CLUSTER COMMANDS CHEAT SHEET

SOFTWARE MODULES

There is a lot of software - compilers, libraries, etc already built on the cluster, ready to be used. So before you try installing something yourself, you may want to check if it's already available. Please note that if you want the software module to be available to a job you're running in the queue, then it needs to be loaded at every login - so use initadd (see below).

module list - shows what software modules you have 'loaded'.
module avail - shows what software is available
module load <*moduleName*> - loads a module just for this login session
module initadd <*moduleName*> - adds the module to your login every time you login
module initls <*moduleName*> - shows which modules are loaded every time you login
module initrm <*moduleName*> - removes a module from your login
module unload <*moduleName*> - unloads a module from this login session only
eg.
module list
module load healpix/3.40-intel   (try it out)
module initadd healpix/3.40-intel (to add it to every login)

If you need any help installing software, please let me know - I'm happy to help.

INSTALLING PYTHON MODULES

Please use '**pip3 install --ignoreinstalled --user** <*moduleName*>' to install it in your home directory, otherwise you'll get errors about not being able to write to parts of the OS that you don't have access to.

SUBMITTING JOBS

showgpus - show what GPUs are available and which queues they are in
addqueue -c *"CommentWithEstimatedRuntime"* **-n** <*numberOfProcesses*> **-m** <*GB RAM needed per process*> **thingToRun**
eg.

addqueue -c "1 week" -m 3 ./doSomeAnalysis
would run a single-process (non-MPI) job with 3GB RAM allocated to it

addqueue -s -q gpulong --gputype=rtx2080with28gb -m 8 ./doSomeAnalysis
would run a process on a machine with an rtx2080 GPU in it

addqueue -c "1 week" -n 32 ./doSomeAnalysis
would submit a 32-core job to the queue with no mention of how many cores to run on each compute node - so they'll be allocated wherever the queue sees fit.

addqueue -c "2 days" -q cmb -n 2x8 ./doSomeAnalysis
will submit your program "doSomeAnalysis" to the cmb queue, requesting 2 compute nodes with 8 cores on each (so 16 processes are started). This would be applicable for an MPI-based job

addqueue -c "1 day" -q cmb -n 1x8 -s ./doSomeAnalysis
would reserve 8 cores on one compute node and start only one process on that machine (-s) - the job is serial. It may use OpenMP or start its own threads to do parallel work, but not MPI with the -s option.

(Just typing addqueue without any options shows you all the options)

CHECKING JOBS

q - starts up a graphical interface to show the jobs, if you've logged in with graphical support (ssh -Y)
q -t - lists your jobs in text-only mode
q -tc 40 - shows your last 40 completed jobs. This can be useful if you don't know why a job stopped prematurely
q -tw <*jobNumber*> - looks at job *jobNumber* and shows what compute nodes it does/doesn't match with and why it hasn't started running. Useful if you're thinking "Why hasn't this job started yet?"
showoutput <*jobNumber*> - shows the text output of your job, and the location of the file containing it. Useful to see how a job is doing.
st -t -c 800 |grep 383445 - looks for completed job 383445 and shows some info about it, possibly with a reason why it failed, if it did.

CANCELLING JOBS

scancel <*jobNumber*> - cancel a job
scancel -u <*myUsername*> - cancel all my jobs