Scripting for HPC – Lab

To get started on labs, untar the lab files into your own directory with:

```
cd && tar zxf ~train100/labs.tgz
```

Measure Speed of Pure Python

In the lab directory called "`python`," compare how long it takes to solve a matrix in pure python versus python that calls the lapack libraries.

```
cd python
module add python
module add numpy
time python pure_invert.py 100
time python numpy_invert.py 100
```

So one is faster. Take a look at the code to see how that comes about. The pure_invert.py file uses the matfunc.py file, while numpy_invert.py uses, ultimately, a library called LAPACK to do its work. How long is each code, and how hard would it be to modify these algorithms if you were doing research with these codes? Which one do you trust?

Use Lua for Parameters and Configuration

This lab shows how to use a scripting language for subroutines inside of Fortran code.

In the directory are:

- program.f90 – the Fortran program
- vals.lua – a file with parameter settings and a function
- config.c – a C file that keeps a reference to the Lua interpreter and allows Fortran to call it.
- Makefile – which compiles config.c and links it to program.f90.

Before you start, load appropriate modules.

```
module swap pgi intel
module add lua
make calculate
```

Now take a look at vals.lua. This is the configuration file, written in a very simple scripting language. The Fortran program will read this and use it if you type "`./calculate vals.lua`". Try making changes to the file.

Run Serial R Many Times

This is called the trivially-parallelizable case. You have a program in R and want to run it many times over. There are many simple ways to do this. We will use a Ranger module called `launcher`.

In the batchr directory is a set of R files to calculate how insects behave near corn fields. First load the R module with "`module load R`". You could run this program with a random number generator seed of 23 with the command:

```
R CMD BATCH –no-save –no-restore "–args 23" main.R out.23
```

We want to do this a bunch of times in one batch job, so we add another module, `launcher`. Your job is to figure out how to run this main.R program 16 times on a single node using the launcher command.

1. How do you find help on the launcher command, such as where it is installed? Read the help now and the README for launcher.
2. The help file talks about a command called bsub, but we are on Ranger where the job submission command is called qsub and the scheduler software is called SGE. Which file do you need to copy to your batchr directory?
3. The file you copied to your batchr directory does not have your account information. If you don't remember the account number, where do you get your account number?
4. Once you have this ready, try running a job as described in the launcher README. You can create a paramlist file by hand. The paramlist file will have many lines like the "R CMD…" line above.

Run an R MPI Job

This lab is more appropriate for people who have used MPI. In this lab, your goal is to get running an R script that has already been converted to use MPI. Getting modules and paths correct will be enough work. Then you can look at the code and demos.

We start with a serial R program and wrap it in a master-worker code, written in R, that uses an R library called Rmpi. In this directory are:

- The serial program – main.R, Kernels.R, MPpesticides.R, MOSpatialParams.R
- The MPI code in R – master.R
- The script to submit to the scheduler – master.sh
- A test script that might be simpler – Submit hello.sh to qsub, and it runs hello.R, with output in hello.Rout.

We will use a custom-compiled version of R, because this one definitely works, not because the system installation of R does not, so unload the system's version of R with "`module del R`".

```
module del R
module add gmp
qsub hello.sh
```

Output will be in hello.Rout. That looks simple, but many things can go wrong because we are asking R to load shared libraries that load MPI. If it doesn't work, check whether you have privileges on installation directories and whether the LD_LIBRARY_PATH is set correctly.

If that works, then try “qsub master.sh”. Now take a look at the code. Start with hello.R, maybe look in demos, and then check out master.R. The Rmpi library makes some unusual choices about its implementation of MPI.