Astroinformatics: Day 1

Adam W. Lavely, Ph.D.

Morning Seminar B

June 4 2018

Slides/Files available: https://goo.gl/Wz72vs
1 Version Control

2 Compute Resources

3 Logging on
Contents

1 Version Control

2 Compute Resources

3 Logging on
What is Version Control?

A way to track multiple versions of a code

**Development**
- Ongoing development
- Multiple developers

**Variations**
- Different purposes
- Different hardware
Types of Version Control

Local-data model
- Single master repo, one development point
- *Examples:* RCS, SCSS

Client-server model
- Single master repo, multiple development points
- *Examples:* SVN, CVS, Vault

Distributed model
- No master repo, multiple development points
- *Examples:* git, ArX
Why use git?

- Most flexibility
- Widely used
- Good documentation

Basic git workflow:

- Modify files - new code, fix bugs, etc
- Stage the files - explicitly state what will be deposited
- Commit your files - store a snapshot
Basic Git Structure

Branches:

master
where ‘current production version’ lives

```bash
git init .
```

other
make a branch for anything else

```bash
git checkout EXISTINGBRANCH

git checkout -b NEWBRANCH
```

- Testing new features
- Non-standard code (cluster, purpose)

What to do with branches:

- Test new features - eventually *merge*
- Develop for non-standard purposes - keep separate
Git has no ‘primary’ location:
Git structure exists *everywhere you have a repo!*

**Up to the user to keep branches as up-to-date as desired:**
- local computer: master, branches
- clusters: master, branches
- github.com: master, branches

**Note:** typical ‘other’ location name is *origin*
Basic Git Commands

Clone
Get exact replica of entire repo

pull/push
Transfer individual branches

init
Initialize a new repo

add
Stage new or modified files

commit
Commit staged files to a repo

checkout
Switch to branch

diff
See differences (branches or versions)

merge
Join two different branches

Note: We’ll do some of this in an afternoon example
**Git Resources**

**Codes available:**
- Online public repos ([github.com](http://github.com))
- Institutional repos ([git.psu.edu](http://git.psu.edu))

**Additional resources:**
- [git-scm.com/doc](http://git-scm.com/doc)
- [try.github.io](http://try.github.io)
- Google is your best friend
Contents

1. Version Control

2. Compute Resources

3. Logging on
Compute Resources for Astroinformatics

A guide to determining selection of computational resources for your astronomy research
Materials online at ....

You can get this slide deck at the following link:

Scales of Compute Resources

- Workstations
- Local Clusters at Your Institutions
- NSF XSEDE Clusters
  - NSF XSEDE Program Overview
  - Resources Available via XSEDE
  - Using Your Access – Getting More Resources
- DOE ALCF, NSF Blue Waters
The Matrix of HPC

- D.I.Y. and Departmental
- Institutional HPC Clusters
- NSF XSEDE Resources
- NSF Blue Waters (NSF)
- Cheyenne (NCAR/UCAR)
- DOE ALCF
- Other Agency Specific (DoE, NASA, etc.)
- Commercial HPC and Cloud Services
Where to start?

- Defining your research needs and wants
- Thinking for the long and short term
- Observing Compliance and Security
- Ease of system use (documentation)
- Easy management of data
- Feasibility
Things to consider ...

- Support structure and balancing local and remote needs
- Backup and recovery options
- Software availability: is it included or an added cost
- Archival Storage (PSU Data Commons)
More Features to Consider ...

- Funding and Allocation Procedures
- Connectivity – Internet2, etc.
- Software – Maturity of Software Stack
- Transfer Tools – Globus, Parallel FTP, etc.
- Hidden Egress (Transfer) Costs
- Container Solutions
- GPUs
- Backup Policies
- Purge / Scratch Policies
What your funding agencies might want to see ....

- Unless you are in hardware development (EE, CSE) or have very specific needs, getting funding to build your own, independent cluster is likely to be difficult.

- Using nationally funded resources or collaborative resources is usually preferred and easier for accounting purposes.

- You don’t have to worry about the daily minutia (i.e. system administration and software management)!
The XSEDE Family of HPC
https://www.xsede.org

A consortium of HPC facilities in the U.S.

Stampede 2 (TACC)
Comet (SDSC)
Xstream (Stanford)
SuperMIC (LSU)
Bridges GPU (PSC)
Bridges Reg. Mem (PSC)
Bridges High Mem (PSC)
Jetstream (IU and TACC)
Comet (SDSC)

Wranger (TACC)
Maverick (UT Austin)
Open Science Grid (OSG)
Traditional XSEDE HPC Machines

PSC's Bridges is a heterogeneous machine designed to support traditional and non-traditional HPC, Big Data and large-memory applications.

SDSC's large Comet cluster is 100% dedicated to XSEDE users. Comet features the next generation Intel "Haswell" processors with AVX2.

Stampede2 is an 18-petaflop national resource featuring 4,200 Knights Landing (KNL) and 1,736 Intel Xeon Skylake (SKX) nodes by the fall of 2017.

Equipped with Intel's Xeon Phi technology, LSU's SuperMIC cluster consists of 380 compute nodes.
XSEDE GPU - HPC Machines

PSC's GPU Bridges provides NVIDIA Tesla K80 and P100 GPUs for substantial, complementary computational power for deep learning, simulations and other applications (Late Summer 2017)

SDSC's Comet GPU cluster provides both NVIDIA K80 and P100 GPU-based resources.

Stanford

The XStream Cray CS-Storm GPU cluster serves researchers with accelerator workload environments. Specifically designed by Cray for GPU computing, or more precisely, heterogeneous parallel computing with CPUs and GPUs, it differs from traditional CPU only based HPC systems as it has almost a Petaflop (PF) of GPU compute power. Each of the 65 nodes has 8 NVIDIA K80 cards or 16 NVIDIA Kepler GPUs.
Jetstream is XSEDE's first user-friendly, scalable cloud environment.

Jetstream, led by the Indiana University Pervasive Technology Institute (PTI), adds cloud-based, on-demand computing and data analysis resources to the national cyberinfrastructure.

Jetstream supports Singularity and Docker container systems.

TACC currently supports Singularity on its Stampede cluster.
Wrangler is the most powerful data analysis system allocated in XSEDE.

The system is designed for large scale data transfer, analytics, and sharing and provides flexible support for a wide range of software stacks and workflows.
Obtaining Your own XSEDE Startup Allocation

Get an XSEDE Portal account

Contact your local XSEDE Campus Champion!
https://www.xsede.org/community-engagement/campus-champions

Apply!
https://portal.xsede.org/allocations/startup
The XSEDE Campus Champion Program

Campus Champions, act as a liaison and a guide to their Institution’s researchers and students with the XSEDE family of resources.

Contact your local XSEDE Campus Champion!

https://www.xsede.org/community-engagement/campus-champions
Applying for XSEDE Research Allocations – Tips for Success

- Start with a Trial or Startup allocation
- Designed for testing/timing of XSEDE systems
- Suitable for your needs?
- Gather timings and parallel scaling data
- Then submit a Research proposal
- Research proposals must have timing/scaling data
  Note - The data is taken seriously
- You must get a startup allocation before getting a research allocation
- You can submit research proposal while a startup allocation is active
Applying for XSEDE Research Allocations – Online Resources

Writing and Submitting a Successful XSEDE Proposal – By Ken Hackworth

Future live seminars on the XSEDE Training Portal

Next allocation deadline: July 15, 2018
(Future 15 Oct 2018, 15 Jan 2019 and 15 April 2019)

https://portal.xsede.org/allocations/policies
National Resources – Blue Waters (NSF)

The Blue Waters system is a Cray XE/XK hybrid machine composed of AMD 6276 "Interlagos" processors (nominal clock speed of at least 2.3 GHz) and NVIDIA GK110 (K20X) "Kepler" accelerators all connected by the Cray Gemini torus interconnect.

Major allocations are given out via the Petascale Computing Resource Allocations (PRAC) program (NSF).

More allocation information can be applied for here:
https://bluewaters.ncsa.illinois.edu/aboutallocations
National Resources – Blue Waters (NSF)

Eligible researchers can also get onto Blue Waters through the Great Lakes Consortium for PetaScale Computation (GLCPC): http://www.greatlakesconsortium.org

Note - The first proposal should be for scaling purposes

More allocation information can be applied for here:
https://bluewaters.ncsa.illinois.edu/aboutallocations
National Resources – (NCAR / UCAR) Cheyenne

These NCAR supercomputing, data storage, and archive systems support the development of climate models, weather forecasting, and other critical research.

Cheyenne is a new 5.34-petaflops, high-performance computer built for NCAR by SGI.

An SGI ICE XA Cluster, the Cheyenne supercomputer features 145,152 latest-generation Intel Xeon processor cores in 4,032 dual-socket nodes (36 cores/node) and 313 TB of total memory.

Allocation information can be found here:
https://www2.cisl.ucar.edu/user-support/allocations
National Resources – DOE – ALCF
Argonne Leadership Computing Facility

The Argonne Leadership Computing Facility (ALCF) is a national scientific user facility that provides supercomputing resources and expertise to the scientific and engineering community to accelerate the pace of discovery and innovation in a broad range of disciplines.

General information can be found here: https://www.alcf.anl.gov
Mira and Theta are the engines that drive scientific discoveries and engineering breakthroughs at the ALCF. At around 10 petaflops each, the systems are among the fastest supercomputers in the world for open science. More than 5 billion computing hours are allocated on ALCF systems each year.

Supporting IBM Blue Gene/Q systems – Cetus and Vesta – are used for debugging and test and development work, respectively.

Cooley, our visualization cluster, helps transform computational data into high-resolution images, videos and animations, helping users to better analyze and understand simulations produced by Mira.

Allocation information can be found here: https://www.alcf.anl.gov/user-guides/how-get-allocation
Preflight Awards

**Purpose:** Start-up and preflight projects. Primarily a "first step" for projects working toward an INCITE or ALCC allocation.

**INCITE - Innovative and Novel Computational Impact on Theory and Experiment Program**

**Purpose:** Supports computationally intensive, large-scale research projects that aim to address "grand challenges" in science and engineering.

Allocation information can be found here: https://www.alcf.anl.gov/user-guides/how-get-allocation
Commercial Cloud Services

Some advantages:

- Almost no spin up time (services are in place)
- High reliability
- Mature systems
- No need for local server room infrastructure
- Instant flexible demand
Commercial Cloud Services

- HPC available through third-party commercial providers
- Good for short-burst and temporary and some specialized needs
- Removes the need for the host institution to maintain equipment
Commercial Cloud Services

Some concerns:

• Software may not be included

• Service costs - accidental usage

• Data egress costs

• Lock in – hard to move / change services

• Lack of control with increasing costs
Public Cloud Services Comparison

https://ilyas-it83.github.io/CloudComparer/
Conclusions

Taking the time to envision your whole research enterprise at the start will pay dividends in the long run.

A hybrid approaching using multiple services is likely the best path for most research operations now and in the near future.

And, when in doubt, ask for help from the experts!
Materials online at ....

You can get this slide deck at the following link:


QUESTIONS?
## The Scheduler Rosetta Stone

<table>
<thead>
<tr>
<th>User Commands</th>
<th>PBS/Torque</th>
<th>Slurm</th>
<th>LSF</th>
<th>SGE</th>
<th>LoadLeveler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job status (by user)</td>
<td><code>qstat -u [user_name]</code></td>
<td><code>squeue -u [user_name]</code></td>
<td><code>bqjobs -u [user_name]</code></td>
<td><code>qstat [-u user_name]</code></td>
<td><code>fjq -u [username]</code></td>
</tr>
<tr>
<td>Queue list</td>
<td><code>qstat -Q</code></td>
<td><code>squeue</code></td>
<td><code>bqueues</code></td>
<td><code>qconf -q</code></td>
<td><code>fclass</code></td>
</tr>
<tr>
<td>Node list</td>
<td><code>pbsnodes -l</code></td>
<td><code>sinfo -N OR sctshow show nodes</code></td>
<td><code>hosts</code></td>
<td><code>ghost</code></td>
<td><code>fstatus -L machine</code></td>
</tr>
<tr>
<td>Cluster status</td>
<td><code>qstat -a</code></td>
<td><code>sinfo</code></td>
<td><code>bqueues</code></td>
<td><code>ghost-q</code></td>
<td><code>fstatus -L cluster</code></td>
</tr>
<tr>
<td>GUI</td>
<td><code>xpbsmon</code></td>
<td><code>svIEW</code></td>
<td><code>xiblist OR xsbatch</code></td>
<td><code>xmon</code></td>
<td><code>xload</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Environment</th>
<th>PBS/Torque</th>
<th>Slurm</th>
<th>LSF</th>
<th>SGE</th>
<th>LoadLeveler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job ID</td>
<td><code>@PBS_JOBID</code></td>
<td><code>@SLURM_JOBID</code></td>
<td><code>@LSB_JOBID</code></td>
<td><code>@JOB ID</code></td>
<td><code>@LOAD_STEP_ID</code></td>
</tr>
<tr>
<td>Submit Directory</td>
<td><code>@PBS_SUBDIR</code></td>
<td><code>@SLURM_SUBMIT_DIR</code></td>
<td><code>@LSB_SUBWD</code></td>
<td><code>@LOAD_STEP_DIR</code></td>
<td><code>@LOAD_STEP_DIR</code></td>
</tr>
<tr>
<td>Submit Host</td>
<td><code>@PBS_HOST</code></td>
<td><code>@SLURM_SUBMIT_HOST</code></td>
<td><code>@LSB_SUBHOST</code></td>
<td><code>@LOAD_HOST</code></td>
<td><code>@LOAD_HOST</code></td>
</tr>
<tr>
<td>Node List</td>
<td><code>@PBS_NODEFILE</code></td>
<td><code>@SLURM_NODELIST</code></td>
<td><code>@LSB_HOSTS/LSB_MCPU_HOST</code></td>
<td><code>@PBS_HOSTFILE</code></td>
<td><code>@LOAD_PROCESSOR_LIST</code></td>
</tr>
<tr>
<td>Job Array Index</td>
<td><code>@PBS_ARRAYID</code></td>
<td><code>@SLURM_ARRAY_TASK_ID</code></td>
<td><code>@LSB_JOBINDEX</code></td>
<td><code>@PBS_HOSTFILE</code></td>
<td><code>@PBS_HOSTFILE</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Job Specification</th>
<th>PBS/Torque</th>
<th>Slurm</th>
<th>LSF</th>
<th>SGE</th>
<th>LoadLeveler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job submission</td>
<td><code>#PBS</code></td>
<td><code>#SBATCH</code></td>
<td><code>#BSUB</code></td>
<td><code>#SBATCH</code></td>
<td><code>#SBATCH</code></td>
</tr>
<tr>
<td>Node Count</td>
<td><code>#PBSnodes</code></td>
<td><code>@PBSnodes</code></td>
<td><code>@PBSnodes</code></td>
<td><code>@PBSnodes</code></td>
<td><code>@PBSnodes</code></td>
</tr>
<tr>
<td>CPU Count</td>
<td><code>-n [size]</code></td>
<td><code>-n [max]</code></td>
<td><code>-n [size]</code></td>
<td><code>-n [max]</code></td>
<td><code>-n [size]</code></td>
</tr>
<tr>
<td>Combine stdioiner</td>
<td><code>-o (both to stdout) OR -e (both to stderr)</code></td>
<td><code>-o (both to stdout) OR -e (both to stderr)</code></td>
<td><code>-e (both to stderr)</code></td>
<td><code>-e (both to stderr)</code></td>
<td><code>-e (both to stderr)</code></td>
</tr>
<tr>
<td>Copy Environmen</td>
<td><code>-V</code></td>
<td><code>-V</code></td>
<td><code>-V</code></td>
<td><code>-V</code></td>
<td><code>-V</code></td>
</tr>
<tr>
<td>Event Notification</td>
<td><code>-m abe</code></td>
<td><code>-m abe</code></td>
<td><code>-m abe</code></td>
<td><code>-m abe</code></td>
<td><code>-m abe</code></td>
</tr>
<tr>
<td>Resource Sharing</td>
<td><code>-l ncpuspolicy=singlejob</code></td>
<td><code>-l ncpuspolicy=singlejob</code></td>
<td><code>-l ncpuspolicy=singlejob</code></td>
<td><code>-l ncpuspolicy=singlejob</code></td>
<td><code>-l ncpuspolicy=singlejob</code></td>
</tr>
<tr>
<td>Account to charge</td>
<td><code>-l group_list=[account]</code></td>
<td><code>-l group_list=[account]</code></td>
<td><code>-l group_list=[account]</code></td>
<td><code>-l group_list=[account]</code></td>
<td><code>-l group_list=[account]</code></td>
</tr>
<tr>
<td>CPUs Per Task</td>
<td><code>-l np=pnum[#CPUs_per_node]</code></td>
<td><code>-l np=pnum[#CPUs_per_node]</code></td>
<td><code>-l np=pnum[#CPUs_per_node]</code></td>
<td><code>-l np=pnum[#CPUs_per_node]</code></td>
<td><code>-l np=pnum[#CPUs_per_node]</code></td>
</tr>
<tr>
<td>Quality Of Service</td>
<td><code>-q [name]</code></td>
<td><code>-q [name]</code></td>
<td><code>-q [name]</code></td>
<td><code>-q [name]</code></td>
<td><code>-q [name]</code></td>
</tr>
<tr>
<td>Job Array</td>
<td><code>-t [array_spec]</code></td>
<td><code>-t [array_spec]</code></td>
<td><code>-t [array_spec]</code></td>
<td><code>-t [array_spec]</code></td>
<td><code>-t [array_spec]</code></td>
</tr>
<tr>
<td>Other Resources</td>
<td><code>-l resource_spec</code></td>
<td><code>-l resource_spec</code></td>
<td><code>-l resource_spec</code></td>
<td><code>-l resource_spec</code></td>
<td><code>-l resource_spec</code></td>
</tr>
</tbody>
</table>


---

29-Apr-2013

Next Steps ....
Get on PSC Bridges

Try ssh-ing over to PSC Bridges using your appropriate terminal program: (mobaXterm, Terminal, etc)

ssh bridges.psc.edu -l username

Password help?

https://www.psc.edu/user-resources/password-policies
**Purpose of Scaling**

**Code performance testing:** Identify ability of code to compute different problem sizes on different computational resources

**You should do scaling:**
- Help diagnose ‘profiling’ issues
- Find range for reasonable efficiency jobs (and max)
- Understand time-scales for analyses
- Show your code is ready and requested time is accurate
  - NSF XSEDE and DoE INCITE proposals
  - XSEDE Campus Champion: Chuck Pavloski - cfp102
Strong Scaling

Find efficiency for a fixed problem

for procs in \{1,2,4,8,16,32,...\}:
    run code with $procs

Notes:

- Choose processors & starting point with intention
  - Full nodes vs. single proc
- Super-linear speed up possible
  - RAM/Cache hits
Strong Scaling - OpenFOAM on Stampede 1.0

The diagram shows the speedup relative to 64 processors as a function of the number of processors. The graph includes two lines:

- The dashed line labeled "Ideal" represents ideal scalability, which would be a 1:1 relationship between processors and speedup.
- The solid line labeled "Actual" shows the actual speedup achieved with OpenFOAM on Stampede 1.0.

The speedup is calculated by dividing the execution time with 64 processors by the execution time with the specified number of processors. The data points indicate that the actual speedup closely follows the ideal line, suggesting good scalability.

The x-axis represents the number of processors, ranging from 0 to 1200, and the y-axis represents the speedup relative to 64 processors, ranging from 0 to 16.
Strong Scaling: Efficiency - OpenFOAM on Stampede

![Graph showing efficiency against number of processors]

- Efficiency using ideal speed up from 64 processors.
- The graph illustrates the decrease in efficiency with increasing numbers of processors.

ICS@PSU
**Weak Scaling**

Find max problem for resource

for procs in \{1,2,4,8,16,32,...\}:
    make problem $procs$ in size
    run code with $procs$

**Notes:**
- Choose sizes with intention
  - Meaningful problems
  - Full nodes
- Used to compare codes/parallelism
- Less common than strong scaling
Restart at 2 pm

Please stay after if you need help with:

- Logging onto Bridges
- Accessing a terminal
- Starting Jupyter