Using GNU parallel: An introduction

Emma McIvor

University of Nottingham

March 21, 2019

Outline

- What is GNU parallel?
- Building a basic example
- Future: build a basic example in Julia?

What is GNU parallel and why use it?

- GNU parallel is a tool that can execute independent jobs (e.g. a script to run a simulation) in parallel using one or more computers
- To run GNU parallel we create a shell file (`.sh`) that wraps the script we want to execute. This is good because it means we are not adding extra complexity into the simulation script (more complexity $\implies$ more chance of introducing errors!)
- GNU parallel allows us to run multiple simulations simultaneously which can be useful for reducing the overall computation time required e.g. doing parameter scans

Setting up GNU parallel

- See README.md on GitHub `https://github.com/emmamcivor/UoN_math_computing_seminar`
- See GNU parallel documentation for additional installation help
- All files used today are available from the GitHub repo so you can practise and make your own scripts to be run in parallel

Running GNU parallel

1. Make a working directory with the script you want to run in parallel e.g. `test.m`
2. Make a parameter list in a `parameter.txt` file with one set of parameters per line
3. Create the shell scripts to run GNU parallel
   - These don't have to be placed in the working directory
   - I have placed them in the level above the working directory but I have added this folder to the local path on my `.bashrc` so I can run these shell scripts from the command line
4. `cd` into working directory and run shell script to run GNU parallel from command line

Example: Create a basic Matlab script

This Matlab script (`test.m`) takes input parameters *a* and *b* and does a simple addition and multiplication which is saved to a unique file. We want to run this for a set of *a* and *b* parameter values.

```
% matlab file to test parallel is working
% a and b are the parameters we feed into the simulations

function test(a,b)

% outputs of simulation
c=a+b;
d=a*b;

% save the outputs to a file in test_save folder with parameters making
% up filename
fn_save=['test_GNU_parallel-a_',num2str(a),'-b_',num2str(b),'.mat'];
save(fn_save,'c','d')

% print to standard out which can be caught by GNU parallel
fprintf(1,"\n\n[DATA]%d,%d,%d,%d\n\n",a,b,c,d);

% I found that I had to exit matlab explicitly but this might not be
% the case for other languages
exit;
end
```

Input parameters ‘a’ and ‘b’

Execute some commands to be saved to a file

Save simulation output to a specific file

Display simulation output

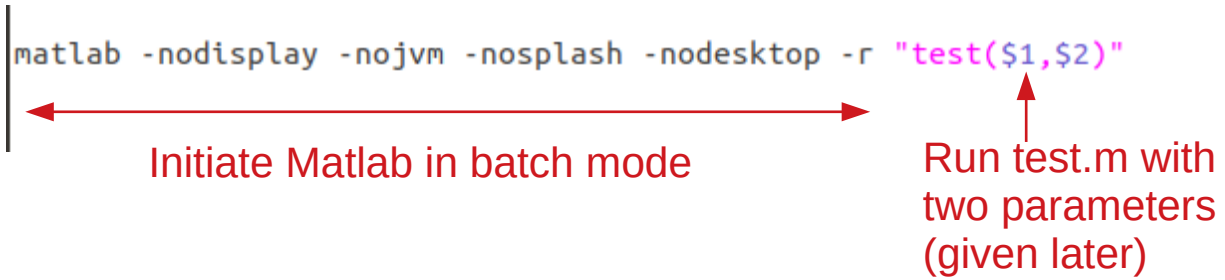
Example: Create a parameter file with all necessary parameters

- Create a file called `parameters.txt` containing all the parameters for the simulation
- Each line is a set of parameters
- I am using a comma separated format (I tell GNU parallel this later)

```
1,2
3,4
5,6
7,8
9,10
```

Example: Create a shell script to run a single instance of the Matlab script

- Create a new shell script (`run_test_matlab.sh`) to initiate the Matlab simulation:



- Make sure this file is executable. If not, on the command line execute `chmod u=rwx run_test_matlab.sh`

Example: Create a shell script to run multiple simulations in parallel

- Create a new shell script (`test_parallel_1host.sh`) to run the Matlab simulations simultaneously/in parallel:

```
#!/usr/bin/bash

(parallel --joblog ./parallel.log --eta --resume --jobs 2 --load 75% --noswap --nice 5 --colsep ',' --arg-file parameters.txt
run_test_matlab.sh {1} {2} & echo $! >&3 ) 3>$HOME/parallel.pid | tee parallel.out
```

- Make sure this file is executable. If not, on the command line execute `chmod u=rwx test_parallel_1host.sh`
- This command is explained in deatil in the README.md of this repository and in GNU parallel documentation

Example: Make sure both shell scripts can be found on local path

This allows us to run the shell script inside any folder containing the Matlab script and parameter list. To do this:

- Suppose shell scripts are in the folder `~/work/math_server_talks`
- Open the `.bashrc` file e.g. `gedit ~/.bashrc`
- Modify the PATH to include this folder e.g.

  ```
  export PATH=$HOME/local/bin:$HOME/work/math_server_talks:$PATH
  ```
- Reload your .bashrc file so that the PATH is updated

  ```
  . ./.bashrc
  ```

Example: Run GNU parallel

- Make sure `test.m` and `parameters.txt` are in the current working directory (~/`work/math_server_talks/example1`)
- Execute `test_parallel_1host.sh` on the command line to run 5 Matlab simulations, 2 at a time.
- As one simulation finishes GNU parallel automatically spawns the next simulation in the queue
- Execute `clean_stdout.sh` to extract the data displayed in standard out and save it in a comma separated list (if you want)