How to run Python scripts multiple times if more computational power/speed is required

Summary

I will provide a tutorial in creating and running a Python script multiple times on a high-performance computing cluster using the SLURM workflow management, aimed at Arts & Humanities researchers.

Prerequisites

- Familiarity with running Python scripts multiple times in Bash script using a for loop on your local machine (see my previous tutorial on GitHub at https://github.com/caterina159/Learning-materials-for-Bash regarding an introduction to for loops in Bash script)
- Access to a large computing system (cluster)
- Familiarity with working on virtual environments
- Familiarity with accessing and exploring resources on GitHub

Process

Set up the virtual environment

To get started, you need to set up an account with a username and a password to connect to the computing system. After setting up the account, it may be necessary to transfer the files you are working with on the cluster.

A way of doing that is to create a virtual environment on the cluster, install the necessary dependencies and either download the necessary files or transfer them from another machine, such as the personal laptop. I created an account on the NCC cluster of Durham University, UK, created a virtual environment, installed the necessary dependencies there (such as a version of Python and some packages for my script, such as Pandas) and then transferred the files from my laptop to the environment. The way in which I created the virtual environment was done using this resource https://www.arch.jhu.edu/python-virtual-environments/. The next step is to create a Bash script using the SLURM workflow management.

Create the script

The script used the same data as my previous tutorial used – csv files with personal information from 17th century historical documents (see the Prerequisites section of this tutorial for a link to the previous tutorial).

However, the script also needs to contain information regarding the resources the cluster will need to run it, such as an estimation of running time (for the script, it was 10 seconds), the type of processor necessary to do the work (a cpu in my case), the resources to be allocated for the task (1 node with 1 core here), the reason for running the script (for example, to debug it, as it was here). If the script runs in a virtual environment, the script should also contain the code the activate the virtual environment. When running multiple scripts on the cluster, a meaningful name for the task of running the script is recommended.

The script

The script runs a Python script in a for loop, because the values of one argument of the Python script are contained in a sequence of numbers.

The script' code

#!/bin/bash

#SBATCH -N 1 – the node

#SBATCH -c 1 – the core

#SBATCH -t 00-00:00:10 – the time

#SBATCH -p cpu – the processor type

#SBATCH --qos=debug – the type of job the script is run for

#SBATCH --job-name=forloopLessYears

source /home3/lxps38/test_env/bin/activate – activate the virtual environment

years=('1610' '1620' '1630' '1640' '1650' '1660' '1670' '1680' '1690' '1700')

for year in ${years[@]}

do

    python network_python.py --birth_year $year --gender male

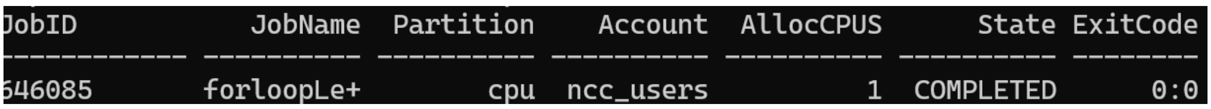
done

Run the script on the cluster

The finished script is then ready to be run, but it will likely be part to a queue, waiting to be run, because multiple users run their scripts on the cluster. To put the script in the queue, the **sbatch** command is used.

The **sacct** command shows the progress of running the script on the cluster. It helps in monitoring the progress and it is especially helpful when having multiple scripts to run or scripts which take longer time of running to complete.

Figure 1 below shows that the job of running my script, starting with the words **forloopLe**, has finished (on the **State** column you have the status of COMPLETED)



```
JobID           JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
646085        forloopLe+        cpu  ncc_users          1  COMPLETED      0:0
```

Figure 1.

If, however, the script fails to run as expected, you need to open the file containing information if the script ran as expected (for this script, see *The script's code* section of this tutorial to find it). Here, it is empty, therefore no errors were found. Also, obviously, you should check the output your script produced to see if it is what you expect it to be.

Take-away points

- SLURM is a workflow management used on high-performance computing clusters, to manage the running of scripts
- Care should be taken to wisely allocate resources when running your scripts, be they time, memory space or processors to ensure that you have enough resources to complete the running of your scripts, but also ensure that you do not rob the other users of their resources