Using GNU parallel: An introduction

Emma McIvor

University of Nottingham

March 20, 2019

What is GNU parallel and why use it?

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

Example: Creating a basic Matlab script in parallel

This Matlab script (`test.m`) does a simple addition and multiplication which is saved to a file:

```
% 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’

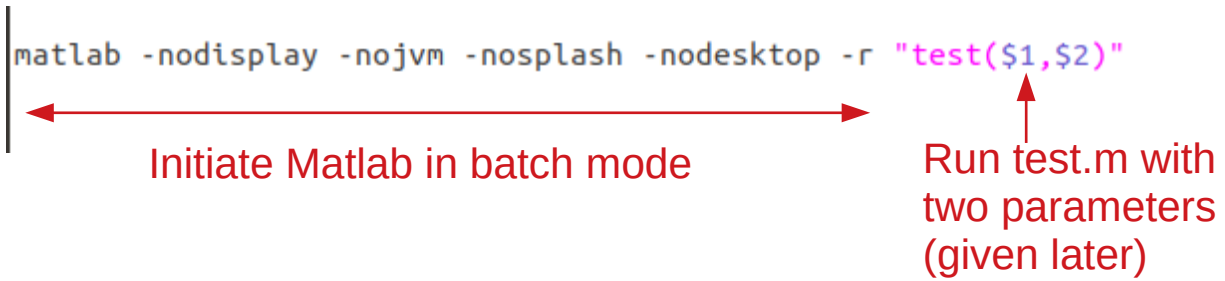
Execute some commands to be saved to a file

Save simulation output to a specific file

Display simulation output

Example: Creating 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: Creating a parameter file with all necessary parameters

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

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

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

- Create a new shell script (`test_parallel_1host.sh`) to run the Matlab simulation simultaneously:

```
#!/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`
- `--jobs 2 --load 50% --noswap --nice 5` means run a maximum of two jobs concurrently on each server, each with a nice value of 5 (sets priority of jobs) and only start a new job if the load on the machine is less than 75% (considers the number of CPUs) don't start jobs if the system is swapping

Example: Add folder with shell scripts to $PATH

- This allows us to run the shell script from any folder

Example: Run GNU parallel

- Make sure `test.m` and `parameters.txt` are in the current working directory
- 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