Putting all pieces together - an R example

Quick Links

HPC Home
Getting an account
Getting started on Prince
Prince How-to Articles
Logging in
Windows
Mac / Linux
Clusters and Storage
Prince (HPC)
Dumbo (Hadoop)
Dalmatia (NYU Abu Dhabi)
Transferring data to/from the clusters
Transferri
g data
to/from
Prince
cluster
using
Globus
Submittin
g jobs
with
sbatch
Available
software
Licensed
Software
Available
on the
HPC
Cluster
Building
Software
Slurm
Tutorial
Tutorials
FAQs
Scratch
Area
Cleanup
Programming for
Biologists
Acknowledgment
Statement
Research
Gallery
HPC
People
HPC
Policies
# Running jobs on the Prince Cluster

## Accessing the Prince Cluster
- From Windows workstation
- From Mac workstation

## Software and Environment Module
- Job script and resource request

## Introduction to job scheduling
- Submitting jobs with `sbatch`
- Requesting resources
- Using computing nodes interactively

## Monitoring batch jobs
- Monitoring batch jobs - `squeue`
Pulling it all together - Preparing, submitting and monitoring a job on Prince

In this section we will prepare, submit and monitor a small R job. Our test case comes from the NYU Data Services “Introduction to R” tutorial.

**Exercise**
Start a terminal session on Prince and replicate this example in it.

**Choose your own example**
After - or instead of - following this example through, prepare and submit a run of something genuinely relevant to your research. This way, if you are doing this tutorial in a classroom, the presenter will be available should you have questions or strike difficulties.

We're using R, so first we'll look for available modules. On Prince:

```bash
$ module avail r
```

```
--------------------------------- /share/apps/modulefiles
---------------------------------
gstreamer/intel/1.10.2  mothur/intel/1.35.1  r/intel/3.3.2
```

There's a few modules starting with r, and a couple of versions of R. We'll use the latest version, 3.1.2.

```bash
$ module purge
$ module list
No Modulefiles Currently Loaded.
$ module load r/intel/3.3.2
```

Take a look at what it did:
$ module list
Currently Loaded Modulefiles:
   1) jdk/1.8.0_111  2) intel/17.0.1  3) openmpi/intel/2.0.1  4) r/intel/3.3.2

... clearly, R uses a lot of other packages. The modulefile has looked after loading the correct ones.

$ module show r/intel/3.3.2
--------------------------------------------------------------------------------------
-----
/share/apps/modulefiles/r/intel/3.3.2.lua:
--------------------------------------------------------------------------------------
-----
whatis("R: a language and environment for statistical computing and graphics")
whatis("Name: r version: 3.3.2 compilers: intel")
load("intel/17.0.1")
load("jdk/1.8.0_111")
load("openmpi/intel/2.0.1")
prepend_path("MANPATH","/share/apps/r/3.3.2/intel/share/man")
prepend_path("PATH","/share/apps/r/3.3.2/intel/bin")
prepend_path("LD_LIBRARY_PATH","/share/apps/r/3.3.2/intel/lib64/R/lib")
prepend_path("PKG_CONFIG_PATH","/share/apps/r/3.3.2/intel/lib64/pkgconfig")
setenv("R_ROOT","/share/apps/r/3.3.2/intel")
setenv("R_INC","/share/apps/r/3.3.2/intel/lib64/R/include")
setenv("R_LIB","/share/apps/r/3.3.2/intel/lib64/R/lib")
family("R")

For our example, we'll get some code and data from /share/apps/examples:

Hint: there are usage examples for a few common packages here

$ mkdir /beegfs/$USER/R-example
$ cd !$
$ cp /share/apps/examples/r/basic/* .

Take a look at the job script:
There are a few steps we can try here:

1. Start an interactive batch session, and run the example.R script interactively
2. Close the interactive session, and submit the batch script as a job:

```bash
$ sbatch my_R_job.s
```

You'll get a job id returned.

Is it running yet?

```bash
$ squeue -u $USER
```

You could watch the output in the run directory:

```bash
$ ls -l $(SCRATCH)/R-example
```

Finally, when the job finishes, you should see a .out file in the directory you submitted from.

**Exercise**

Experiment with sbatch options for the job name, output and error file merging and location, resource limits.