Using Canadian Supercomputers to Run Time-Consuming R Jobs

Victoria Celio and Rob Cribbie
A Little History

- **SHARCNET** is a community that consists of 19 academic institutions in Ontario
  - Including York University
  - Began in the early 2000s
- **Compute Canada** also began in the early 2000s
  - Canada's national high-performance computing (HPC) system
  - SHARCNET was part of Compute Canada
- **Digital Research Alliance of Canada (The Alliance)**
  - As of March, 2022, SHARCNET and Compute Canada have now been superseded by the Digital Research Alliance of Canada
# Supercomputer Systems Across Canada

<table>
<thead>
<tr>
<th>Supercomputer System</th>
<th>University</th>
<th>Additional Information</th>
</tr>
</thead>
</table>
| Béluga/Narval        | McGill University           | - 100,000 CPU cores; 1300 GPUs  
- Comparable to 30,000 high-end laptops bundled into the same computer room |
| Arbutus              | University of Victoria      | - 1.6 PB of storage; 7,640 CPU cores across 290 nodes  
- Arbutus can store the equivalent of 10 million eight-drawer filing cabinets worth of text and process calculations  
- 1000x faster than a desktop computer                                        |
| Cedar                | Simon Fraser University     | - 3.6 petaFLOPS of computing power; up to 3 terabytes of memory; can run jobs of up to 1280 CPU cores each  
- When launched, Cedar was Canada’s most powerful academic supercomputer and one of the world’s top 100 supercomputers |
| Graham               | University of Waterloo      | - 41,548 cores and 520 GPU devices, spread across 1,185 nodes of different types; can run jobs up to 1024 cores each |
| Niagara              | University of Toronto       | - 76th fastest supercomputer in the world  
- Cluster of 2,016 Lenovo SD530 servers, each with 40 Intel Skylake at 2.4 GHz cores or 40 Intel CascadeLake cores at 2.5 GHz |
How Can Supercomputers Benefit Us?

- Have you ever run a large simulation study that took hours, days or even weeks to run?
  - 1000s of iterations, models with bootstrapping and numerous conditions
    - E.g., Nataly Beribisky’s dissertation simulation study took one week *per model*!

- It is likely that you can significantly reduce the length of time it takes to run your study by using Digital Research Alliance of Canada’s supercomputers (clusters)
  - That’s their purpose!
Example of the Benefit of Supercomputers

- Monte Carlo simulation study
  - $N_{\text{sims}} = 5000$
- Simple mediation model
- Confidence interval for the indirect effect obtained via bootstrapping ($N_{\text{boot}} = 5000$)
- Cedar completion time: 4.8 hours
- Laptop completion time: $\sim 100$ hours
Steps for using *The Alliance*

1. Preliminary steps
2. Logging-in
3. Running R in login node
4. Creating job/R Scripts
5. Transferring files
6. Submitting jobs
7. Viewing output
Preliminary Steps: Terminal Commands

• Familiarize yourself with basic terminal commands
• We will be working with the ‘Terminal’ in RStudio, so knowing basic commands will be essential
## Preliminary Steps: Terminal Commands

- **Basic Terminal commands:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cd</code></td>
<td>Change directory (e.g., ‘cd folder1’)</td>
</tr>
<tr>
<td><code>cd ..</code></td>
<td>Go back one directory</td>
</tr>
<tr>
<td><code>pwd</code></td>
<td>Display the current directory</td>
</tr>
<tr>
<td><code>ls</code></td>
<td>Lists all the files in the specified directory</td>
</tr>
<tr>
<td><code>module load NameOfModule</code></td>
<td>Loading module onto a cluster</td>
</tr>
<tr>
<td><code>sbatch FileName.sh</code></td>
<td>Running the job script</td>
</tr>
<tr>
<td><code>squeue -u YourUserName</code></td>
<td>Checking the status of your jobs</td>
</tr>
<tr>
<td><code>Exit</code></td>
<td>Exit the terminal</td>
</tr>
</tbody>
</table>
Preliminary Steps: Alliance Account

• Create a Digital Research Alliance of Canada account
  • Go to: https://ccdb.alliancecan.ca/
  • Click on ‘register’

• If you are a graduate student, you will need a ‘Sponsor’
  • If your supervisor has a Digital Research Alliance of Canada account, they can be your ‘Sponsor’
Preliminary Steps

Faculty Account

Student Account
Preliminary Steps: RStudio Terminal

Open RStudio

- Go to Terminal
Open RStudio

• Go to Terminal

• Note: make sure you are in the home directory
Steps for using *The Alliance*

1. Preliminary steps
2. Logging-in
3. Running R in login node
4. Creating job/R Scripts
5. Transferring files
6. Submitting jobs
7. Viewing output
Logging in to a Cluster

- **Login Node**: nodes that you are automatically directed to when you login to a cluster, where you can prepare scripts and do computationally small tasks (e.g., run a small R job)
  - Note: we are going to prepare scripts in RStudio, but you can prepare scripts directly within a cluster
- **Compute Nodes**: nodes where computationally heavy tasks are completed
  - This is the focus of this presentation
# Supercomputer Systems Across Canada

<table>
<thead>
<tr>
<th>Supercomputer System</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Béluga</td>
<td>beluga.alliancecan.ca</td>
</tr>
<tr>
<td>Narval</td>
<td>narval.alliancecan.ca</td>
</tr>
<tr>
<td>Cedar</td>
<td>cedar.alliancecan.ca</td>
</tr>
<tr>
<td>Graham</td>
<td>graham.alliancecan.ca</td>
</tr>
<tr>
<td>Niagara</td>
<td>niagara.alliancecan.ca</td>
</tr>
</tbody>
</table>

We will be using Cedar for this presentation, no specific reason why we are working with Cedar over any of the other clusters.
Logging in to a Cluster

Different methods for logging in from the Terminal in RStudio

- Username login
  - ssh USERNAME@ADDRESS
  
  ```
  Patricias-MacBook-Air:~ patriciacelio$ ssh vcelio@cedar.alliancecan.ca
  ```

- SSH key login:
  - ssh -i path/to/PrivateKey USERNAME@ADDRESS
  - *Note: refer to supplementary materials regarding how to create a SSH key*

  ```
  Patricias-MacBook-Air:~ patriciacelio$ ssh -i /Users/patriciacelio/.ssh/quant-key vcelio@cedar.alliancecan.ca
  ```

*Note: you also have the option to set-up duo authentication*
Logging in to a Cluster

• By default, you are provided with three folders:

  • **scratch**
    • Useful for temporary space for large files – not backed up
  • **projects**
    • Large adjustable quota per project
    • Daily backup
    • Folder that you should use for your regular work
  • **nearline**
    • Manage infrequently-used files; does not count against storage quota
Steps for using *The Alliance*

1. **Preliminary steps**
2. **Logging-in**
3. **Running R in login node**
4. **Creating job/R Scripts**
5. **Transferring files**
6. **Submitting jobs**
7. **Viewing output**
Running R Commands in the Login Node

- If you want to run basic R operations within a cluster login node, you can do so by opening R within the terminal in RStudio (after logging in)
- You will be asked which version of R you would like to run

```
[cribble@cedar5 ~]$ R
[miij] Please select a module to run R:

  MODULE  PARENT(S)
  1 r/4.3.1  StdEnv/2020
  2 r/4.2.2  StdEnv/2020
  3 r/4.2.1  StdEnv/2020
  4 r/4.1.2  StdEnv/2020
  5 r/4.1.0  StdEnv/2020
  6 r/4.0.5  StdEnv/2020
  7 r/4.0.2  StdEnv/2020
  8 r/4.0.0  StdEnv/2020
  9 sagemath/9.3  StdEnv/2020
 10 openfoam/5.0  nixpkgs/16.09 intel/2018.3 openmpi/3.1.4
 11 openfoam/4.1  nixpkgs/16.09 intel/2018.3 openmpi/3.1.4
 12 openfoam/3.0.1  nixpkgs/16.09 intel/2018.3 openmpi/3.1.4

Make a selection (1-68, q aborts) [1]: 1
[miij] loading StdEnv/2020 r/4.3.1 ...
```

R version 4.3.1 (2023-06-16) -- "Beagle Scouts"
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'license()' or 'licence()' for distribution details.

       Natural language support but running in an English locale

R is a collaborative project with many contributors. Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications.

Type 'demo()', for some demos, 'help()' for on-line help, or 'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> []
Running R Commands in the Login Node

• E.g., installing R packages
  • Note: I have not had any issue including “install.packages()” commands within my scripts, but I have read that some clusters are not directly connected to the internet so in that case you may need to use this option

```r
> install.packages("lavaan", repos="https://cloud.r-project.org/")
Installing package into '/home/cribbie/R/x86_64-pc-linux-gnu-library/4.3'
(as ‘lib’ is unspecified)
trying URL 'https://cloud.r-project.org/src/contrib/lavaan_0.6-17.tar.gz'
Content type 'application/x-gzip' length 921430 bytes (899 KB)
downloaded 899 KB
* installing *source* package ‘lavaan’...
 ** package ‘lavaan’ successfully unpacked and MD5 sums checked
 ** using staged installation
 ** R
 ** data
 ** moving datasets to lazyload DB
 ** inst
 ** byte-compile and prepare package for lazy loading
 ** help
 ** installing help indices
 ** building package indices
 ** testing if installed package can be loaded from temporary location
 ** testing if installed package can be loaded from final location
 ** testing if installed package keeps a record of temporary installation path
* DONE (lavaan)
The downloaded source packages are in
 '/tmp/Rtmpoa8jiks/downloaded_packages'
```
Running R Commands in the Login Node

• But you can also run basic R commands as well
• You should not run any R jobs in the login node that take significant computing time (e.g., > 2 minutes)

```r
> 2+3
[1] 5
> a <- c(2,6,4)
> var(a)
[1] 4
```
Steps for using *The Alliance*

1. Preliminary steps
2. Logging-in
3. Running R in login node
4. Creating job/R Scripts
5. Transferring files
6. Submitting jobs
7. Viewing output
Creating a Job Script

- What are Job Scripts?
  - Provides the scheduler (Slurm) with important information regarding the resources required to complete your job
  - This information is used by the scheduler to assign your job
  - The more resources required, the further back in the queue your job will be placed
Creating a Job Script

• There are two options for creating a job script

  • Within the Terminal
    • nano FileName.sh
    • Save: ctrl + x + y
      • Note: create file in the same storage as script

  • In any text editor (e.g., RStudio)
    • Save the script with .sh extension
    • Transfer it to the cluster folder with your R script
      • Note: when you create a job script in RStudio, it will be in DOS/MAC format; to convert it to unix format (which is required by the clusters) we can use dos2unix (e.g., dos2unix jobscript.sh)
Creating a Job Script

Must include at the start of every job script

```bash
#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03-00:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R
```
Creating a Job Script

Number of cores (default 1 core; for simulation studies use maximum cores available)

```bash
#!/bin/bash
#$SBATCH --cpus-per-task=32
#$SBATCH --mem-per-cpu=4000
#$SBATCH --time=03:00:00:00 # DD-HH:MM
#$SBATCH --account=def-cribbie
#$SBATCH --mail-user=cribbie@yorku.ca
#$SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R
```
Creating a Job Script

Approximate memory required (default 256MB)

```bash
#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03-00:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R
```
Creating a Job Script

Maximum time required to run job (don’t underestimate!)

```bash
#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03:00:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R
```
Creating a Job Script

#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03-00:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load R/4.3.1
Rscript play_7.R
Creating a Job Script

```bash
#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R
```

Include to get an email notification of the start and end of the job.
Creating a Job Script

#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03-00:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R

Modules required
Creating a Job Script

```bash
#!/bin/bash
#SBATCH --cpus-per-task=32
#SBATCH --mem-per-cpu=4000
#SBATCH --time=03-00:00:00 # DD-HH:MM
#SBATCH --account=def-cribbie
#SBATCH --mail-user=cribbie@yorku.ca
#SBATCH --mail-type=ALL
module load r/4.3.1
Rscript play_7.R
```

Name of R script to run
Creating an R Script

• Create your R script as usual
  • To maximize efficiency, you will want to use scripts that make use of parallel processing (e.g., `foreach` loop instead `for` loop)
  • Optionally, you can include a location to save objects from the output
    • You can save scalars, vectors or matrices, etc., and also save multiple objects

• The following R script will save the object ‘z’ to a file called `play22.Rdata`, as well as printing the output/object in the `.out` file

```r
library(psych)
y <- rnorm(100)
x <- rep(c(0,1),each=50)
x <- factor(x)

z <- t.test(y ~ x)$p.value
z  # object will appear in the .out file
save(z, file = "play22.Rdata")  # object will be saved in directory
```
Steps for using *The Alliance*

1. Preliminary steps
2. Logging-in
3. Running R in login node
4. Creating job/R Scripts
5. Transferring files
6. Submitting jobs
7. Viewing output
Transferring Files

Goal: transfer files to and from your personal computer to one of the clusters

- **Globus**: a service that allows researchers to securely transfer files between two “resources”
  - I.e., between your computer and The Alliance clusters

See https://docs.alliancecan.ca/wiki/Globus
Transferring Files

• Step 1 – Set-up a Globus account (using Compute Canada login)

• Step 2 – Download Globus Connect Personal on your computer
  • Globus Connect Personal allows you to access folders on your computer, and make transfers to The Alliance clusters
    • Or, technically, between clusters
    • Note: you will not download the program, only a file that will run in the background
  • You will be asked to set a name for your personal ‘collection’
    • E.g., Robs Collection
Transferring Files

• Step 2 – Download **Globus Connect Personal** on your computer
  • Note for Windows users: by default, the only folder that you can transfer to and from with Globus is `c:\\user\documents`
  • However, you can add any folder you like, including Google Drive, Onedrive, etc.
Transferring Files

• Step 3 – Select your collections
  • I.e., the relevant folder on your computer and the relevant folder on the cluster
    • See https://docs.alliancecan.ca/wiki/Globus

• Step 4 – Initiate transfer
Transferring Files

The image shows a file management interface with two paths highlighted:

- `/My Drive/Research/Cribbie_Celio/SharncNet/SharncNet/`
- `/home/cribbie/projects/def-cribbie/cribbie/Play/`

The paths are being compared, likely for the purpose of transferring files between the two locations.
Transferring Files

• Quick note for Windows users
  • There is a program called MobaXterm that is very user friendly for transferring files back and forth between your computer and The Alliance clusters
  • It also has a terminal so you can complete all transfers and commands in one program

Just drag and drop files from the left (your computer) to the right (cluster)
Steps for using *The Alliance*

1. Preliminary steps
2. Logging-in
3. Running R in login node
4. Creating job/R Scripts
5. Transferring files
6. Submitting jobs
7. Viewing output
Submitting a Job

• Make sure all of your files are in the same directory (or provide proper directory information)
  • E.g., job script + R script

• You will most likely have to convert your job script file from dos to unix
  • dos2unix JobScriptName.sh

• Command to submit a job:
  • sbatch JobScriptName.sh

[vcelio@cedar1 vcelio]$ sbatch qmforum5.sh
Submitted batch job 26757945
Steps for using *The Alliance*

1. Preliminary steps
2. Logging-in
3. Running R in login node
4. Creating job/R Scripts
5. Transferring files
6. Submitting jobs
7. Viewing output
Viewing Output

- Check the status of your job
  - `squeue -u YourUserName`

```
[vcelio@cedar1 vcelio]$ squeue -u vcelio

+-----------------+----------+----------+----------+----------------+---------------+----------+----------+-------------+-------------+----------+----------+----------+-----------------+------------------------+---------------------+------------------------+-----------------------------+-----------------------+---------------+---------------+-----------------+----------------------------+
| JOBID | USER       | ACCOUNT  | NAME     | ST  | TIME_LEFT | NODES | CPUS | TRES_PER_N | MIN_MEM | NODELIST | RESERVATION | REASON        |
|-------|------------|----------|----------|-----|-----------|-------|------|------------|---------|----------|-------------|---------------|------------------------+------------------------+------------------------+-----------------------------+-----------------------+---------------+---------------+-----------------+----------------------------+
| 26757945 | vcelio    | def-cribble_ | QM_Forum_Test | PD | 2:00 | 1     | 1    | N/A        | 256M    | (None)         |             | (None)         |             | (None)     |
```

- Transfer the output to your computer
  - Using one of the methods mentioned in previous slides
  - General format: `slurm-ProjectID.out`

```
[vcelio@cedar1 vcelio]$ ls
```
```r
x <- sample(100)
output <- numeric()

for(i in 1:length(x)){
  print(c("original", x[i]))
  output[i] <- (x[i] + 1)/2
  print(c("new", output[i]))
}
```

```
[1] "original" "66"
[1] "new" "33.5"
[1] "original" "74"
[1] "new" "37.5"
[1] "original" "34"
[1] "new" "17.5"
[1] "original" "33"
[1] "new" "17"
[1] "original" "31"
[1] "new" "16"
[1] "original" "29"
[1] "new" "15"
[1] "original" "3"
[1] "new" "2"
[1] "original" "58"
```
Exit the cluster and terminal

• Command: exit

• Execute command twice (i.e., to leave cluster and then the terminal)
Additional Notes

Close Globus Connect Personal
Additional Resources

- An introduction to command line
- Running R script with SharcNet
- Using Secure Shell (SSH)
- Creating a key pair on iOS and Linux
- Information about Modules
- Information about Globus
- Information about running jobs on SharcNet
- Information about SharcNet
- Other video resources provided by SharcNet
- Additional Information about SharcNet
- Resource Document

Note: The Alliance offers weekly new user seminars
Demonstration

```r
nsim <- 5000
n <- 50
alpha <- 0.05
results <- logical(length = n)

for (i in 1:nsim) {
  results[i] <- shapiro.test(rnorm(n))$p.value <= alpha
}

rej <- mean(results)
print(rej)
```
Thanks for Listening!