluster. Go to this page, and click on "Join" next to Niagara and Mist. Note that there is no default storage allocation on Niagara.

Project Identifier Groupname[RAPI] (Owner)	Allocations (as owner or member of the RAP)	Opportunistic Use
Beluga		
def-fuenma [zhf-914-aa]	1 TB Project Storage	Compute (CPU, GPU)
def-fuenma-ab [zhf-914-ab]	1 TB Project Storage	Compute (CPU, GPU)
Cedar		
def-fuenma [zhf-914-aa]	1 TB Project Storage	Compute (CPU, GPU)
def-fuenma-ab [zhf-914-ab]	1 TB Project Storage	Compute (CPU, GPU)
Graham		
def-fuenma [zhf-914-aa]	1 TB Project Storage	Compute (CPU, GPU)

```
#!/bin/bash
#SBATCH --account=def-someuser   # replace this with your own account
#SBATCH -mem-per-cpu=4G         # memory per cpu
#SBATCH -time=0-01:00            # time (DD-HH:MM)
#SBATCH -cpus-per-task=4
module load gcc/9.3.0 StdEnv/2023 r/4.3.1   # modules



Rscript Slow.R > stdout.txt 2>stderr.txt
```

Monitor current jobs

- `squeue` [list jobs pending or running]
  - `squeue -u $USER` [all my jobs the scheduler is managing, including running and pending]
  - `squeue -u $USER -t RUNNING` [?]

```
$ sq
   JOBID     USER      ACCOUNT       NAME  ST   TIME_LEFT NODES CPUS    GRES MIN_MEM NODELIST (REASON)
  123456   smithj   def-smithj   simple_j   R        0:03     1    1  (null)      4G cdr234  (None)
  123457   smithj   def-smithj   bigger_j  PD  2-00:00:00     1   16  (null)     16G (Priority)
```

Cancel a job

- `scancel 1234567` [cancel jobID 1234567]
- `scancel -u $USER` [cancel all my jobs running or pending]
- `scancel -t PENDING -u $USER` [cancel my pending jobs]

Summarize completed jobs

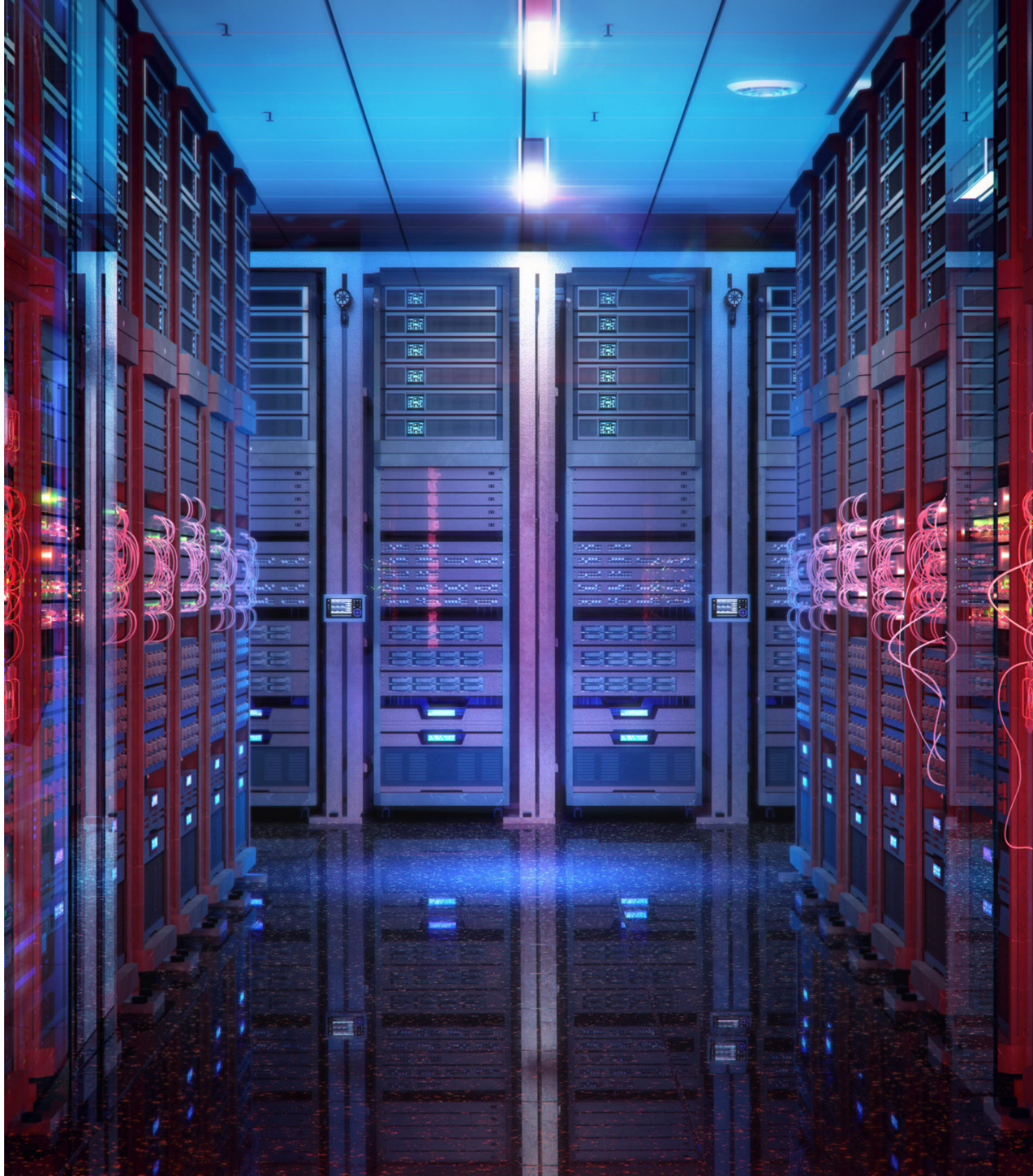
```
$ seff 12345678
Job ID: 12345678
Cluster: cedar
User/Group: jsmith/jsmith
State: COMPLETED (exit code 0)
Cores: 1
CPU Utilized: 02:48:58
CPU Efficiency: 99.72% of 02:49:26 core-walltime
Job Wall-clock time: 02:49:26
Memory Utilized: 213.85 MB
Memory Efficiency: 0.17% of 125.00 GB
```

Practice

If you have set up the account, submit simple_job.sh, monitor its status. You can cancel it, or check it when it's finished.

```
#!/bin/bash
#SBATCH --time=00:15:00
#SBATCH --account=def-someuser
echo 'Hello, world!'
sleep 30
```

Conclusion



Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

The general workflow revisit

- Apply for an account if you haven't!
  - Sponsorship from your supervisor
- Upload your script to one of these clusters: Béluga, Cedar, Graham, Narval
  - Make sure your code runs! Debugging is not straightforward here
  - (Optional: find the bottleneck and improve the efficiency of your code, such as benchmark and parallelize)
  - Make sure all input/output streams use relative paths
- Create connection with login node, the "receptionist"
- Prepare the materials to run your code
  - For example `>install.packages( ‘‘Rcpp’’ )`
- Submit job
  - You specify the time, memory, number of CPUs, which script you want to run
  - The scheduler will allocate the resource and decide when to do it
  - Nodes "workers" will receive and run the code
- Check if the job is done.
- Download the output.

Next steps

- One thing we did not cover: environment modules. A module contains the information to make an application available in the user's login session.
- Find out the modules your R version based on
  - `module spider r` [list available r versions]
  - `module spider r/4.3.1` [list the modules r 4.3.1]
  - `module load StdEnv/2023 r/4.3.1`
- Install r packages
  - `module load StdEnv/2023 r/4.3.1`
  - `R`

  ```
  > install.packages(..)
  >q()
  ```
- In the job script, include `module load gcc/9.3.0 StdEnv/2023 r/4.3.1` before run your R script

  https://docs.alliancecan.ca/wiki/Utiliser_des_modules/en

More workshops

- Calcul Québec training events and spring school
- Workshops provided by the Alliance

Reference

- https://docs.alliancecan.ca/wiki/Getting_started
- https://docs.alliancecan.ca/wiki/Technical_support
- https://docs.alliancecan.ca/wiki/Utiliser_des_modules/en
- https://docs.alliancecan.ca/wiki/Running_jobs
- Datacamp

Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

iWOMEN in ARC

McGill initiative in Computational Medicine

Thank you!