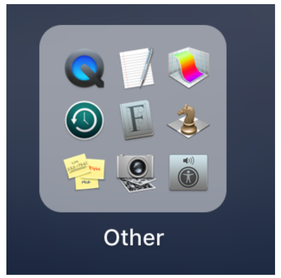
2/18/2020 Week 6 Module 1
Running molecular dynamics on a remote computer

- Useful molecular dynamics simulations and other scientific calculations often require too much computer power to run on a local laptop or desktop
- In this module, you will learn how to
  - log into a remote computer
  - install OpenMM on the remote computer
  - transfer data to and from a remote computer
  - submit a calculation to a queue on a computing cluster
- We will use remote computers belonging to XSEDE, the “Extreme Science and Engineering Discovery Environment”, an organization that manages computing resources for open science throughout the United States

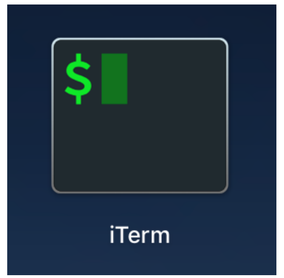
Review: The UNIX Terminal

- Predecessor and very similar to the terminal used in
  - LINUX - many scientific computing clusters
  - Mac OS X - this lab and many of your own laptops/computers
- Useful for
  - accessing programs that run with a command line interface (CLI)
  - automation

- To start the terminal on a Mac:



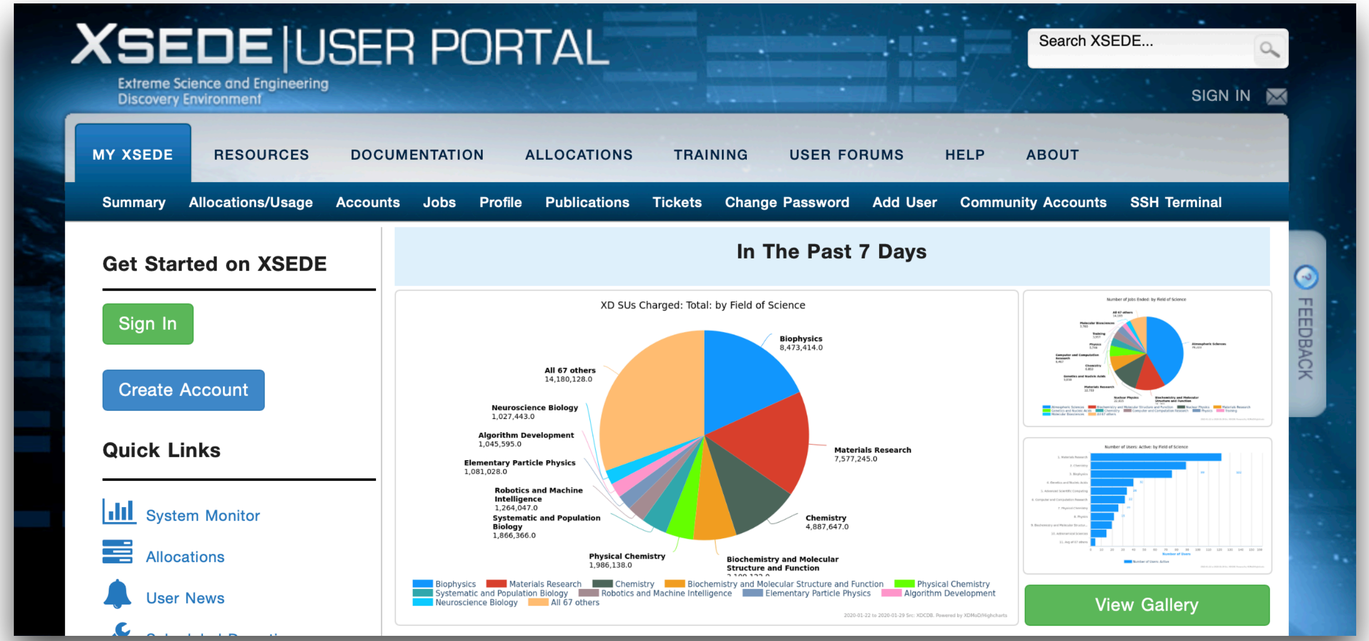
- I actually usually use another program:



Review: UNIX Terminal Essentials

- Try the following commands
  - echo $SHELL - echo reports the value of a variable. $SHELL is the terminal interface you are using, and will affect the details of all other commands. bash is a popular shell.
  - File operations
    - ls – list files and directories
    - cp – copy files
    - rm – remove files and directories
    - mv – rename or move files and directories to another location
  - Directory management
    - pwd – confirm current directory
    - cd – change directory
    - mkdir – make new directory
    - rmdir – remove directory
- Also see
  - https://www.unixtutorial.org/basic-unix-commands
  - An Introduction to Linux (https://www.youtube.com/watch?v=IVquJh3DXUA). Work with the terminal starts at 3:35.

- A lot of what we will work through today is covered in the“Getting started with XSEDE” tutorial.
- This can be found at http://portal.xsede.org/. Click on the tabs marked “Documentation” and “Get Started”.
- You should already have
  - an XSEDE account and access to the class allocation
  - signed up for Multi-Factor Authentication with Duo (https://portal.xsede.org/mfa)
- So you can start from “Login to your Allocated Resources” on the left pane.



Cluster computing

On 7/22/2019, your professor visited the
Texas Advanced Computing Center (TACC),
home of an XSEDE computing cluster, Stampede2.

- The XSEDE supercomputers are not especially fast computers, but are computing clusters, groups of computing nodes that work together
- The most common way for computational scientists to utilize these computers is to submit a batch job to the queue
  - a batch job contains
    - information about the job, e.g.
      - allocation
      - maximum duration
      - type and amount of resource, e.g. CPUs or GPUs
    - a series of commands, like if you typed them into the terminal
  - The queue
    - is usually managed by a master node
    - prioritizes jobs
    - distributes jobs to worker nodes

Logging In

- We will follow the “Single Sign On via SSH Login Hub”, so you can enter
  - ssh username@login.xsede.org
  - your XUP passcode
  - then ask for a push to your phone
  - more information is at https://portal.xsede.org/web/xup/single-sign-on-hub
- From the login portal, you can log into bridges/comet by entering
  - `gsissh bridges’ or
  - `gsissh comet’
- To log out of a remote computer, press control-D

```
Minh-IIT-MBP2018:[~]: ssh daveminh@login.xsede.org
Please login to this system using your XSEDE username and password:
password:
Duo two-factor login for daveminh

Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-4139
 2. Phone call to XXX-XXX-4139

Passcode or option (1-2): 1
Success. Logging you in...
Last login: Sat Feb  1 13:41:10 2020 from 207.237.205.39

#  Welcome to the XSEDE Single Sign-On (SSO) Hub!
#
#  This system is for use by authorized users only, and is subject to the XSEDE
#  Acceptable Use Policy, described at https://www.xsede.org/usage-policies.
#  All activities on this system may be monitored and logged.
#
#  Your storage on this system is limited to 100MB. Backup is not provided.
#
#  From this system, you may login to other XSEDE system login hosts on which
#  you currently have an active account. To see a list of your accounts, visit:
#  https://portal.xsede.org/group/xup/accounts
#
#  To login to an XSEDE system login host, enter: gsissh <login-host>
#  where <login-host> is the hostname, alias or IP address of the login host.
#  The following default gsissh host aliases have been defined:
#
#           bridges  comet  mason  osg  rmacc-summit  stampede
#           stampede2  supermic  wrangler-iu  wrangler-tacc
#
#  For example, to login to the Comet system at SDSC, enter: gsissh comet
#
#  E-mail help@xsede.org if you require assistance in the use of this system.

[daveminh@ssohub ~]$ gsissh comet
Last login: Sat Feb  1 10:46:54 2020 from 149.165.168.51
Rocks 7.0 (Manzanita)
Profile built 13:03 03-Dec-2019

Kickstarted 14:18 03-Dec-2019

                    WELCOME TO
      ______________  __  _____________
       -----/ ____/ __ \/  |/  / ____/_  __/
         --/ /   / / / / /|_/ / __/   / /
          / /___/ /_/ / /  / / /___  / /
          \____/\____/_/  /_/_____/ /_/

comet-ln3:[~]:
```

The Bridges File System

- As described in the Bridges user guide under “File Spaces”, Bridges allows you to store data in
  - your home directory ($HOME) - shared across nodes, backed up, smaller quota (10 GB)
  - a “scratch” directory ($SCRATCH) - shared across nodes, not backed up, faster than $HOME, larger quota shared by class
  - the worker node for a job ($LOCAL) - only accessible by local node, not backed up and immediately deleted after job, faster than shared disk space, many terabytes of available space
  - the local memory for a job ($RAMDISK) - only accessible by local node, not backed up and immediately deleted after job, fastest storage
  - $(VARIABLE_NAME) is a variable that contains the actual location. The variable is available while logged onto Bridges
- Try `cd $SCRATCH’, ‘pwd’, `cd $HOME’, and ‘pwd’

Installing OpenMM

- The easiest way to install OpenMM is through the conda package manager
- Conda is available on bridges, but needs to be loaded by entering `module load anaconda'
- Then you need to create an conda environment that you can write to by entering `conda create --name openmm'

```
br005:[~]: conda create --name openmm
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.7.12
  latest version: 4.8.1

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: /home/dminh/.conda/envs/openmm



Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate openmm
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

- To allow you to activate the environment, enter `conda init bash' and then `bash'
- Then to activate the environment, enter `conda activate openmm'
- Finally, to install OpenMM, enter
  - `conda install -c omnia -c conda-forge openmm'
  - `conda init bash'
- OpenMM may be tested using `python -m simtk.testInstallation'

```
(openmm) br006:[~]: conda install -c omnia -c conda-forge openmm
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.7.12
  latest version: 4.8.1

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: /home/dminh/.conda/envs/openmm

  added / updated specs:
    - openmm


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    _libgcc_mutex-0.1          |      conda_forge           3 KB  conda-forge
    _openmp_mutex-4.5          |            0_gnu         435 KB  conda-forge
    ca-certificates-2019.11.28 |       hecc5488_0         145 KB  conda-forge
    certifi-2019.11.28         |           py37_0         148 KB  conda-forge
    cython-0.29.14             |   py37he1b5a44_0         2.2 MB  conda-forge
    ld_impl_linux-64-2.33.1    |       h53a641e_8         589 KB  conda-forge
    libblas-3.8.0              |      14_openblas          10 KB  conda-forge
```

Transferring Data

- After logging into the XSEDE Single Sign-On Hub, you can transfer data
  - using various commands in the terminal, including rsync, scp, and sftp
  - using specialized programs, as specified in the user manuals, but this requires installing programs, which you might not want to do
- On the terminal, I recommend rsync because you don't need to copy (and wait for) the whole directory
- A good template is: rsync -Cuavz --port 2222 source_path destination_path
  - Cuavz are options, which can be seen by `man rsync'
  - port 2222 means that you go through the XSEDE SSO
  - the paths have the format user_name@directory/file_name
    - if user_name@ is omitted, it assumes the same user name as on the current system
    - if directory is omitted, the home directory is assumed
    - if file_name is omitted, the whole directory is assumed

- Try copying a local directory with the ubiquitin pdb file and python script for OpenMM

```
Minh-IIT-MBP2018:[~/Documents/GitHub/Chem456/static_files/tutorials]: rsync -Cua
vz --port 2222 ubq-md dminh@data.bridges.psc.edu:/home/dminh/
building file list ... done
ubq-md/
ubq-md/1ubq.pdb
ubq-md/MD_ubq.py

sent 14309 bytes  received 70 bytes  9586.00 bytes/sec
total size is 52573  speedup is 3.66
Minh-IIT-MBP2018:[~/Documents/GitHub/Chem456/static_files/tutorials]:
```

If you don't have them on your local computer, you can download them from github:
https://github.com/daveminh/Chem456/tree/master/static_files/tutorials/ubq-md

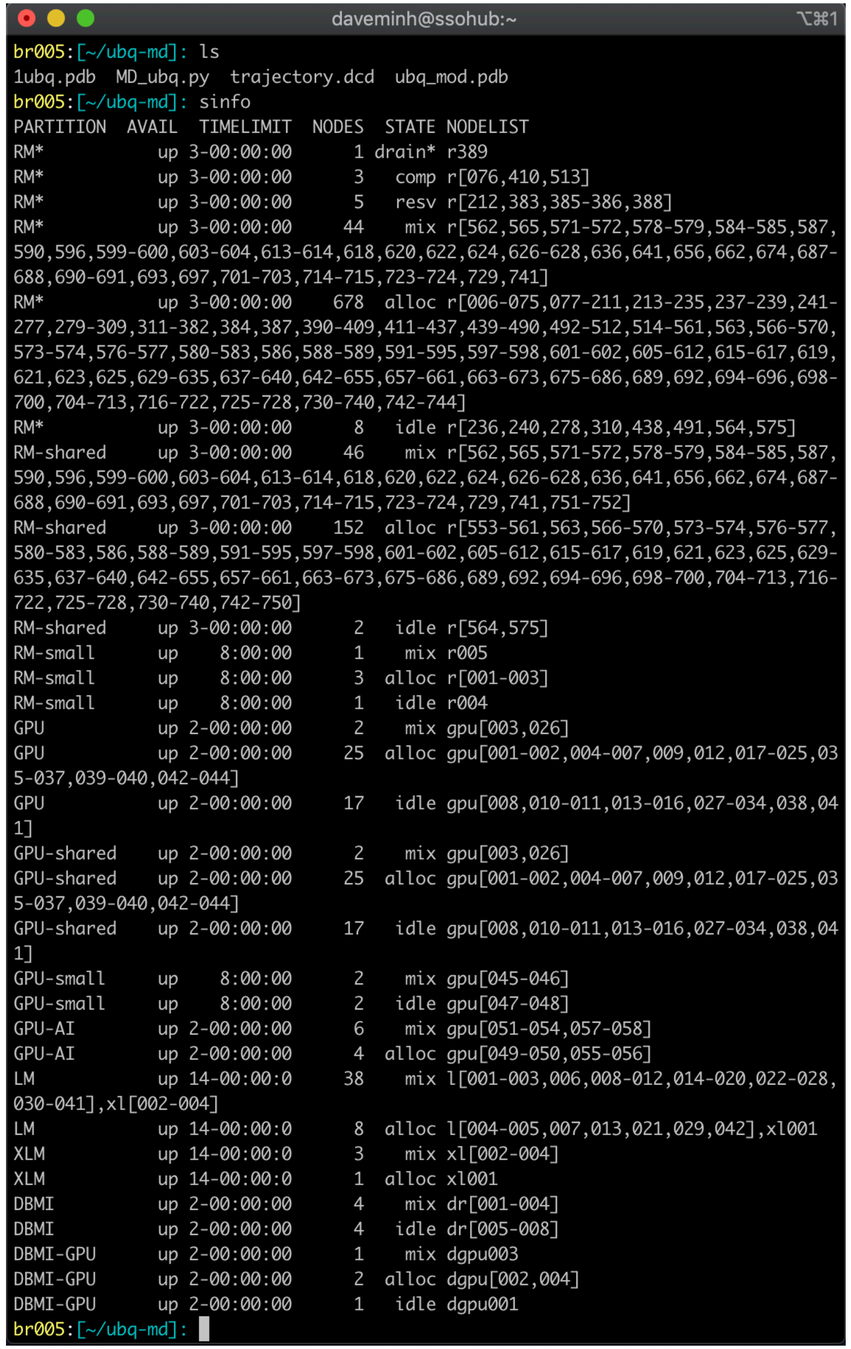
- Once you have installed OpenMM and copied the data, you can simply run the python script on the login terminal
- However, this is not the way that you're supposed to do things
- If you try to run a big calculation on the login terminal the system administrators will get mad at you!

```
(openmm) br006:[~]: python -m simtk.testInstallation

OpenMM Version: 7.4.1
Git Revision: 068f120206160d5151c9af0baf810384bba8d052

There are 2 Platforms available:

1 Reference - Successfully computed forces
2 CPU - Successfully computed forces

Median difference in forces between platforms:

Reference vs. CPU: 6.30481e-06

All differences are within tolerance.
(openmm) br006:[~]: ls
scripts  software  ubq-md
(openmm) br006:[~]: cd ubq-md/
(openmm) br006:[~/ubq-md]: ls
1ubq.pdb  MD_ubq.py
(openmm) br006:[~/ubq-md]: python MD_ubq.py
Minimizing...
Running Production...
#"Progress (%)" "Step"  "Potential Energy (kJ/mole)"    "Temperature (K)"
"Speed (ns/day)"        "Time Remaining"
10.0%   100     -12937.54104464231      178.8691182900329       0       --
20.0%   200     -12786.424474924024     206.88709731573758      21.9    0:06
30.0%   300     -12425.520220420438     214.78270910359848      21.8    0:05
40.0%   400     -12333.981042607265     242.67256593500656      21.8    0:04
50.0%   500     -12170.023875048062     252.83736423813997      21.8    0:03
60.0%   600     -12043.26256936375      262.77890676191413      21.7    0:03
70.0%   700     -12022.84441006196      274.6817644781113       21.8    0:02
80.0%   800     -11904.676295196514     279.1694256963495       21.8    0:01
90.0%   900     -11873.317454425185     274.09491413581435      21.7    0:00
100.0%  1000    -11727.128648981274     275.21880048454915      21.7    0:00
Done!
```

Submitting jobs to Bridges

- As described in the Bridges user guide under “Running Jobs”, you can submit a job to the cluster using a batch script
- Bridges uses the SLURM (Simple Linux Utility for Resource Management) scheduler
- A batch job consists of
  - a line that describes the shell, e.g. `!#/bin/bash’
  - SLURM directives that describe job properties, starting with `#SBATCH’
    - many directives can be command-line arguments, but it is generally better to save job properties into the job script
    - other schedulers will have different directives
  - executable commands, details of which will depend on the shell
  - optionally, comments after `#’

- SLURM has its own commands (https://slurm.schedmd.com/quickstart.html)
- Let's try two simple ones
  - sinfo - information about the Bridges partitions. We want "RM-small" for our first job. Shorter jobs can be squeezed in between longer jobs.
  - squeue - lists jobs currently on the queue



- Let’s write a batch script called MD_ubq.job and put it in the ~/ubq-md/ directory. You can
  - use a terminal text editor like `vi’
  - create it on your local computer and transfer it to bridges

MD_ubq.job

```
#!/bin/bash
#SBATCH -N 1
#SBATCH -p RM-small
#SBATCH -t 00:30:00
#SBATCH --ntasks-per-node 1

# Set up the environment
source ~/.bashrc
module load anaconda3
conda activate openmm

# Run MD
cd ~/ubq-md
python MD_ubq.py

```

- Once you have created the job file, you can submit it to the queue using `sbatch MD_ubq.job'
- After submitting the job you can log out or be disconnected from the server and it will still run
- You can use `squeue -u *user_name*' to check the status of your job(s)
- The calculation will complete the same way as an interactive job
- The output that would be sent to the terminal in an interactive job is sent to a file

```
br006:[~/ubq-md]: sbatch MD_ubq.job
Submitted batch job 7589729
br006:[~/ubq-md]: squeue -u dminh
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REA
SON)
           7589729  RM-small MD_ubq.j    dminh  R       0:06      1 r005
```

```
br006:[~/ubq-md]: ls -ltr
total 312
-rw-r--r-- 1 dminh mc4s8bp  50895 Jan 31 18:44 1ubq.pdb
-rw-r--r-- 1 dminh mc4s8bp   1678 Jan 31 18:50 MD_ubq.py
-rw-r--r-- 1 dminh mc4s8bp    221 Feb  5 17:17 MD_ubq.job
-rw-r--r-- 1 dminh mc4s8bp  99863 Feb  5 17:23 ubq_mod.pdb
-rw-r--r-- 1 dminh mc4s8bp 148796 Feb  5 17:23 trajectory.dcd
-rw-r--r-- 1 dminh mc4s8bp    727 Feb  5 17:23 slurm-7589729.out
br006:[~/ubq-md]: more slurm-7589729.out
Minimizing...
Running Production...
#"Progress (%)" "Step"  "Potential Energy (kJ/mole)"    "Temperature (K)"
"Speed (ns/day)"        "Time Remaining"
10.0%   100     -12891.608289052907     182.75168707244657      0       --
20.0%   200     -12679.091283728229     208.12074466763545      8.22    0:16
30.0%   300     -12544.950513910171     222.9911557564418       8.27    0:14
40.0%   400     -12417.294668972278     241.63949664412493      8.29    0:12
50.0%   500     -12182.148295443869     250.84879213741047      8.31    0:10
60.0%   600     -12086.427954357983     250.25410151752132      8.31    0:08
70.0%   700     -12201.427649847661     275.4539256800823       8.3     0:06
80.0%   800     -12071.165875336525     278.7590849918891       8.29    0:04
90.0%   900     -11779.891281989974     278.2551306095054       8.28    0:02
100.0%  1000    -11804.03142004821      289.69894096208446      8.27    0:00
Done!
```

Some Complications

- For your project, you will want different options in your scripts to
  - run MD on GPUs
  - run calculation for longer
  - not overwrite previous data
  - store data on $LOCAL or $SCRATCH