OSU Center for Genome Research and Biocomputing (CGRB), MCB 599: "Introduction to Unix/Linux" (INX_U18)
"Cheat Sheet": Day 9, **Pages:** 3, **Revision:** 2018-08-06, **Instructor:** Matthew Peterson (matthew@cgrb.oregonstate.edu)

Command/Concept	Usage/Syntax	Description	Example

`qsub`	`qsub <script_to_run>`	Submit a "batch" job to the SGE queuing system ("the hard way"). See the CGRB's `SGE_Batch` (below) for an easier submission method. SGE stdout (`*.o<job id>`) and stderr (`*.e<job id>`) files will be placed in your `$HOME` directory by default.	# For numerous options
			`qsub -help`
			# Simple script
			`cat sgetest.sh`
			`#!/bin/tcsh`
			`echo "SGE Test!"`
			# Simple submission
			`qsub sgetest.bash`

`qstat`	`qstat`	Status of "running" jobs in SGE. "state" will show:
		r Job is running on a compute node
		qw Waiting in queue to be matched to a node
		eqw Error, delete job and try resubmitting
		Completed jobs will no longer appear in `qstat`
			# Show your running jobs
			`qstat`
			# Show all jobs on system
			`qstat -u '*'`

`qdel`	`qdel <job number>`
	`qdel -u <username>`
		Delete a submitted job. You can only delete your jobs, not another user's jobs.	# Deletes a job you own
			`qdel <job number>`
			# Deletes all of your jobs
			`qdel -u <username>`

`qhost`
`SGE_Avail`
	`qhost`
	`qhost -q`
	`SGE_Avail`
		Show resources (machines and queues) available in SGE. The CGRB's `SGE_Avail` wrapper script shows resources and queues available specifically to you.	# Show resources for all
			# available machines in SGE
			`qhost`
			# All resources by queue
			`qhost -q`
			# Resources available to you
			`SGE_Avail`

`qrsh`	`qrsh -l h="<hostname>"`
	`qrsh -pe thread <int> -l h="<hostname>"`
		Interactively login to a machine reserving 1 CPU (thread) by default from SGE. **Note**: This is turned off on the majority of CGRB machines. If your lab or department owns a machine you may be granted access to use `qrsh` to login interactively to it.	`# Interactively login to`
			`# 'server100' reserving 1 CPU`
			`qrsh -l h="server100"`
			`# Interactively login to`
			`# 'server100' reserving 4`
			`# CPUs from SGE`
			`qrsh -pe thread 4 -l h="server100"`

Queue types	`SGE_Avail` or `qhost -q`	A machine can have 3 queue properties under "ARCH" or "QTYPE": **B** Allow for "batch" submission via `qsub` or `SGE_Batch` **I** Allow for interactive use with `qrsh` **P** Allow for multiple-CPU jobs (almost all of them)	`SGE_Avail` or `qhost -q`

`SGE_Batch`	Interactive: `SGE_Batch`
	Command-line:
	`SGE_Batch`
	`-P <reserve # CPUs>`
	`-f <free RAM>`
	`-m <max RAM>`
	`-F <max file size>`
	`-q <queue to use>`
	`-c <command to run>`
	`-r <run output dir>`
		CGRB's `qsub` wrapper script ("the easy way" to submit jobs to SGE). Running the command by itself loads an interactive menu; options **c** (command) and **r** (run output directory) are required. **h** to show setup and **s** submits job. `*.sh` (script file created by `SGE_Batch`), SGE std**o**ut and std**e**rr files are placed in the **r**un output directory you specify and not your `$HOME`	`# See usage/examples`
			`SGE_Batch -help`
			`# Submit sgetest.sh`
			`# job requesting resources:`
			`# 4 CPU slots, 5 GB of RAM`
			`# Kill if uses > 5 GB of RAM`
			`# Kill if it creates a single file`
			`# greater than 100 GB`
			`# Put logs in $PWD/mytest`
			`SGE_Batch`
			`-c sgetest.sh -P 4`
			`-f 5G -m 5G -F 100G`
			`-r mytest`

Array jobs	`SGE_Batch -t <range>`	`SGE_Batch` can submit an "array of jobs" from one script. You specify the range of numbers via `-t` (or **t** interactively), e.g., `-t 1-20` or if by twos, `2-20:2` The total number of jobs launched by `SGE_Batch` is equal to the range of numbers you specified, e.g., 20
		Each SGE job of the "array" is assigned one of the numbers and accessible in your script via `$SGE_TASK_ID`
		All jobs will write their SGE std**e**rr and std**o**ut files to the same **r**un output directory
			# Simple script echoing its
			# assigned ID
			`cat sgearray.sh`
			`#!/bin/tcsh`
			`echo "My assigned ID number: $SGE_TASK_ID"`
			`sleep 60`
			# Submit 10 array jobs
			# Using 1 CPU and 1 GB of
			# RAM each
			`SGE_Batch`
			`-c sgearray.sh`
			`-t 1-10`
			`-P 1 -f 1G -m 1G`
			`-r arraytest`
			# Repeatedly run until you
			# see all 10 jobs queued up
			`Qstat`

SGE_Plot	`SGE_Plotdir` `<run output dir>`	Displays RAM usage and execution time. If the run output directory contains the results of an "Array Jobs" run it will produce ASCII plots of this information.	# Plot the previous array jobs # test (producing 10 jobs) `SGE_Plotdir arraytest | less`
"10/100/1000" rule	`SGE_Batch` and `SGE_Plotdir`	Run tests on small subsets of data to extrapolate usage requirements! Bioinformatic algorithms behave differently and *may* scale in different ways, e.g., linearly, logarithmically, etc. Test, test, test!	Run `SGE_Batch` using "Array Jobs" with different sized data sets, e.g., 1/1000th, 1/100th, 1/10th to compare memory usage and CPU run times, e.g., via `SGE_Plotdir`