Introduction to Computing at CARC

1 hour 15 minute version with Mesa Star and Jupyter Notebooks

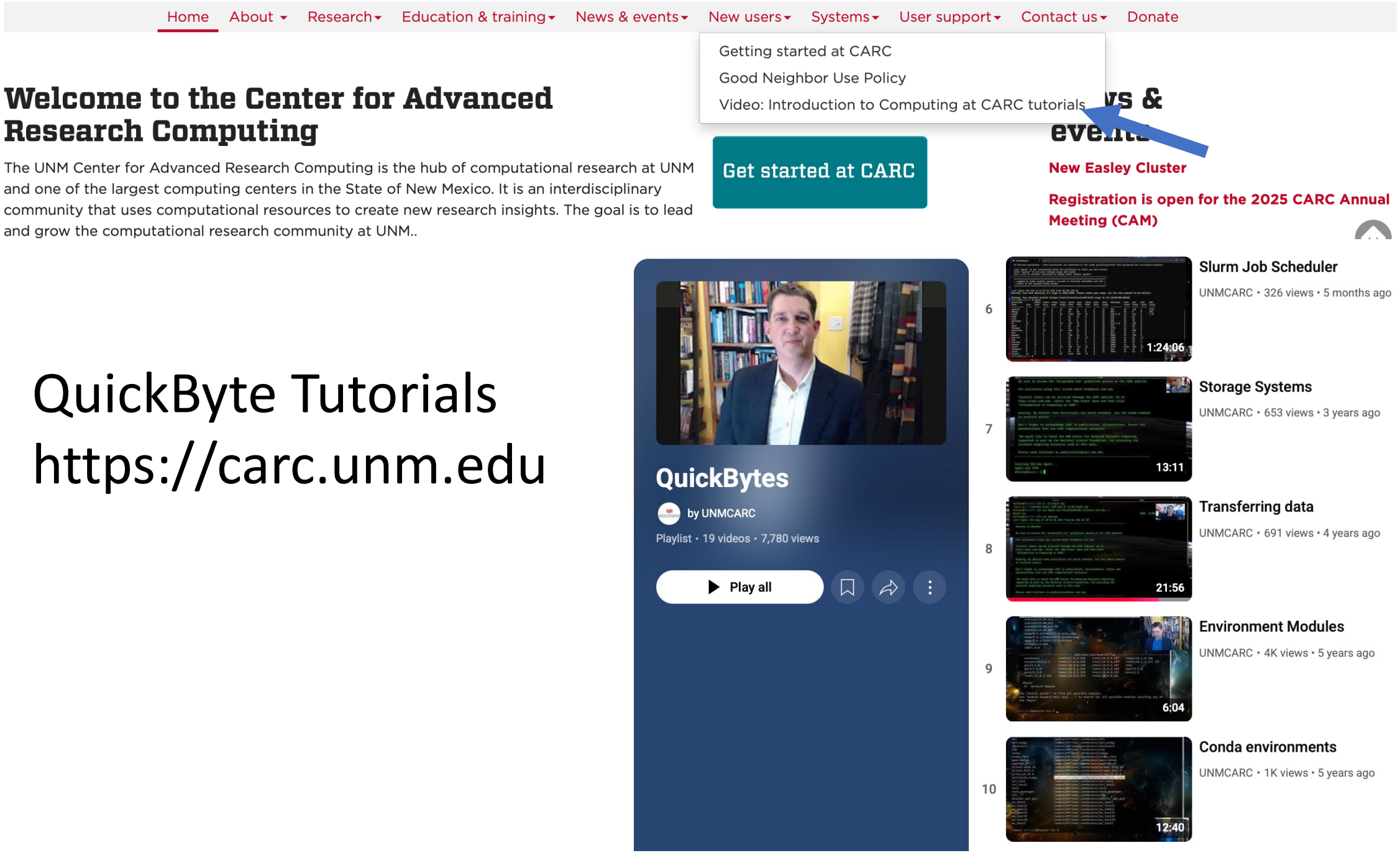
Matthew Fricke

Version 0.1

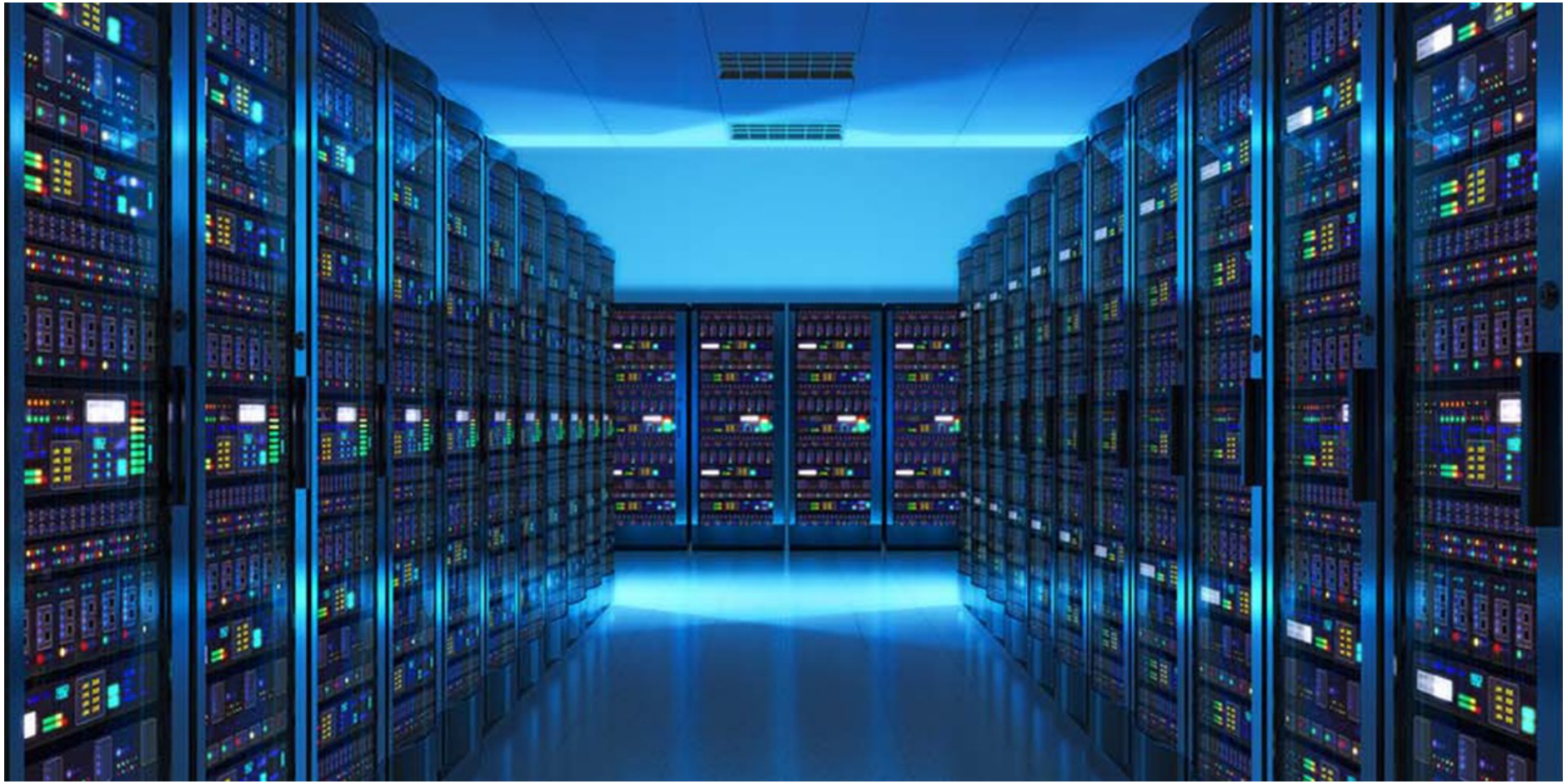
Goals

1. High Performance Computing
2. BASH
3. SLURM scheduler
4. Running Mesa Star interactively and in batch
5. JupyterHub

- We wont cover file transfer, storage systems, module system, conda, PBS. (These are all covered in depth in the video tutorials)


Home
About
Research
Education & training
News & events
New users
Systems
User support
Contact us
Donate
Getting started at CARC
Good Neighbor Use Policy
Video: Introduction to Computing at CARC tutorials
Welcome to the Center for Advanced Research Computing
The UNM Center for Advanced Research Computing is the hub of computational research at UNM and one of the largest computing centers in the State of New Mexico. It is an interdisciplinary community that uses computational resources to create new research insights. The goal is to lead and grow the computational research community at UNM..
Get started at CARC
New Easley Cluster
Registration is open for the 2025 CARC Annual Meeting (CAM)
QuickByte Tutorials
https://carc.unm.edu
QuickBytes
by UNMCARC
Playlist • 19 videos • 7,780 views
Play all
6
Slurm Job Scheduler
UNMCARC • 326 views • 5 months ago
1:24:06
7
Storage Systems
UNMCARC • 653 views • 3 years ago
13:11
8
Transferring data
UNMCARC • 691 views • 4 years ago
21:56
9
Environment Modules
UNMCARC • 4K views • 5 years ago
6:04
10
Conda environments
UNMCARC • 1K views • 5 years ago
12:40

HPC Cluster



High Performance Computing

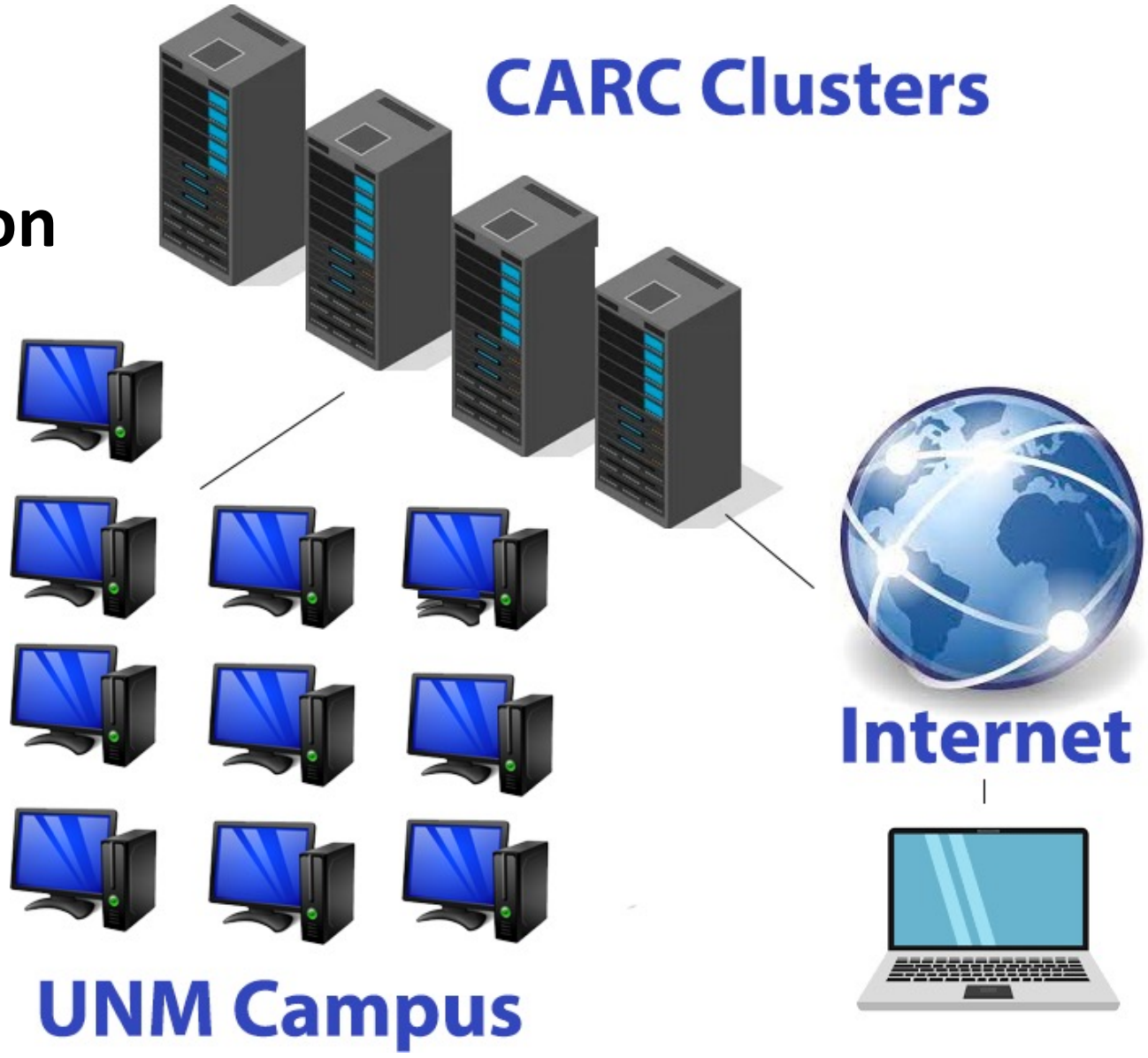
- What is high performance computing?
  - Really just means something that is a lot more powerful than your desktop or laptop.
  - Hardware:
    - That might mean more and faster processors to do the calculations more quickly (eg 400 CPUs instead of 4)
    - More RAM so you can work on bigger problems (3,000 GB instead of 8)
    - Bigger file systems so you can process larger datasets
    - More and bigger GPUs to accelerate your computations (12 GPUs at a time instead of 1)
  - People:
    - Someone else to manage the systems and keep them running and secure
    - Someone to answer your questions and help with problems

Logging into Easley

First login to the **workstation** in front of you.

You will always login to CARC cluster remotely.

These clusters don’t even have monitors.



Logging into Easley

First login to the Linux **workstation** in front of you.

If you have logged in before, use your existing password

Otherwise, your initial password is Welcome2CARC2025

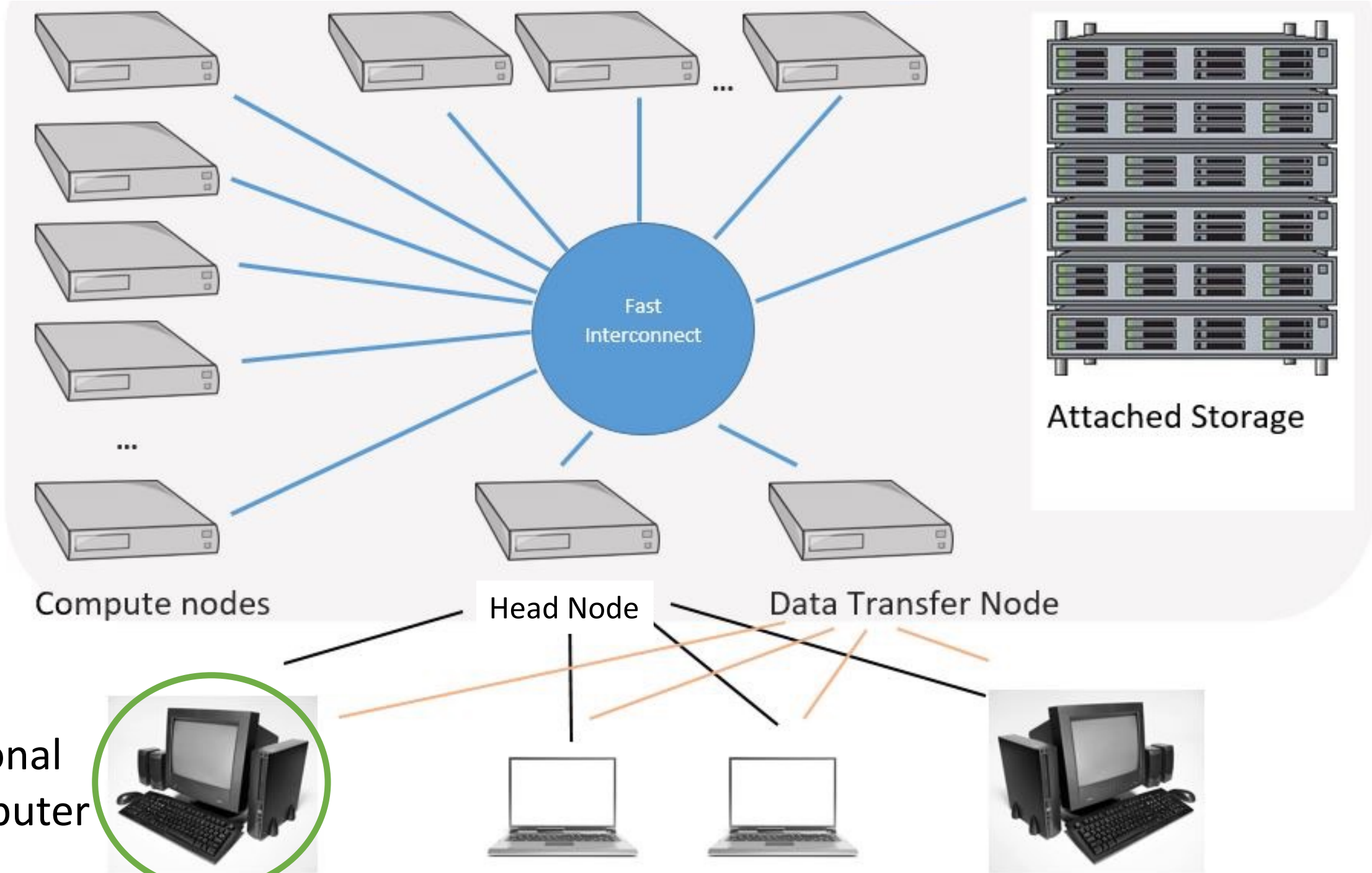
This is an “important step” so don’t let me move on until you have logged in

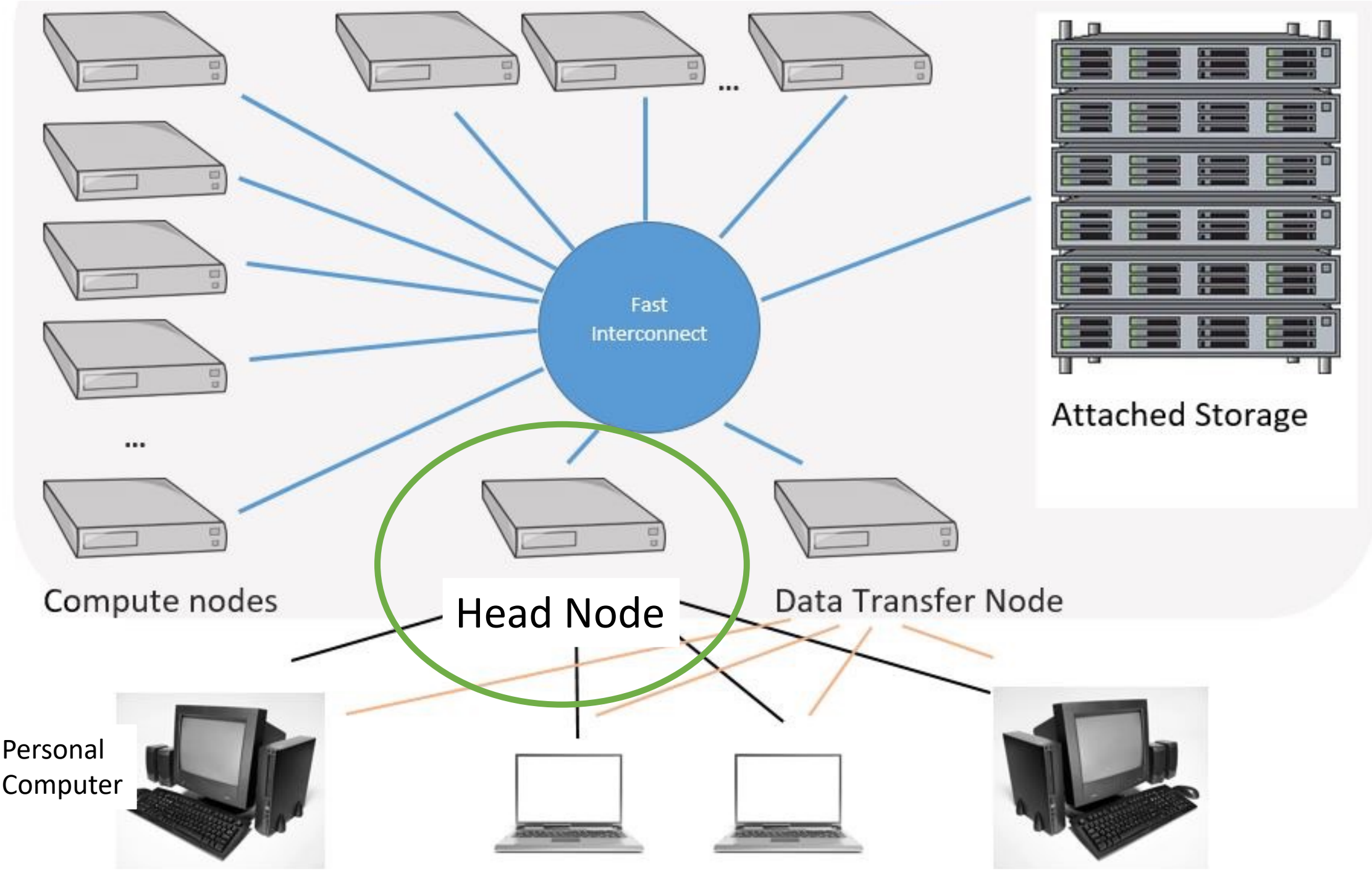
First login to the Linux **workstation** in front of you.

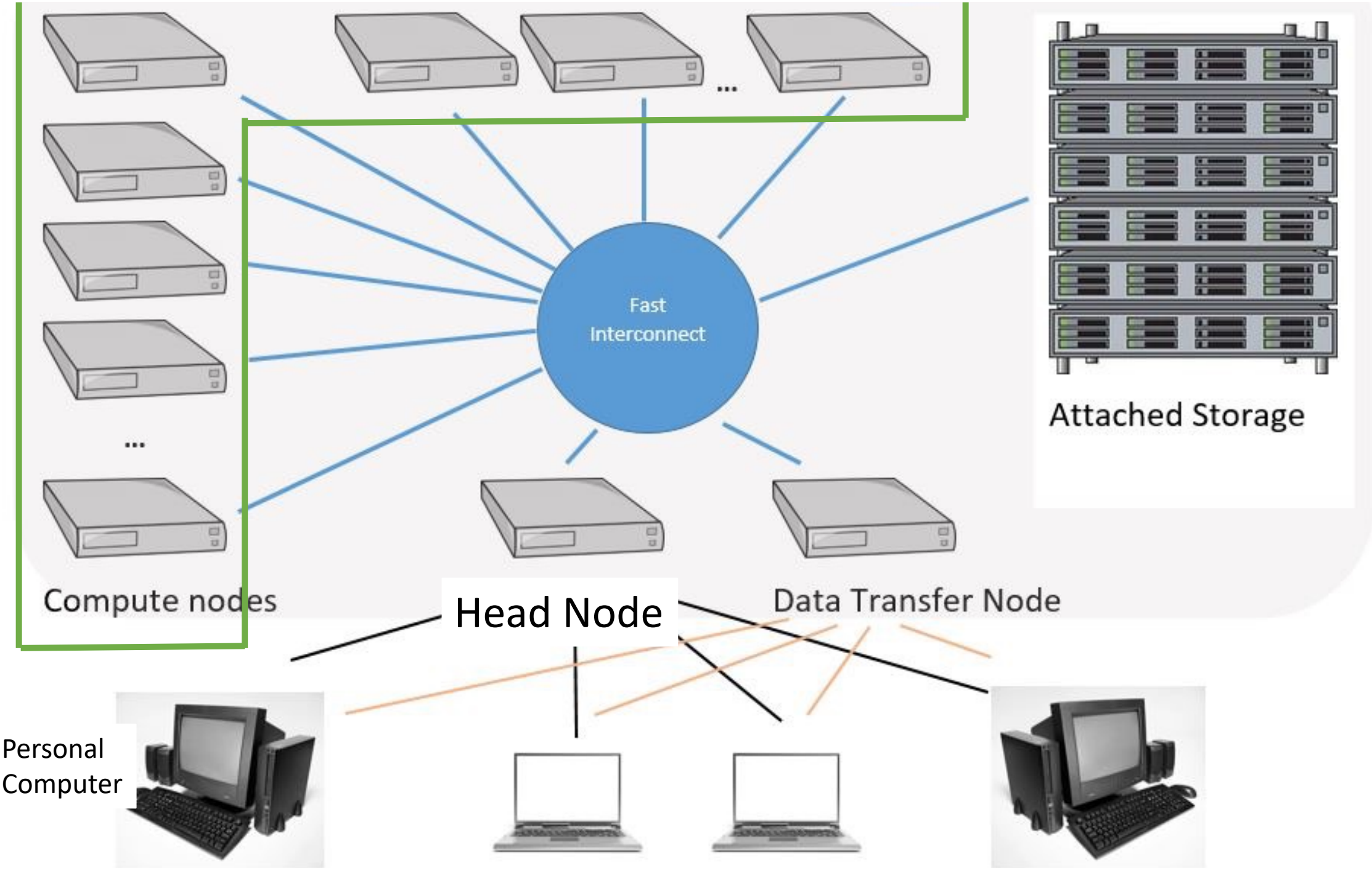
If you have logged in before, use your existing password

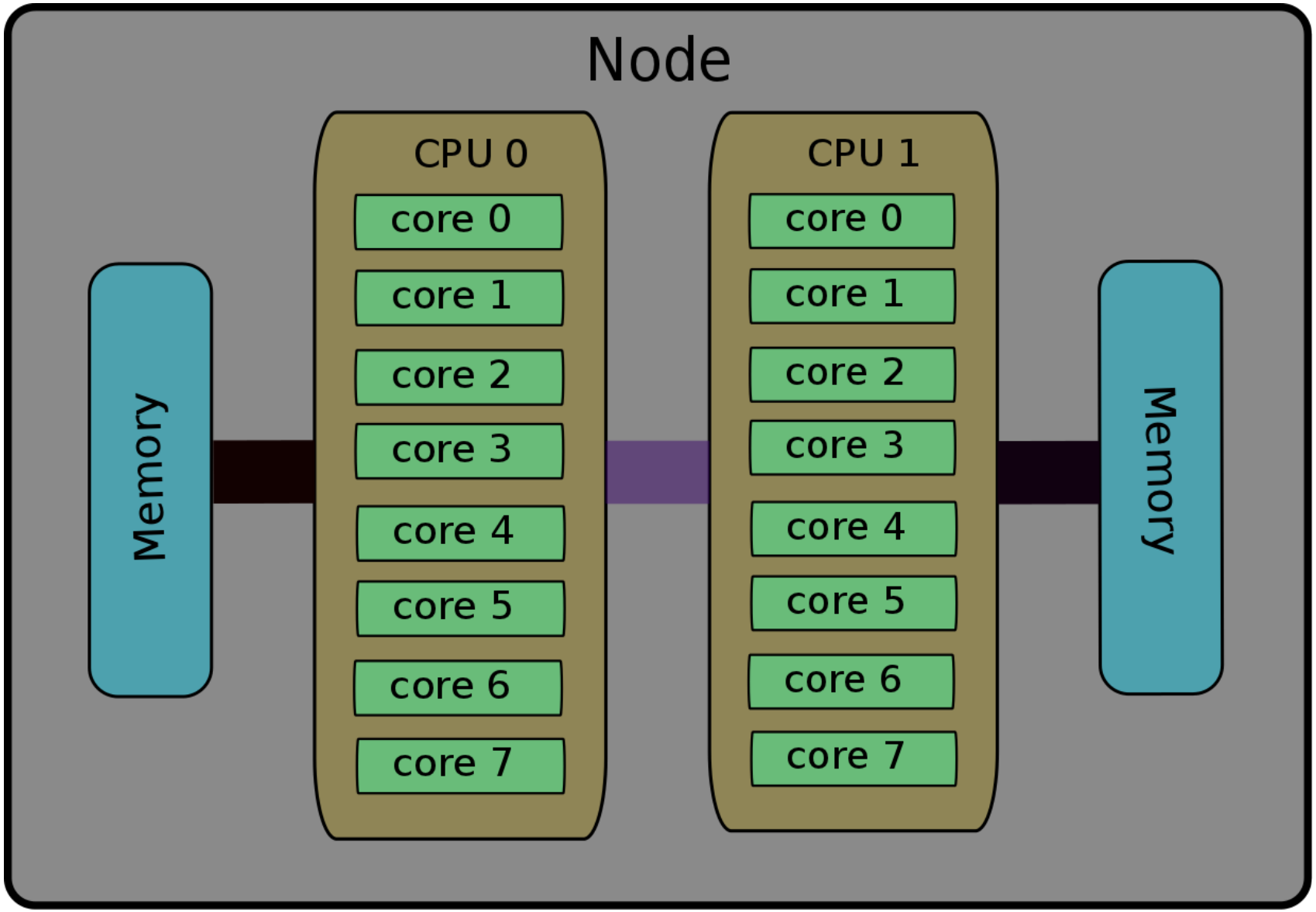
Otherwise, your initial password is Welcome2CARC2025

Username	First Name	Last Name	Email	Role	Status
ebhort27	Eleanor Blue	Hort	ebhort27@unm.edu	User	Active
hduan	Huaiyu	Duan	duan@unm.edu	Manager	Active
ipavel	Izabela	Pavel	ipavel@unm.edu	User	Active
jrosentel	Joseph Kenneth	Rosentel	jrosentel@unm.edu	User	Active
snovanthony	Kristina	Anthony-Semenova	snovanthony@unm.edu	User	Active
sstamer	Sarah	Stamer	sstamer@unm.edu	User	Active
turtle	Anson	Kost	anson@unm.edu	User	Active
woodkensia	Charles	Woodkensia	woodkensia@unm.edu	User	Active

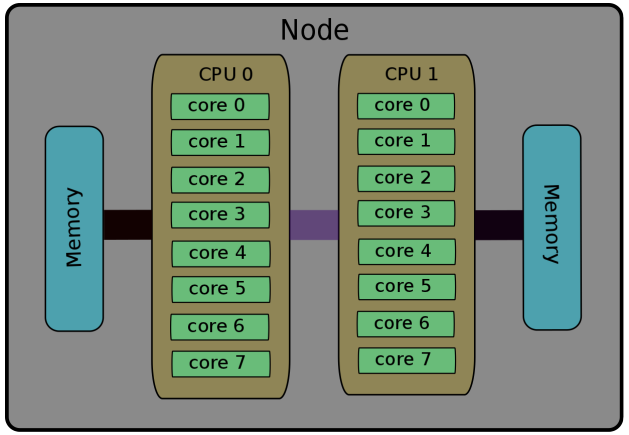

Fast
Interconnect
Attached Storage
Compute nodes
Head Node
Data Transfer Node
Personal
Computer


Fast
Interconnect
Attached Storage
Compute nodes
Head Node
Data Transfer Node
Personal
Computer

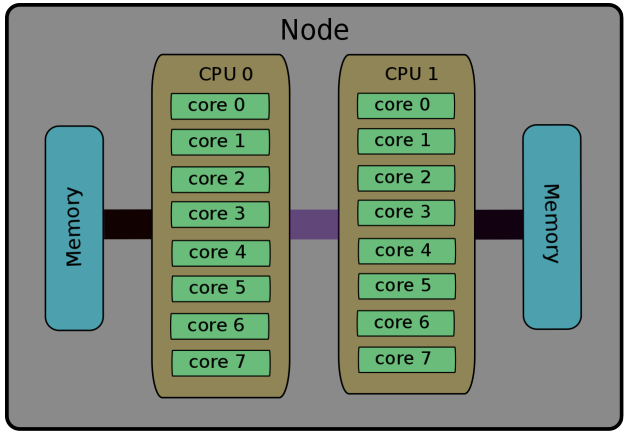

Fast
Interconnect
Attached Storage
Compute nodes
Head Node
Data Transfer Node
Personal
Computer


Node
CPU 0
core 0
core 1
core 2
core 3
core 4
core 5
core 6
core 7
CPU 1
core 0
core 1
core 2
core 3
core 4
core 5
core 6
core 7
Memory
Memory

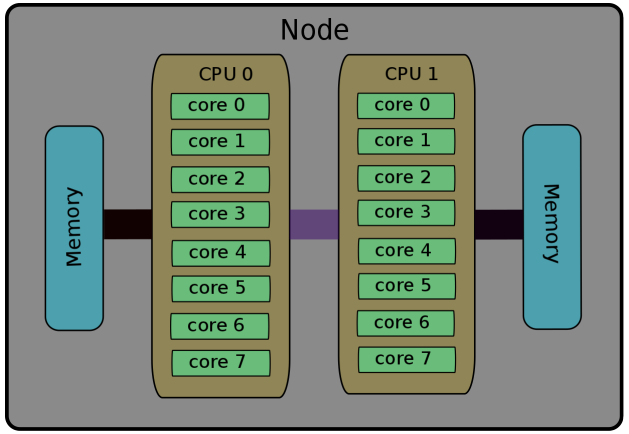
Head Node (hopper)



Compute Nodes

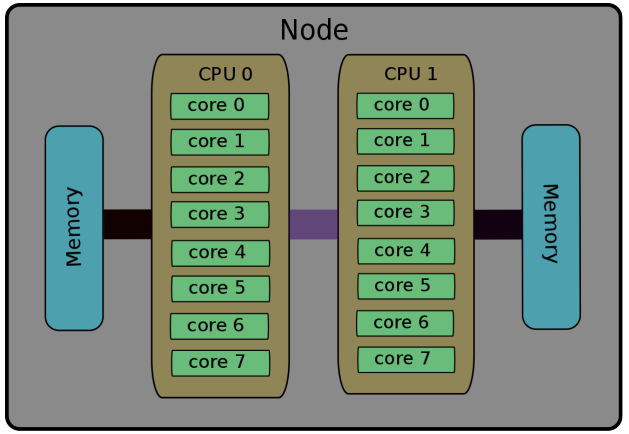


hopper001



hopper002

...



hopper067


Activities
Dec 24 16:35
Type to search
Rocky

Logging into Easley

Replace “vanilla” with your username in these slides:

```
ssh vanilla@easley.alliance.unm.edu
```

You may be prompted for a password…

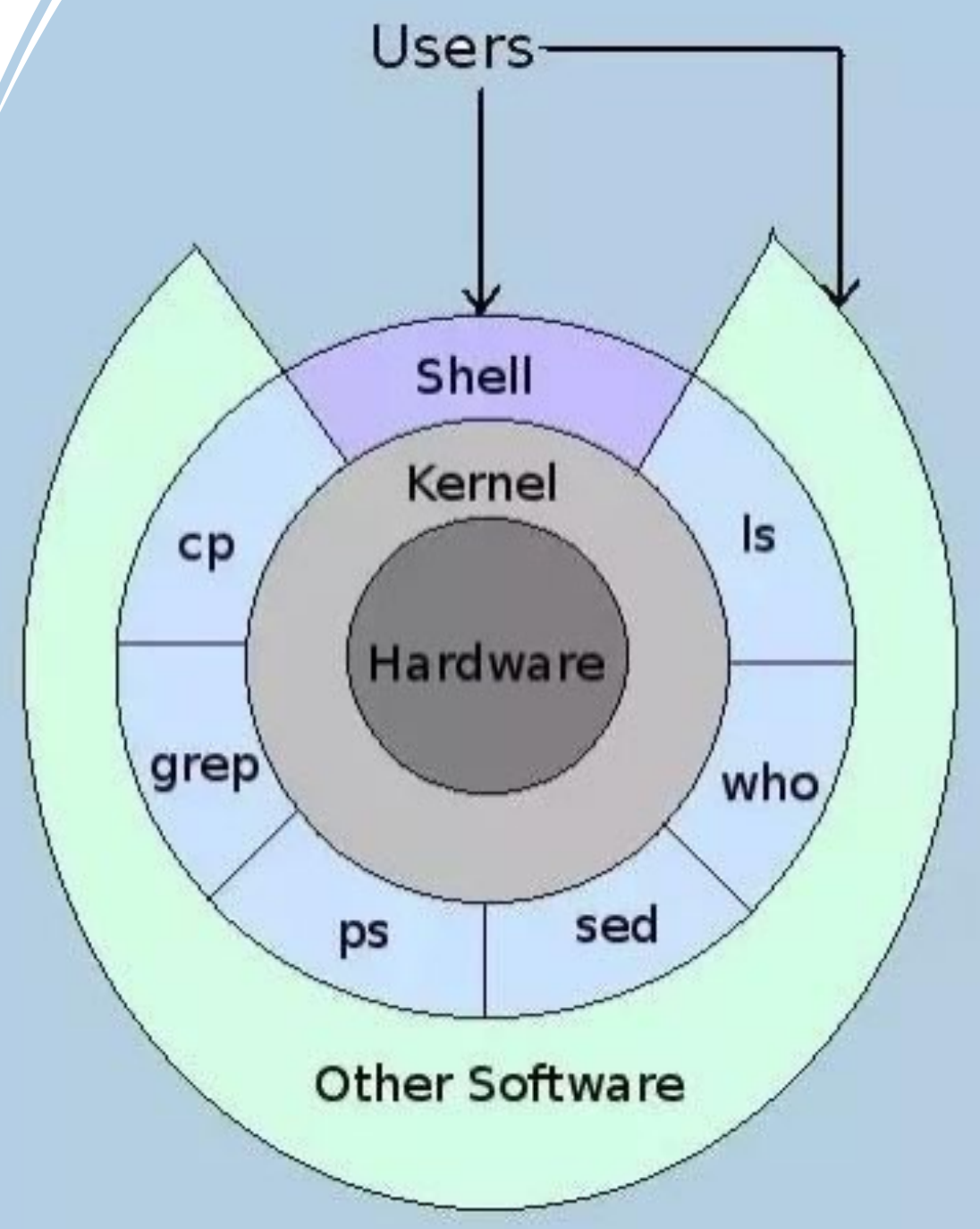
Don’t let me move on until you are able to login.

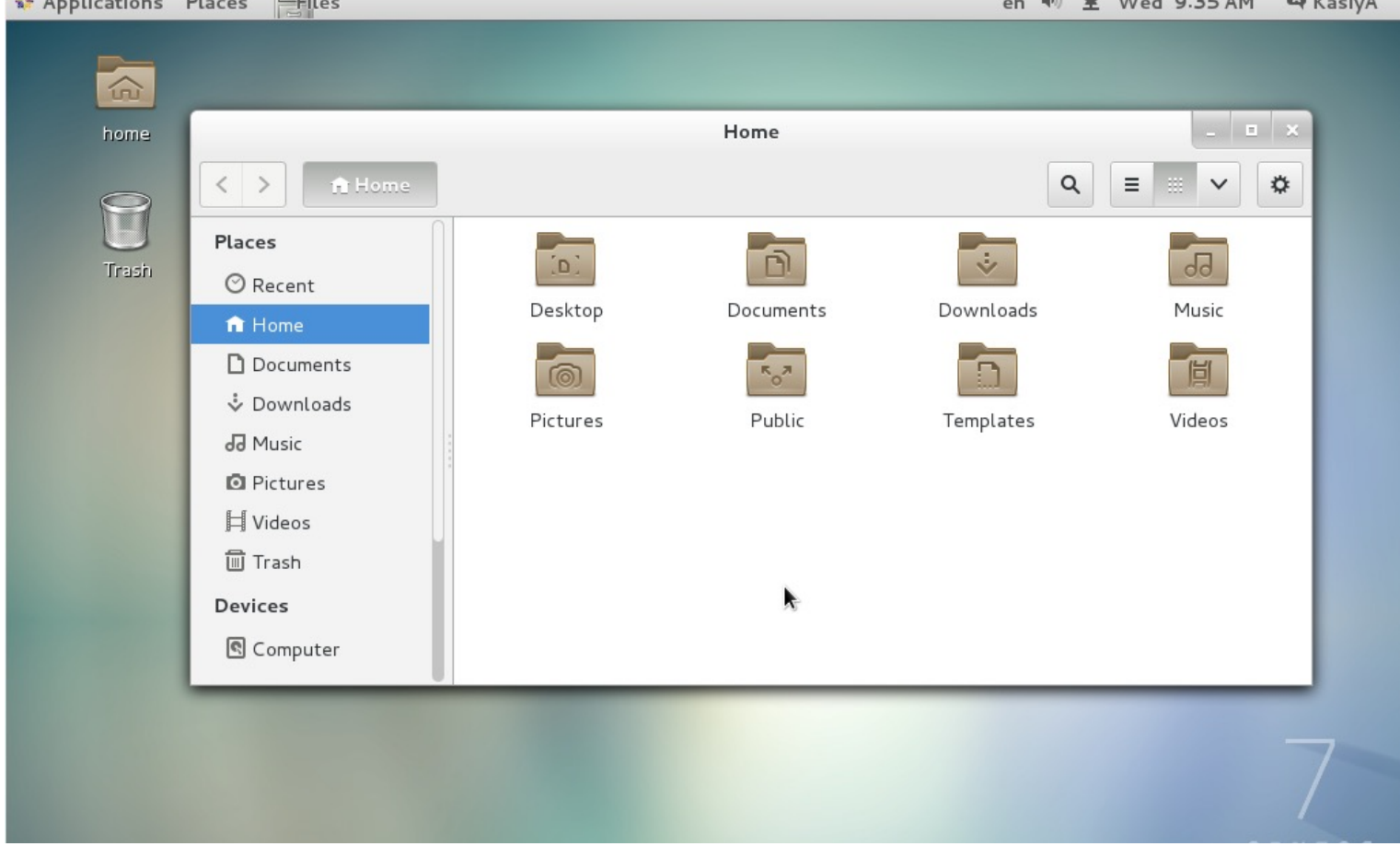
Logging into Easley

```
  Welcome to Easley
  ▶️ By using CARC systems you agree to the CARC Good
Neighbor Guide
  ▶️ Tutorials and the Good Neighbor Guide are at
https://goto.unm.edu/carc-start
  ▶️ Need help? Email: help@carc.unm.edu


The Easley cluster is currently in open beta;
unannounced restarts may occur.

vanilla@easley:~ $
```

Linux and the BASH Shell

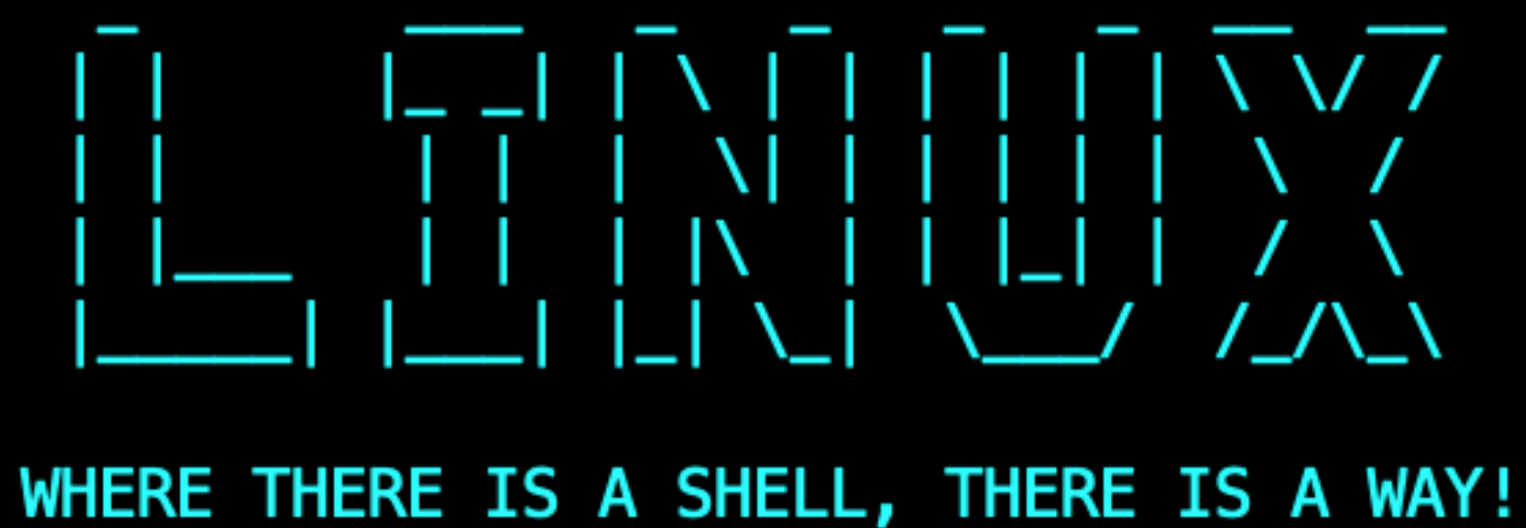
- The Kernel manages access to the hardware in a computer.
- An Operating System (OS) is the Kernel plus useful programs provided by the OS.
- The “shell” is the outermost layer of the OS.
- It is where the user interacts with the OS.



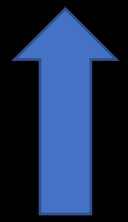


Graphical Shells (GUIs)

Linux and the BASH Shell

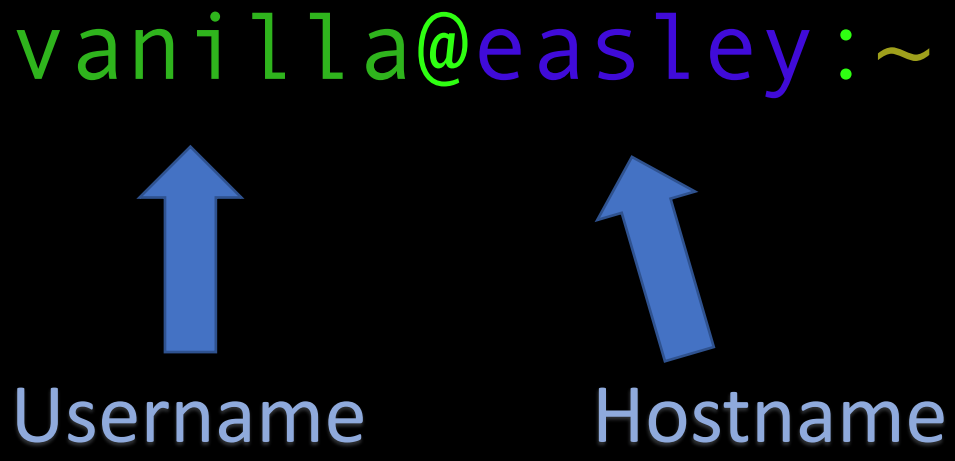


```
vanilla@easley:~ $
```



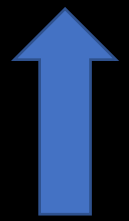
Username

Understanding the BASH prompt…



Understanding the BASH prompt…

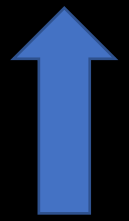
```
vanilla@easley:~ $
```



This is the current working directory.
"~" is short for home directory

Understanding the BASH prompt…

```
vanilla@easley:~ $
```



“$” means this user is standard user
(i.e. not a system administrator)

Understanding the BASH prompt…

```
[vanilla@hopper ~]$ git clone https://lobogit.unm.edu/CARC/workshops.git
Cloning into 'workshops'...
remote: Enumerating objects: 132, done.
remote: Counting objects: 100% (75/75), done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 132 (delta 33), reused 74 (delta 32), pack-reused 57
Receiving objects: 100% (132/132), 57.58 KiB | 3.60 MiB/s, done.
Resolving deltas: 100% (51/51), done.
```

Rather than make you write scripts lets just download some we wrote for this workshop…

```
vanilla@easley:~$ cd workshops
vanilla@easley:~/workshops$
```

The “cd” command changes the current directory
Notice the BASH prompt has changed

Shortcuts:
~ = home directory
.. = the directory above
- = the previous current directory
. = current directory, useful when we want to run a program that’s in the current directory

```
vanilla@easley:~$ cd workshops
vanilla@easley:~/workshops$ ls
crystallography  intro_workshop    namd              README.md
escape           machine_learning  quantum_computing  starccm
gaussian         mesastar          radio_astronomy
vanilla@easley:~/workshops$
```

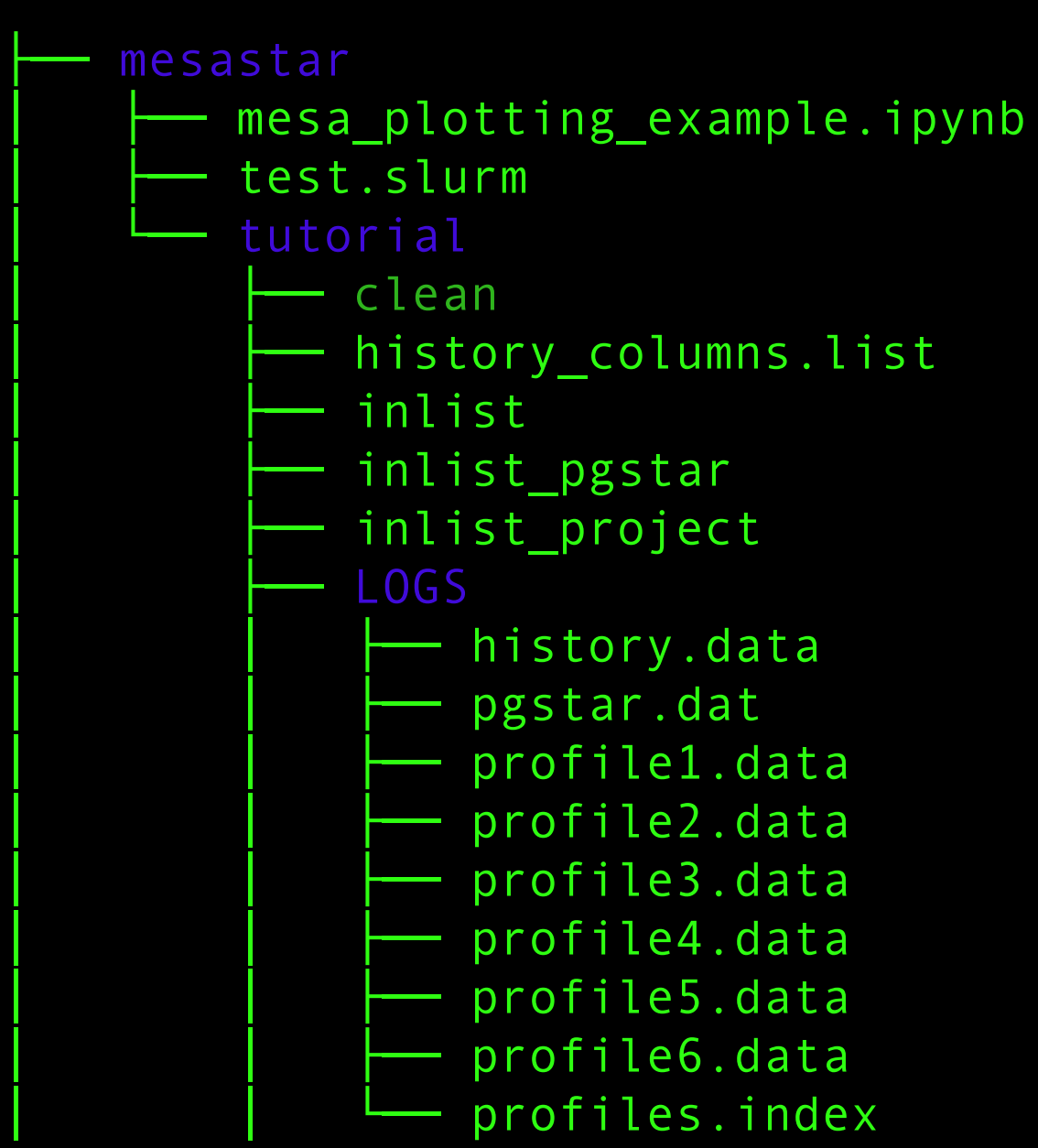
The “ls” command lists the contents of the current directory

```
vanilla@easley:~/workshops$ pwd
/users/vanilla/workshops
```

The “pwd” command lists *path* to the current directory

```
vanilla@easley:~/workshops$ tree
.
├── crystallography
│   ├── data
│   │   └── thaumatin.pdb
│   └── slurm
│       ├── integrate_xray_images.slurm
│       └── solve_structure.slurm
├── escape
│   ├── fit_insar_data.ipynb
│   └── kml_functions.py
├── gaussian
│   ├── AcetylChloride.com
│   └── gaussian16_linda_acetylchloride.sh
├── intro_workshop
│   ├── code
│   │   ├── calc_pi_mpi.f90
│   │   ├── calcPiMPI.py
│   │   ├── calc_pi_serial.f90
│   │   ├── calcPiSerial.py
│   │   └── vecadd
│   │       ├── Makefile
```

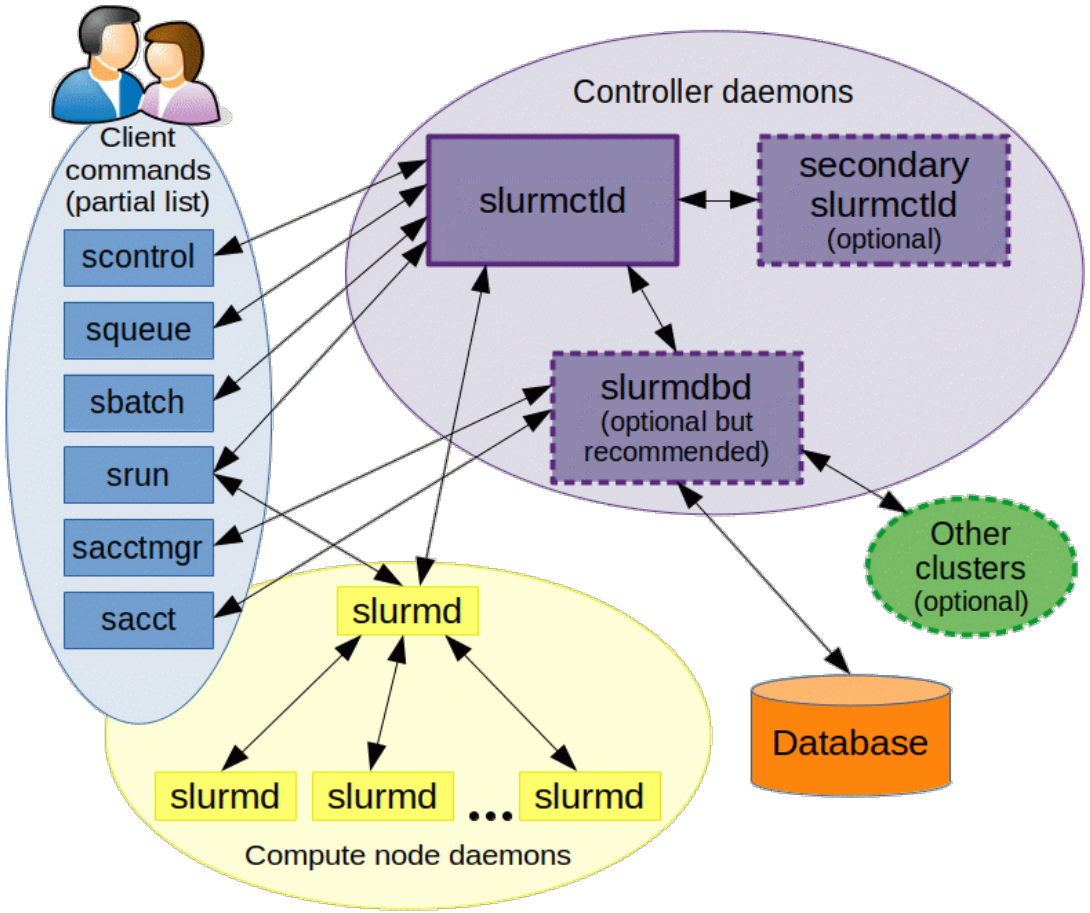
The “tree” command shows you the directory structure



You will have a “mesastar” directory

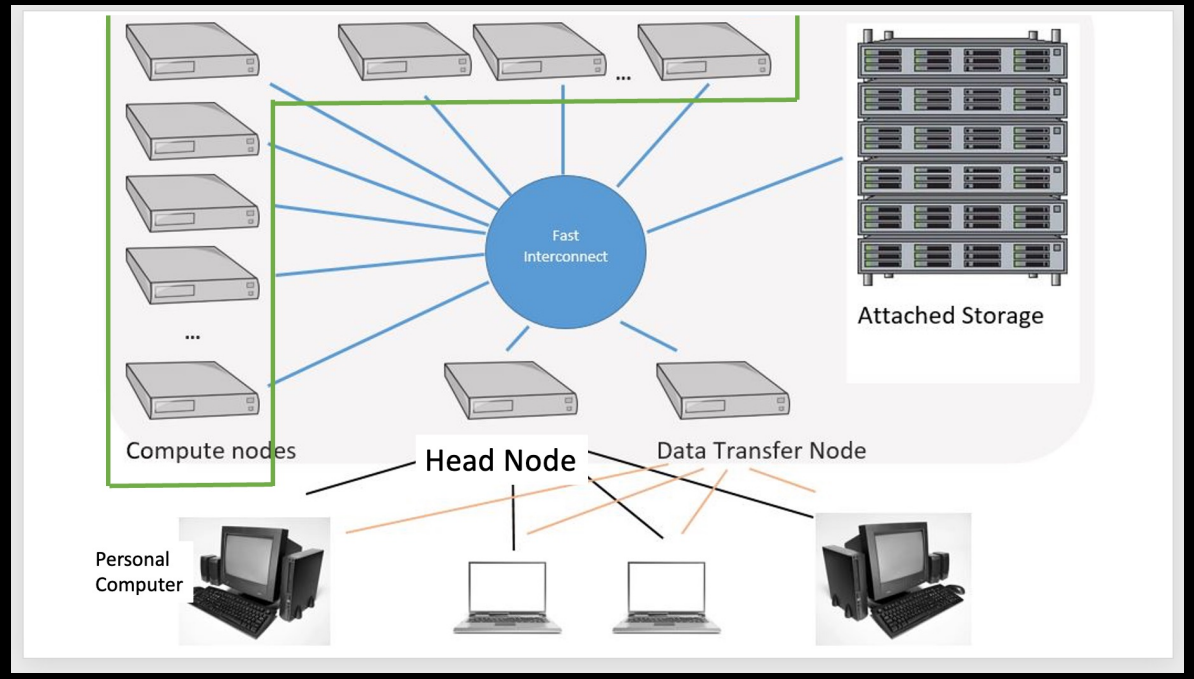
Simple Linux Utility for Resource Management

slurm
workload manager




ENJOY
Slurm
SODA
IT'S HIGHLY ADDICTIVE!
VOTED #1 SOFT DRINK OF THE 31ST CENTURY!

```
vanilla@easley:~/workshops$ qgrok
queues       free   busy   offline  jobs   nodes  CPUs   idle_CPUS  GPUs   CPUs/node  memory/node  time_limit   CPU_limit  GPU_limit  RAM_limit
-----        -----  -----  -------  -----  -----  -----  ---------  -----  ---------  -----------  -----------  ---------  ---------  ---------
general      4      36     8        10     48     3072   256        0      64         252G         2-00:00:00   512        0          2059472M
bigmem       2      0      0        0      2      128    128        0      64         2.0T         2-00:00:00   32         0          2063765M
h100         0      4      0        4      4      256    181        8      64         1008G        2-00:00:00   64         2          1031578M
l40s         1      2      0        2      3      192    127        12     64         252G         2-00:00:00   96         3          171624M
interactive  1      6      0        0      7      448    308        20     64         252G-1008G   4:00:00      64         1
debug        7      42     8        0      57     3648   692        20     64         252G-1008G   1:00:00      128        2
scavenger    12     43     8        1      63     4032   1072       44     64         252G-1008G   2-00:00:00   4032       44
liulab       5      1      0        4      6      384    380        24     64         252G         7-00:00:00   384        24
```



Compute nodes are grouped into partitions/queues

```
vanilla@easley:~/workshops$ quotas

Home Directory (/users/vanilla):
Disk quotas for user vanilla (uid 659):
     Filesystem   space   quota   limit   grace   files   quota   limit   grace
chama:/home/homes
                 1527M    100G    200G           14913   4295m   4295m
----------------------------
Centerwide user scratch (/carc/scratch/users/vanilla)

Quota information for storage pool Default (ID: 1):

      user/group     ||           size          ||    chunk files
     name     |  id  ||    used    |    hard    ||  used   |  hard
--------------|------||------------|------------||---------|---------
       vanilla|  1512||  592.71 GiB | 1024.00 GiB ||    32784|unlimited

Centerwide scratch quota for project vanilla2016174 (/carc/scratch/projects/vanilla2016174)

Quota information for storage pool Default (ID: 1):
      user/group     ||           size          ||    chunk files
     name     |  id  ||    used    |    hard    ||  used   |  hard
--------------|------||------------|------------||---------|---------
vanilla2016174|2016142||  190.97 GiB| 1024.00 GiB||    23704|unlimited
```

```
vanilla@easley:~/workshops$ sinfo --partition debug
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      8 drain* easley[041-048]
debug        up    1:00:00      5    mix easley[051-055]
debug        up    1:00:00     37  alloc easley[001-017,019-035,039-040,056]
debug        up    1:00:00      7   idle easley[018,036-038,049-050,057]
vanilla@easley:~/workshops$
```

sinfo reports information about partitions

```
vanilla@easley:~/workshops$ sinfo --partition debug
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      8 drain* easley[041-048]
debug        up    1:00:00      5    mix easley[051-055]
debug        up    1:00:00     37  alloc easley[001-017,019-035,039-040,056]
debug        up    1:00:00      7   idle easley[018,036-038,049-050,057]
vanilla@easley:~/workshops$
```

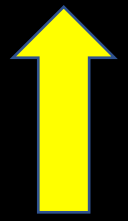
The debug queues are intended for testing your programs.

And for interactive jobs.

```
vanilla@easley:~/workshops$ sinfo --partition debug
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      7   idle easley[018-025]
```

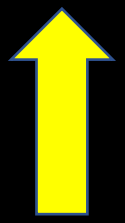
Name

```
vanilla@easley:~/workshops$ sinfo --partition debug
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      7   idle easley[018-025]
```



You can run a “job” for up to 1 hrs.

```
vanilla@easley:~/workshops$ sinfo --partition debug
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      7   idle easley[018-025]
```

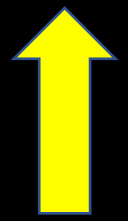


There are 7 nodes in this partition.

```
vanilla@easley:~/workshops$ sinfo --partition debug
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      7   idle easley[018-025]
```

The names of the nodes in the partition

```
vanilla@easley:~/workshops$ hostname
easley
vanilla@easley:~/workshops$
```



Running on the Head Node.
The head node's name is "easley" the same as the cluster name.

Never run computations on the head node

Always use compute nodes

```
vanilla@easley:~/workshops$ hostname
hopper
vanilla@easley:~/workshops$ man hostname
```

"man" displays manual pages for Linux commands

```
vanilla@easley:~/workshops$ man hostname
(‘q’ to quit)

vanilla@easley:~/workshops$ man man
(‘q’ to quit)
```

“man” displays manual pages for Linux commands

```
vanilla@easley:~/workshops$ man sinfo
```

```
sinfo(1)                                                    Slurm
Commands                                                    sinfo(1)

NAME
       sinfo - View information about Slurm nodes and partitions.

SYNOPSIS
       sinfo [OPTIONS...]

DESCRIPTION
       sinfo is used to view partition and node information for a system running
Slurm

OPTIONS
       -a, --all
              Display  information  about  all partitions. This causes information
to be displayed about partitions that are configured as hidden and partitions
that are unavailable to the user's group.
```

```
vanilla@easley:~/workshops$ sinfo --all
PARTITION   AVAIL  TIMELIMIT  NODES  STATE NODELIST
general*       up 2-00:00:00      8 drain* easley[041-048]
general*       up 2-00:00:00     36  alloc easley[001-017,019-035,039-040]
general*       up 2-00:00:00      4   idle easley[018,036-038]
bigmem         up 2-00:00:00      2   idle easley[049-050]
h100           up 2-00:00:00      4    mix easley[051-054]
l40s           up 2-00:00:00      1    mix easley055
l40s           up 2-00:00:00      1  alloc easley056
l40s           up 2-00:00:00      1   idle easley057
interactive    up    4:00:00      5    mix easley[051-055]
interactive    up    4:00:00      1  alloc easley056
interactive    up    4:00:00      1   idle easley057
debug          up    1:00:00      8 drain* easley[041-048]
debug          up    1:00:00      5    mix easley[051-055]
debug          up    1:00:00     37  alloc easley[001-017,019-035,039-040,056]
debug          up    1:00:00      7   idle easley[018,036-038,049-050,057]
scavenger      up 2-00:00:00      8 drain* easley[041-048]
scavenger      up 2-00:00:00      6    mix easley[051-055,058]
scavenger      up 2-00:00:00     37  alloc easley[001-017,019-035,039-040,056]
scavenger      up 2-00:00:00     12   idle easley[018,036-038,049-050,057,059-063]
liulab         up 7-00:00:00      1    mix easley058
liulab         up 7-00:00:00      5   idle easley[059-063]
```

```
vanilla@easley:~/workshops$ srun --partition debug hostname
```

Tell slurm to run a program
on a compute node…

```
vanilla@easley:~/workshops$ srun --partition debug hostname
```

Run the program on a compute node in the debug partition.

```
vanilla@easley:~/workshops$ srun --partition debug hostname
```

The program to run.

```
vanilla@easley:~/workshops$ srun --partition debug hostname
To request GPUs, add --gpus-per-node X or --gpus X,
where X is the desired number of GPUs.
Job 24897 running on easley054
easley054
```

Output of program

```
vanilla@easley:~/workshops$ squeue
```

```
vanilla@easley:~/workshops$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
             24343   general   Nb16D1   suryab  R    4:51:48      2 easley[004-005]
             24342   general   Nb16D1   suryab  R    4:55:23      2 easley[014-015]
             24341   general   Nb16D1   suryab  R    5:27:58      2 easley[016-017]
             24340   general   Nb16D1   suryab  R    5:40:26      2 easley[002-003]
             23822   general site_add zaarmijo  R 1-16:06:20      1 easley001
             23703   general    gmx_1 cstevens  R 1-16:20:37      1 easley019
             23700   general    gmx_2 cstevens  R 1-16:36:07      1 easley028
             24042   general cav1_fin schrient  R   22:00:49      8 easley[006-013]
             24043   general cav1_fin    mgenk  R   21:59:30      8 easley[030-035,039-040]
             24893   general  med_100 keenankh  R      32:13      8 easley[020-027]
             23432      h100 run1p_30 saulange PD       0:00      1 (QOSMaxMemoryPerUser)
             23866      h100 run4q_10 saulange PD       0:00      1 (QOSMaxMemoryPerUser)
             24775      h100     demo  sebasjr  R    9:46:03      1 easley052
             23385      h100  Control michel88  R 1-00:24:42      1 easley051
             23431      h100 run1q_30 saulange  R 1-01:32:39      1 easley053
             24026      h100   prod3S  aungzaw  R   14:53:22      1 easley054
             24030      l40s      cst   fredre  R   22:53:19      1 easley056
             23817      l40s d2527-pr wbricker  R 1-16:38:18      1 easley055
   23780_[57-70%1]    liulab runprod.   eugene PD       0:00      1 (JobArrayTaskLimit)
   23783_[57-70%1]    liulab runprod.   eugene PD       0:00      1 (JobArrayTaskLimit)
   23785_[57-70%1]    liulab runprod.   eugene PD       0:00      1 (JobArrayTaskLimit)
```

```
vanilla@easley:~/workshops$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
             24343   general   Nb16D1   suryab  R    4:51:48      2 easley[004-005]
             24342   general   Nb16D1   suryab  R    4:55:23      2 easley[014-015]
             24341   general   Nb16D1   suryab  R    5:27:58      2 easley[016-017]
             24340   general   Nb16D1   suryab  R    5:40:26      2 easley[002-003]
             23822   general site_add zaarmijo  R 1-16:06:20      1 easley001
             23703   general    gmx_1 cstevens  R 1-16:20:37      1 easley019
             23700   general    gmx_2 cstevens  R 1-16:36:07
             24042   general cav1_fin schrient  R   22:00:49
             24043   general cav1_fin    mgenk  R   21:59:30
             24893   general  med_100 keenankh  R      32:13
             23432      h100 run1p_30 saulange PD          00
                                      nge PD       0:00
                                      r  R    9:46:03
                                      8  R 1-00:24:42
                                      e  R 1-01:32:39      1 easley053
                                      w  R   14:53:22      1 easley054
                                      e  R   22:53:19      1 easley056
                                      r  R 1-16:38:18      1 easley055
                                      e PD       0:00      1 (JobArrayTaskLimit)
                                      e PD       0:00      1 (JobArrayTaskLimit)
                                      e PD       0:00      1 (JobArrayTaskLimit)
```

PD means programs that are waiting their turn.

Shows you what the slurm scheduler is doing right now.

Here we can see that user ’erowland’ has a lot of programs waiting to run.

```
vanilla@easley:~/workshops$ squeue -t R --all
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
             24343   general   Nb16D1   suryab  R    4:54:47      2 easley[004-005]
             24342   general   Nb16D1   suryab  R    4:58:22      2 easley[014-015]
             24341   general   Nb16D1   suryab  R    5:30:57      2 easley[016-017]
             24340   general   Nb16D1   suryab  R    5:43:25      2 easley[002-003]
             23822   general site_add zaarmijo  R 1-16:09:19      1 easley001
             23703   general    gmx_1 cstevens  R 1-16:23:36      1 easley019
             23700   general    gmx_2 cstevens  R 1-16:39:06      1 easley028
             24042   general cav1_fin schrient  R   22:03:48      8 easley[006-013]
             24043   general cav1_fin    mgenk  R   22:02:29      8 easley[030-
035,039-040]
             24893   general  med_100 keenankh  R      35:12      8 easley[020-027]
             24775      h100     demo  sebasjr  R    9:49:02      1 easley052
             23385      h100  Control michel88  R 1-00:27:41      1 easley051
             23431      h100 run1q_30 saulange  R 1-01:35:38      1 easley053
             24026      h100   prod3S  aungzaw  R   14:56:21      1 easley054
```

Interactive vs Batch Mode

Interactive Mode

- Srun is **“interactive”**. You request hardware, run your program, and get the output on your screen right away.

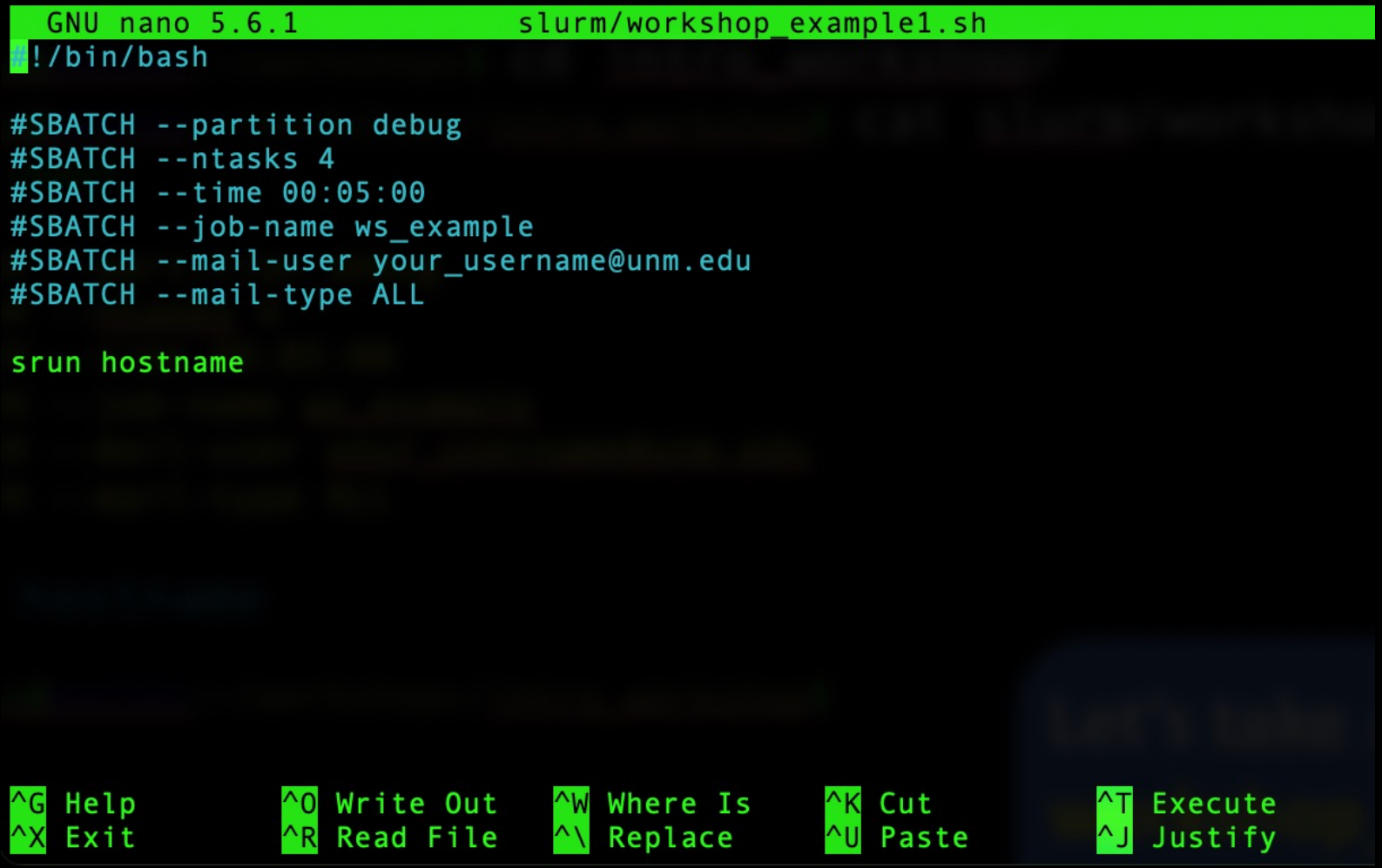
Batch Mode

- Most programs at an HPC center are run in “batch” mode.
- Batch mode means we write a shell script that the SLURM scheduler runs for us. The script requests hardware just like we did with salloc and then runs the commands in the script.
- Whatever would have been written to the screen is saved to a file instead.

```
vanilla@easley:~/workshops$ cd intro_workshop/
vanilla@easley:~/workshops/intro_workshop$ cat slurm/workshop_example1.sh
#!/bin/bash

#SBATCH --partition debug
#SBATCH --ntasks 4
#SBATCH --time 00:05:00
#SBATCH --job-name ws_example
#SBATCH --mail-user your_username@unm.edu
#SBATCH --mail-type ALL

srun hostname

vanilla@easley:~/workshops/intro_workshop$
```

Let's take a look at the workshop_example1.sh script in the slurm directory…

```
vanilla@easley:~/workshops/intro_workshop$ nano slurm/workshop_example1.sh
```



“nano” is a simple text editor

Change the –mail-user line to use your email

Control-x will exit and prompt you to save

```
vanilla@easley:~/workshops/intro_workshop$ sbatch slurm/workshop_example1.sh
Submitted batch job 24898
vanilla@easley:~/workshops/intro_workshop$
```

We submit our slurm shell script with the sbatch command.

```
vanilla@easley:~/workshops/intro_workshop$ sbatch slurm/workshop_example1.sh
Submitted batch job 24898
vanilla@easley:~/workshops/intro_workshop$
```

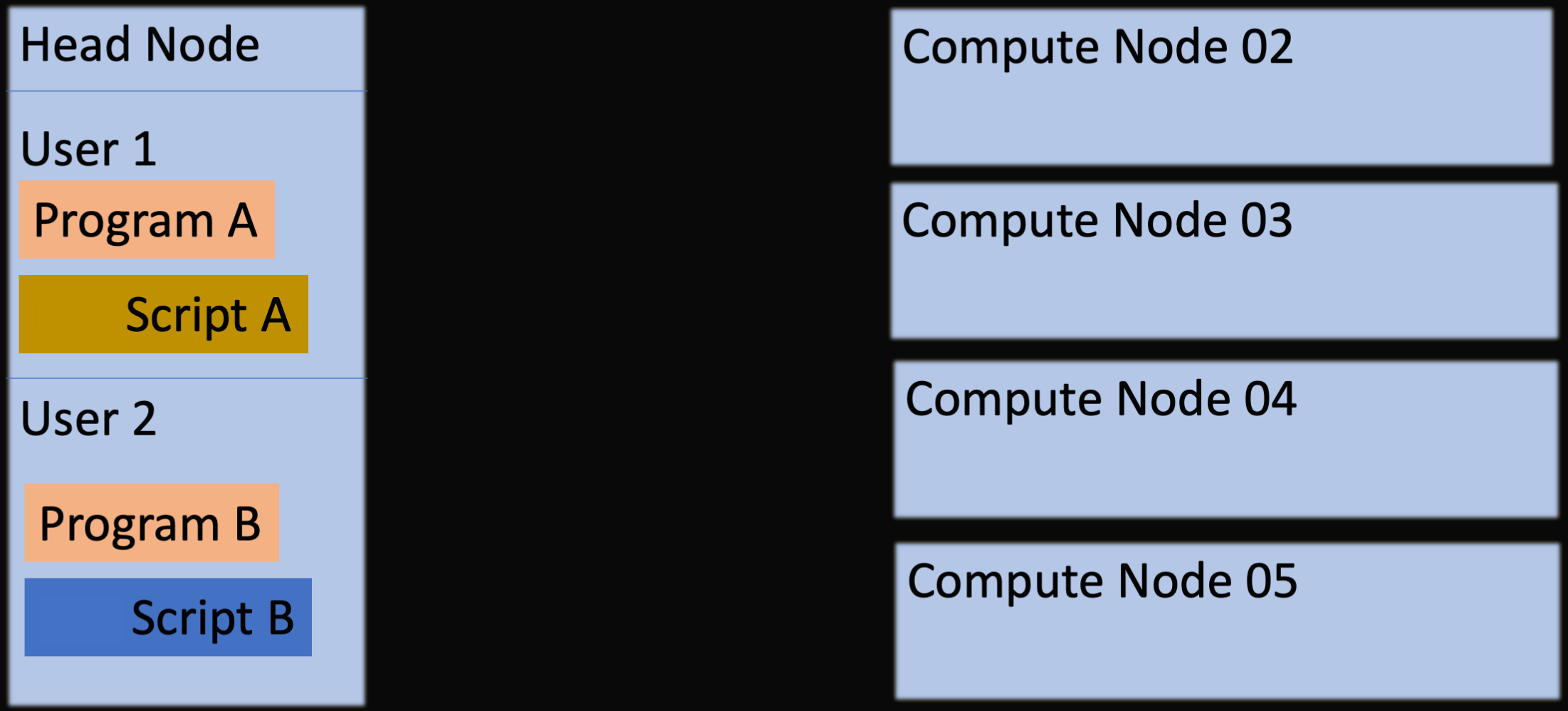
We submit our slurm shell script with the sbatch command.

Notice that the only output we get is a job id.

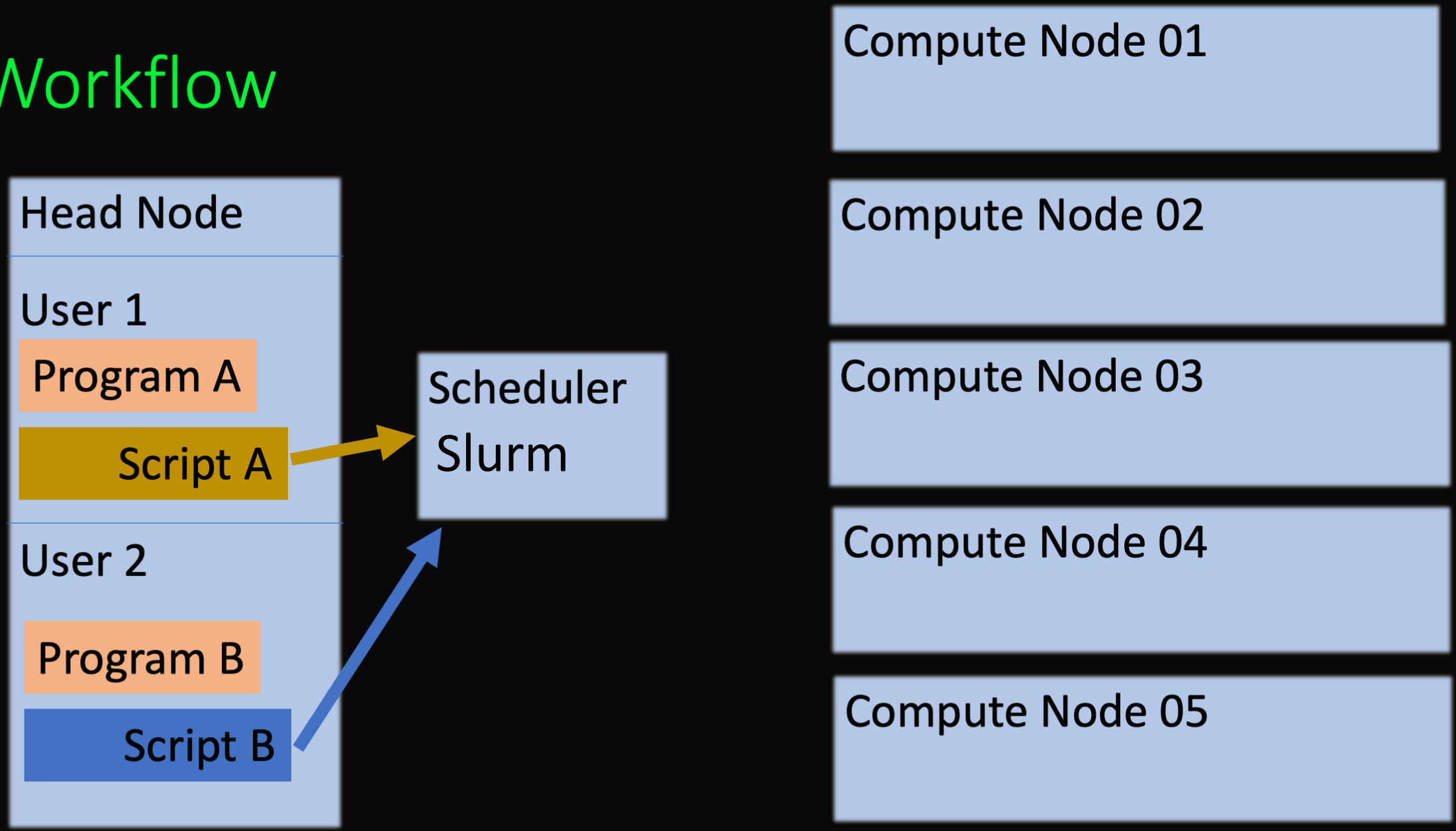
This indicates that the script was successfully sent to the scheduler.

The commands in the script will run as soon as the hardware requested is available.

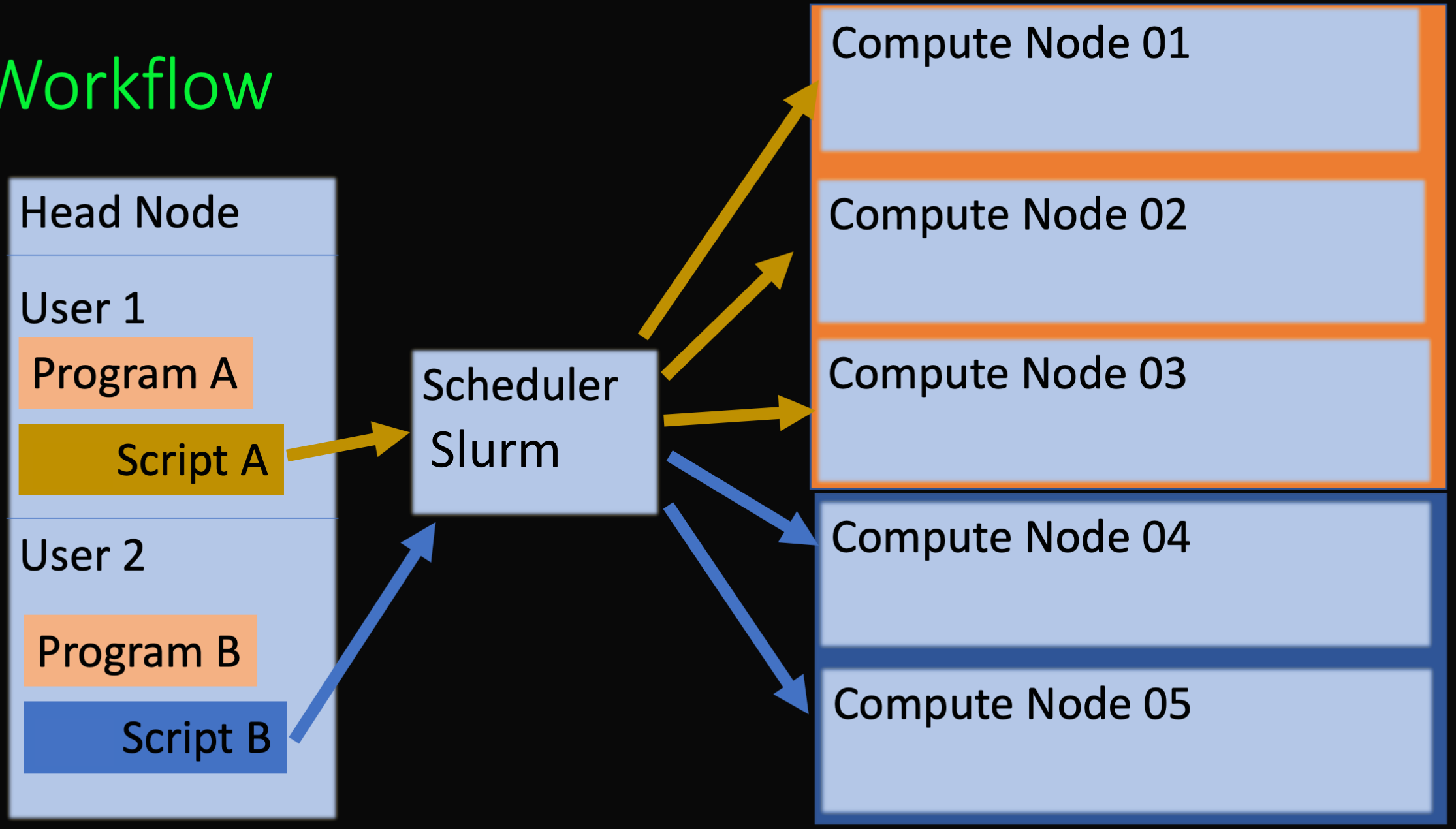
Workflow


Head Node
User 1
Program A
Script A
User 2
Program B
Script B
Compute Node 01
Compute Node 02
Compute Node 03
Compute Node 04
Compute Node 05
Shared filesystems – All nodes can access the same programs and write output

Workflow

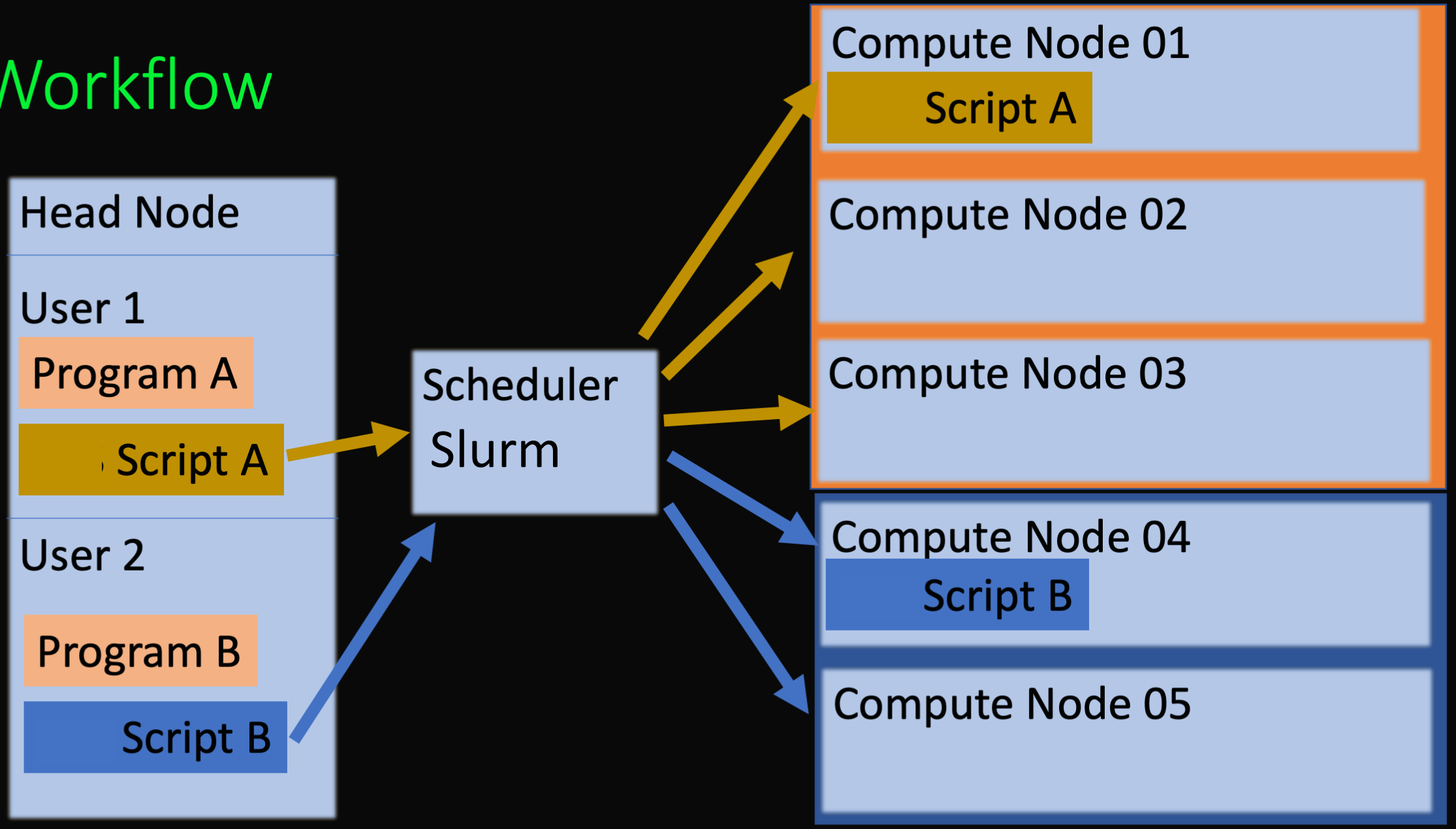

Head Node
User 1
Program A
Script A
User 2
Program B
Script B
Scheduler
Slurm
Compute Node 01
Compute Node 02
Compute Node 03
Compute Node 04
Compute Node 05
Shared filesystems – All nodes can access the same programs and write output

Workflow



Shared filesystems – All nodes can access the same programs and write output

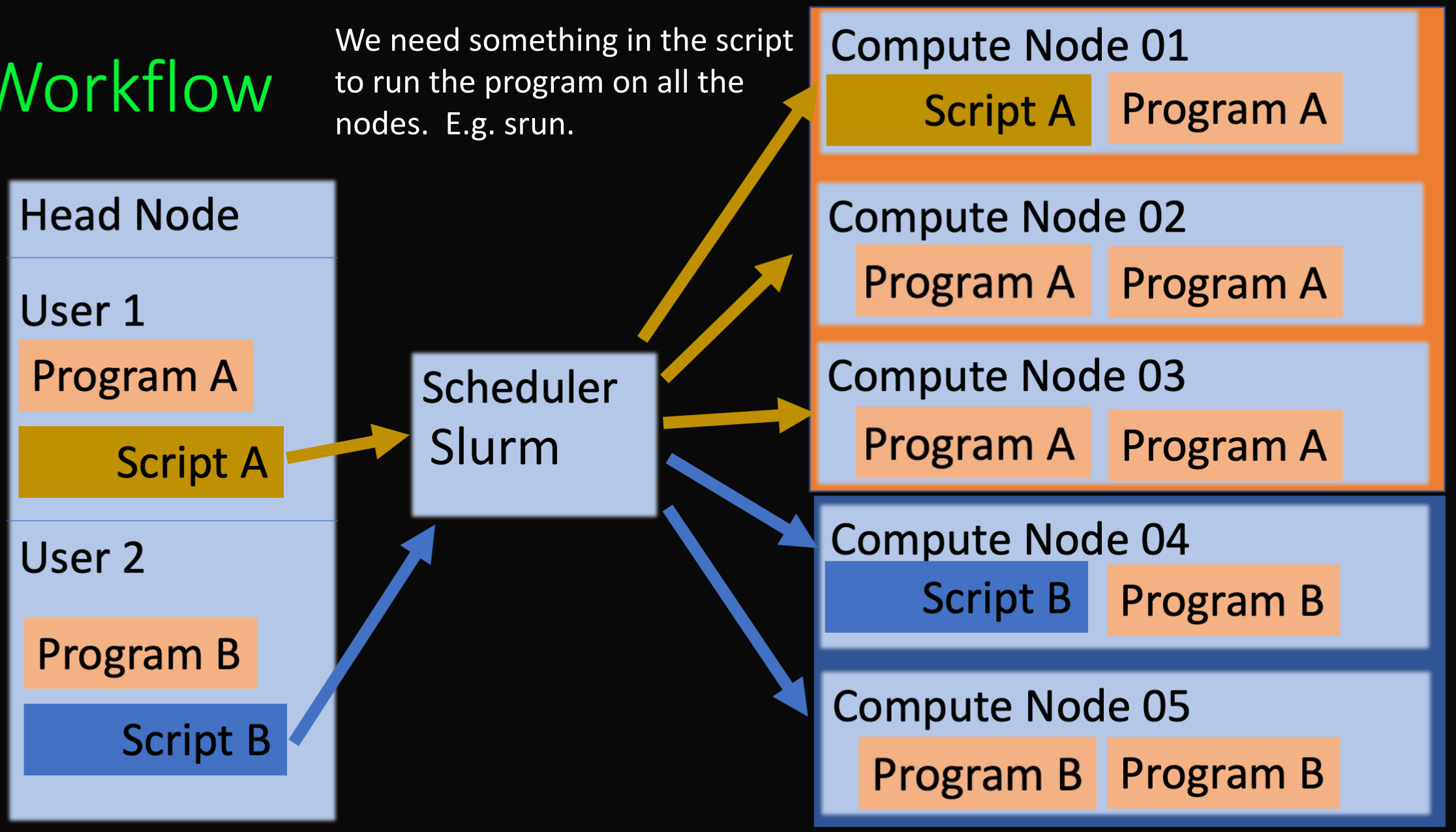
Workflow



Shared filesystems – All nodes can access the same programs and write output

Workflow

We need something in the script to run the program on all the nodes. E.g. srun.



Shared filesystems – All nodes can access the same programs and write output

```
vanilla@easley:~/workshops/intro_workshop$ ls
code  conda_envs  data  pbs  slurm  slurm-24898.out
```

The **hostname** command is very fast so everyone's job should finish in a few seconds.

When it is finished you will have a new file named slurm-{your job id}.out.

```
vanilla@easley:~/workshops/intro_workshop$ ls
code  conda_envs  data  pbs  slurm  slurm-24898.out
```

The **hostname** command is very fast so everyone's job should finish in a few seconds.

When it is finished you will have a new file named slurm-{your job id}.out.

```
[vanilla@hopper intro_workshop]$ ls
code  data  pbs  slurm  slurm-5252.out
```

When it is finished you will have a new file named slurm-{your job id}.out.

```
vanilla@easley:~/workshops/intro_workshop$ cat slurm-24898.out
To request GPUs, add --gpus-per-node X or --gpus X, where X is the desired nu
Job 24898 running on easley018
easley018
easley018
easley018
easley018
easley018
```

The output that would have gone to screen is saved to a file instead

Search ⌘ + K

Contents

Modules for Experiments in Stellar Astrophysics

Modules for Experiments in Stellar Astrophysics (MESA) is an advanced, open-source 1D stellar evolution code. The code is developed and maintained by a globally distributed team of researchers. MESA is written primarily in Fortran with a modular, flexible design that facilitates easy interfacing.

🚀 New to MESA? Start with the Quickstart guide to set up and run your first models.

📖 Experienced users may find the Reference and Defaults page most useful for exploring the code's features and input parameters. The Test Suite provides example problems for testing and showcasing MESA, serving as a project starting point but often requiring modifications for science-grade accuracy.

Check out the full documentation below.

Contents

https://docs.mesastar.org

Mesa Star

```
vanilla@easley:~/workshops/intro_workshop$ cd ..
vanilla@easley:~/workshops$
```

Mesa Star – searching for software on Easley

```
vanilla@easley:~/workshops/intro_workshop$ cd ..
vanilla@easley:~/workshops$
```

```
vanilla@easley:~/workshops$ module spider mesastar

----------------------------------------------------------------------------
  mesastar: mesastar/24.08.1
----------------------------------------------------------------------------

    This module can be loaded directly: module load mesastar/24.08.1

    Help:
      MESA STAR
```

Mesa Star – loading software on Easley

```
vanilla@easley:~/workshops$ module load mesastar
You are on the easley head node. Be sure to run
MESA STAR on a compute node.
E.g srun --partition debug ~/mesastar/tutorial/rn
```

Loading a module gives us access to that software

Mesa Star – loading software on Easley

```
vanilla@easley:~/workshops$ cd mesastar
```

```
vanilla@easley:~/workshops/mesastar$ ls
mesa_plotting_example.ipynb  test.slurm  tutorial
```

I've setup an example mesastar project in the "mesastar" directory with an example slurm script to run the project and a jupyter notebook for plotting.

Mesa Star – loading software on Easley

```
vanilla@easley:~/workshops$ cd mesastar/

vanilla@easley:~/workshops/mesastar$ ls
mesa_plotting_example.ipynb  test.slurm  tutorial

vanilla@easley:~/workshops/mesastar$ ls tutorial
clean                 inlist_pgstar   make     re           src
history_columns.list  inlist_project  mk       README.rst   star
inlist                LOGS            photos   rn
vanilla@easley:~/workshops/mesastar$
```

Project layout. The code used in this example is FORTRAN.

•**Inlist -** A top-level control file that typically includes or references other inlists (like inlist_project, inlist_pgstar). Sets global controls (e.g. paths, physics options, run controls).

•**Inlist_project**
Controls the physics and numerical setup of the star (e.g. initial mass, metallicity, stop conditions). Most important for the physical simulation itself.

•**inlist_pgstar -** Configures PGstar, the interactive graphical plotting module of MESA. Determines which plots are shown during runtime, layout, colours, etc.

There a some commands to make compiling and recompiling the code simple

To compile the code run

./mk

To clean and recompile

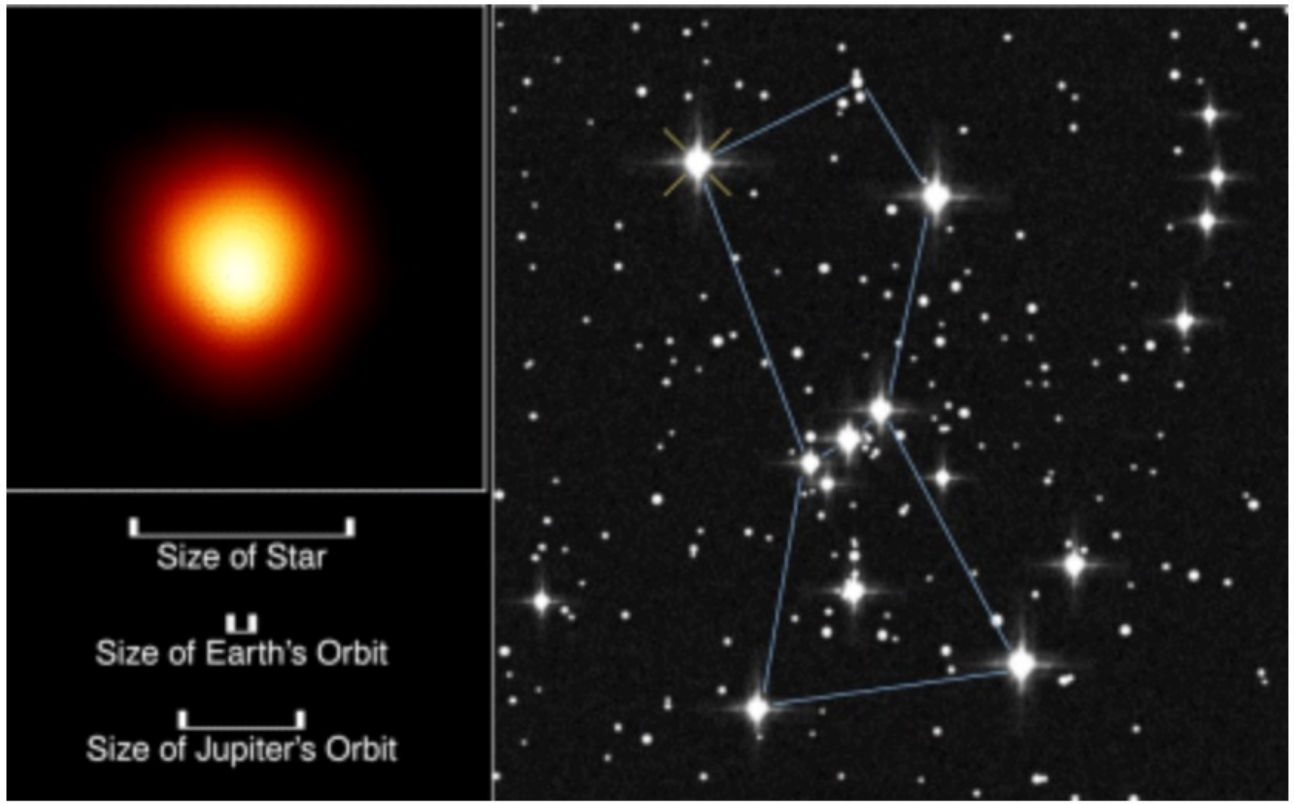
./clean
./mk

The model:

15 M⊙ Stellar Model Overview
- **Initial Mass:** 15 solar masses (Msun)
- **Metallicity (Z):** 0.02 (solar)
- **Start Phase:** Pre-main sequence
- **Opacities:** Type II opacities enabled (Z-dependent)
- **Equation of State:** Default MESA EOS

Termination Conditions
- Stop near **zero-age main sequence (ZAMS)**:
- When L_{\text{nuc}} / L > 0.99
- Stop when **central hydrogen** X_{H1} < 10^{-3}

Numerics and Output
- Uses 'dedt' energy equation for conservation
- Uses gold solver tolerances for accuracy



Atmosphere of Betelgeuse - Alpha Orionis

Hubble Space Telescope - Faint Object Camera

January 15, 1996; A. Dupree (CfA), NASA, ESA

Mesa Star – Compiling the Example Project

```
vanilla@easley:~/workshops/mesastar$ cd tutorial/

vanilla@easley:~/workshops/mesastar/tutorial$ ./mk
gfortran -Wno-uninitialized -fno-range-check -fmax-errors=7 -fdiagnostics-col
gfortran -Wno-uninitialized -fno-range-check -fmax-errors=7 -fdiagnostics-col
gfortran -Wno-uninitialized -fno-range-check -fmax-errors=7 -fdiagnostics-col
gfortran -fopenmp -o ../star  run_star_extras.o run_star.o  run.o  -L/opt/loc
/opt/local/mesastar/mesasdk/bin/../lib/gcc/x86_64-pc-linux-gnu/13.3.0/../../.
vanilla@easley:~/workshops/mesastar/tutorial$
```

Compile the mesastar example.
Recall . Is the shortcut for the current directory

Mesa Star – Compiling the Example Project

```
vanilla@easley:~/workshops/mesastar/tutorial$ ls
clean                  inlist_pgstar   make     re           src
history_columns.list   inlist_project  mk       README.rst   star
inlist                 LOGS            photos   rn
```

We have compiled a new program called “star” will take the inlist files as input. rn will call star.

You can modify the FORTRAN source code for more advanced analysis and recompile star.

Mesa Star – Running interactively

```
vanilla@easley:~/workshops/mesastar/tutorial$ srun --partition interactive rn
To request GPUs, add --gpus-per-node X or --gpus X, where X is the desired numb
Job 24903 running on easley057
DATE: 2025-08-28
TIME: 08:32:47
 reading user weak rate file /opt/local/mesastar/24.08.1/data/rates_data/rate_t
 version_number r24.08.1
 read inlist_project
```

Run the analysis interactively with srun.
This only uses one CPU, but mesastar can use many to run faster.

Mesa Star – Running in Batch Mode

```
vanilla@easley:~/workshops/mesastar/tutorial$ cd ..
vanilla@easley:~/workshops/mesastar$ cat test.slurm
#!/bin/bash
#SBATCH --job-name mesa_run
#SBATCH --partition debug
#SBATCH --cpus-per-task 8
#SBATCH --time=00:05:00
#SBATCH --output=mesa_%j.out
#SBATCH --error=mesa_%j.err
#SBATCH --mem 32G
module load mesastar

# Set OpenMP environment variables
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
export OMP_DISPLAY_ENV=TRUE

cd tutorial

# Run MESA
srun ./rn
```

Mesa Star – Running in Batch Mode

```
vanilla@easley:~/workshops/mesastar$ sbatch test.slurm
Submitted batch job 24904

vanilla@easley:~/workshops/mesastar$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(RE
             24904     debug mesa_run  vanilla  R       0:53      1 easley018

vanilla@easley:~/workshops/mesastar$ ls
mesa_24904.err  mesa_plotting_example.ipynb  tutorial
mesa_24904.out  test.slurm
```

Mesa Star – Running in Batch Mode

```
vanilla@easley:~/workshops/mesastar$ sbatch test.slurm
Submitted batch job 24904
vanilla@easley:~/workshops/mesastar$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(RE
             24904     debug mesa_run  vanilla  R       0:53      1 easley018
vanilla@easley:~/workshops/mesastar$ ls
mesa_24904.err  mesa_plotting_example.ipynb  tutorial
mesa_24904.out  test.slurm
vanilla@easley:~/workshops/mesastar$ tail mesa_24904.out
  lg_dt_yrs    lg_Tcntr    lg_R     lg_L3a     lg_Lneu     lg_Mdot    He_core
    age_yrs    lg_Dcntr    lg_L     lg_LZ      lg_Lphoto   lg_Dsurf   CO_core
------------------------------------------------------------------------------

        207   7.533310  3.078E+04   4.293451   4.293451  15.000000  15.000000
 3.9013E+00   7.533310   0.690863 -28.971924   3.137050 -99.000000   0.000000
 9.0641E+04   0.787282   4.289535 -11.439140 -99.000000  -8.905833   0.000000

DATE: 2025-08-28
TIME: 08:41:45
```

Mesa Star – Output files

```
vanilla@easley:~/workshops/mesastar$ ls tutorial/LOGS/
history.data  profile1.data  profile3.data  profile5.data  profiles.index
pgstar.dat    profile2.data  profile4.data  profile6.data
```

History tracks the star's properties over time (e.g. luminosity, temperature, age).

Profile captures the star's internal structure at a single timestep (e.g. composition, temperature, pressure vs. radius).

Mesa Star – Longer run (at least an hour to run simulating 12 Myr+)

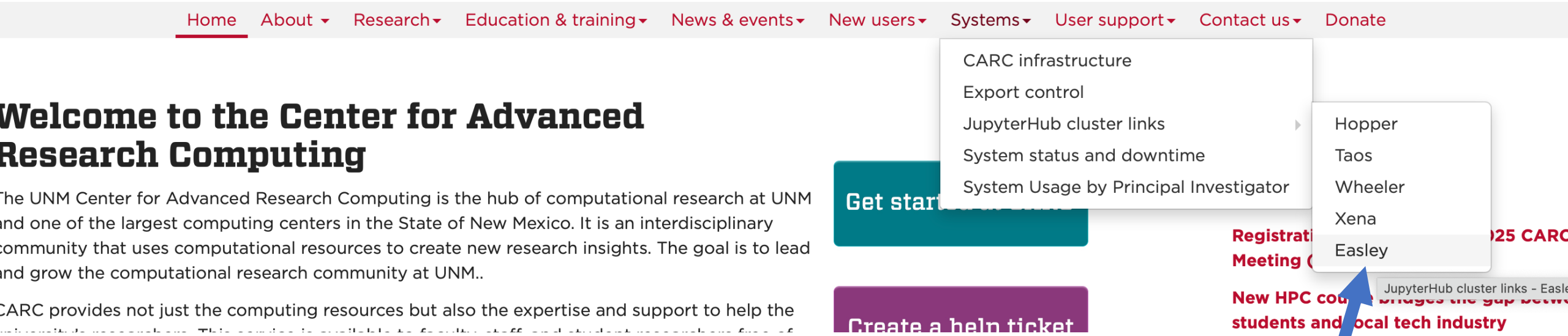
```
vanilla@easley:~/workshops/mesastar$ sbatch test_long.slurm
Submitted batch job 24904
vanilla@easley:~/workshops/mesastar$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(RE
             24904     debug starsim vanilla  R       0:53      1 easley018
vanilla@easley:~/workshops/mesastar$ ls
mesa_24905.err  mesa_plotting_example.ipynb  tutorial
mesa_24905.out  test.slurm
vanilla@easley:~/workshops/mesastar$ tail mesa_24905.out
  lg_dt_yrs    lg_Tcntr    lg_R     lg_L3a     lg_Lneu     lg_Mdot    He_core
    age_yrs    lg_Dcntr    lg_L     lg_LZ      lg_Lphoto   lg_Dsurf   CO_core
------------------------------------------------------------------------------

        207   7.533310  3.078E+04   4.293451   4.293451  15.000000  15.000000
 3.9013E+00   7.533310   0.690863 -28.971924   3.137050 -99.000000   0.000000
 9.0641E+04   0.787282   4.289535 -11.439140 -99.000000  -8.905833   0.000000

DATE: 2025-08-28
TIME: 08:41:45
```

JupyterHub for Plotting – Installing matplotlib

```
vanilla@easley:~ $ module load miniconda3
This Miniconda3 installation uses the conda-forge channel. Conda-forge packages
are generally open-source and free for research and educational use. Be sure to
review the license agreements of software packages you install and additional
channels you enable.

vanilla@easley:~ $ conda create --name mesa_plot matplotlib pandas ipykernel
Retrieving notices: done
Channels:
 - conda-forge
 - nodefaults
Platform: linux-64
Collecting package metadata (repodata.json): /
```

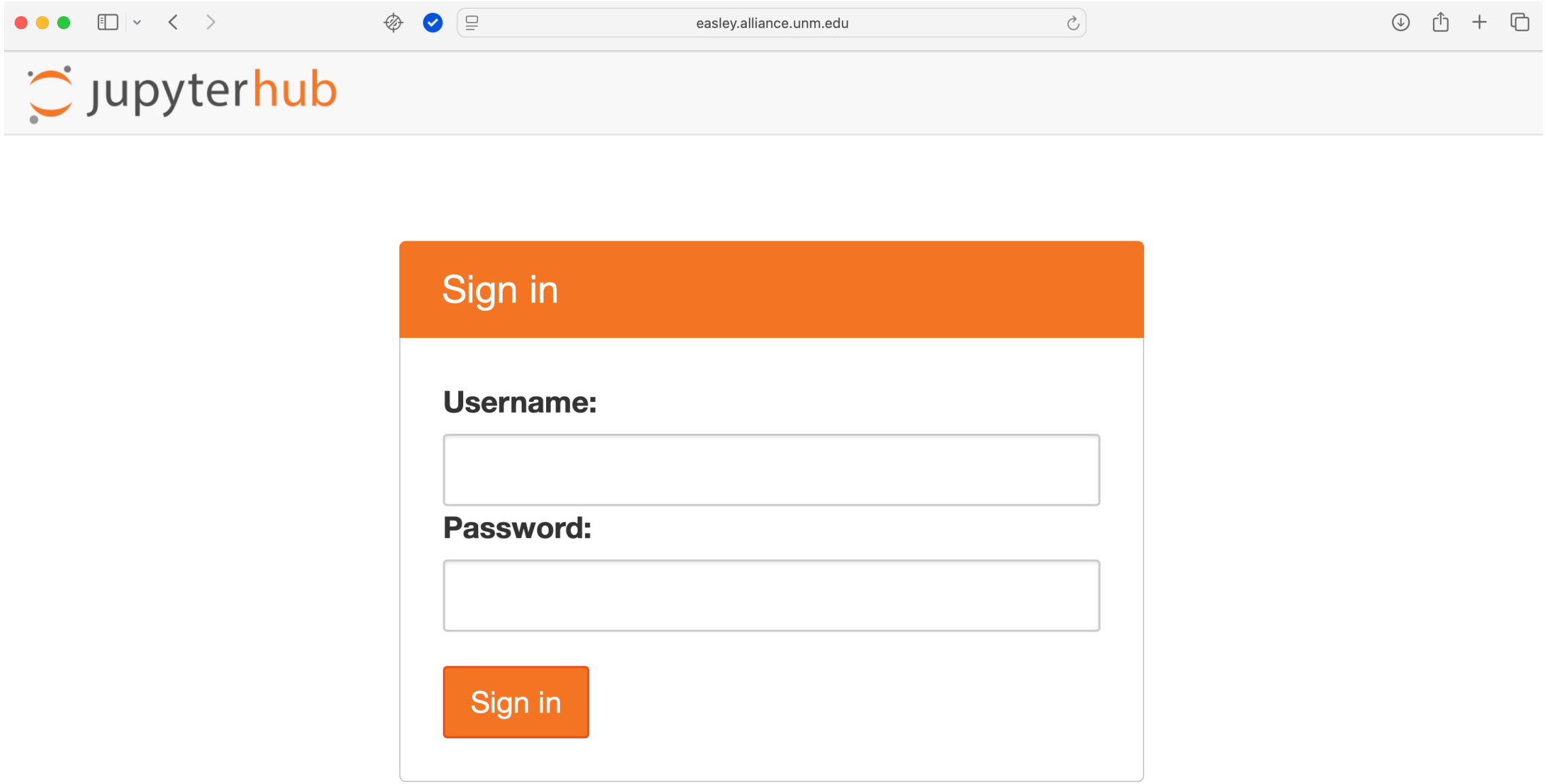
You may be prompted to add conda information to your BASH login script with the commands
conda init bash
source ~/.bashrc

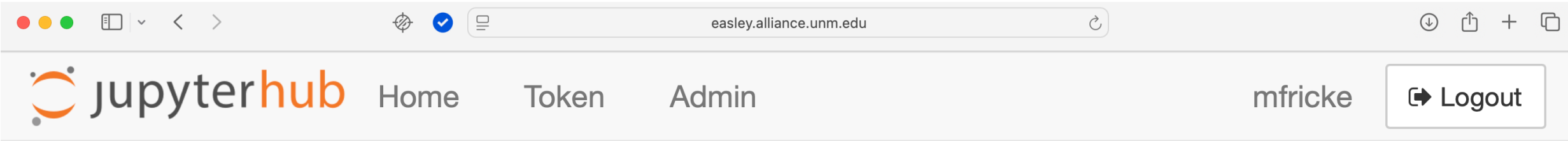
http://carc.unm.edu



python™

julia


easley.alliance.unm.edu
jupyterhub
Sign in
Username:
Password:
Sign in



Server Options

Select a job profile:

Interactive Queue, 1 hours, 1 core, 4GB RAM

Start

The Jupyter Notebook is running as a batch job under Slurm

```
vanilla@easley:~/workshops/mesastar$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(RE
             24910 interacti jupyterh  vanilla  R       0:17      1 easley057
vanilla@easley:~/workshops/mesastar$
```

vtst_src	4 years ago
working	6 years ago
workshops	12 hours ago
workshops2	a year ago
workshops3	3 years ago
workshops4	2 years ago
workshops_old	12 hours ago
workshops_old2	6 months ago
workshops_old3	12 hours ago
workshops_prev	a year ago
workshops_quantum	5 months ago

jupyterhub

Logout | Control Panel

crystallography	12 hours ago	
escape	12 hours ago	
gaussian	12 hours ago	
intro_workshop	an hour ago	
machine_learning	12 hours ago	
mesastar	27 minutes ago	
namd	12 hours ago	
quantum_computing	12 hours ago	
radio_astronomy	12 hours ago	
starccm	12 hours ago	
README.md	12 hours ago	47 B

easley.alliance.unm.edu

jupyterhub

Logout | Control Panel

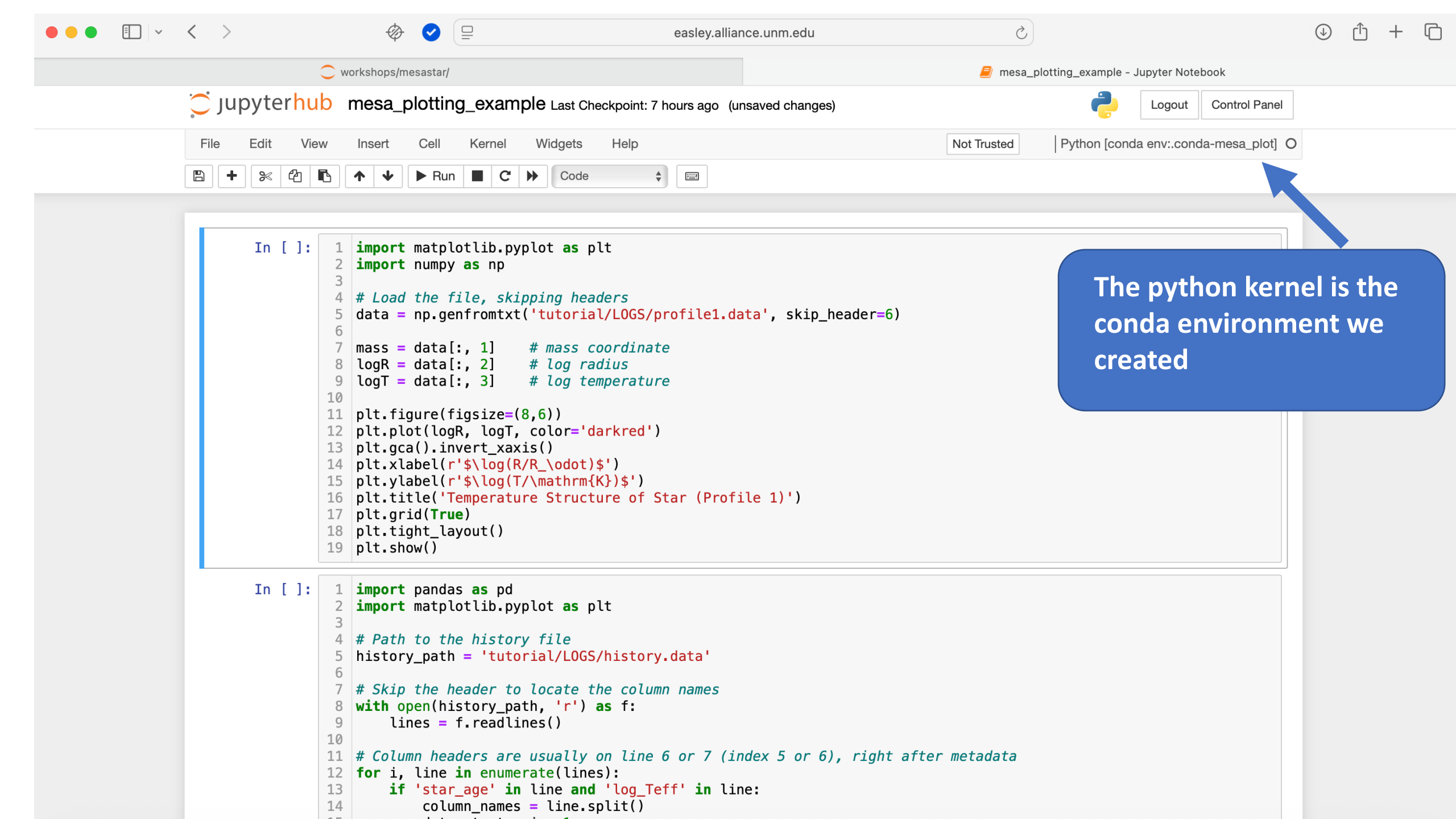
Files | Running | Clusters

Select items to perform actions on them.

Upload | New

0 / workshops / mesastar

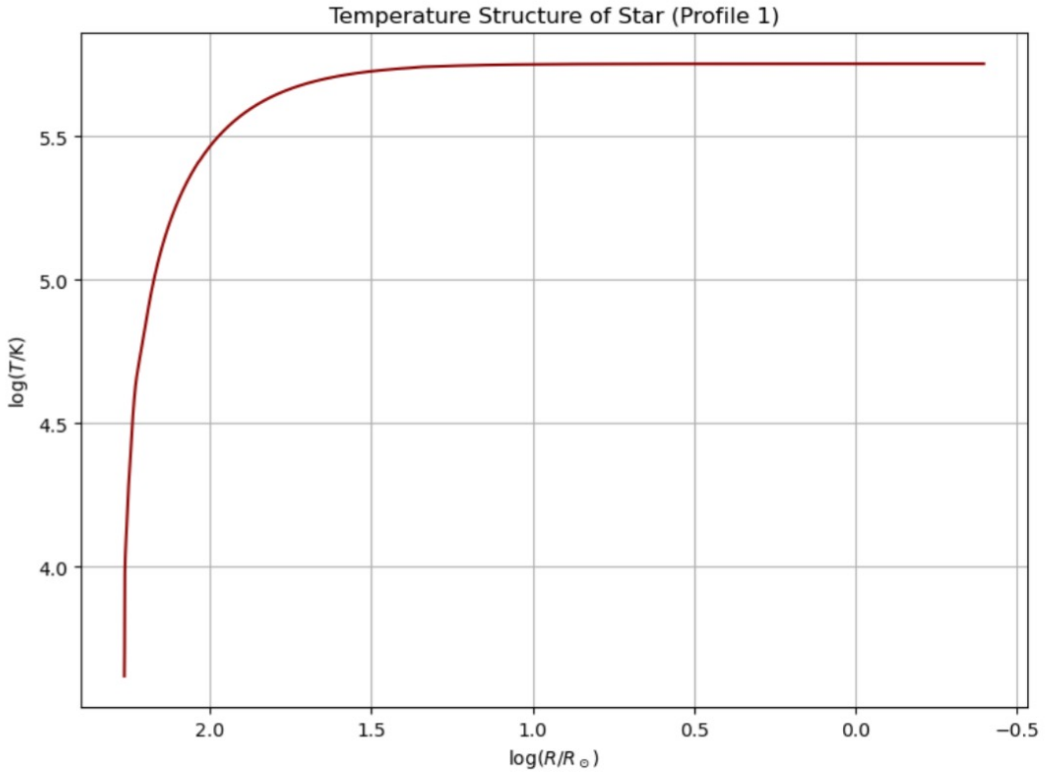
	Name	Last Modified	File size
..		seconds ago	
tutorial		43 minutes ago	
mesa_plotting_example.ipynb		7 hours ago	74.6 kB
mesa_24904.err		28 minutes ago	776 B
mesa_24904.out		27 minutes ago	265 kB
test.slurm		32 minutes ago	357 B


easley.alliance.unm.edu
workshops/mesastar/
mesa_plotting_example - Jupyter Notebook
jupyterhub
mesa_plotting_example
Last Checkpoint: 7 hours ago
(unsaved changes)
Logout
Control Panel
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python [conda env:.conda-mesa_plot]
Run
Code
In [ ]:
import matplotlib.pyplot as plt
import numpy as np
# Load the file, skipping headers
data = np.genfromtxt('tutorial/LOGS/profile1.data', skip_header=6)
mass = data[:, 1] # mass coordinate
logR = data[:, 2] # log radius
logT = data[:, 3] # log temperature
plt.figure(figsize=(8,6))
plt.plot(logR, logT, color='darkred')
plt.gca().invert_xaxis()
plt.xlabel(r'$\log(R/R_\odot)$')
plt.ylabel(r'$\log(T/\mathrm{K})$')
plt.title('Temperature Structure of Star (Profile 1)')
plt.grid(True)
plt.tight_layout()
plt.show()
In [ ]:
import pandas as pd
import matplotlib.pyplot as plt
# Path to the history file
history_path = 'tutorial/LOGS/history.data'
# Skip the header to locate the column names
with open(history_path, 'r') as f:
lines = f.readlines()
# Column headers are usually on line 6 or 7 (index 5 or 6), right after metadata
for i, line in enumerate(lines):
if 'star_age' in line and 'log_Teff' in line:
column_names = line.split()
The python kernel is the conda environment we created

Temperature Profile Plot: Python script reads a MESA profile.data file and plots stellar temperature vs. radius (log T vs. log R), revealing the star's internal temperature gradient.

In [2]:

```python
import matplotlib.pyplot as plt
import numpy as np

# Load the file, skipping headers
data = np.genfromtxt('tutorial/LOGS/profile1.data', skip_header=6)

mass = data[:, 1]    # mass coordinate
logR = data[:, 2]    # log radius
logT = data[:, 3]    # log temperature

plt.figure(figsize=(8,6))
plt.plot(logR, logT, color='darkred')
plt.gca().invert_xaxis()
plt.xlabel(r'$\log(R/R_\odot)$')
plt.ylabel(r'$\log(T/\mathrm{K})$')
plt.title('Temperature Structure of Star (Profile 1)')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Hertzsprung–Russell Diagram

Protostar phase: first 90 kyr of contraction toward the main sequence. From MESA's history.data file: tutorial/LOGS/history.data.

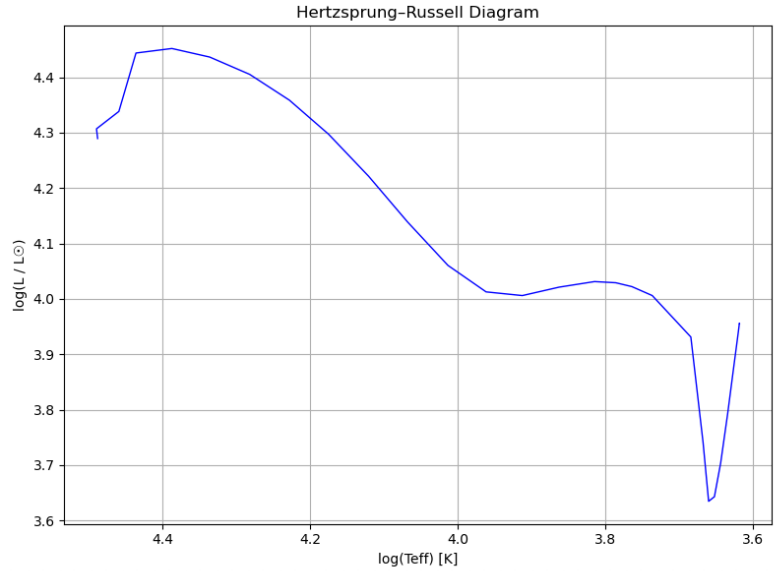
File Edit View Insert Cell Kernel Widgets Help Trusted Python [conda env:.conda-mesa_plot]

Run Code

In [3]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Path to the history file
history_path = 'tutorial/LOGS/history.data'

# Skip the header to locate the column names
with open(history_path, 'r') as f:
    lines = f.readlines()

# Column headers are usually on line 6 or 7 (index 5 or 6), right after metadata
for i, line in enumerate(lines):
    if 'star_age' in line and 'log_Teff' in line:
        column_names = line.split()
        data_start = i + 1
        break

# Read the table using pandas
df = pd.read_csv(history_path, skiprows=data_start, delim_whitespace=True, names=co

# Plot H-R diagram
plt.figure(figsize=(8, 6))
plt.plot(df['log_Teff'], df['log_L'], color='blue', linewidth=1)
plt.gca().invert_xaxis()  # Hotter (higher T) on the left
plt.xlabel('log(Teff) [K]')
plt.ylabel('log(L / L⊙)')
plt.title('Hertzsprung–Russell Diagram')
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
/tmp/ipykernel_2184284/412598046.py:19: FutureWarning: The 'delim_whitespace' keyword
in pd.read_csv is deprecated and will be removed in a future version. Use ``sep='\s+'`
` instead
  df = pd.read_csv(history_path, skiprows=data_start, delim_whitespace=True, names=col
umn_names)
```



Hertzsprung–Russell Diagram
(from history file under tutorial_long)

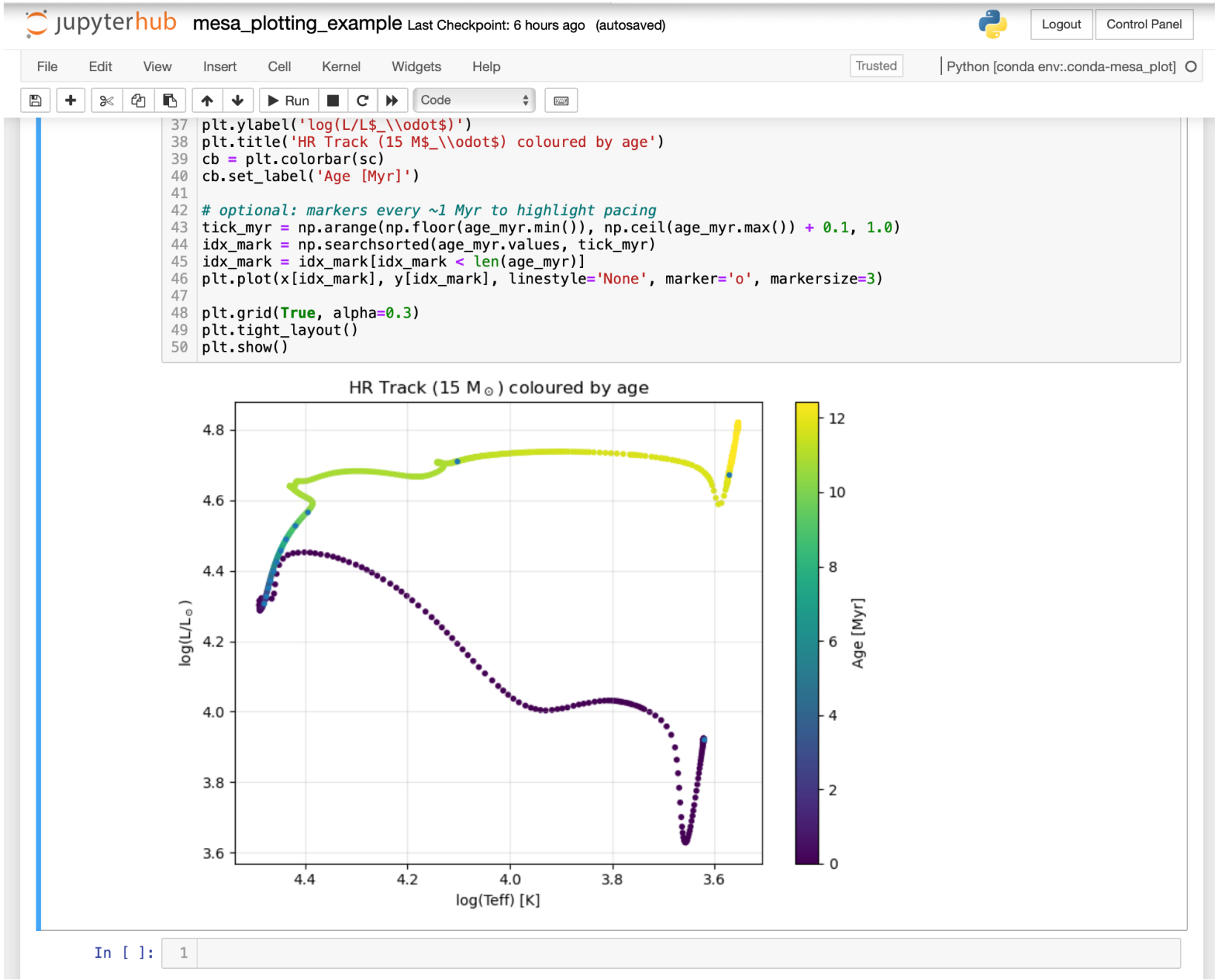
- **Pre–Main Sequence (Pre-MS)**: 0–0.1 Myr – star contracting before sustained fusion

- **Zero-Age Main Sequence (ZAMS)**: ≈0.1 Myr – hydrogen fusion ignites in the core

- **Main Sequence (MS)**: 0.1–10 Myr – stable core hydrogen burning

- **Terminal-Age Main Sequence (TAMS)**: 10–11 Myr – core hydrogen exhausted

- **Red Supergiant (RSG), Helium burning**: 11–12 Myr – star expands and ignites helium in the core



Hertzsprung–Russell Diagram
(from history file under tutorial_long)

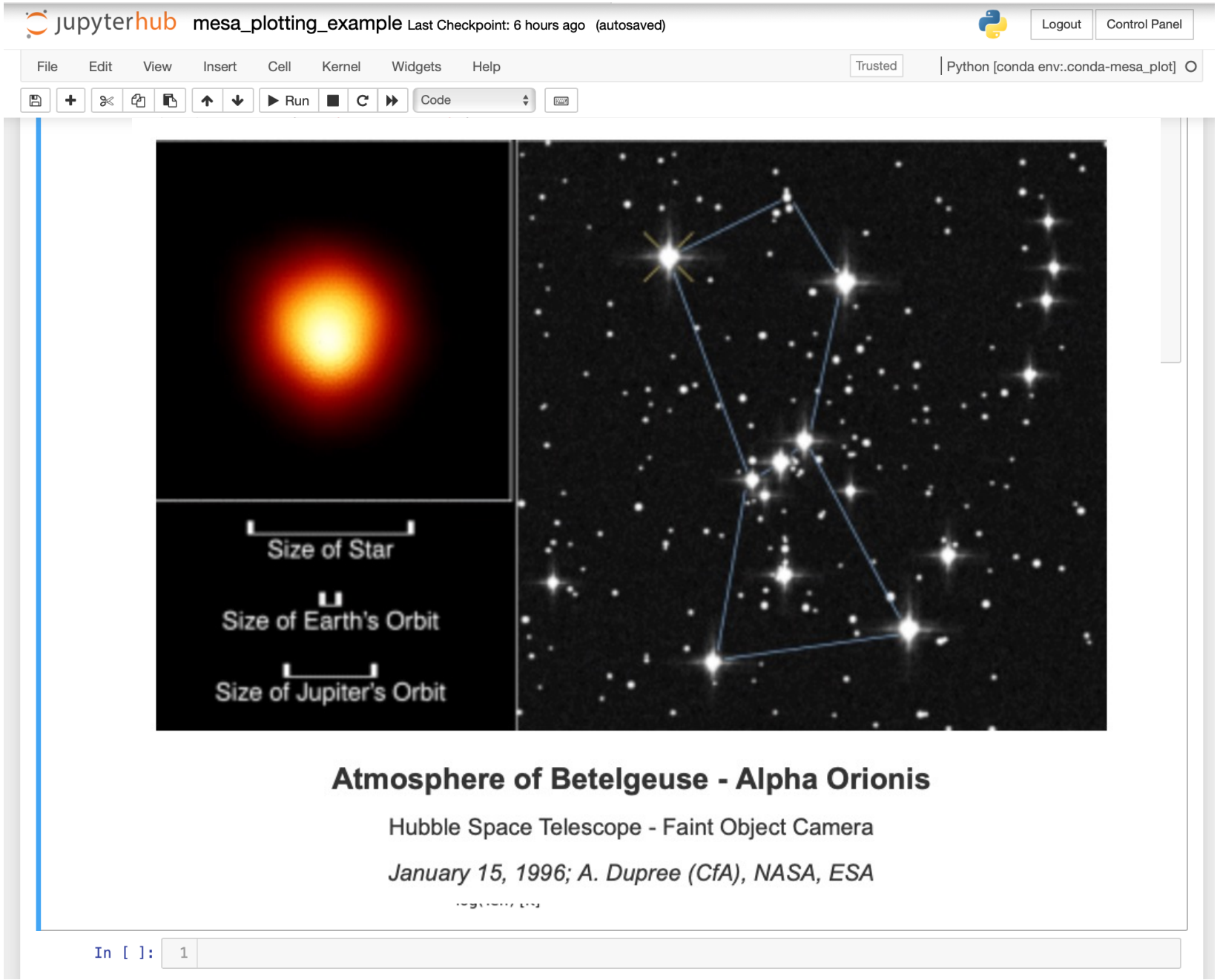
- **Pre–Main Sequence (Pre-MS)**: 0–0.1 Myr – star contracting before sustained fusion

- **Zero-Age Main Sequence (ZAMS)**: ≈0.1 Myr – hydrogen fusion ignites in the core

- **Main Sequence (MS)**: 0.1–10 Myr – stable core hydrogen burning

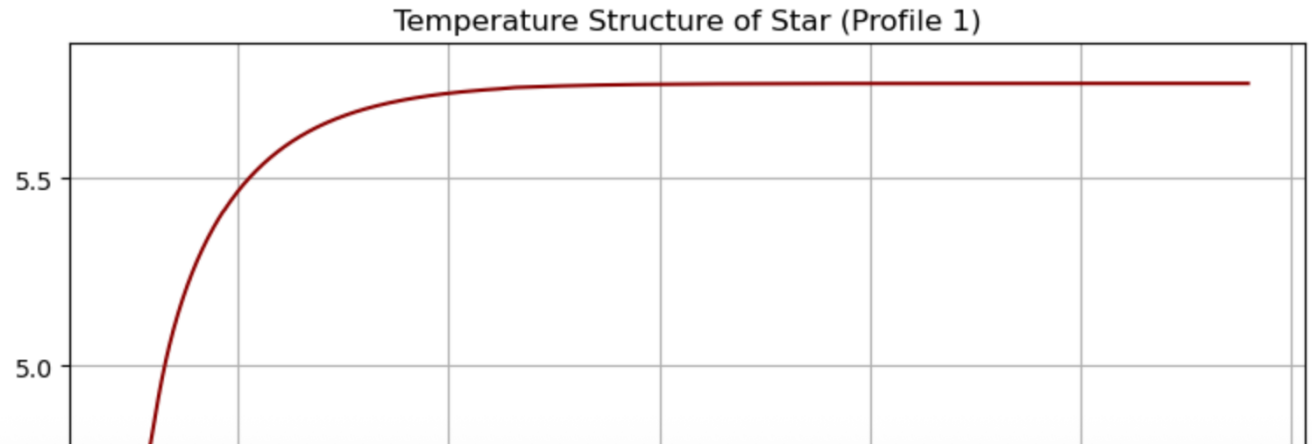
- **Terminal-Age Main Sequence (TAMS)**: 10–11 Myr – core hydrogen exhausted

- **Red Supergiant (RSG), Helium burning**: 11–12 Myr – star expands and ignites helium in the core



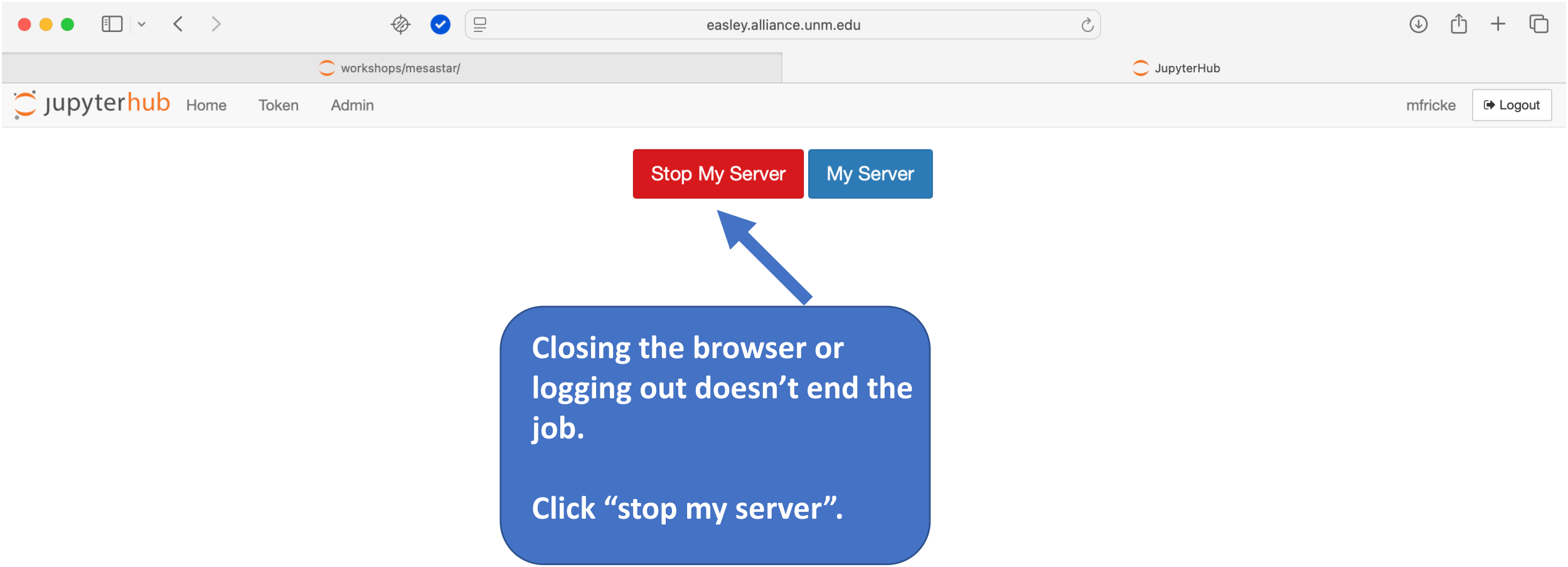
In [1]:

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  # Load the file, skipping headers
5  data = np.genfromtxt('tutorial/LOGS/profile1.data', skip_header=6)
6
7  mass = data[:, 1]    # mass coordinate
8  logR = data[:, 2]    # log radius
9  logT = data[:, 3]    # log temperature
10
11 plt.figure(figsize=(8,6))
12 plt.plot(logR, logT, color='darkred')
13 plt.gca().invert_xaxis()
14 plt.xlabel(r'$\log(R/R_\odot)$')
15 plt.ylabel(r'$\log(T/\mathrm{K})$')
16 plt.title('Temperature Structure of Star (Profile 1)')
17 plt.grid(True)
18 plt.tight_layout()
19 plt.show()
```



When finished we should relinquish the hardware.

Closing the browser or logging out doesn't end the job.


easley.alliance.unm.edu
workshops/mesastar/
JupyterHub
jupyterhub
Home
Token
Admin
mfricke
Logout
Stop My Server
My Server
Closing the browser or logging out doesn't end the job.
Click "stop my server".

Useful Slurm Commands

squeue --me --long	shows information about jobs you submitted
squeue --me --start	shows when slurm expects your job to start
scancel jobid	cancels a job
scancel --u $USER	cancels all your jobs
sacct	shows your job history
seff jobid	shows how efficiently the hardware was used

You have learned

- how to run programs using the SLURM scheduler
- the difference between interactive and batch jobs
- how to check the status of your jobs
- how to ask for the hardware resources you need
- you ran MesaStar interactively and in batch mode and plotted the results in Jupyterhub