Using R on HPG2

Packages and job arrays

by

Hunter R. Merrill

September 19, 2016

Objectives

Learning objectives of tutorial:

- Install and use an `R` package on HPG2
- Submit multiple (independent) `R` jobs in parallel

This exercise will:

- Install the `R` package `ISLR` on HPG2
- Illustrate the sampling distribution of the mean in parallel

Installing & Using the Package

Step 1. download the `ISLR` package source from here:

cran.r-project.org/web/packages/ISLR/index.html

Step 2. Put it in your working directory on HPG2

Step 3. Create the directory `/R` in your working directory

Step 4. install the package: `R CMD INSTALL -l R ISLR_1.0.tar.gz`

Your package can be loaded with `library(ISLR)`, although you might have to specify the location:

```
library(ISLR, lib.loc = "/path/to/package")
```

Setup for multiple jobs

This is generally how my working directory looks for parallel jobs:

- directory `R-` contains any `R` packages I've installed for this set of jobs
- directory `infiles-` contains all input files for the parallel jobs
- directory `outfiles-` this is where all job output will go
- Common files for all jobs (data, `C++` source code, etc)

Workflow for multiple jobs

This is my workflow for parallel jobs:

- A template file and a code generation script are placed on the cluster
- The code generation script reads in the template file, replaces dummy characters appropriately, and writes out copies in the `infiles` directory
- The `sbatch` command is called with the `array` option to submit multiple jobs simultaneously
- All results from every job is collected in a single file.

SLURM Submission Script

```
#!/bin/sh
#SBATCH --job-name=my_job
#SBATCH --output=outfiles/my_job_%j.out
#SBATCH --mail-type=FAIL
#SBATCH --mail-user=youremail@somewhere.huh
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=1gb
#SBATCH --time=1:00:00
#SBATCH --qos=bliznyuk-b
```

SLURM Submission Script (cont.)

```
pwd;hostname;date

cat /proc/cpuinfo | head

module load gcc/5.2.0 R/3.2.2

Rscript infiles/Code_${SLURM_ARRAY_TASK_ID}.R

date
```

Example

Template file:

```
rm(list=ls()) #clear anything in memory
setwd("/ufrc/bliznyuk/hmerrill/StatLearning") #set wd

n = 100 #set sample size
set.seed(AA) #set random seed- important!

x = rnorm(n) #generate n normal random variables
results_AA = mean(x) #calculate the mean
```

Note: A script `CodeGeneration.R` is used to generate the 100 input files with `AA` replaced appropriately.

Example (cont.)

```
M = "20160919_results.csv" #name of file
does.M.exist = file.exists(M) #does the file already exist?
if (!does.M.exist) file.create(M) #if not, create it
write.table(results_AA, #write out results...
            file = M,
            sep=",",
            row.names=FALSE,
            append=TRUE,
            col.names=!does.M.exist)
```

Submitting the Jobs

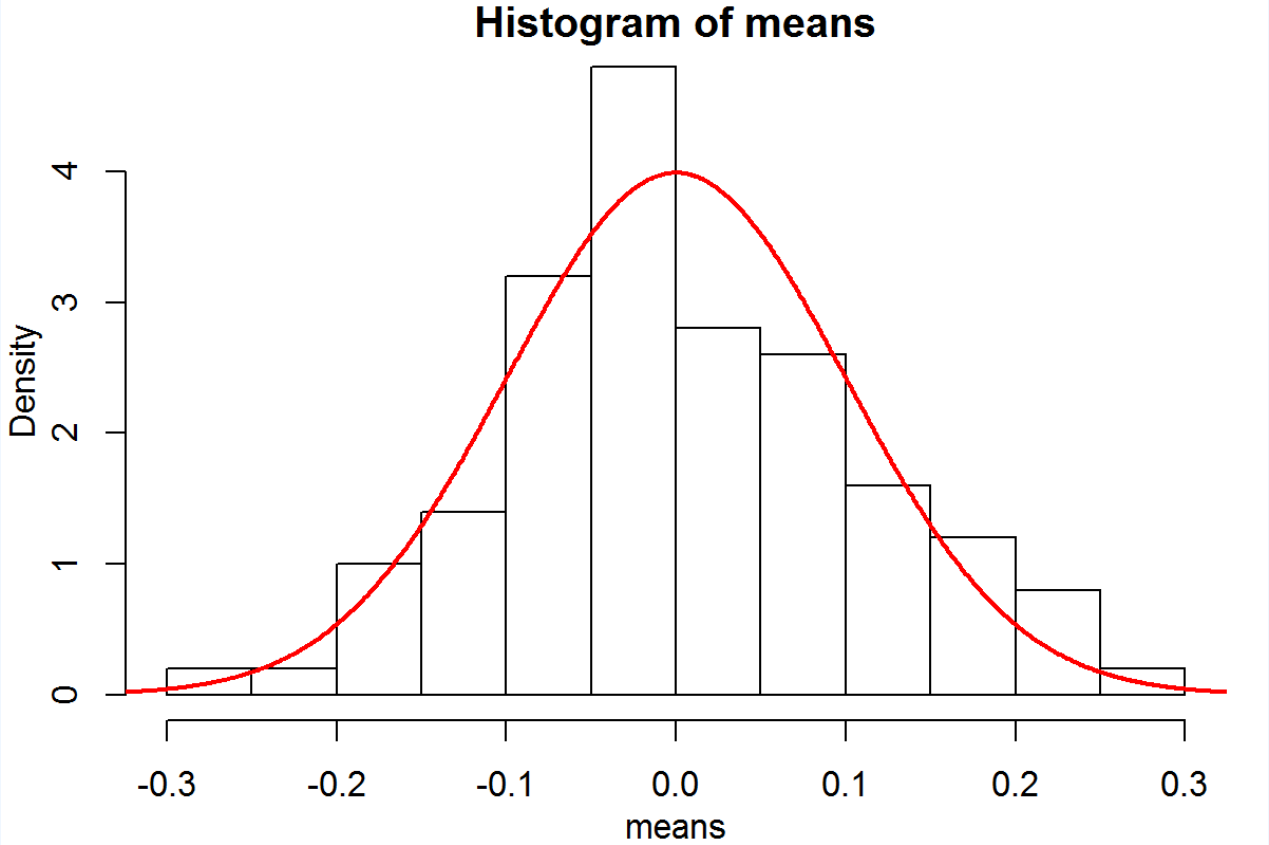
Step 1. `module add R; Rscript CodeGeneration.R`

- This first loads `R` then runs the code generation script

Step 2. `sbatch --array=1-100%50 Run_job.job`

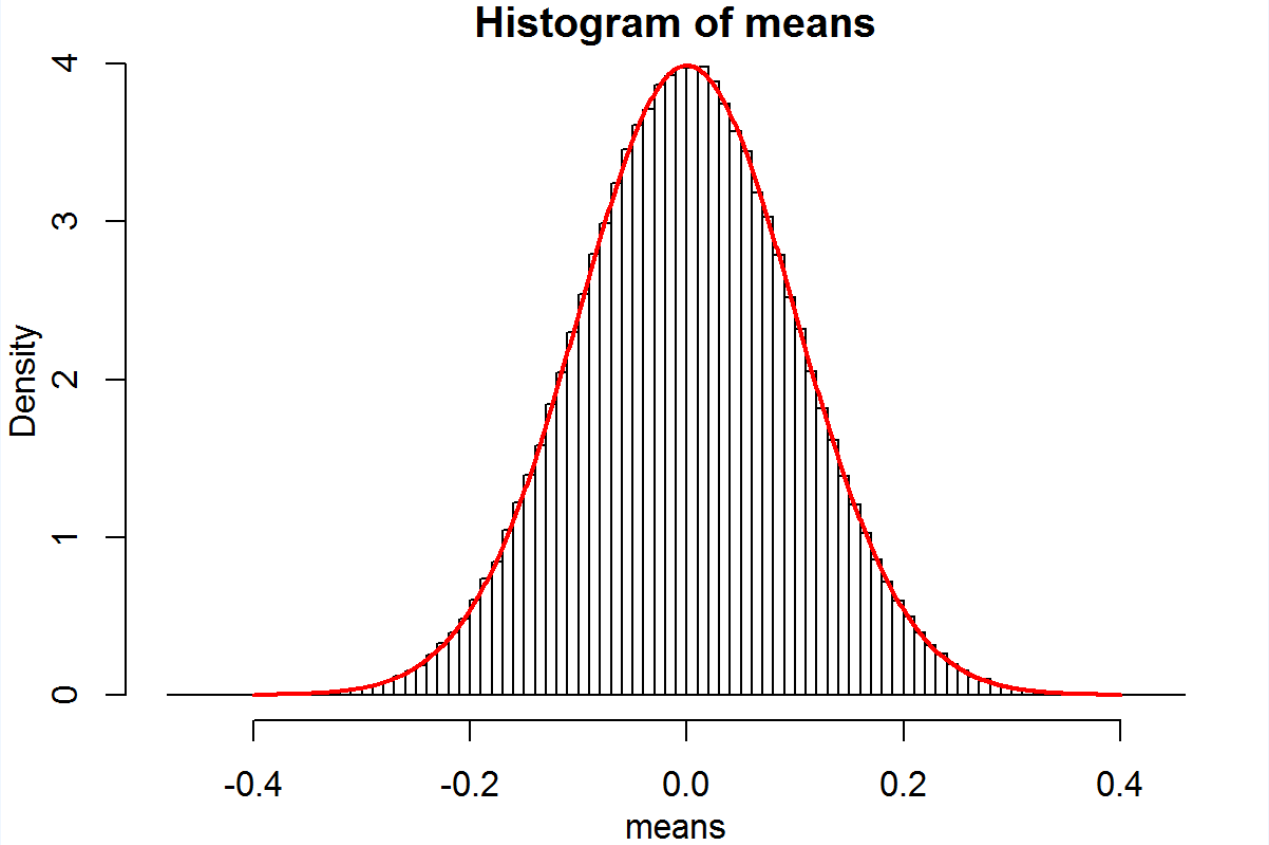
- This submits the jobs 1 through 100 described in the job file, but with the restriction that only 50 jobs run at a time

Results



A histogram of the 100 computed means, along with the theoretical sampling distribution, $\mathrm{N}(0, 1/100)$, in red.

Results



A histogram of 1×10^6 computed means, along with the theoretical sampling distribution, $\mathrm{N}(0, 1/100)$, in red.