Practical 12: Getting started on saguaro Documentation

Release 1.0

Oliver Beckstein

April 03, 2013
# CONTENTS

1 Running jobs on saguaro .......................................................... 3
   1.1 Etiquette ........................................................................... 3
   1.2 Log in ............................................................................... 3
   1.3 Copying files .................................................................... 4
   1.4 Software on saguaro .......................................................... 4
   1.5 Queuing system ................................................................ 5

2 Example run ............................................................................ 9

3 Exercise/Assignment ................................................................ 11
   3.1 Assignment 11: Simulating Argon at multiple temperatures on saguaro ........................................... 11
   3.2 Optional: Least square fitting in Python .......................................................... 11

4 Indices and tables .................................................................. 13
Practical 12: Getting started on saguaro Documentation, Release 1.0

2 CONTENTS
RUNNING JOBS ON SAGUARO

1.1 Etiquette

In general:

- Do not run simulations or any other CPU-intensive calculations on the head node for longer than about a minute.
  All computation must be done through the batch queuing system.
- Your disk space is limited (about 1 GB) so you need to copy off data to a local machine (laptop, iMac) and delete data.
  You can use the /scratch directory (unlimited) but files there are deleted automatically after 30 days.
- Don’t snoop around in other peoples’ directories or copy files without express permission.
- The use of University computer equipment is only allowed to carry out work assigned to you by your instructor, line manager, or laboratory head. Private use is not allowed. The Computer, Internet, and Electronic Communications Policy (ACD 125) of the University applies and must be complied with.

In particular for this course:

- Do not run jobs for longer than 24h per run.
- Do not use more than 4 cores (“nodes”) per run unless you have my explicit permission.
- Do not run more than 2 jobs concurrently unless you have my explicit permission.
- Only use the course billing account “phy598s113” for work related to the class (see below under Queuing system).

(The course only has a limited number of CPU-hours available and these rules help to prevent someone—possibly accidentally—wasting the whole course’s allocation of 10,000 CPU-h.

By the way, one CPU-h costs $0.01, i.e. a 4-core run for 24h is about $1.)

1.2 Log in

Log in to the head node of the saguaro cluster via ssh:

ssh -l ASURITE saguaro.fulton.asu.edu

You need to provide your ASU password to log in.

This is your own home directory and your own work space, tied to your username (your ASURITE id). Only you can write here.
Make a directory for today’s practical in your home directory on saguaro:

```
mkdir P12
```

**Note:** Any Linux or Mac OS X system will have `ssh` (and `scp`) installed. However, Windows users will have to install a ssh client. The free PuTTY ssh client is highly recommended.

### 1.3 Copying files

Use the `scp` command to transfer individual files (in the following, `ASURITE` is a placeholder for your own ASURITE ID, which acts as your user name on saguaro):

```
scp saguaro_gromacs.pbs ASURITE@saguaro.fulton.asu.edu:P12
```

or whole directories:

```
scp -r Argon_input_files ASURITE@saguaro.fulton.asu.edu:P12
```

You can also use `scp` to copy results back:

```
scp -r ASURITE@saguaro.fulton.asu.edu:P12/MD .
```

**See Also:**

- On saguaro you can also use `curl` to get files directly from URLs.
- The `rsync` command is in many ways more comfortable than `scp` but it’s also more complicated.

### 1.4 Software on saguaro

#### 1.4.1 Gromacs 4.5.5 on saguaro

I compiled two versions of Gromacs on saguaro: one to run simulations (the "MPI version"), the other one to run a short analysis (the "serial version")

You have my explicit permission to use these version of Gromacs and look around in my Library directory.

**Serial version for quick analysis**

The serial version of Gromacs should only be used for trying out very short runs or a quick analysis. If it takes longer than a minute then it should be submitted as a job (but if you submit analysis, make sure that you only use a single core, i.e. `#PBS -l nodes=1` in your script).

To use the serial (i.e. non parallel) version of Gromacs:

```
/home/obeckste/Library/Gromacs/versions/serial-4.5.5/bin/GMXRC
```

You can then run `grompp`, `g_msd`, etc...
**MPI version (for simulations)**

I compiled a parallel (MPI-enabled) version of Gromacs 4.5.5 for saguaro. To use it:

```
module load openmpi/1.4.5-intel-12.1
./home/obeckste/Library/Gromacs/versions/4.5.5/bin/GMXRC
```

Note that this will *not work on the login node* because use of the MPI libraries is restricted to compute nodes. However, it *will* work as part of a Gromacs PBS script for saguaro.

“MPI” stands for “message passing interface” and is a protocol that supports writing parallel code that can run on thousands of CPUs. We are using the Open MPI implementation of the MPI library.

**1.4.2 Modules**

The module system makes many different software packages available. The `module` command can be used to learn which programs are available: See

```
module avail
```

for a list.

The `module load` command then loads the software into your environment, e.g.

```
module load openmpi/1.4.5-intel-12.1
```

and

```
module list
```

shows what you have loaded.

- We need the MPI library and Intel compilers for our version of gromacs, hence `module load openmpi/1.4.5-intel-12.1`.
- A2C2 staff compiled a version of gromacs (`module load gromacs/4.5.4`) but it uses double precision arithmetic and is only half as fast as the version we are using.

**1.5 Queuing system**

Instead of directly running a calculation, you write a small shell script and hand this script over to a *batch queuing system*. Saguaro uses the OpenPBS queuing system.

**1.5.1 Workflow**

The typical workflow is

1. prepare input files in a work directory (If you will generate substantial amounts of data (>500MB) then this should be done in a scratch directory in `/scratch/ASURITE`)
2. adapt a queuing system script (see below for an example `saguaro_gromacs.pbs`)
3. submit the job to the queuing system:
   ```
   qsub saguaro_gromacs.pbs
   ```
4. monitor the status of your jobs
1.5.2 Important queuing system commands

- **qsub**: submit a script to the queuing system, known as “submitting a job”; when the job is launched successfully, the job id will be printed.

- **qstat**: check the status of your job(s); shows a list of job ids and job names together with their status (“Q” for still waiting in queue, “R” for running, “C” for complete).

- **qdel**: terminate a running job: `qdel JOB_ID` (you will only be billed for the CPU-h the job has consumed so far).

1.5.3 Note on allocations and CPU-h

The system keeps track of how many CPU-h are being used. They all come out of the course’s account, `phy598s113`. You **always have to provide the account name when running a queuing system script**. You do this (see below) by either providing the `-A` account flag to `qsub`:

```
qsub -A phy598s113 saguaro_gromacs.pbs
```

or (simpler!) by adding a line to the script that automatically sets the flag:

```
#PBS -A phy598s113
```

near the top of the script.

In fact, you can add many additional `qsub` options to a script by starting a line with `#PBS`. See `man qsub` on saguaro for more options.

1.5.4 Gromacs PBS script for saguaro

You can use the following *queuing system script* to run our version of Gromacs on saguaro:

```
#!/bin/bash
#PBS -N GMX_MD
#PBS -l nodes=4
#PBS -l walltime=00:10:00
#PBS -A phy598s113
#PBS -j oe
#PBS -o md.$PBS_JOBID.out

WALL_HOURS=0.167
DEFFNM=md
TPR=$DEFFNM.tpr
```

# host: saguaro
# queuing system: PBS

# max run time in hours, 1 min = 0.0167
WALL_HOURS=0.167

DEFFNM=md
TPR=$DEFFNM.tpr
LIBDIR=/home/obeckste/Library

cd $PBS_O_WORKDIR
.
$LIBDIR/Gromacs/versions/4.5.5/bin/GMXRC
module load openmpi/1.4.5-intel-12.1

MDRUN=$LIBDIR/Gromacs/versions/4.5.5/bin/mdrun_mpi

# -noappend because apparently no file locking possible on Lustre
# (/scratch)
mpiexec $MDRUN -s $TPR -deffnm $DEFFNM -maxh $WALL_HOURS -cpi -noappend

You will have to change parameters, depending on how you want to use it.

- give your job a name (instead of “GMX_MD”) — very useful when you run many jobs and need to check on them with qstat.

- adjust the run time of the job (both in the -l walltime=HH:MM:SS line and in the WALL_HOURS=hours (where hours is a decimal number). Your job will only run this long but it will shut down cleanly (mdrun will stop itself after 0.99*hours).

- modify the default filename variable DEFFNM and the filename of your TPR file (TPR)

- Note that your output files will look like md.part0001.xtc: This is due to the -noappend flag for mdrun, which we need for “continuation runs” (-cpi flag), i.e. continuing a simulation seamlessly after it ran out of time. If you’re confident that your simulation will complete in the allocated time then you may remove the -noappend flag.

1.5.5 PBS and accounts

Saguaro has OpenPBS installed. Note that there many different queuing systems that all implement slightly different version of qsub and friends so you need to read the local man pages (man qsub).

Some other useful commands on saguaro:

showq

shows the current queue (all waiting jobs).

showq -i

all jobs.

You can see how many hours the course has still got with the

mybalance

command. If you see other projects then that means that you are also enlisted in another research group with allocations on saguaro. In this case, check which one is your default project (the one that gets billed if you don’t use the -A account flag):

mybalance -d

You can also see the default project with

glsuser $USER
EXAMPLE RUN

Let’s run our Argon simulation on saguaro. You need to prepare the binary run input file (the “TPR file”) `md.tpr` as described in the last practical. It contains all information for Gromacs to run the simulation. Right now you can use one I prepared.

Let’s practice the work flow:

Get the file on your local computer (iMac, workstation):

```
```

```
tar -jxvf Argon_input_files.tar.bz2
```

```
cd Argon_input_files
```

Copy the TPR file to saguaro:

```
scp md.tpr ASURITE@saguaro.fulton.asu.edu:P12
```

Log in to saguaro (`ssh`) and now on saguaro, set up the queuing system script.

```
cd P12
```

```
```

- change the jobname to “Argon”
- change the run time to 5 minutes
- change the number of nodes to 2

Run the job:

```
qsub saguaro_gromacs.pbs
```

Check status:

```
qstat
```

and also look at the output files such as `md.5805254.newmoab.local.out`.

When done, copy everything back to your workstation (e.g. use a different terminal or log out from saguaro):

```
scp -r ASURITE@saguaro.fulton.asu.edu:P12 .
```
EXERCISE/ASSIGNMENT

3.1 Assignment 11: Simulating Argon at multiple temperatures on saguaro

1. Prepare four argon simulations at four different temperatures (choose sensible values for yourself) for 0.5 ns each.
   (I suggest you create a separate directory for each simulation, e.g. T40, T94, T120, T273.)
2. Submit to saguaro as 4-core jobs (you may submit all of them in one go). Make sure that your time limit will not kill the job prematurely. ¹
3. Analyze the diffusion coefficient and plot $D(T)$.
   • discard the first 100 ps (see the $-b$ flag of g_msd)
   • plot all four MSD graphs together in one figure
4. Record results on your LJ fluid wiki page (on Blackboard (Course Documents -> Practicals -> Practical 12 -> LJ Fluid in Gromacs Wiki).
   • plot
   • table with temperature $T$ and $D$.

You could write a small shell script with a for loop that takes a template MDP file, fills in the temperature and generates the TPR file. Or you could do it manually — it’s up to you. Whatever gets the job done!

3.2 Optional: Least square fitting in Python

If you don’t want to use g_msd to do the least square fit to the MSD, you can do it yourself using the following code that makes use of SciPy’s scipy.optimize.leastsq() function

```
# simple linear fitting with SciPy

import numpy
import scipy
import scipy.optimize
```

¹ You can estimate the required walltime by running one job first with a much longer walltime than you would expect to require (e.g. 30 minutes in this case) and time it (look at the end of the log file).
An even better approach is to use the performance information in the log file of the job you ran in the practical (... it ran at X ns/d, you are now running the same system for 0.5 ns, which requires Y days...).
def fit_linear(x, y, p0):
    """Fit a straight line to the data in *x* and *y*.
    The fitted function is `'y = m*x + t'`.
    """
    :Arguments:
        *x*  array of x values
        *y*  array of y values
        *p0* initial guess [slope, offset] e.g. `[1.0, 0.0]`
    :Returns: Parameters `[m, t]`
    """
    def errfunc(p, x, y):
        return p[0]*x + p[1] - y  # residuals
    
    p, msg = scipy.optimize.leastsq(errfunc, p0[:], args=(x, y))
    print(msg)
    return p

How does your diffusion coefficient compare with the one from g_msd?²

² Note: to convert nm²/ps to cm²/s multiply by 1e-2.
INDICES AND TABLES

- genindex
- modindex
- search