


An Introduction to Parallel Programming in Python

Nick Featherstone
CU Research Computing



Who are we?
Why are we here?

- Nuts & Bolts of Parallel Python
 - Programming with NumPy
 - IPyParallel
 - MPI4PY

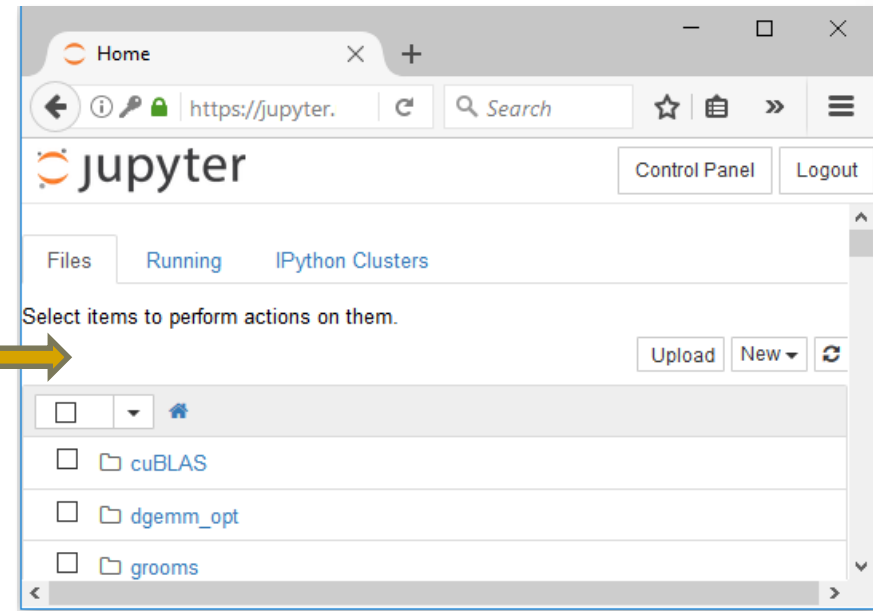
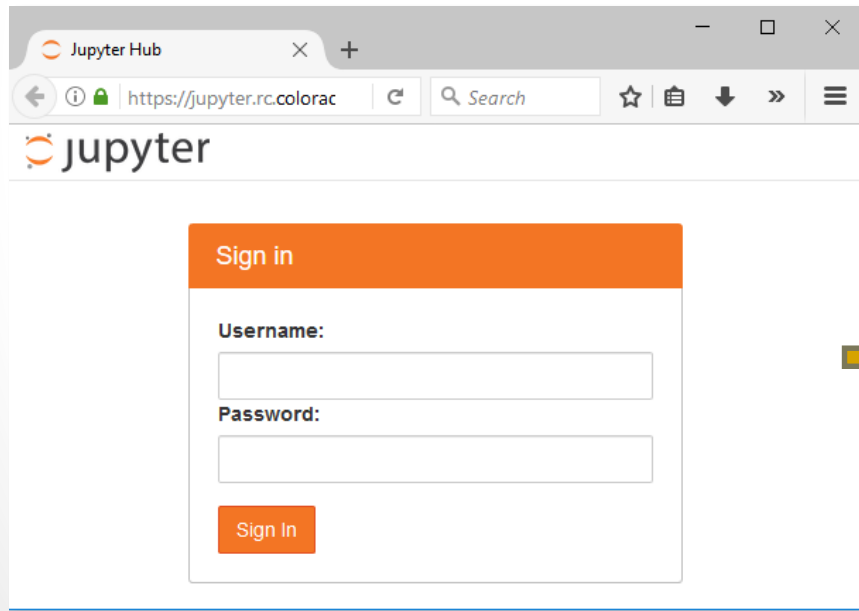
Workshop Notes

- Literally a ‘nuts and bolts’ introduction to parallel programming in Python
- This is, far and away, a non-exhaustive introduction...
- When finished, you should have enough tools in your toolbox to start parallelizing your own projects
- Each session has three directories:
 - examples: working programs that we discuss
 - exercises: partially working programs that you modify
 - exercises/solutions: working solutions to exercises

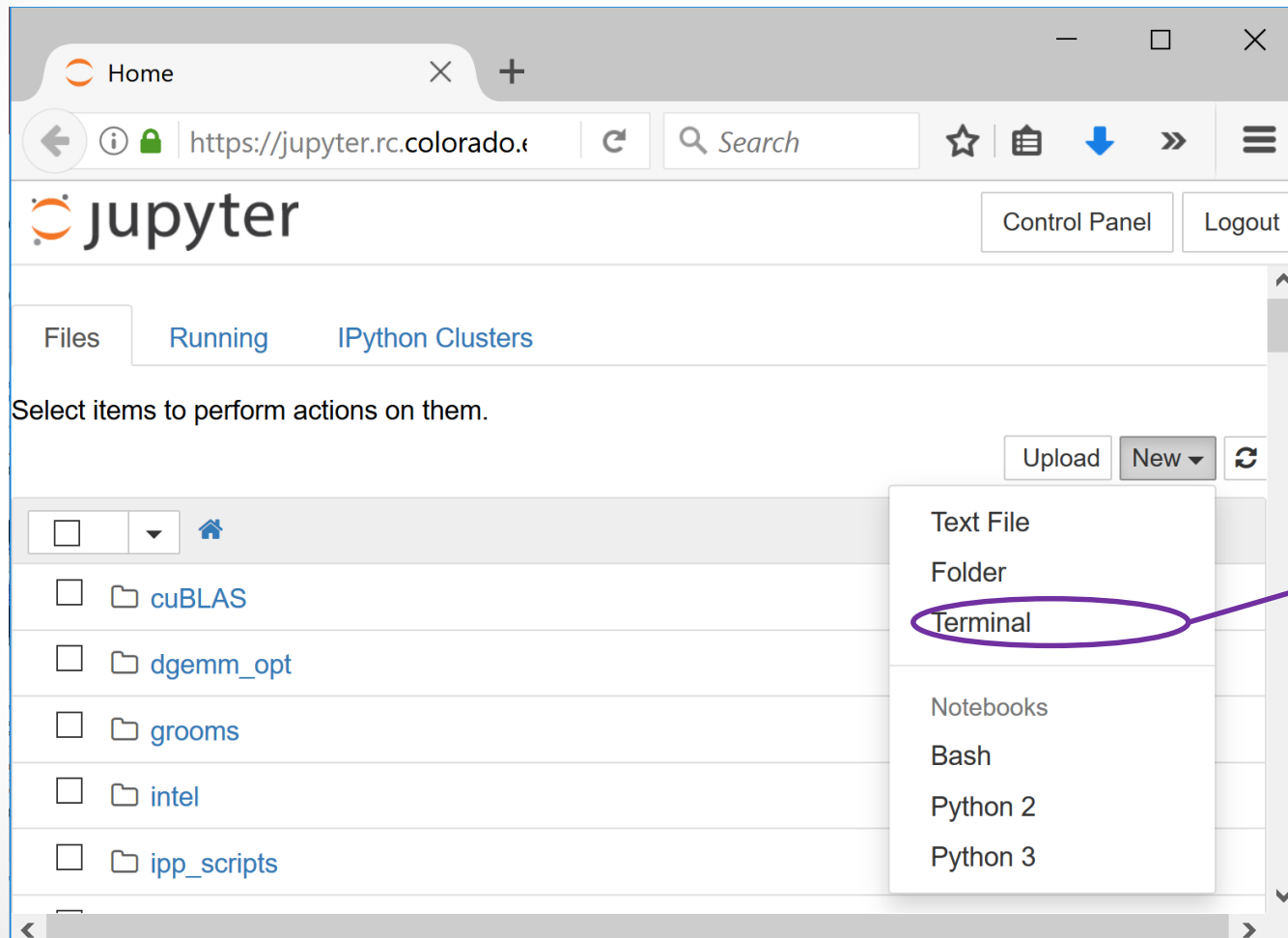
Getting Started

- Login to the RC Jupyter Hub:

`https://jupyter.rc.colorado.edu`



Getting Started...



Open a terminal

Getting Started...

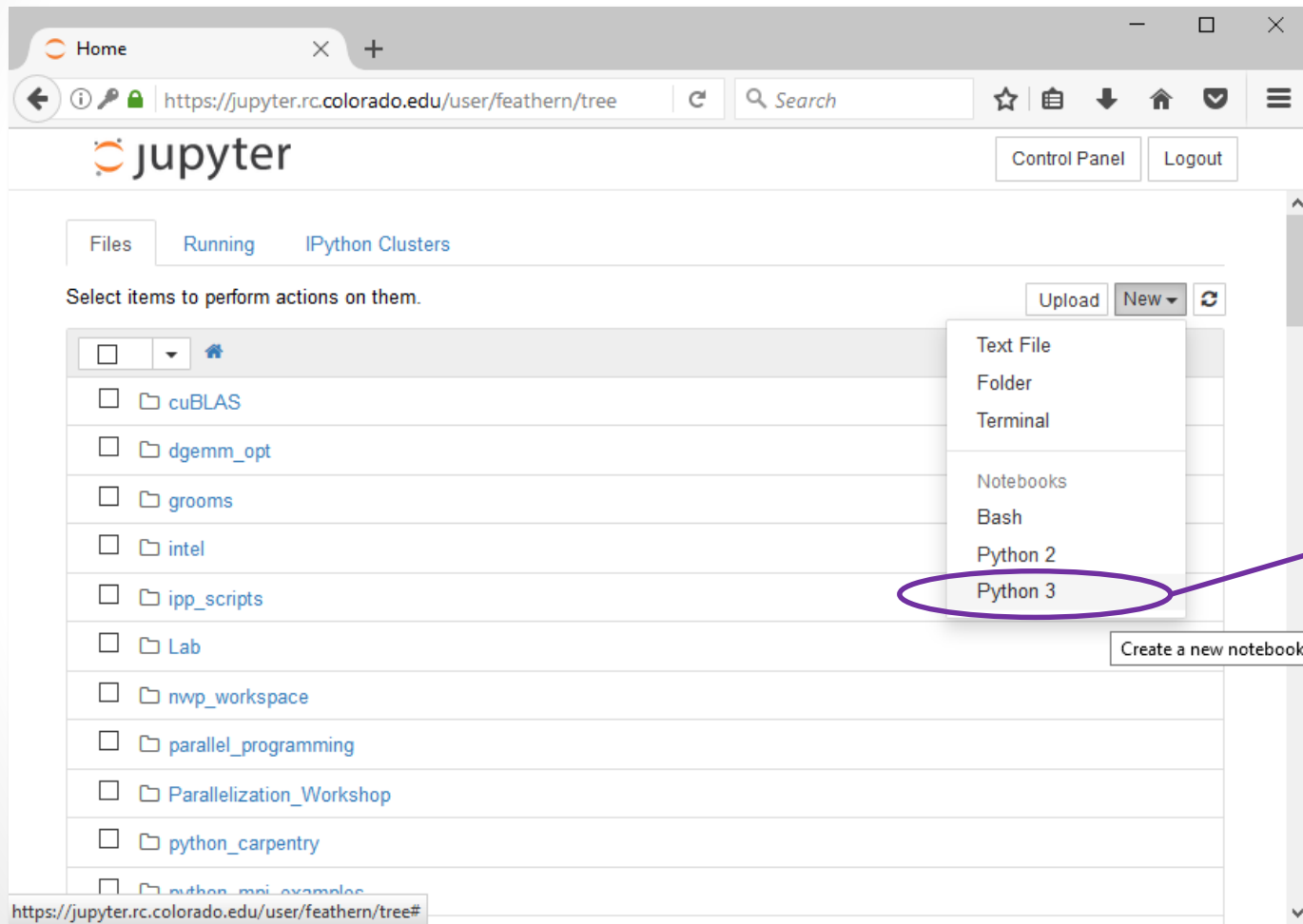
- Clone the repository (type all one line):

```
git clone git@github.com:
```

```
ResearchComputing/Parallelization_Workshop.git
```

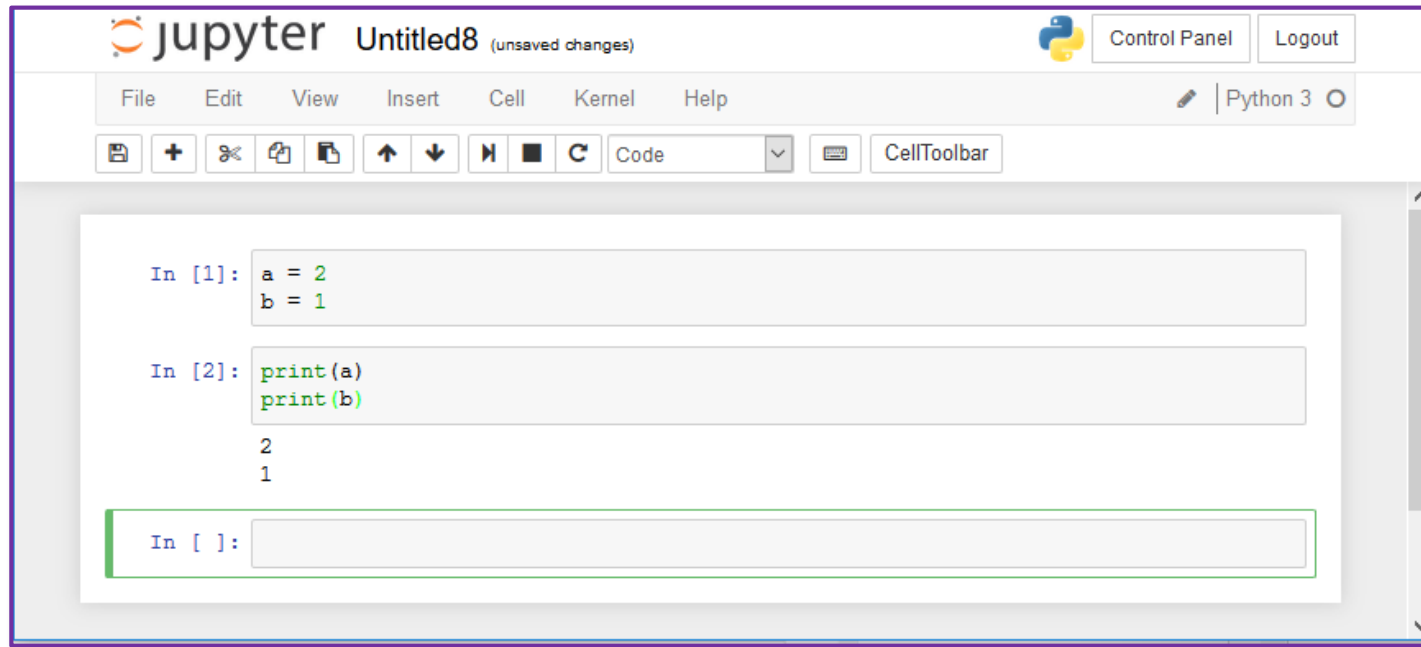
- Type 'exit'
- Close your terminal tab

Getting Started...



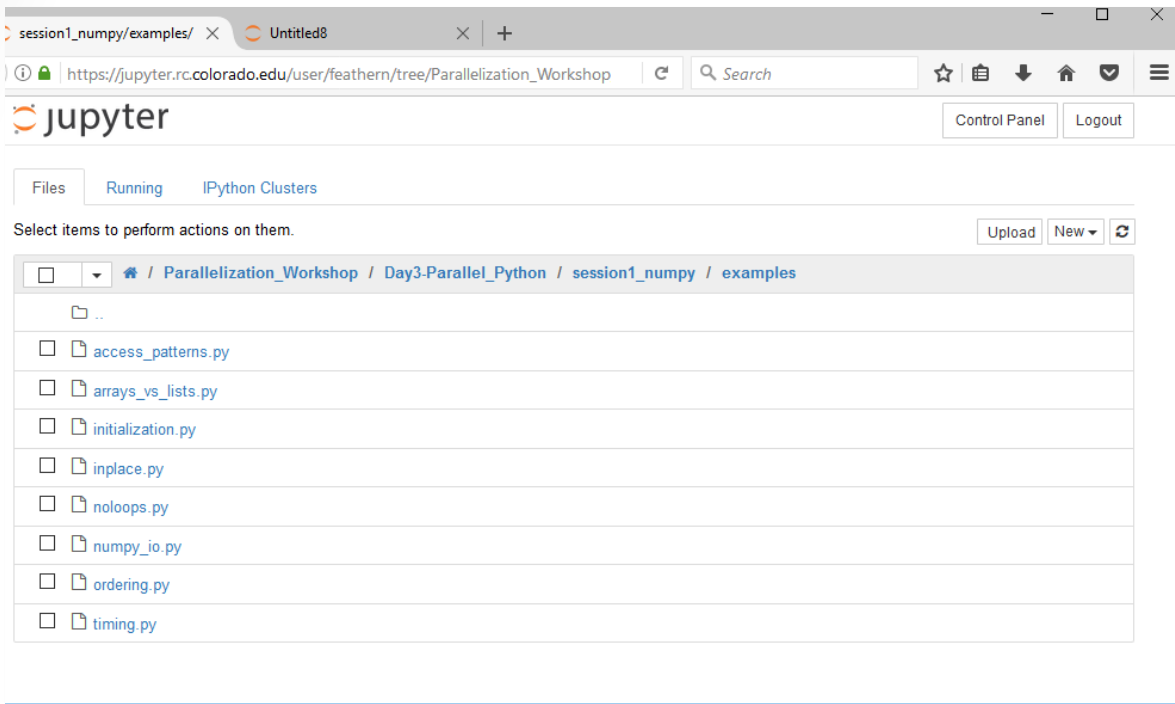
Start a Python 3
Notebook

So how does this work?



- Pressing “enter” moves to next line
- Pressing “shift” + “enter” executes code block
- Variables remain in memory between blocks...

File browser tab remains open...



Open this file:

Parallelization Workshop /
Day3-Parallel_Python /
session1_numpy /
examples /
timing.py

Workflow for today:

- Open file in file browser
- Cut + paste into notebook tab
- “shift” + “enter”

Timing in Python...

- Timing via “time” module
- Let’s look at **timing.py**
- `time()` returns seconds elapsed since some reference time.

```
import time
```

usage pattern

```
t0 = time.time()
```

```
... code you want to time ...
```

```
t1 = time.time()
```

```
dt = t1-t0
```

```
print ('Calculation time in seconds: ', dt)
```

[Open this file:](#)

Parallelization Workshop /
Day3-Parallel_Python /
session1_numpy /
examples /
timing.py