Quick start: Run code on Argon cluster

1) Login into argon server, the command is “ssh [your hawkid]@argon.hpc.uiowa.edu”. If you are using out-of-campus network, use port 40 by adding “p -40” in the command. For example in my case, I use “ssh -p 40 dthong@argon.hpc.uiowa.edu".

```
HONGs-MacBook-Pro:~ dathong$
HONGs-MacBook-Pro:~ dathong$ ssh -p 40 dthong@argon.hpc.uiowa.edu
ECDSA host key for IP address '128.255.1.87' not in list of known hosts.
dthong@argon.hpc.uiowa.edu's password:
Duo two-factor login for dthong

Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-5072
 2. Phone call to XXX-XXX-5072
 3. SMS passcodes to XXX-XXX-5072 (next code starts with: 2)

Passcode or option (1-3): 1
```

2) Create a simple program for test, here I create a python file called “hello.py”. You can do it by any text editor, I used vim. Start by typing “vim hello.py”:

```
[dthong@argon-login-1 ~]$ vim hello.py
```

```
print("start...")
f = open("hello_world_output.txt","w")
print("writing...")
f.write("hello_world")
f.close()
print("done")
```

3) There are 3 ways you can run this code, the simplest way is running python explicitly on the current node by typing “python3 [script name]”

```
[dthong@argon-login-1 ~]$ python3 hello.py
start...
writing...
done
[dthong@argon-login-1 ~]$
```

You should be able to see the output file afterwards.

```
-rw-r--r--  1 dthong its-rs-user 141 Sep 22 19:00 hello.py
-rw-r--r--  1 dthong its-rs-user  11 Sep 22 19:02 hello_world_output.txt
```

Note: If your python is not the right version, you can type “module avail” and load your desired module. For example here, I’m loading python 3.6.4

```
[dthong@argon-login-1 ~]$ module avail

 python/3.6.4_parallel_studio-2017.4
 python/3.6.4
 python/3.7.0_openmpi-2.1.2_parallel_studio-2017.4


[dthong@argon-login-1 ~]$ module load python/3.6.4
```

The second way is to run as an underlying process, which means you can log out and get the results later. Instead of using “python3 [script name]”, you type “nohup python3 [script name] > [log_file] &”:

```
[dthong@argon-login-1 ~]$ nohup python3 hello.py > log_hello &
[1] 32435
[dthong@argon-login-1 ~]$ nohup: ignoring input and redirecting stderr to stdout

[1]+  Done                    nohup python3 hello.py > log_hello
```

The file “log_hello” will record all of your “print” outputs in the program. This can be helpful for you to collect the results.

The third way (recommended) is to create a job and submit it. This can help you take advantage of all Argon provided resources.

+ Create a script, say “myjob.sh” which contains only the running command:

```
[dthong@argon-login-1 ~]$ vim myjob.sh

python3 hello.py

[dthong@argon-login-1 ~]$ chmod 777 myjob.sh
```

“chmod 777” will assign all read-write-execute rights to all users. You can pick what rights you want. Type “chmod —help” for more information.

Now submit the job to Argon’s scheduler. Here, I just use the “qsub” command. There are many other options for you. More details can be looked up on https://wiki.uiowa.edu/display/hpcdocs/Argon+Cluster.

```
[dthong@argon-login-2 ~]$ qsub myjob.sh
Your job 72932 ("myjob.sh") has been submitted
```

After the job finishes, Argon will generate the output files for you:

```
-rw-r--r--  1 dthong its-rs-user   0 Sep 22 19:40 myjob.sh.e72932
-rw-r--r--  1 dthong its-rs-user  11 Sep 22 19:40 hello_world_output.txt
-rw-r--r--  1 dthong its-rs-user  25 Sep 22 19:40 myjob.sh.o72932
```

For example, open file “myjob.sh.o…”. You will see exactly the output of your program.

```
start...
writing...
done
```

4) Download and run the script from our course webpage:

Because most of our homework require you to adjust the existing code. This section gives you example how to download and run the code from our webpage (or any websites). You can upload the code from your local machine to Argon cluster too.

First, download the file you want to run from website. For example if I want to download “mlnnSGD.py” file, i’ll copy its link:

CS:4980:006 Deep Learning Assignment 3: Due 9/20/2018

ı program for creating multi layer general neural networks: mlnnSGD.py and its application mnist.py to the MNIST data set mnist.pkl.gz .

fore we update the weights for each mini batch, we mutiply the weig . For the default architecture owing values for *miu*: 0.001, 0.005, 0.01, 0.05, and 0.1, by using the l ectively. Please record the error can you draw from this experiment?

vork with the architecture (784, 60, 30, 10), using the quadratic cost f 0 and the number of epochs = 10

Open Link in New Tab
Open Link in New Window
Open Link in Incognito Window
Save Link As...
Copy Link Address

Then in Argon linux console, i’ll just use “wget” command to download it. For example I would like to download to [my home directory]/deep_learning/hw3:

```
[dthong@argon-login-2 hw3]$ pwd
/Users/dthong/deep_learning/hw3
[dthong@argon-login-2 hw3]$
[dthong@argon-login-2 hw3]$
[dthong@argon-login-2 hw3]$ wget -P ./ http://www.cs.uiowa.edu/~hzhang/c196/codes/mlnnSGD.py
```

```
[dthong@argon-login-2 hw3]$ ls -lrt
total 24
-rw-r--r-- 1 dthong its-rs-user 8959 Aug 31 15:32 mlnnSGD.py
[dthong@argon-login-2 hw3]$
```

Repeat the same steps for mnist.py and mnist.pkl.gz. Note: Because you can't use matplotlib to visualize on Argon, you need to comment out all visualization code.

Update: how to run it with our new tutorial script

If you took the Argon cluster training from Dr. Sai Ramadugu (saikumar-ramadugu@uiowa.edu), you would have been introduced the new submission script sleeper.sh. To use this script, just follow these steps:

+Change the name of the script to submission.sh (optional, just "sleeper" name sounds sleepy to me)
+Load the right python module:

```
[dthong@argon-login-2 hw3]$ module load python/3.6.4
[dthong@argon-login-2 hw3]$
```

After changing the configuration as you want, update the submission.sh:

```
/bin/echo Running on compute node: `hostname`.
/bin/echo In directory: `pwd`
/bin/echo Starting on: `date`

#Sleep (wait and do nothing) for 60 seconds
#sleep 60

#Print the end date of the job before exiting
#echo Now it is: `date`
python mnist.py
#####End Compute Work#####
```

Then submit it:

```
[dthong@argon-login-2 hw3]$ qsub submission.sh
Your job 193831 ("sleeper") has been submitted
[dthong@argon-login-2 hw3]$
```

After finishing, open the output file, in my case it is submission.o193831:

```
Running on compute node: argon-mm-compute-5-14.hpc.
In directory: /Users/dthong/deep_learning/hw3
Starting on: Fri Sep 28 15:19:24 CDT 2018
The 75th digit is 7.
Creating Network = [784, 20, 10]
weight shapes: [(784, 20), (20, 10)]
Epoch 0 : 5230 / 10000
Epoch 1 : 3490 / 10000
Epoch 2 : 3434 / 10000
Epoch 3 : 3370 / 10000
Epoch 4 : 3369 / 10000
Epoch 5 : 3355 / 10000
Epoch 6 : 3346 / 10000
Epoch 7 : 3332 / 10000
Epoch 8 : 3341 / 10000
Epoch 9 : 3334 / 10000
run time: 22.886232614517212 seconds
```

Congrats! Now you can submit the job, let it run, get a meal then return to get the results!