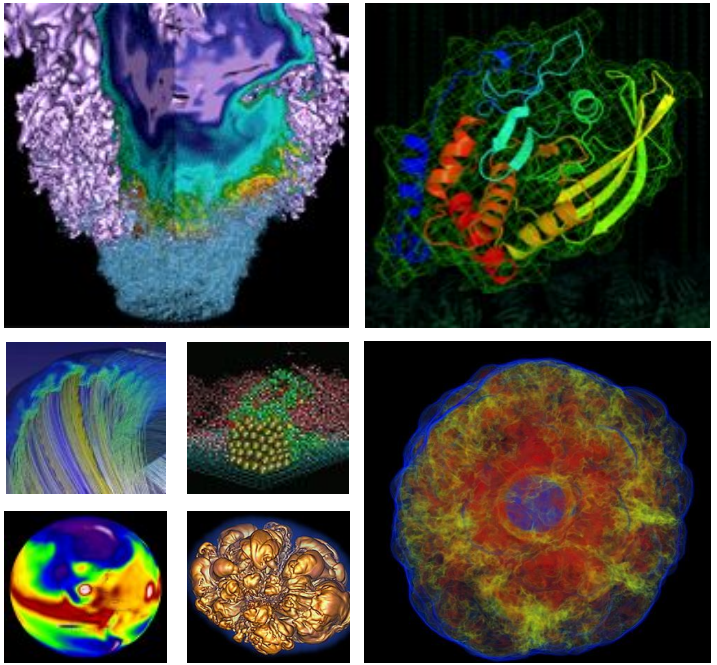
Python and Jupyter at NERSC



Rollin Thomas

Data Architect

Data and Analytics Services, NERSC

New User Training, 2019-06-21

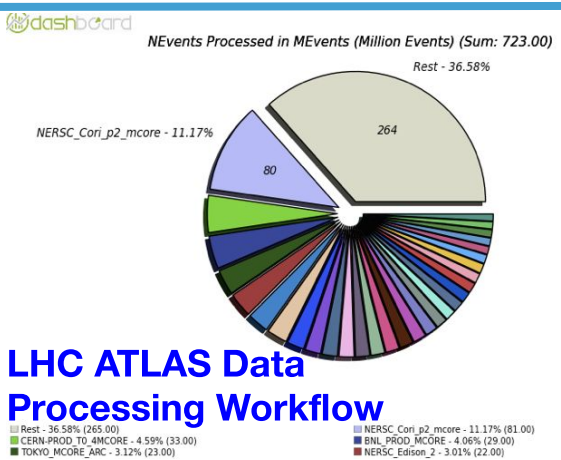
U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

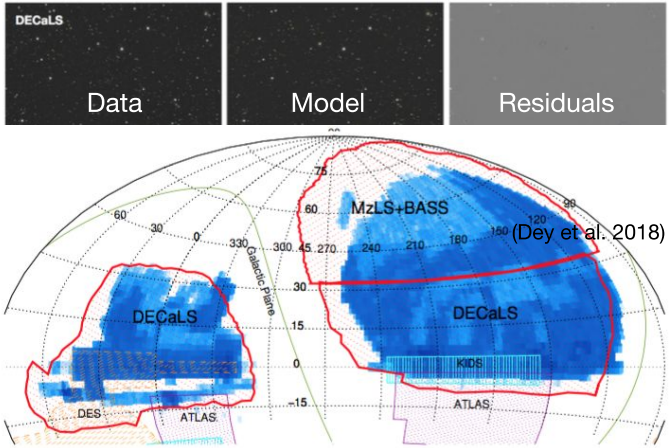
Science via Python@NERSC

The Materials Project

Powering Workflows to Understand Properties of Materials



LHC ATLAS Data Processing Workflow

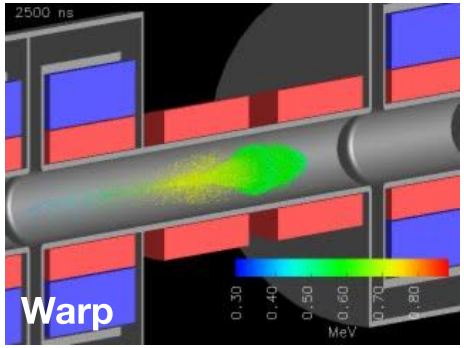


(Dey et al. 2018)

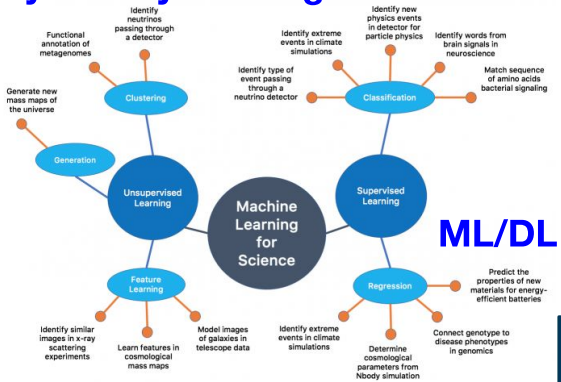
Sky Survey Catalogs for Cosmology

NBODYKIT

Modeling Dark Matter and Dark Energy



PIC Code for Plasmas and High Current Particle Beams



ML/DL

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

NERSC Python Documentation

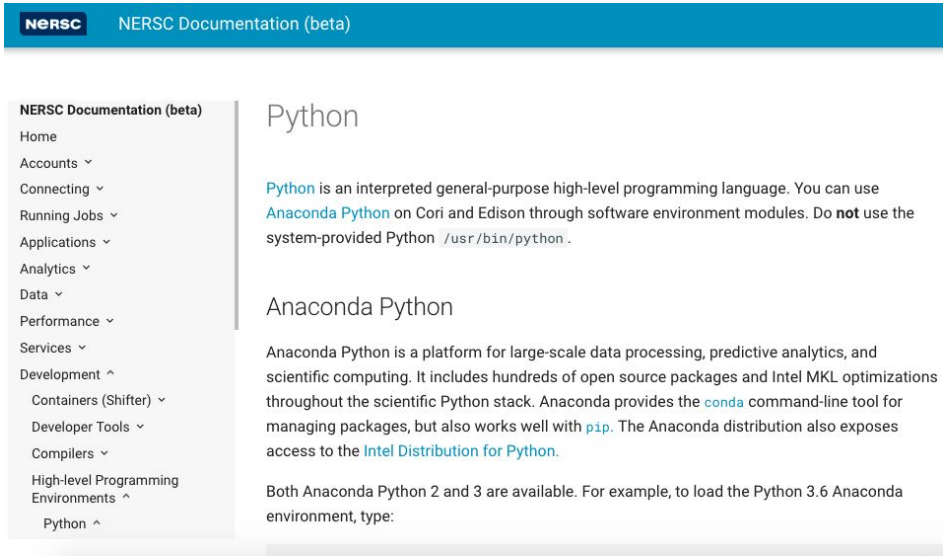
Good docs advise on how to use Python at NERSC.

Updates are ~continuous.
[Main page.]

Frequently Asked Questions
[FAQ page.]
Suggest new questions!

Advice/gotchas for KNL users
[KNL page.]

Advice on [optimizing Python.]



New site: [docs.nersc.gov]

Use Environment Modules

Environment modules:

Environment modules project:
http://modules.sourceforge.net/

Always* “module load python”

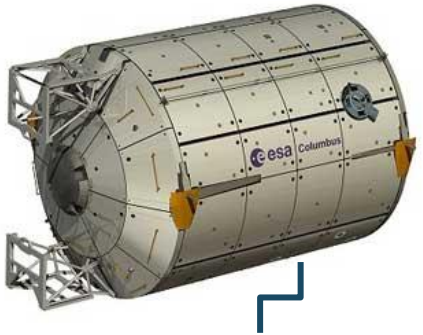
Do not use /usr/bin/python.

Using #!/usr/bin/env python is OK!

What is there?

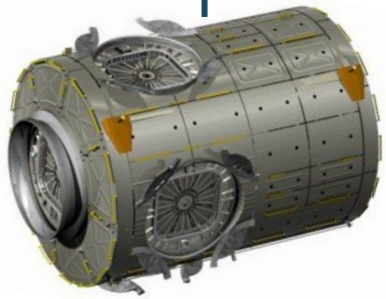
module avail python

*** Unless you install your own somehow.**
(Totally fine, see later in the talk.)



python/2.7-anaconda-4.4

python/3.6-anaconda-4.4



NERSC's Python is Anaconda

NERSC's builds of Python on Cori have been retired for a while.

Modules Now Leverage Anaconda Python

- Distro for large-scale data analytics and scientific computing.
- Handy package management and deployment (conda tool).
- Conda environments replace virtualenv.

Hundreds of useful packages (400+ already installed)

- Threaded Intel MKL comes for free.
- Now with some ML tools too.
- Additional "channels" and you can still use pip.

https://docs.anaconda.com/anaconda/

They are monolithic; with some add-on modules (h5py-parallel).

NERSC Python Modules

Recommended environment modules at NERSC for Python users:

```
module load python/2.7-anaconda-4.4
module load python/3.6-anaconda-4.4
```

Default is 2.7 up to no later than *6 months!* from now:

```
module load python
 [= module load python/2.7-anaconda-4.4]
```

Conda Environments

Conda makes it easy to create tailored environments with the packages you need.

```
module load python/3.6-anaconda-4.4
conda create -n myenv python=2 numpy
[installation outputs]
source activate myenv
```

And pip is OK to use too. Note,
"--no-cache-dir" is handy
Don't bother with --user, just pip in your conda env.

Doing Things Yourself

Project-wide Anaconda installation, e.g. at /global/common/software/<project-name>

```
module unload python
unset PYTHONSTARTUP

wget https://repo.continuum.io/miniconda/Miniconda2-latest-Linux-x86_64.sh
/bin/bash Miniconda2-latest-Linux-x86_64.sh -b -p $PREFIX
source $PREFIX/bin/activate
    <or export PATH=$PREFIX/bin:$PATH>
conda install basemap yt…
```

Doing Things Yourself

Building your own mpi4py or parallel h5py?

Do not conda install ...

Do not pip install ...

Link to Cray MPICH, using compiler wrappers

```
wget https://bitbucket.org/mpi4py/mpi4py/downloads/mpi4py-3.0.0.tar.gz
tar zxvf mpi4py-3.0.0.tar.gz
cd mpi4py-3.0.0
module swap PrgEnv-intel PrgEnv-gnu
python setup.py build --mpicc=$(which cc)
python setup.py install
```

Parallelism with Python

Within a node:

Use OpenMP-threaded math libs.

Multiprocessing is OK too.

Multi-node parallelism:

Best supported by mpi4py.

Dask, PySpark work too.

Hybrid parallelism:

Best route is mpi4py + threaded math libs.

Handling MPI with mpi4py

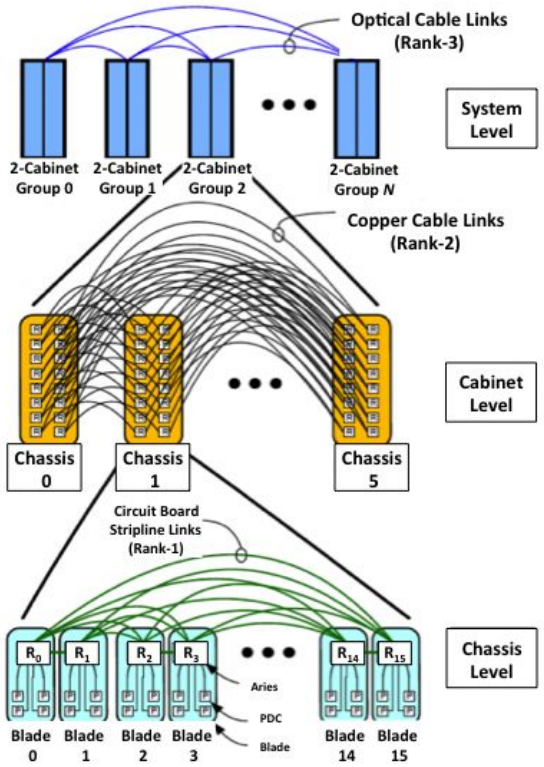
Cluster parallelism with MPI via mpi4py:

- MPI-1/2/3 specification support
- OO interface ~ MPI-2 C++ bindings
- Point-to-point and collectives
- Picklable Python objects & buffers

Build mpi4py & dependents with Cray MPICH:

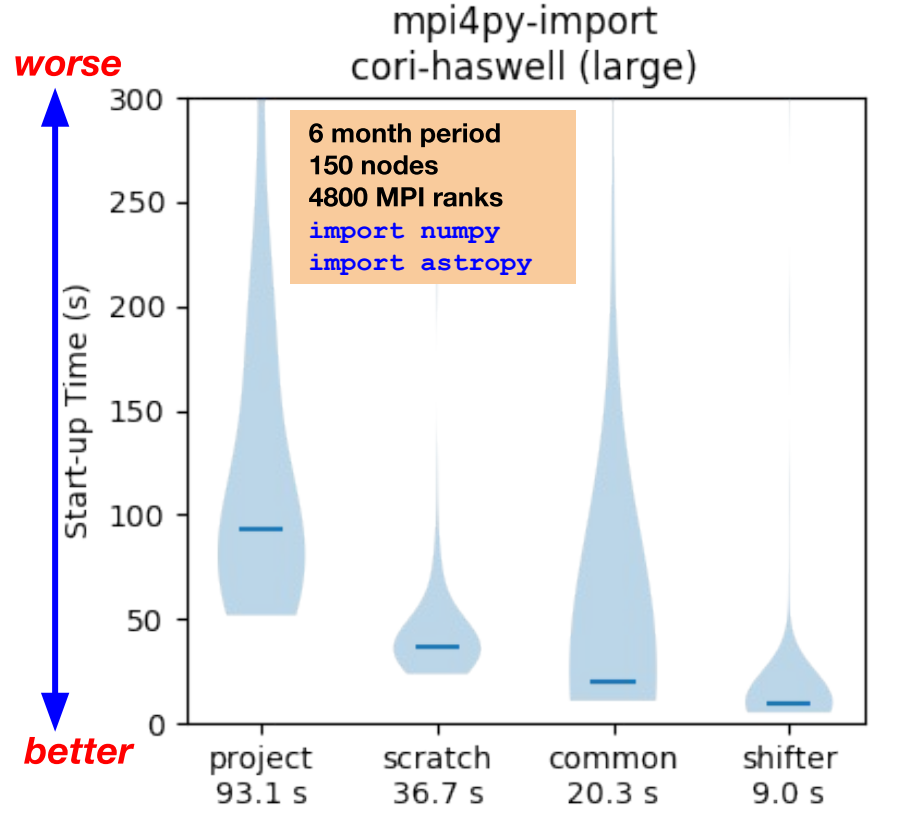
```
python setup.py build --mpicc=cc
python setup.py install
```

Cray-provided Compiler wrapper



Cori Aries Interconnect

Python "Slow Launch" at Scale



[Median launch time incl. MPI_Init()]

Python's import is metadata intensive,
⇒ catastrophic contention at scale
⇒ *it matters where you install your env*

Project (GPFS):
For sharing large data files

Scratch (Lustre):
OK, but gets purged periodically!

Common (GPFS):
RO w/Cray DVS client-side caching
Open to users now, was only staff

Shifter (Docker Containers):
Metadata lookup only on compute
Storage on compute is RAM disk
ldconfig when you build image

SHIFTER

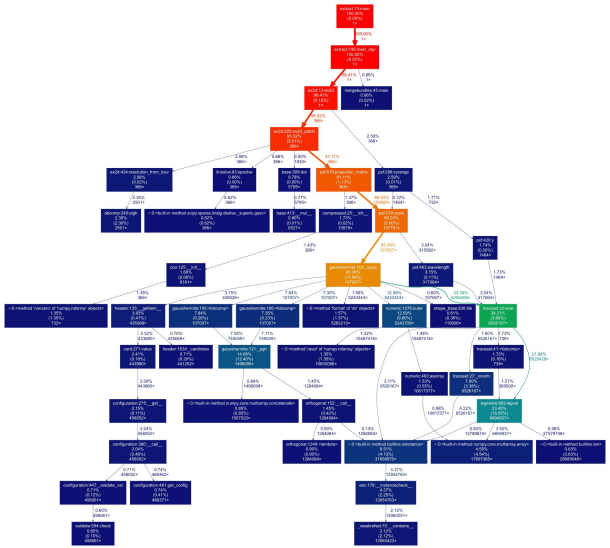
Profiling, Debugging

Ye olde stand-bye, `print()`!

`srun -u python -u <script-name>` …

Unbuffer both srun and python.

Can be a lot of messy output to parse.



Good for general exploration (standard lib):

cProfile plus snakeviz or gprof2dot

MPI processes? [see an example here]

Good for a deeper dive on one function (package):

line_profiler

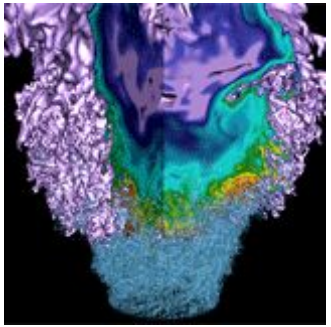
High-performance instrumented timer (mixed-language, MPI, package):

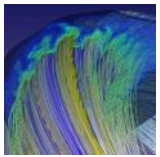
TiMemory

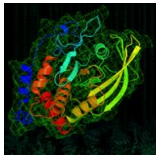
High-performance tools (mixed-language, MPI):

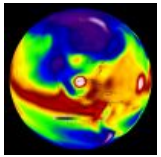
Intel VTune (some collection methods) on Intel Python, and Tau

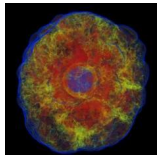
Jupyter at NERSC

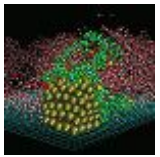












jupyter + NERSC

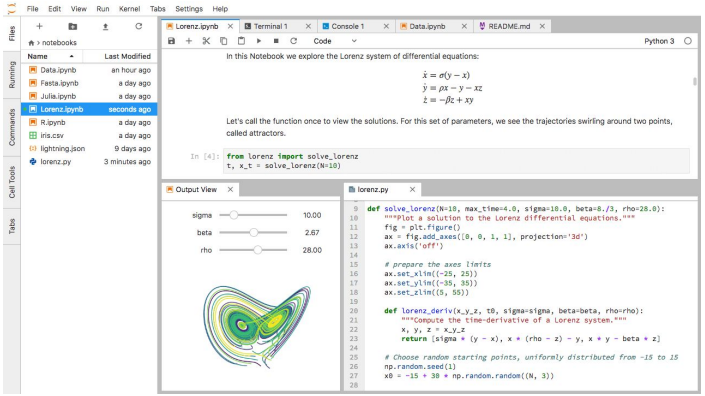
Using Jupyter at NERSC

Jupyter Notebook: "Literate Computing."
Code, text, equations, viz in a narrative.

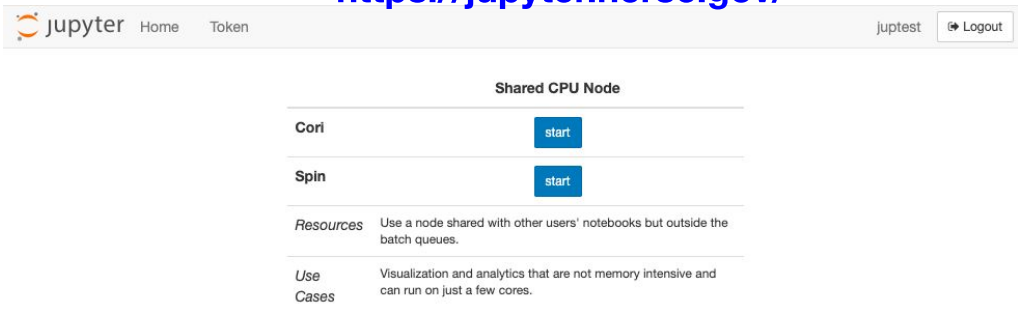
Use JupyerHub for Jupyter at NERSC.
https://jupyter.nersc.gov/

Cori Shared CPU Node:
- Launches notebooks on Cori
- Can see Cori $SCRATCH
- Same Python env as ssh login
- Can submit jobs via `%sbatch`

Spin Shared CPU Node:
- External to Cori
- Can't see $SCRATCH
- But can see /project, $HOME



https://jupyter.nersc.gov/



Your Own Jupyter Kernel

Most common Jupyter question:

"How do I take a conda environment and use it from Jupyter?"

Several ways to accomplish this, here's the easy one.

```
$ module load python
$ conda create -n myenv python=3.6
$ source activate myenv
(myenv) $ conda install ipykernel <other-packages>...
(myenv) $ python -m ipykernel install --user --name myenv-jupyter
```

This creates a "kernelspec" file.

Point your browser to jupyter.nersc.gov.
(You may need to restart your notebook server via control panel).
Kernel "myenv-jupyter" should be present in the kernel list.

The kernelspec File

```
(myenv) rthomas@cori01:~> cat \
    $HOME/.local/share/jupyter/kernels/myenv-jupyter/kernel.json
{
 "argv": [
  "/global/homes/r/rthomas/.conda/envs/myenv/bin/python",
  "-m",
  "ipykernel_launcher",
  "-f",
  "{connection_file}"
 ],
 "display_name": "myenv-jupyter",
 "language": "python"
}
```

Additional Customization

```
{
 "argv": [
  "/global/homes/r/rthomas/.conda/envs/myenv/bin/python",
  "-m",
  "ipykernel_launcher",
  "-f",
  "{connection_file}"
 ],
 "display_name": "myenv-jupyter",
 "language": "python",
 "env": {
  "PATH": …,
  "LD_LIBRARY_PATH": …,
 }
}
```

Additional Customization

```
{
 "argv": [
  "/global/homes/r/rthomas/jupyter-helper.sh",
  "-f",
  "{connection_file}"
 ],
 "display_name": "myenv-jupyter2",
 "language": "python",
}
```

Meanwhile, in jupyter-helper.sh:

```
#!/bin/bash
export SOMETHING=123
module load texlive
exec python -m ipykernel "$@"
```

A Shifter Kernelspec

```
{
  "argv": [
    "shifter",
    "--image=continuumio/anaconda3:latest",
    "/opt/conda/bin/python",
    "-m",
    "ipykernel_launcher",
    "-f",
    "{connection_file}"
  ],
  "display_name": "my-shifter-kernel",
  "language": "python"
}
```

Image name

Path in the image

SHIFTER

Debugging Jupyter Stuff


```
(myenv) rthomas@cori01:~> cat ~/.jupyter.log
```

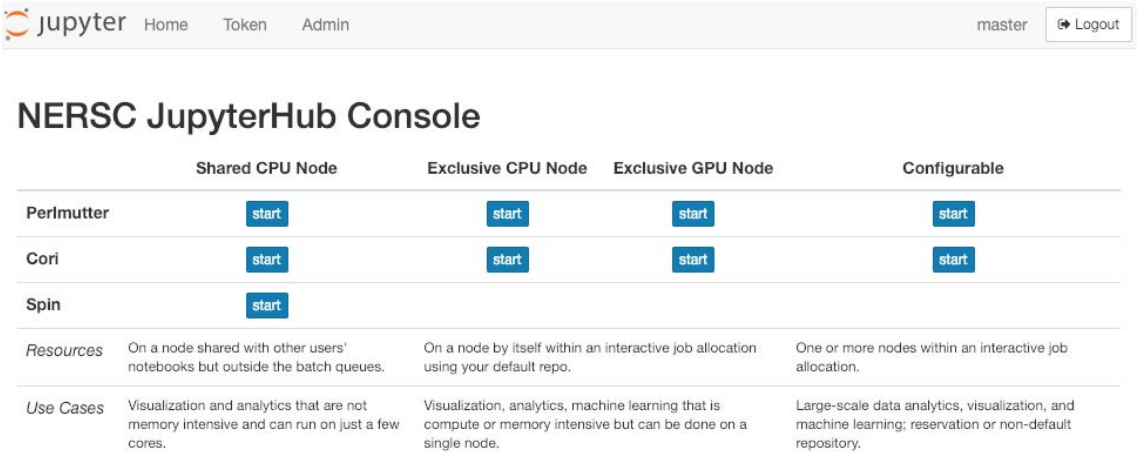
← YOUR FRIEND

```
[I 2018-03-19 16:00:08.175 SingleUserNotebookApp manager:40] [nb_conda_kernels] enabled, 5 kernels found
[I 2018-03-19 16:00:08.248 SingleUserNotebookApp extension:53] JupyterLab beta preview extension loaded from
/usr/common/software/python/3.6-anaconda-4.4/lib/python3.6/site-packages/jupyterlab
[I 2018-03-19 16:00:08.248 SingleUserNotebookApp extension:54] JupyterLab application directory is
/global/common/cori/software/python/3.6-anaconda-4.4/share/jupyter/lab
[I 2018-03-19 16:00:09.123 SingleUserNotebookApp handlers:73] [nb_anacondacloud] enabled
[I 2018-03-19 16:00:09.129 SingleUserNotebookApp handlers:292] [nb_conda] enabled
[I 2018-03-19 16:00:09.181 SingleUserNotebookApp __init__:35] ✓ nbpresent HTML export ENABLED
[W 2018-03-19 16:00:09.181 SingleUserNotebookApp __init__:43] ✗ nbpresent PDF export DISABLED: No module
named 'nbbrowserpdf'
[I 2018-03-19 16:00:09.186 SingleUserNotebookApp singleuser:365] Starting jupyterhub-singleuser server
version 0.8.0.rc1
[I 2018-03-19 16:00:09.190 SingleUserNotebookApp log:122] 302 GET /user/rthomas/ →
/user/rthomas/tree/global/homes/r/rthomas? (@128.55.206.24) 0.62ms
[I 2018-03-19 16:00:09.194 SingleUserNotebookApp notebookapp:1445] Serving notebooks from local directory: /
[I 2018-03-19 16:00:09.194 SingleUserNotebookApp notebookapp:1445] 0 active kernels
[I 2018-03-19 16:00:09.194 SingleUserNotebookApp notebookapp:1445] The Jupyter Notebook is running at:
[I 2018-03-19 16:00:09.194 SingleUserNotebookApp notebookapp:1445] http://0.0.0.0:56901/user/rthomas/
[I 2018-03-19 16:00:09.194 SingleUserNotebookApp notebookapp:1446] Use Control-C to stop this server and shut
down all kernels (twice to skip confirmation).
[I 2018-03-19 16:00:09.236 SingleUserNotebookApp log:122] 302 GET /user/rthomas/ →
/user/rthomas/tree/global/homes/r/rthomas? (@::ffff:10.42.245.15) 0.39ms
```

Near Future Jupyter Support

Working on:

- **Expanding resources to support Jupyter**
- **New ways to launch parallel workloads managed through Jupyter**
- **Expanding JupyterLab interface to:**
  - **Track and monitor batch jobs**
  - **New viewers**



Python and Jupyter at NERSC

Python & Jupyter: integral elements of NERSC's Data Intensive Science portfolio.

We want users to have a:

familiar	Python environment
productive	Python experience
performant	Python software stack

Always looking for:

New ways to empower Python & data science users.
Feedback, advice, and even help:
https://help.nersc.gov/
rcthomas@lbl.gov

Key Take-Aways

- Read our really pretty good documentation on Python
- Use *conda* environments to customize Python
- Process-level parallelism is easier to push than thread-level
- Build mpi4py and h5py against Cray MPICH
- To run a *lot* of Python processes, use Shifter (next talk)

- Go to https://jupyter.nersc.gov to use Jupyter at NERSC
- Use a kernel-spec to use a conda environment in your notebook
- You can customize those kernelspec files in many ways
- We work on making Jupyter work for you

Thank You