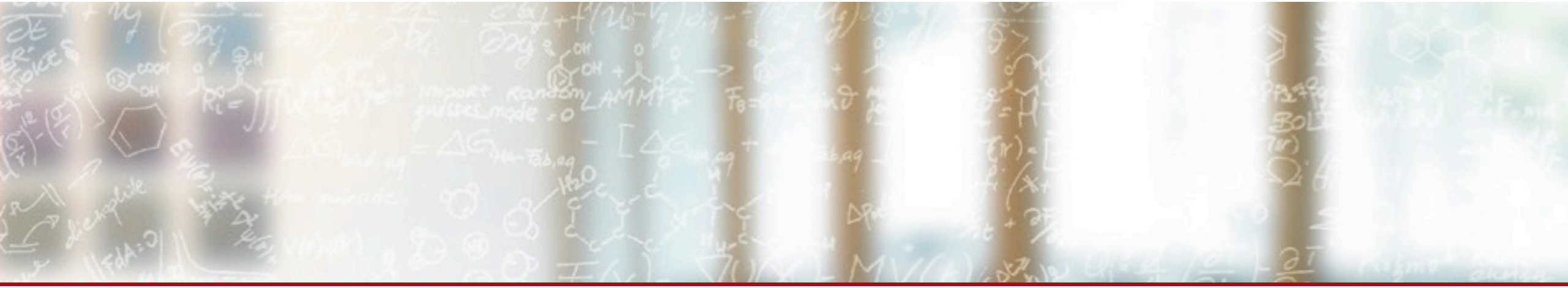
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



Interactive Computing with Jupyter on Piz Daint, using Python, ParaView and Julia

Tim Robinson, Jean Favre and Harmen Stoppels, CSCS

Apr 26, 2021

Agenda

09:00 – 10:45	**Jupyter, JupyterLab and JupyterHub on Piz Daint**
10:45 – 11:00	*Break*
11:00 – 12:00	**ParaView and Jupyter, Part I**
12:00 – 13:00	*Lunch*
13:00 – 14:00	**ParaView and Jupyter, Part II**
14:00 – 14:15	*Break*
14:15 – 16:15	**Programming in Julia**

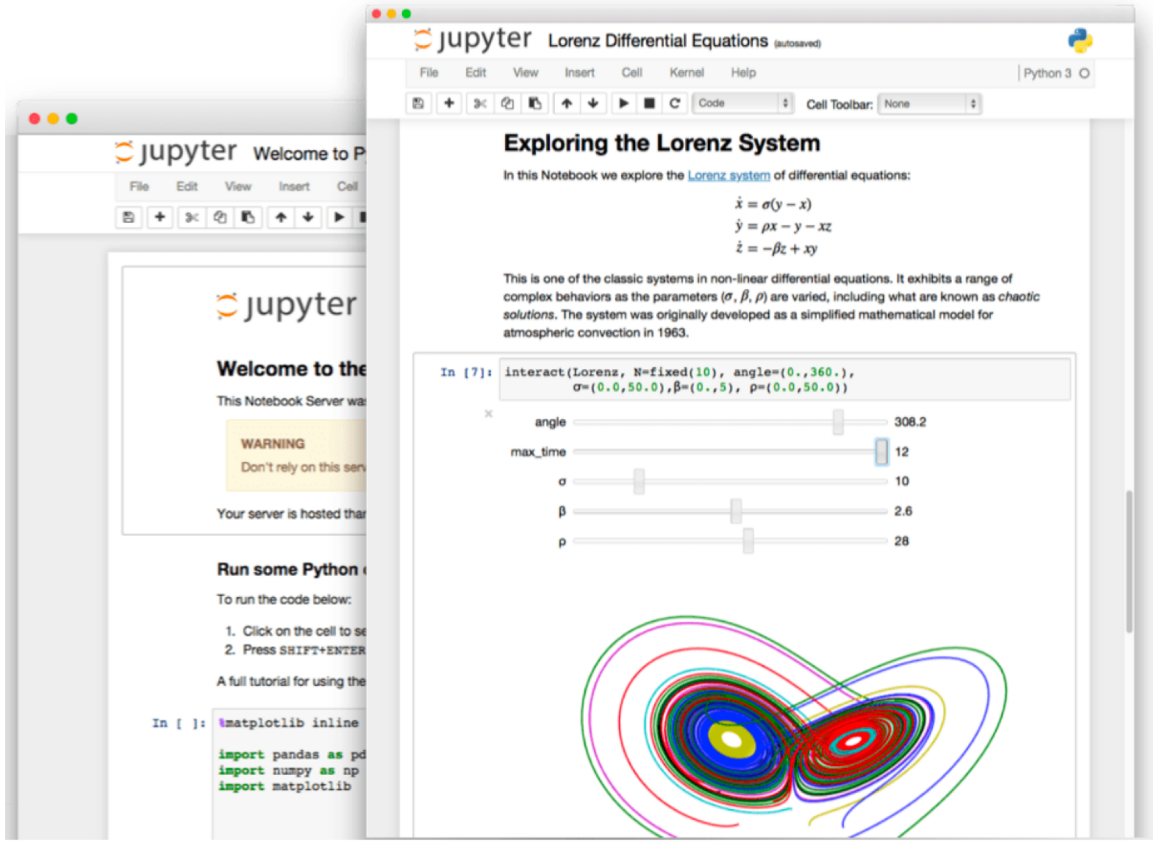
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Part I: Jupyter: Jupyter Notebook, JupyterLab and JupyterHub

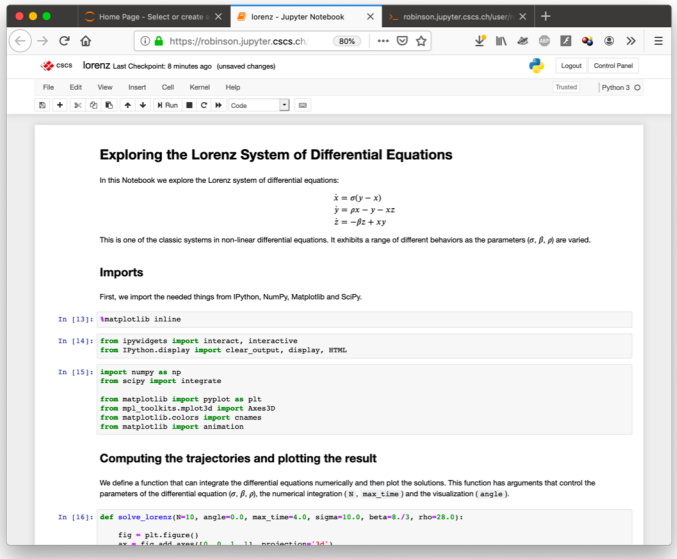
Jupyter Notebook

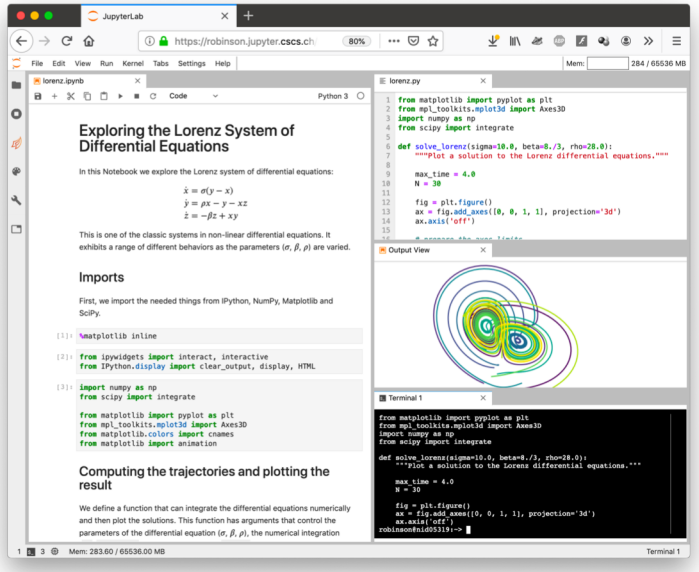
- Jupyter Notebook is an open-source web app for creating reproducible computational narratives
- Documents can contain live code, equations, narrative text, visualizations, and rich media
- Documents can be shared or exported to PDF, HTML, LaTeX, etc.
- Working environment includes
  - File browser
  - Terminal
  - Support for many languages: Python, R, Julia, C++, …
  - Extensible design
  - Many server/client plugins
- Notebook is attached to a “kernel”, which does the actual computation



JupyterLab

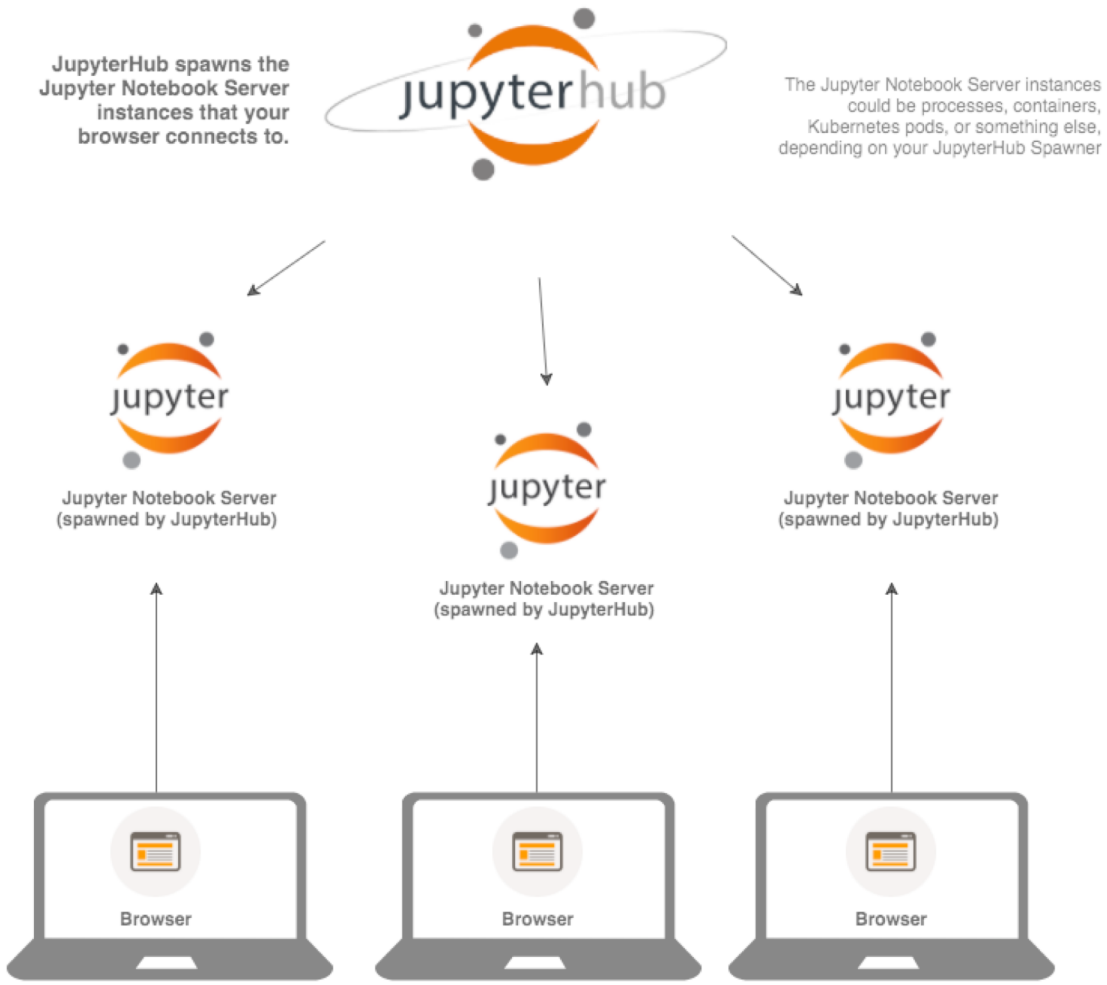
- JupyterLab is the next-generation user interface for Jupyter
- Higher degree of interaction between notebooks, documents, text editors and other activities (arrange with tabs/splitters)
- More advanced interactive development environment
- Uses the same notebook document format
- JupyterLab is the default interface at CSCS
- Classic notebook interface is still accessible by modifying the URL from /lab to /tree
  - Or: Help -> Launch Classic Notebook





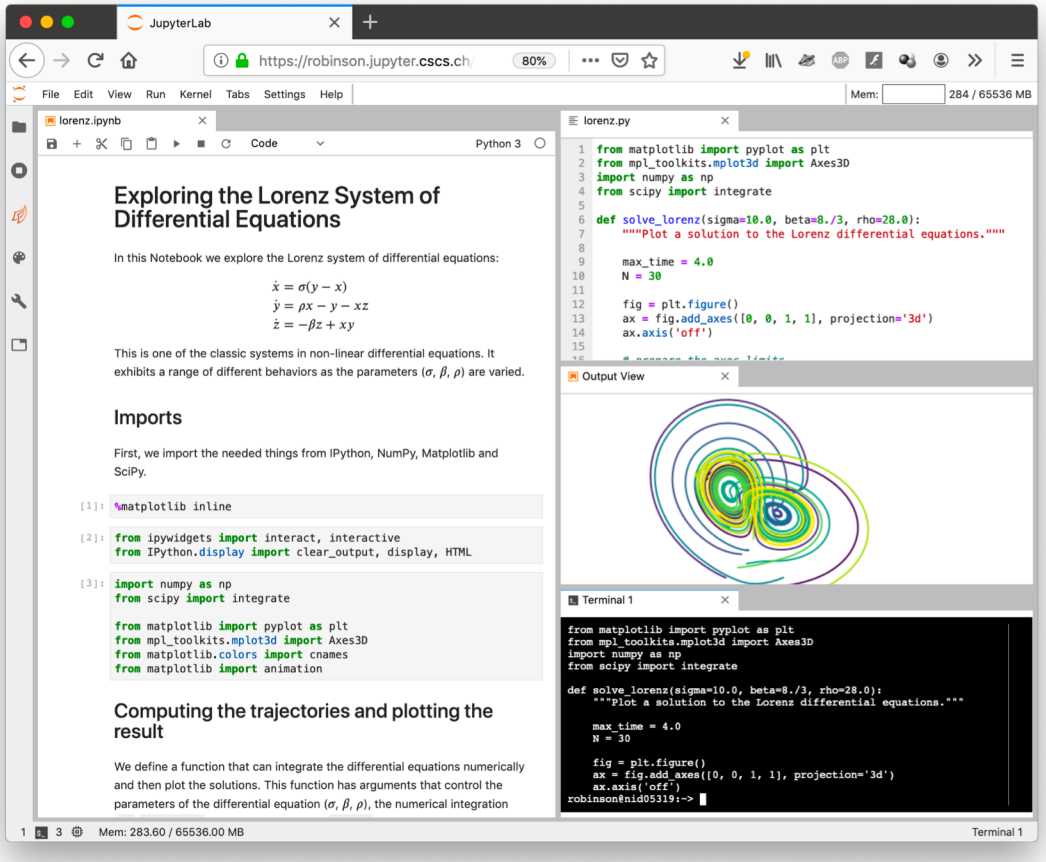
JupyterHub

- Multi-user server for Jupyter Notebooks (designed for classrooms, research labs, Universities...)
- Spawns, manages and proxies multiple instances of the single-user Jupyter Notebook server
- Main components:
  - **Hub** (tornado process)
  - **configurable http proxy** (node-http-proxy) that receives the requests from the client's browser
  - multiple **single-user Jupyter notebook servers** (Python/IPython/tornado) that are monitored by Spawners
  - an **authentication class** that manages how users can access the system

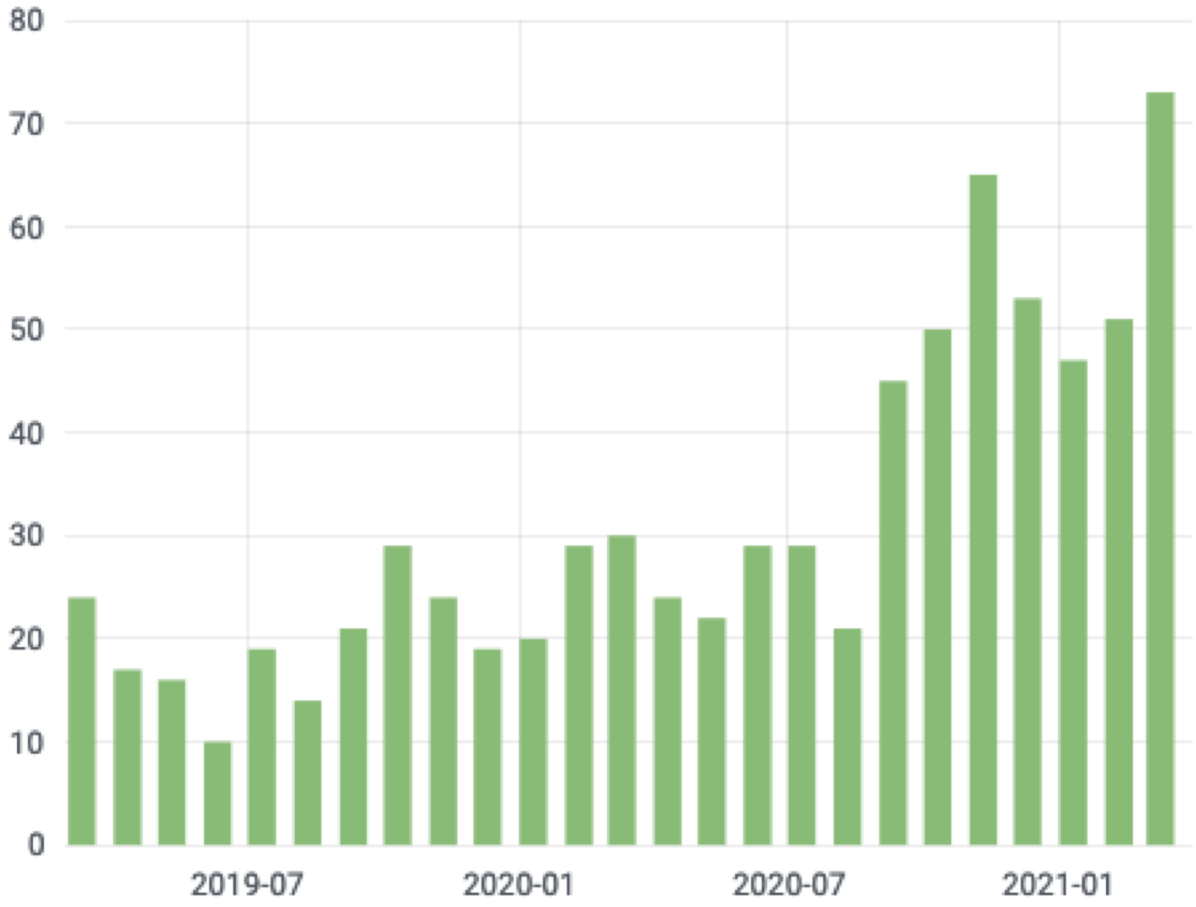


JupyterHub for Piz Daint

- JupyterHub for Piz Daint is accessible at:
  https://jupyter.cscs.ch
- Available to anyone with a compute allocation on Piz Daint
- Servers are launched on the compute nodes (GPU or multicore)
- You have dedicated access to the full compute capability of the node(s)
- Special Slurm reservations for interactive computing grow and shrink automatically according to demand
- In most cases you should have to wait less than five minutes for your allocation



JupyterHub monthly users on Piz Daint



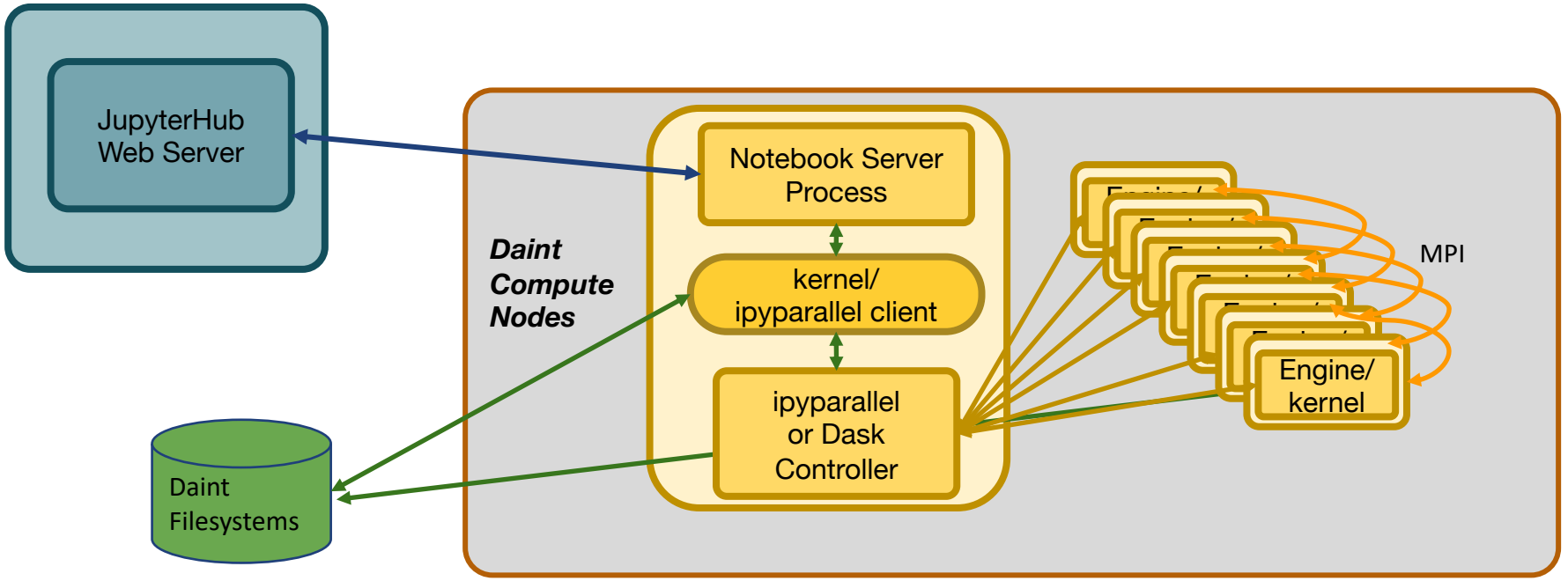
Challenges with interactive computing on HPC

- Shared parallel filesystems
- Network connectivity
- Slurm
- Warm swapping
- All kinds of other possible issues - usually these are temporal!

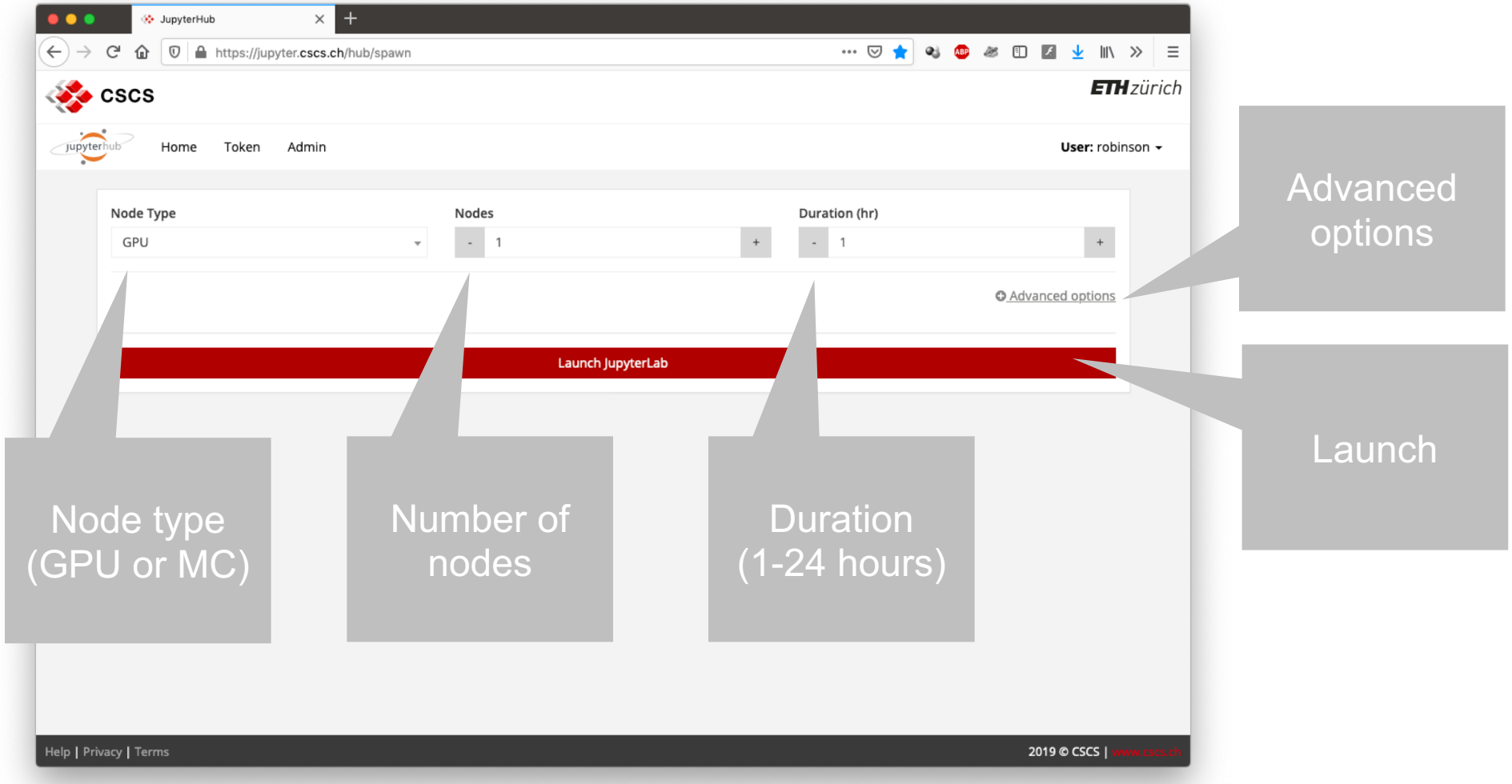
- Debugging problems:
  - Check for any maintenance announcements on the spawner page
  - Check your own internet connectivity
  - Check for old JupyterHub sessions in browser tabs (especially if kernels are not starting or are slow to connect or restart)
  - Inspect the Jupyter log file in your $SCRATCH directory
  - If problems persist, submit a ticket at https://support.cscs.ch
    - Include as much detailed information as possible – precise time periods, Slurm jobid, what was working, what was not working...

JupyterHub implementation at CSCS

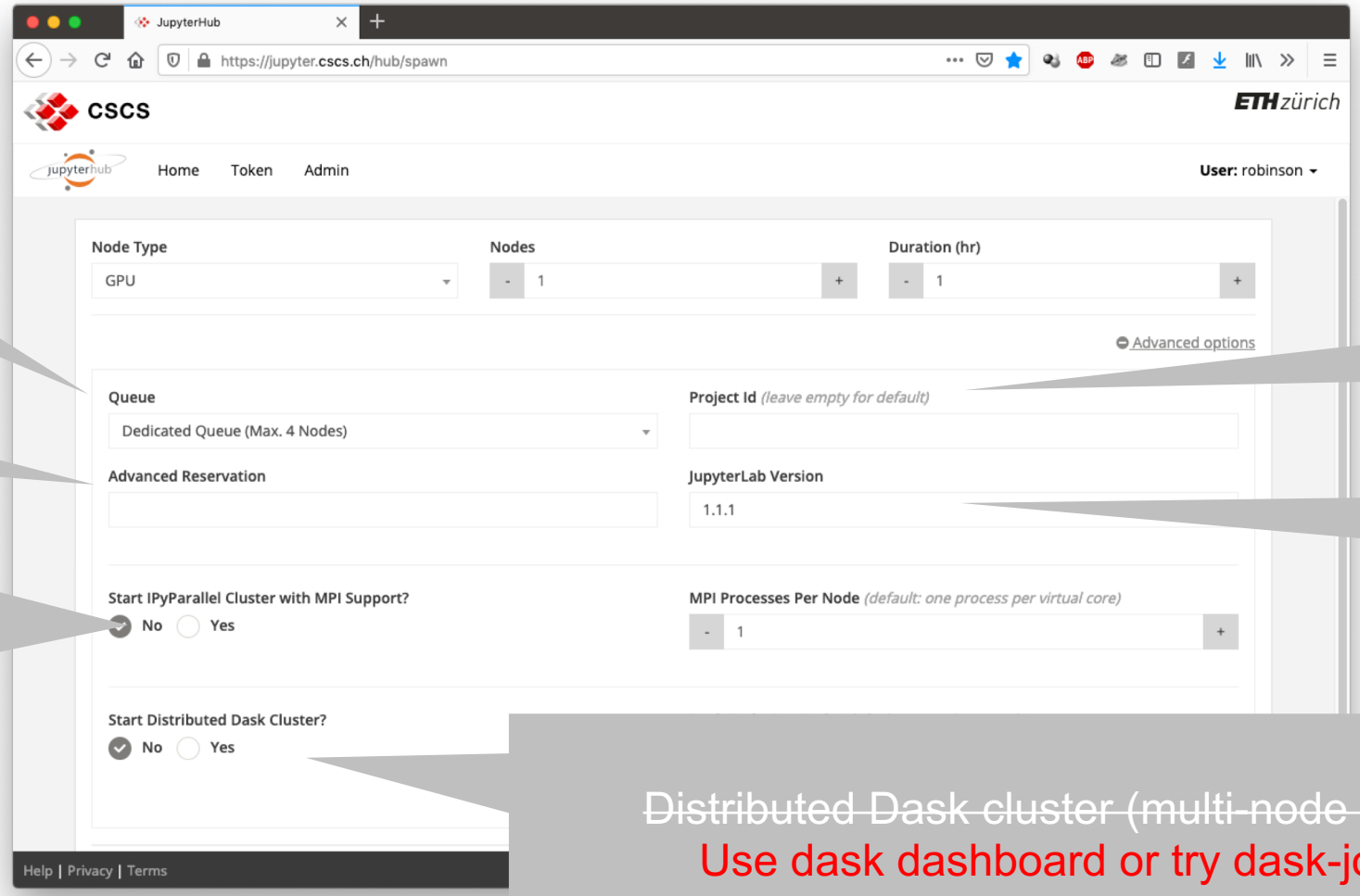
jupyter.cscs.ch



Basic options



Advanced options



Queue

Reservation

~~IPyParallel (multi-node notebooks)~~
Replaced by ipcluster magic

Project Id (account)

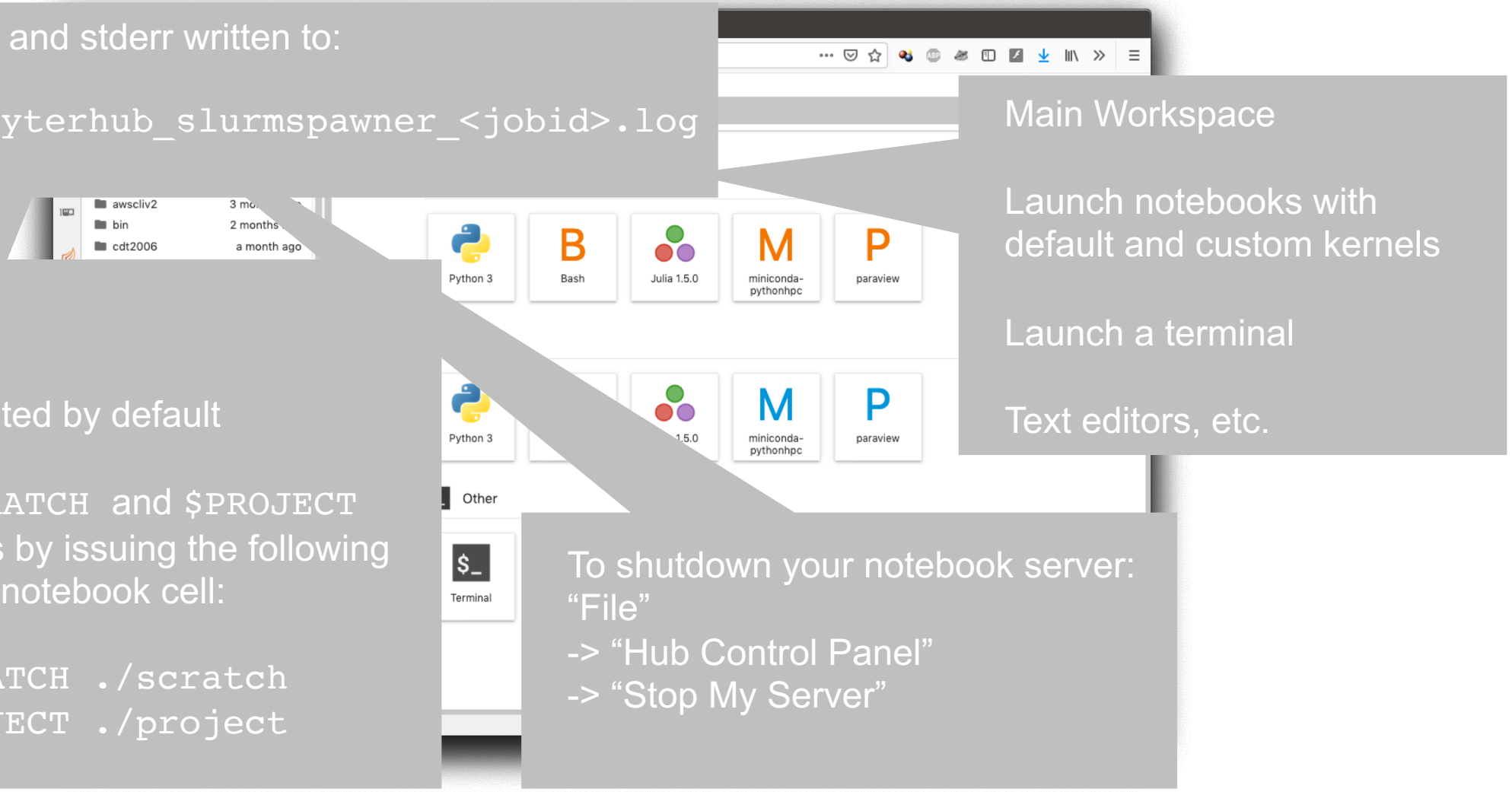
JupyterLab version

~~Distributed Dask cluster (multi-node notebooks)~~
Use dask dashboard or try dask-jobqueue

JupyterLab interface

Standard output and stderr written to:

`$SCRATCH/jupyterhub_slurmspawner_<jobid>.log`



File Browser

`$HOME` is mounted by default

To access `$SCRATCH` and `$PROJECT` create sym links by issuing the following commands in a notebook cell:

```
!ln -s $SCRATCH ./scratch
!ln -s $PROJECT ./project
```

Main Workspace

Launch notebooks with default and custom kernels

Launch a terminal

Text editors, etc.

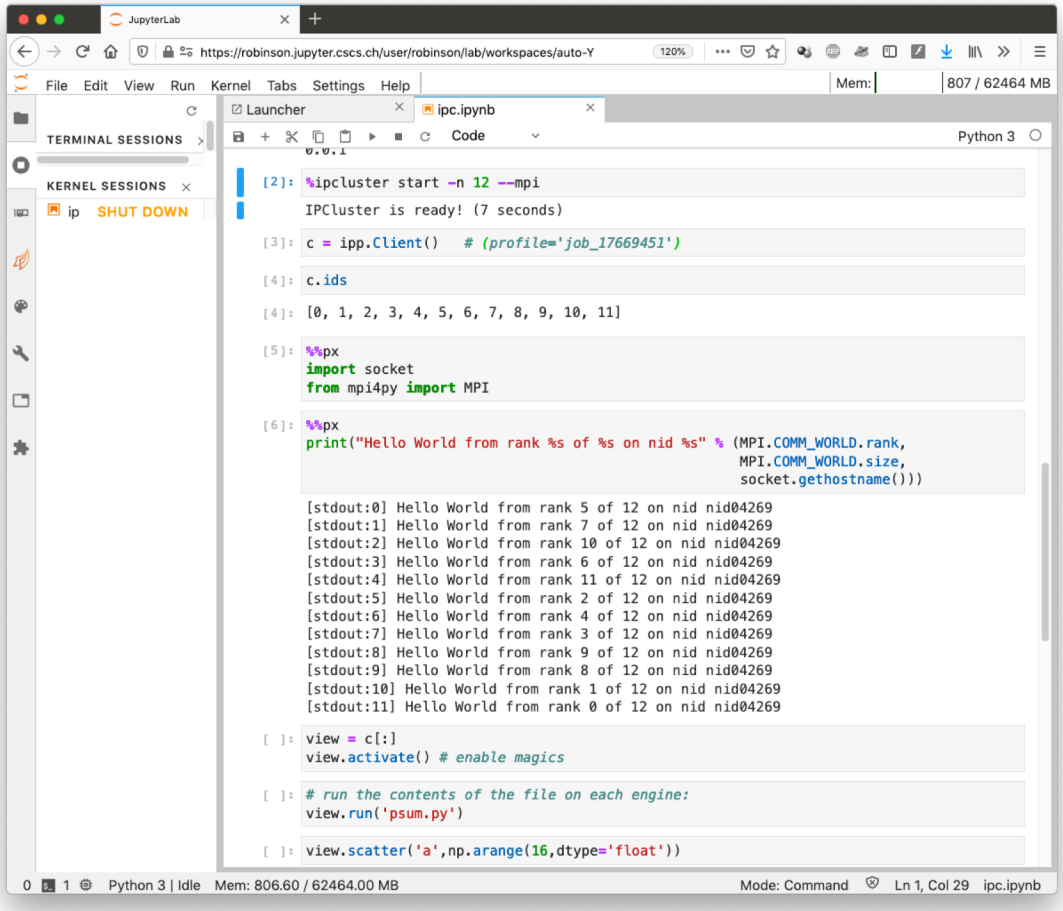
To shutdown your notebook server:
“File”
-> “Hub Control Panel”
-> “Stop My Server”

Getting the Python packages you need

- **pip install**, direct from the notebook (simple)
- **Virtual environments and kernels** (better)
- **module load** commands in $HOME/.jupyterhub.env (can be useful)
- Install **miniconda** and use **conda** environments (if you prefer conda)
- **Containers** (work in progress; potentially offers more flexibility)

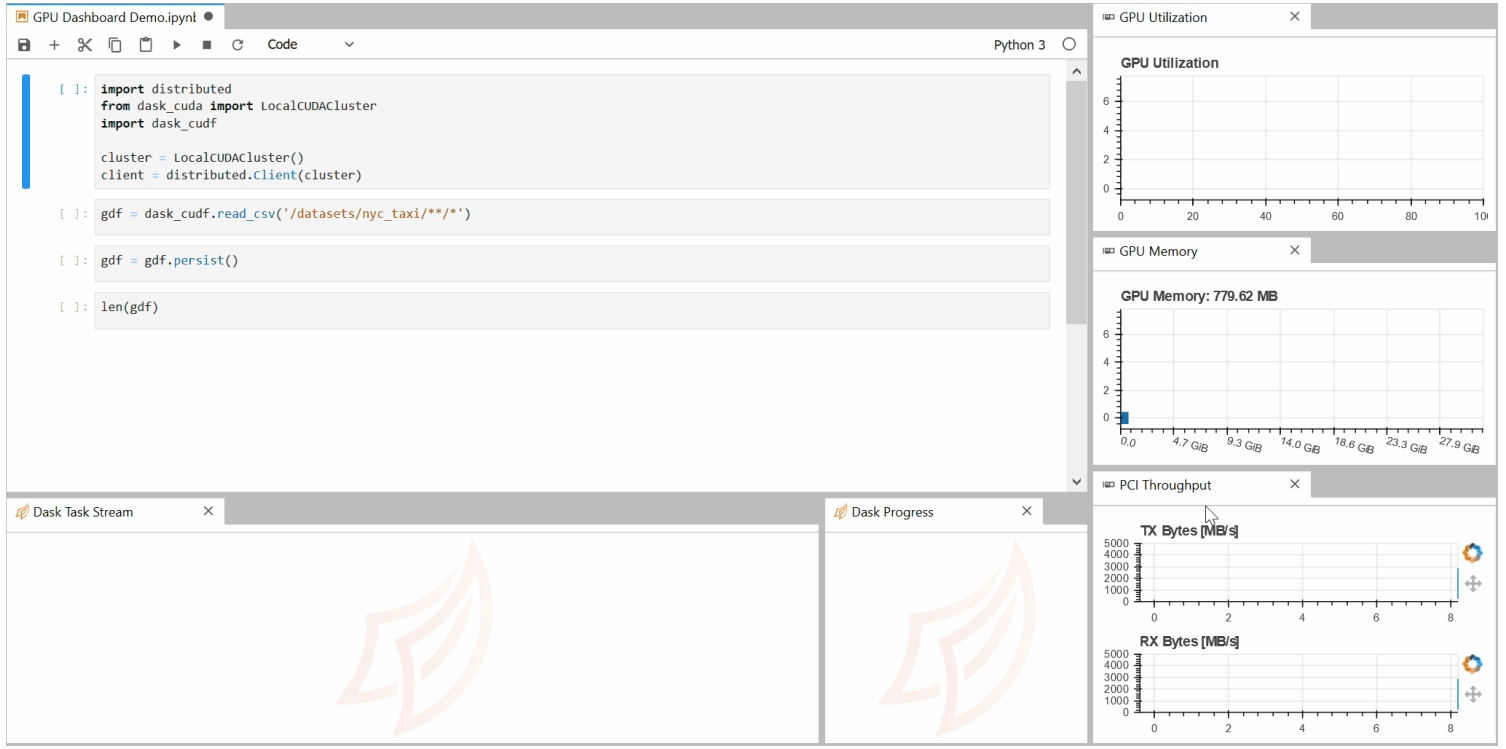
IPyParallel and ipcluster magic

- IPCluster magic commands for automatically starting and stopping ipyparallel clusters (ipcontroller and ipengines)
- `module load ipcmagic` in $HOME/.jupyterhub.env
- `import ipcmagic` in notebook
- Makes it easier to use MPI directly in the notebook

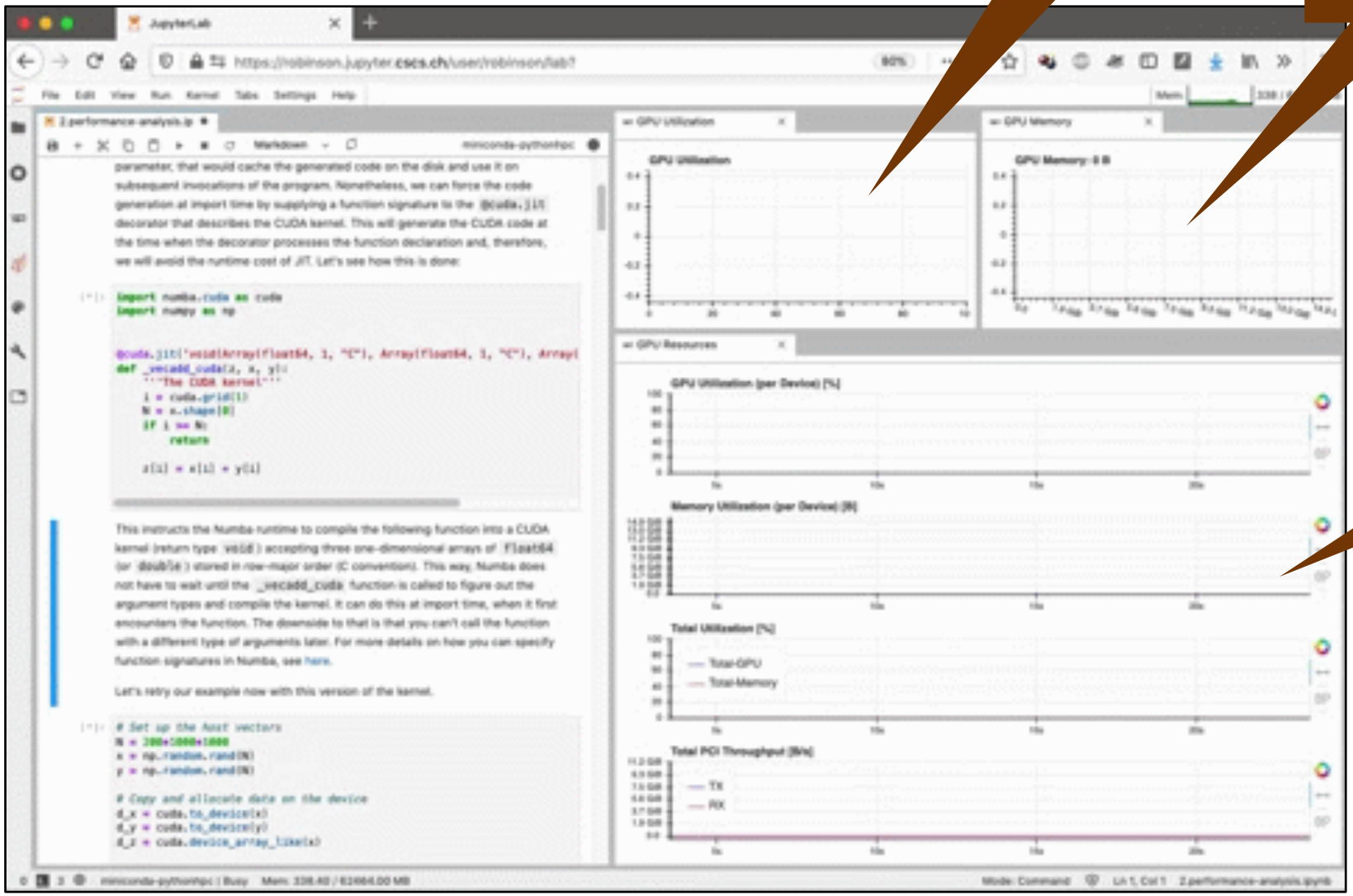


NVDashboard and Dask Distributed JupyterLab Extensions

- NVDashboard is a JupyterLab extension for displaying dashboards of GPU usage (from the NVIDIA RAPIDS team)
- Dask JupyterLab extension provides a way to manage Dask clusters, as well as embedding Dask's dashboard plots directly into JupyterLab panes



GPU dashboards



Let’s crack on…