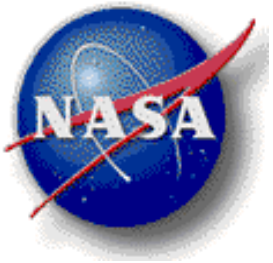
JupyterHub

at the
NCCS



Matt Stroud
Savannah Strong Finch

Agenda

Basic Information

- What is JupyterHub?
- Why is Jupyter useful?
- How have we implemented it in HPC environments?

Access & Usage

- Requesting Access
- Navigating the Interfaces
- Setting Up Custom Environments/Kernels

Demo

- Launching a Session
- NCCS Innovation Lab Demo Notebook
- eViz/iViz

Future & Questions

- Future Plans for JupyterHub Environments
- Additional Resources
- Questions & Open Discussion

What is Jupyter?

JupyterHub

The process responsible for launching and managing access to user's JupyterLab sessions.

"JupyterHub brings the power of notebooks to groups of users. It gives users access to computational environments and resources without burdening the users with installation and maintenance tasks. Users - including students, researchers, and data scientists - can get their work done in their own workspaces on shared resources which can be managed efficiently by system administrators."

JupyterLab

The IDE served to users with support for the notebook format, over 40 programming languages, interactive data visualization tools, and a friendlier interface to our HPC systems.

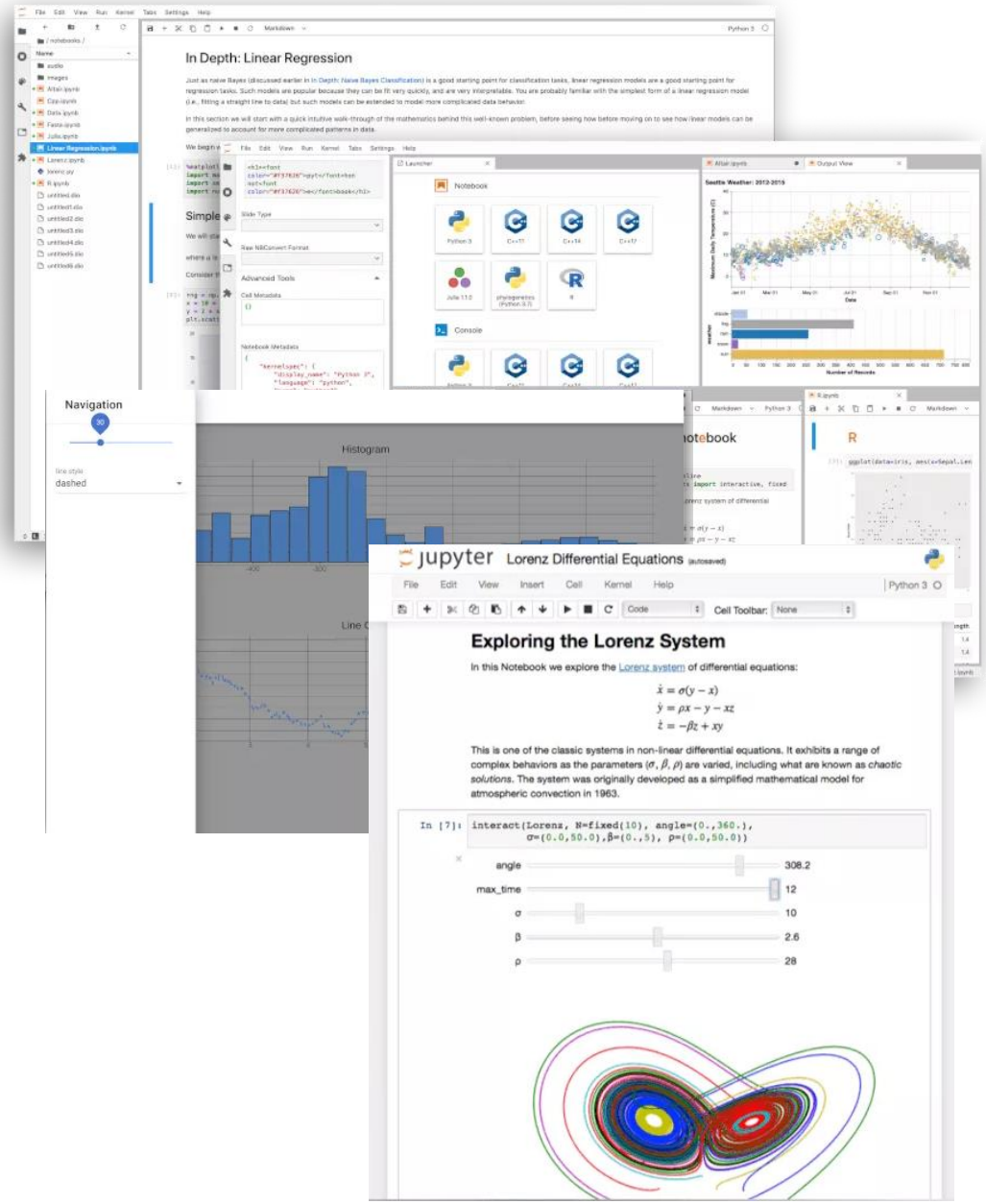
"JupyterLab is a highly extensible, feature-rich notebook authoring application and editing environment, and is a part of Project Jupyter, a large umbrella project centered around the goal of providing tools for interactive computing with computational notebooks."

Source: jupyter.org

Why Jupyter?

The Jupyter Notebook and associated tools are especially useful for collaborative projects, scientific computing, rich and interactive visualizations, contextualizing large data sets, and analyzing output.

At the NCCS we have been working to improve accessibility to our HPC platforms, as well as to accelerate scientific workflows. We believe that interactive tools like JupyterHub are a step in this broader direction.



Source: jupyter.org

JupyterHub on NCCS Systems

We currently have JupyterHubs available on both Discover and ADAPT.

ADAPT has multiple hub instances implemented to support specific user groups. This includes:

- Prism GPU Cluster
- ICESat-2
- Roman Space Telescope
- ABoVE

Our systems have a couple of notable complications that are useful to understand.

Each of these hub instances is tied directly into a back-end Slurm cluster. This allows resources to be allocated directly to user sessions while also serving other jobs and users on a given pool of compute. This can pose unique architectural challenges for our various systems and can sometimes result in odd behaviors when using JupyterHub.

We also work with agency Launchpad integration and network access requirements to help ensure site and account security. This creates some additional steps for users trying to gain access to the hubs.

Requirements to Access

1. **NAMS Request:** A request can be made for either 'NCCS Discover JupyterHub' to access the Discover hub, or 'NCCS Web services' at https://idmax.nasa.gov for any of the ADAPT hubs. (Note: You will only be able to submit NAMS requests from the NASA network/VPN.)

2. **System Request:** Once you have submitted your NAMS request, please send a ticket to support@nccs.nasa.gov stating which specific JupyterHub you would like access to, and we will ensure that you have the necessary LDAP permissions to access the systems.

3. **VPN Access:** Some of our hubs will only be accessible from the NASA network/VPN. This only applies to the RomanST, ABoVE, and the generic ADAPT hubs.

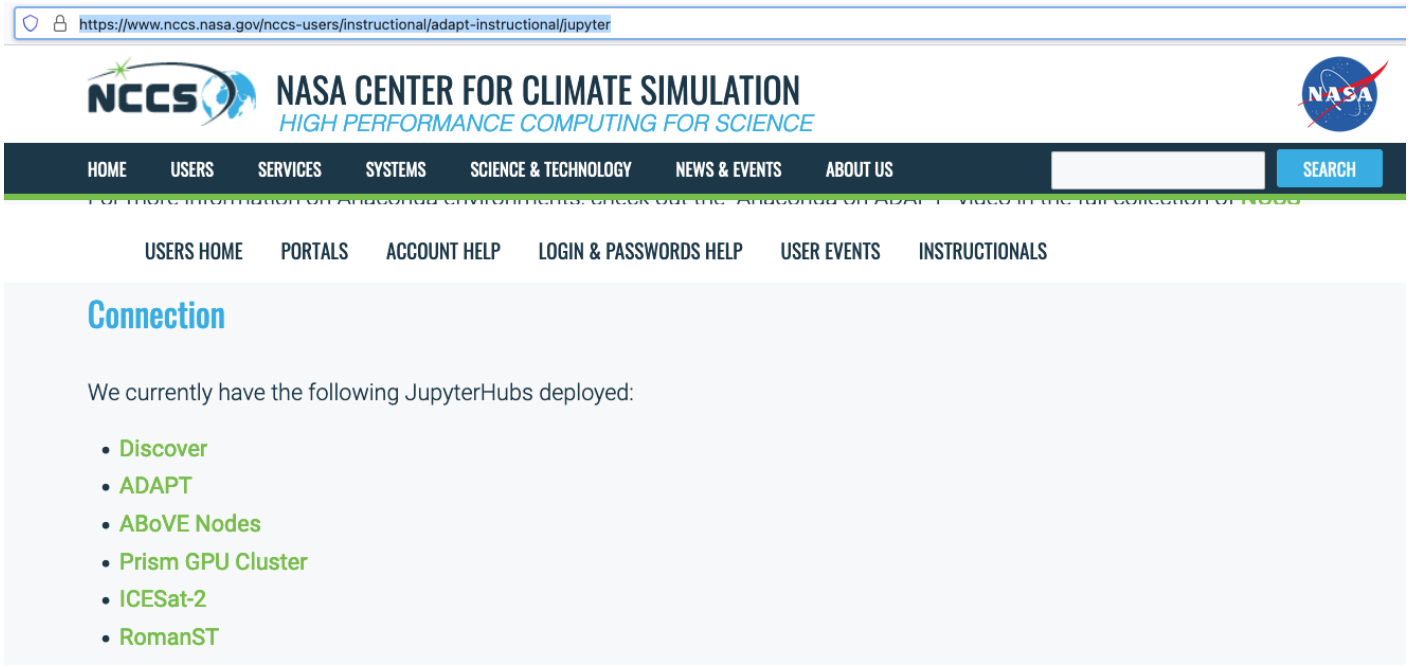
If your group would like a JupyterHub for your team's ADAPT VMs that aren't listed, please send a request to support@nccs.nasa.gov for further inquiry.

Navigate to desired JupyterHub

- You can access JupyterHubs by going to

https://www.nccs.nasa.gov/nccs-users/instructional/adapt-instructional/jupyter

And clicking on one of the desired connections:

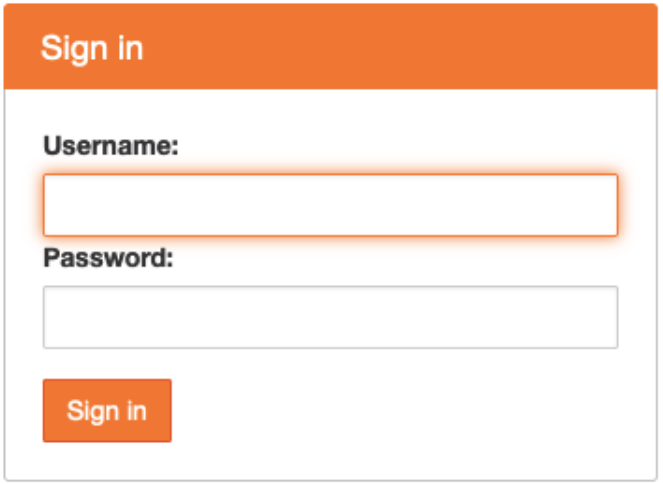


- **Or** by going directly to the URL. Example: https:jh-discover.nccs.nasa.gov

Logging in

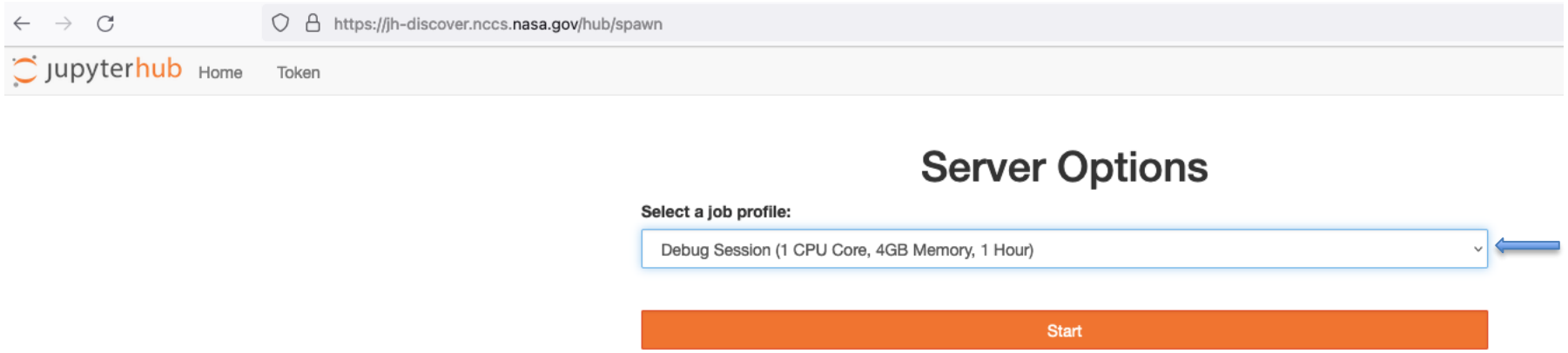
After authentication..

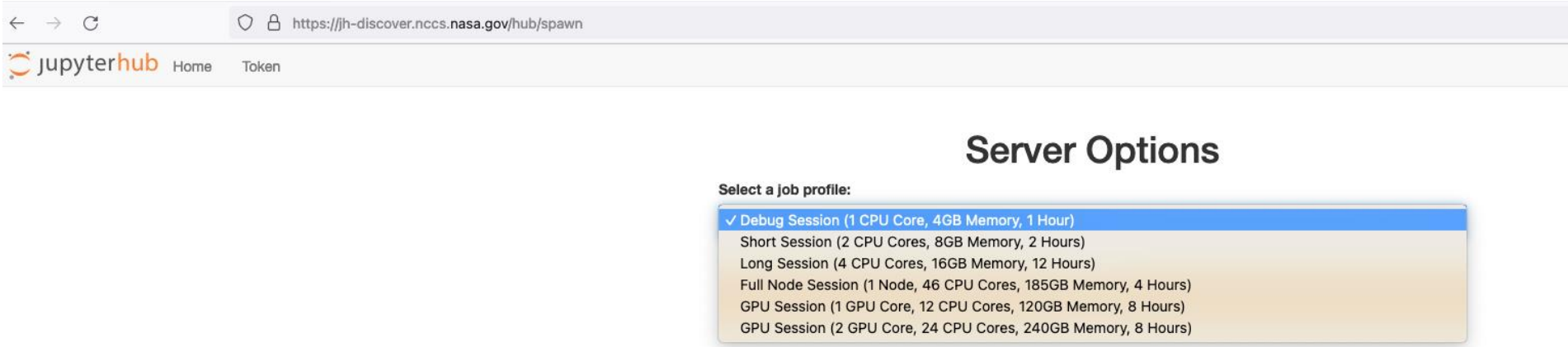
You will be presented with a Sign In page.
(Enter your LDAP username and password)



Start Server

Select your desired Server Option



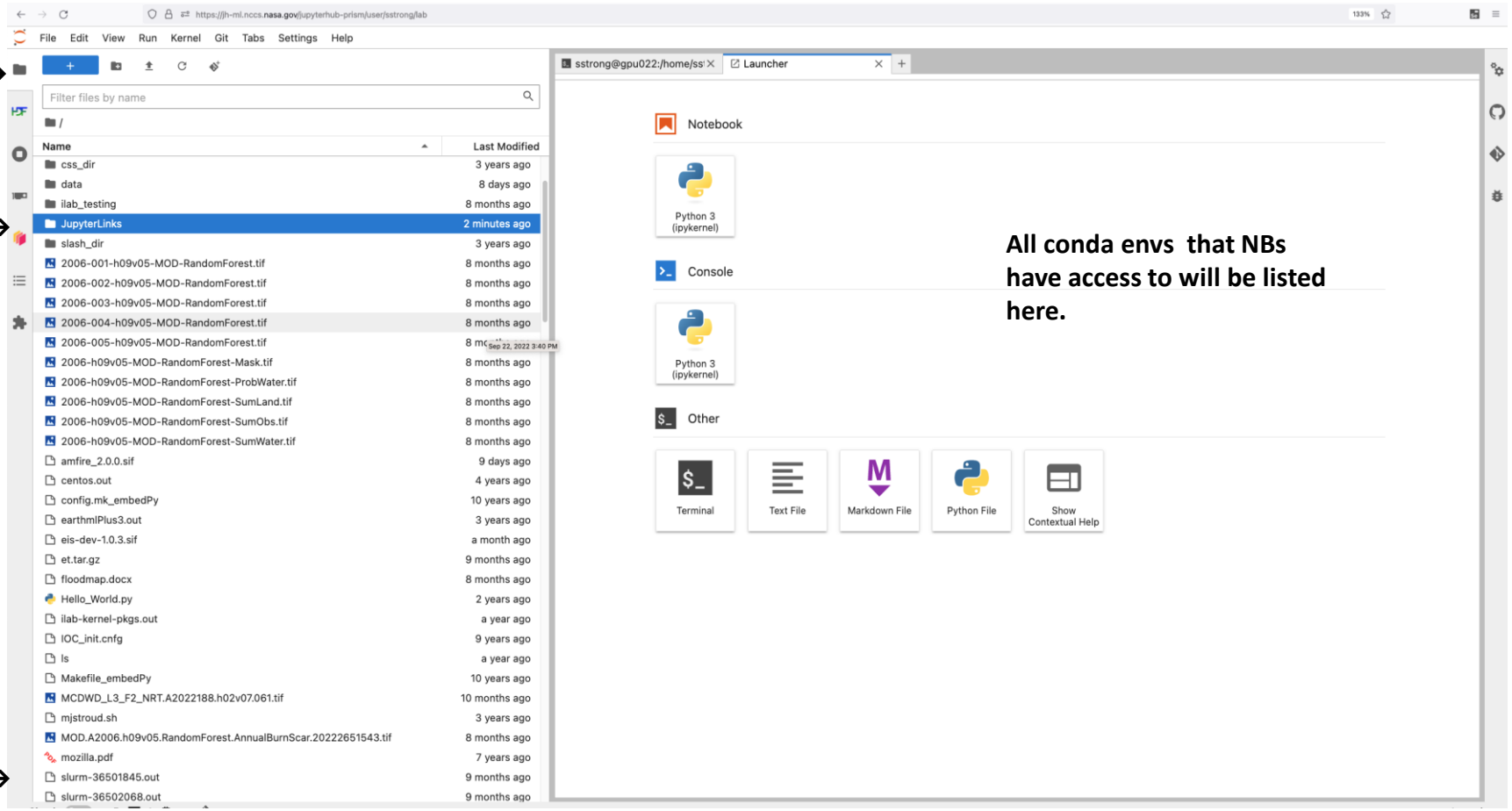


Jupyter lab Console

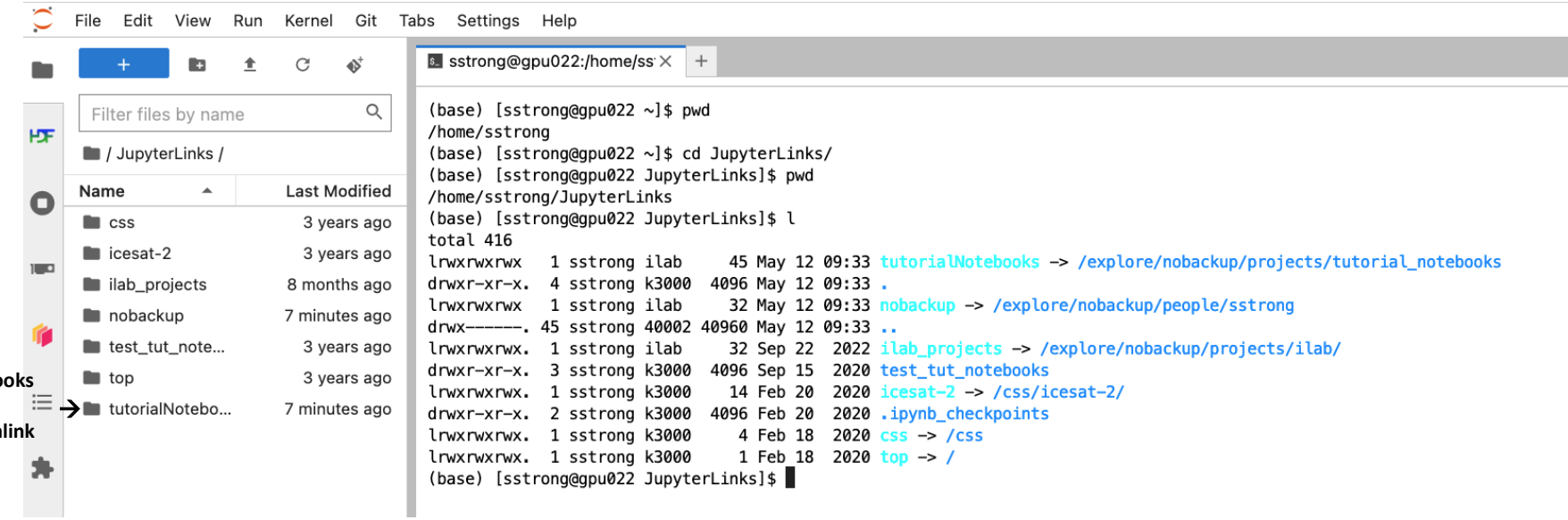
Default directory is /home/<username>

JupyterLinks Dir will be visible If it doesn't exist user session will create it upon startup. User may add other symlinks there as well.

Multiple slurm output files contain log from the user session.



Jupyter lab Console

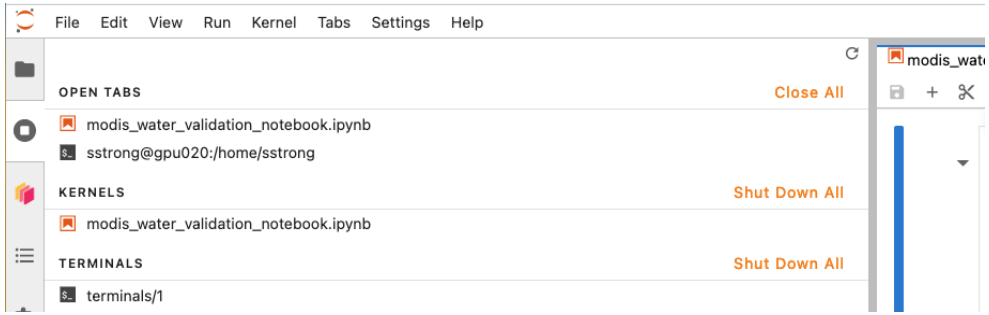


tutorialNotebooks is a system generated symlink to publicly available Notebooks.

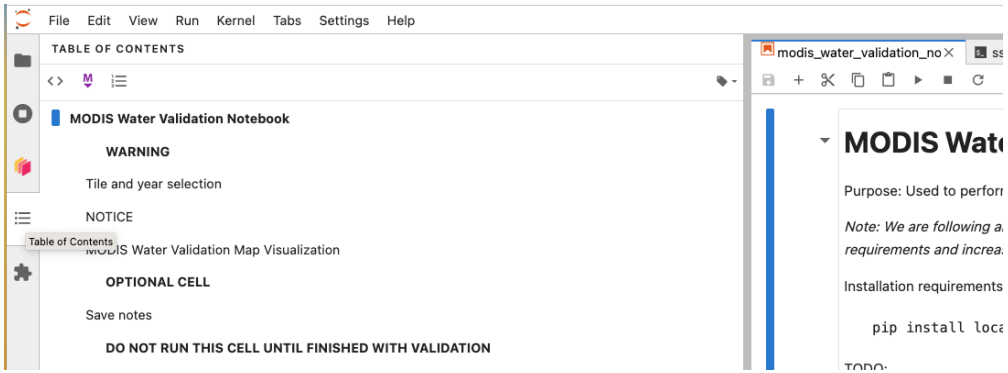
Jupyter lab Console

Explore left side menu

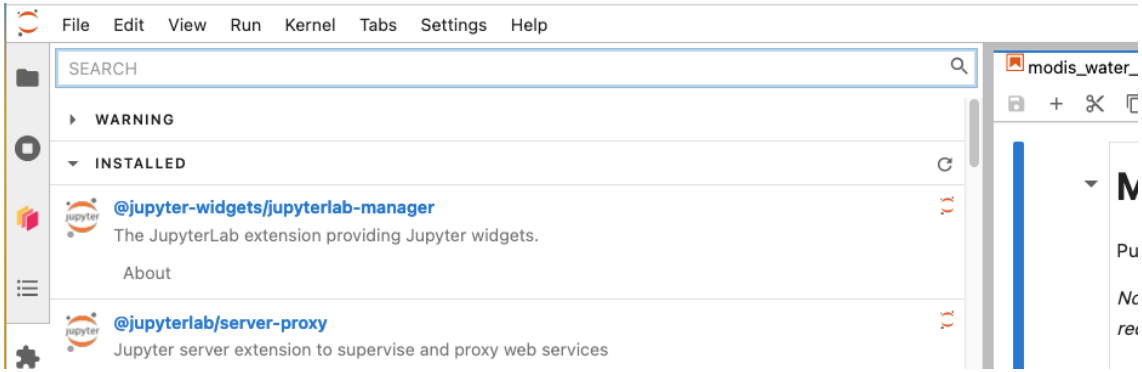
Show what kernels are running →



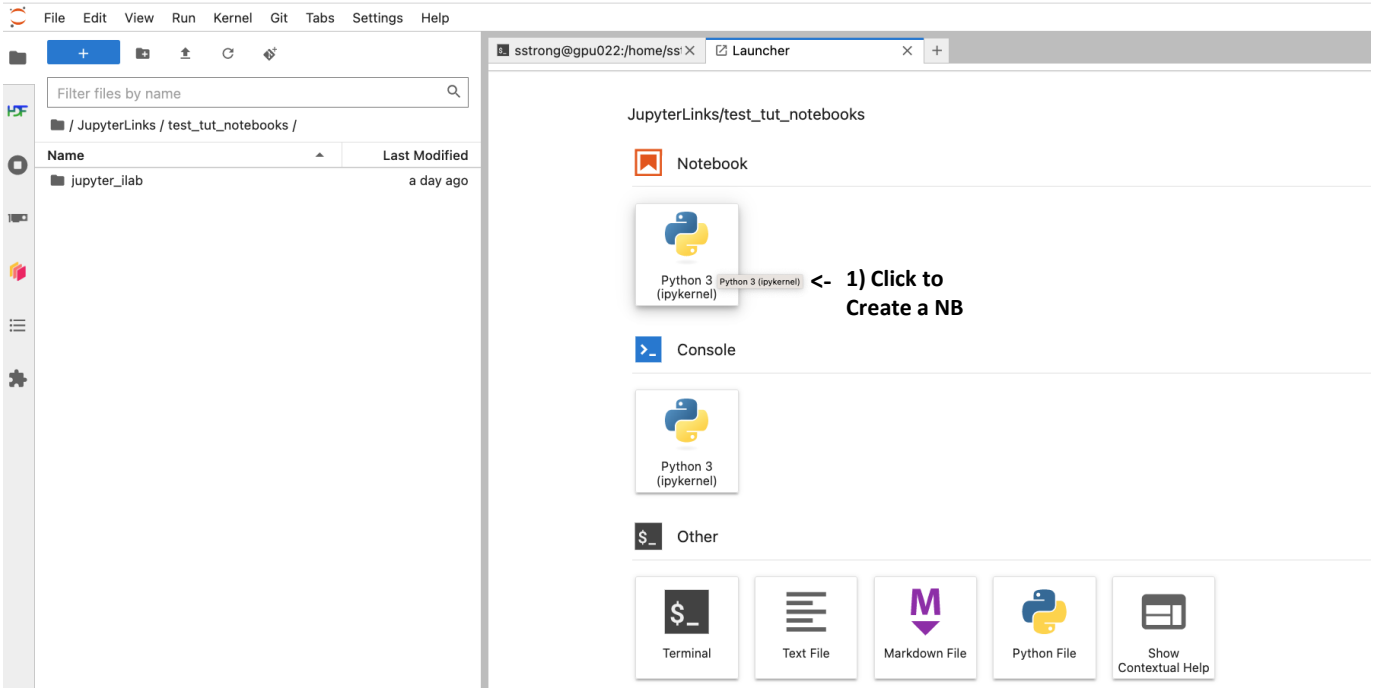
Markup →

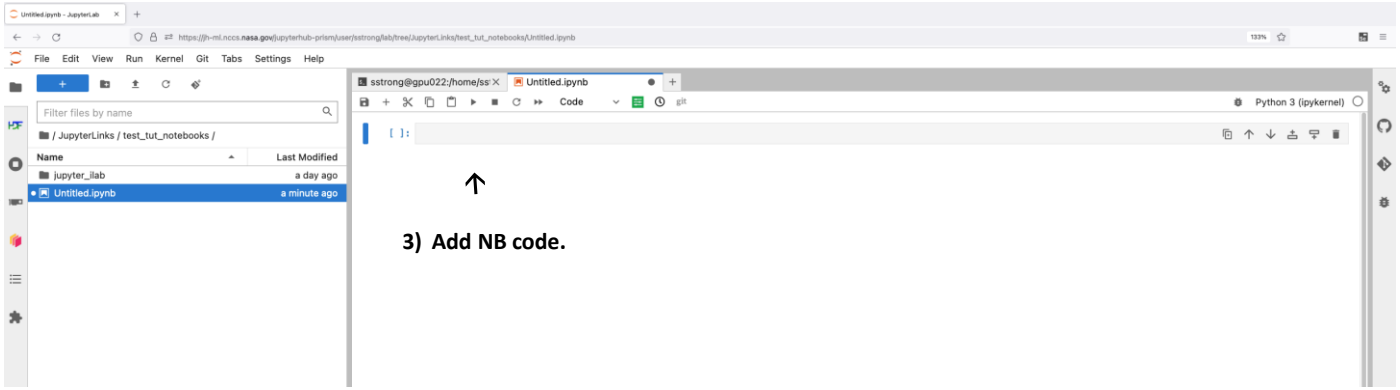


Installed plugins →



Jupyter lab Create a NB

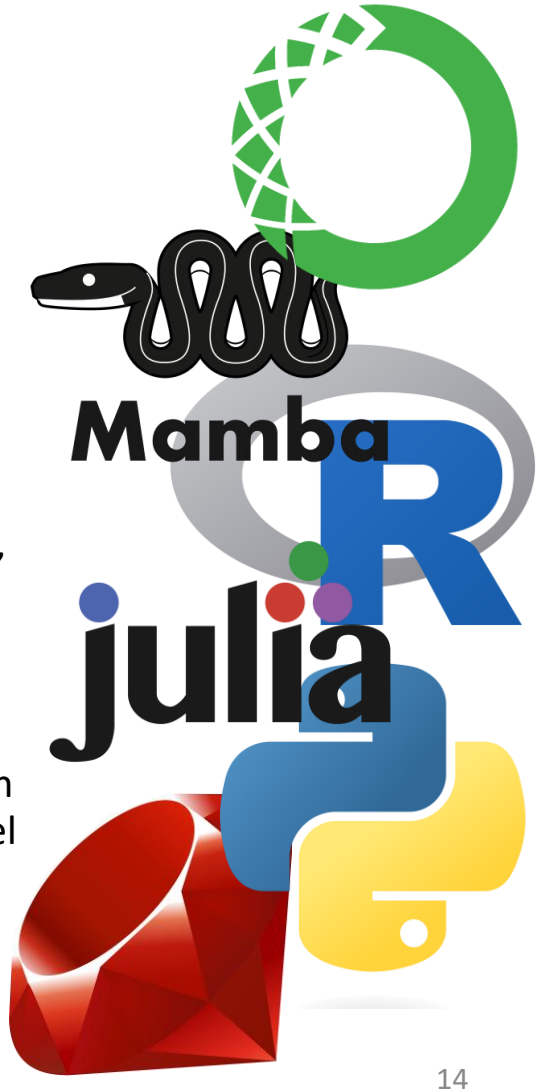




2) Creates an Untitled NB in current dir.

Environments & Kernels

- Many projects will require a distinct set of libraries and software for running code against. Package management tools such as Anaconda are available on our systems to help users build custom environments to support their various projects. While we do have various global environments available, it is highly recommended that users learn how to create and customize their own environments.
- Alternative tools exist for creating environments such as Mamba, or language native tools such as Python's venv, Julia's Pkg environments, Renv, to name a few. While Anaconda tends to be the easiest for integrating new environments into Jupyter sessions, these other tools can also be made to work with some additional configuration steps.
- A kernel, in the context of Jupyter, is simply an environment that has been made available for use in Jupyter. The IPython kernel for example is the Python execution backend for Jupyter. While python is the most common, other languages also use an interactive kernel to bridge a software environment to your Jupyter Notebook (IRKernel, IJulia, Iruby, etc).



Simple Python Environment Example

We use Lmod modules to provide Anaconda for users without dependency on installing the package management pieces in user space.

```
$ ml anaconda
```

With Anaconda loaded, we can then create our environment and we include ipykernel so that the environment will be usable as a kernel in your JupyterLab session.

```
$ conda create -n MyEnvironment -c conda-forge python ipykernel -y
```

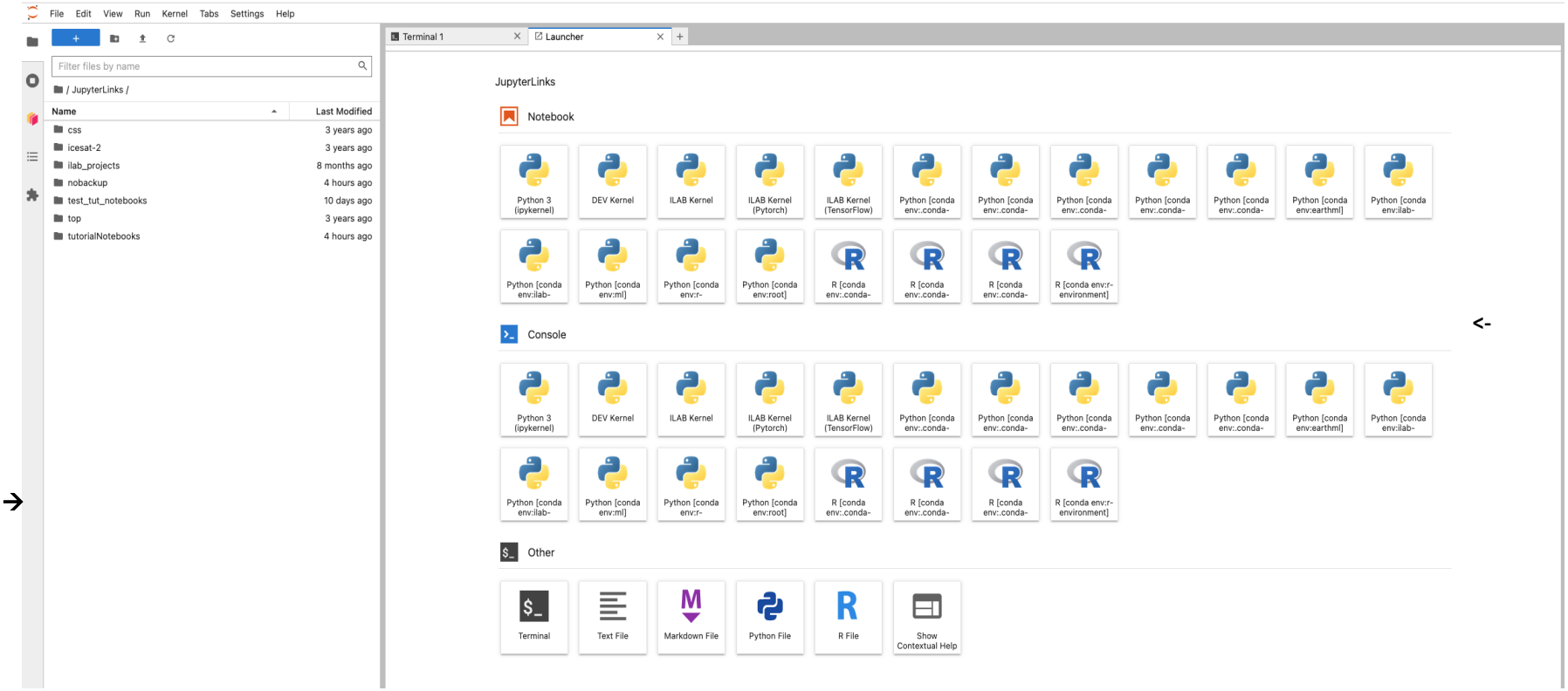
We can then activate our environment and run python.

```
$ conda activate MyEnvironment
(MyEnvironment) $ python
```

We can also use Anaconda to install or remove different packages, or manage environments as a whole.

```
(MyEnvironment) $ conda install -c conda-forge numpy -y
(MyEnvironment) $ conda remove numpy -y
(MyEnvironment) $ conda deactivate
$ conda env list
….
$ conda env remove -n MyEnvironment -y
```

Jupyter lab Conda Envs



Common Environment/Kernel Issues

- Anaconda environments are installed into user home directories by default. Home quotas are small and environments can become large quickly. My preferred method for solving this issue is to relocate ~/.conda to $NOBACKUP and to create a symlink in your home directory, but other methods can be used to get around this (custom prefix on install, .condarc config, environment variables).
- On occasion Anaconda will prompt a user to run ‘conda init’ which will modify their ~/.bashrc (or cshrc, etc.) to preload Anaconda. This process should be handled by loading the module, and if done through login scripts instead can result in kernels not showing up in your Jupyter session.
- As mentioned, environments require an interactive kernel package to be used within Jupyter. An environment will appear to exist from the command line, but will not be visible in your available kernel list until such a package is installed.

Demo

Innovation Lab

Tutorial Notebook

eViz

“An easy-to-use visualization framework for earth system models.”

EViz consists of two Python-based tools, eViz and iViz, that provide a simple way to visualize a wide variety of data sources. eViz is a CLI-driven static plotting tool configurable with YAML files. iViz is a Jupyter Notebook-based tool with a Bokeh backend for providing an interactive exploration of data sources.

https://www.nccs.nasa.gov/sites/default/files/eviz_tech_talk.pdf

https://www.nccs.nasa.gov/nccs-users/instructional/instructional-videos

Resources & Documentation

- Jupyter: https://jupyter.org/
- JupyterLab: https://jupyterlab.readthedocs.io/en/latest/
- Anaconda: https://www.anaconda.com/
- NCCS Documentation:
  - https://www.nccs.nasa.gov/nccs-users/instructional/adapt-instructional/jupyter
  - https://www.nccs.nasa.gov/nccs-users/instructional/instructional-videos

For any further assistance using JupyterHub on NCCS systems, or for any software related requests contact us at: support@nccs.nasa.gov

Future Plans

- JupyterLab 4.0
- Allocation customization
- Anaconda/Environment management improvements
- Access and reliability for hubs behind www-proxy-dev

Questions
&
Open Discussion

Thanks for Attending!

- Recording will be made available after the conclusion of the tech talk.
- Slides will be made available on our website at: https://www.nccs.nasa.gov/nccs-users/user-events/tech-talks
- Suggestions for future tech talk topics or feedback regarding today's discussion can be put in the chat of this meeting
- Join us again next week on 6/29 at 12PM for another tech talk discussing explainable AI/ML.