

# SDSC HPC-DSI 2022: Interactive Computing

August 2, 2022

Mary Thomas  
SDSC

EXPANSE  
COMPUTING WITHOUT BOUNDARIES

# EXPANSE

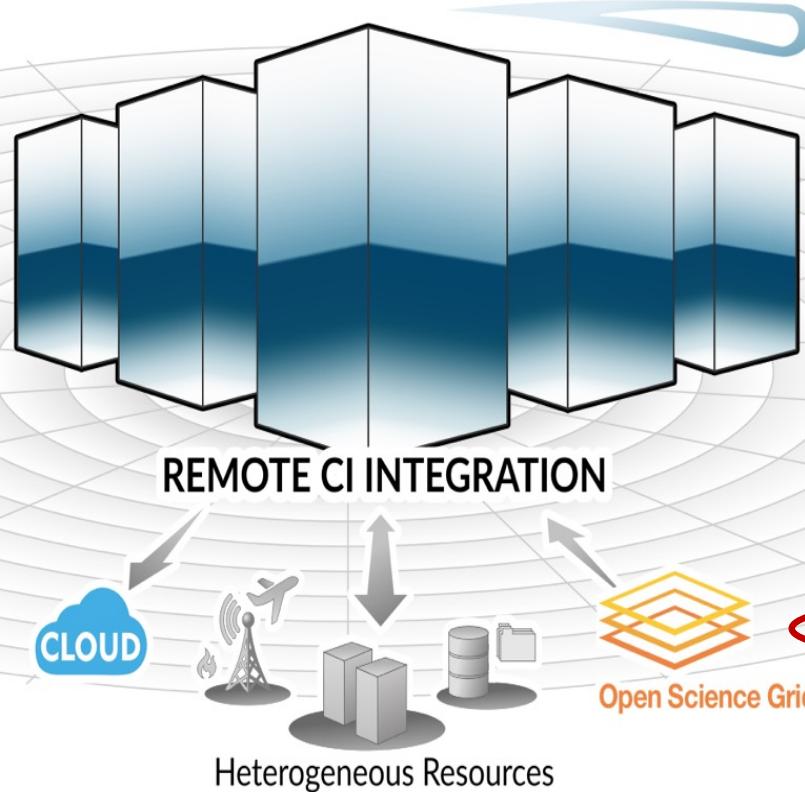
COMPUTING WITHOUT BOUNDARIES  
5 PETAFLOP/S HPC and DATA RESOURCE

## HPC RESOURCE

13 Scalable Compute Units  
728 Standard Compute Nodes  
52 GPU Nodes: 208 GPUs  
4 Large Memory Nodes

## DATA CENTRIC ARCHITECTURE

12PB Perf. Storage: 140GB/s, 200k IOPS  
Fast I/O Node-Local NVMe Storage  
7PB Ceph Object Storage  
High-Performance R&E Networking



## LONG-TAIL SCIENCE

Multi-Messenger Astronomy  
Genomics  
Earth Science  
Social Science

## INNOVATIVE OPERATIONS

Composable Systems  
High-Throughput Computing  
Science Gateways  
**Interactive Computing** (circled in red)  
Containerized Computing  
Cloud Bursting

For more details see the Expanse user guide @ [https://www.sdsc.edu/support/user\\_guides/expanse.html](https://www.sdsc.edu/support/user_guides/expanse.html)  
and the "Introduction to Expanse" webinar @ [https://www.sdsc.edu/event\\_items/202006\\_Introduction\\_to\\_Expanse.html](https://www.sdsc.edu/event_items/202006_Introduction_to_Expanse.html)

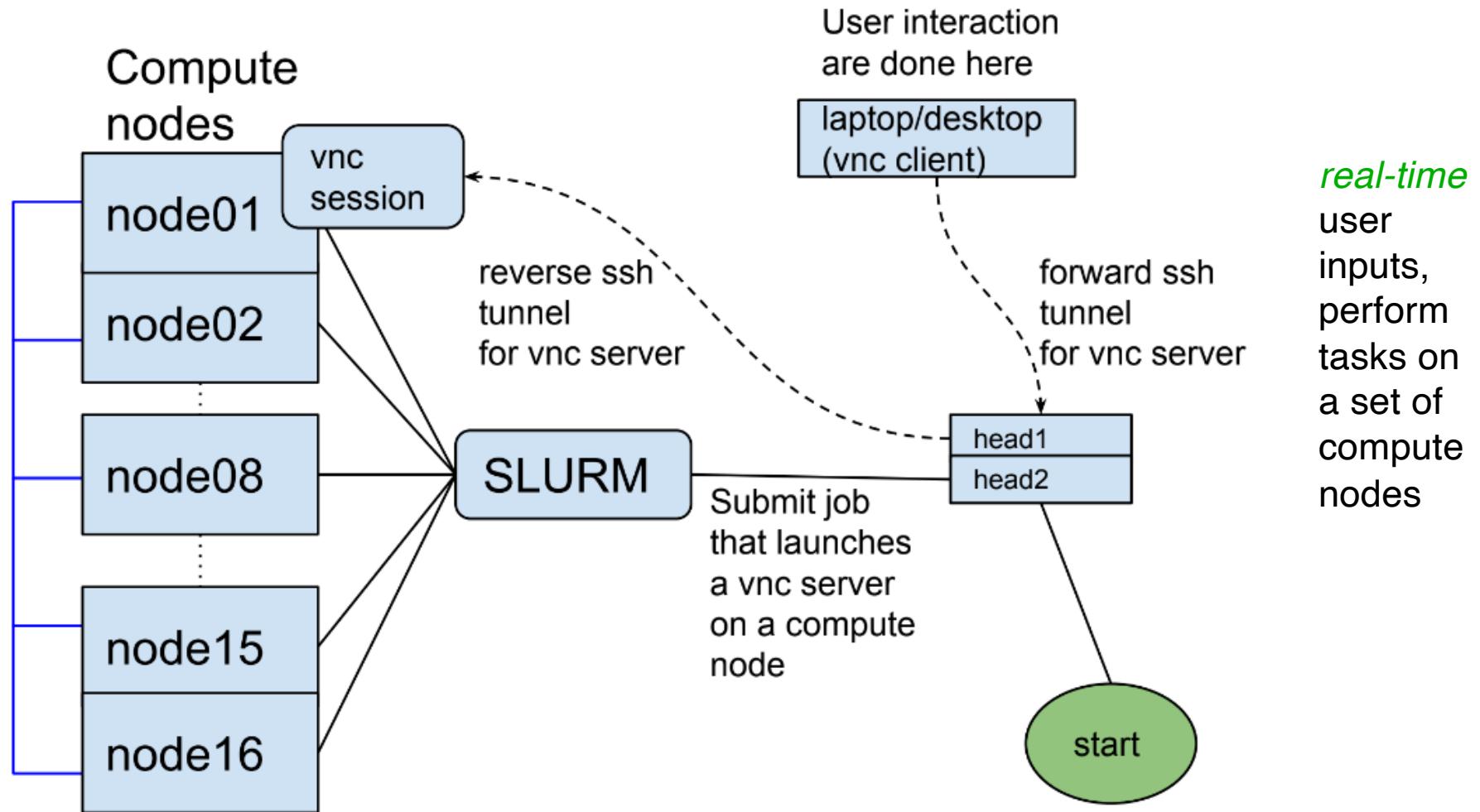
# Outline

- What is Interactive High-Performance Computing?
- Accessing Interactive HPC Nodes
- Running Interactive Apps
- Running Interactive Apps Using the Expanse Portal
- Notebook Security

# What is Interactive HPC Computing

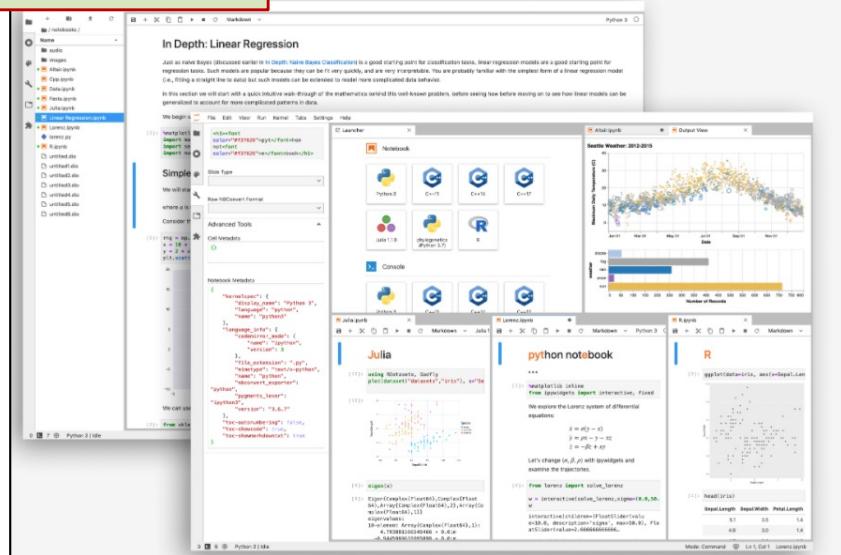
- In **computer science**, **interactive computing** refers to software which accepts input from the user as it runs.
  - **Interactive** software includes commonly used programs, such as word processors or spreadsheet applications.
- **Interactive HPC computing** involves *real-time* user inputs to perform tasks on a set of compute node(s) including:
  - Code development, real-time data exploration, and visualizations.
  - Used when applications have large data sets or are too large to download to local device, software is difficult install, etc.
  - User inputs come via command line interface or application GUI (Jupyter Notebooks, Matlab, R-studio).
  - Actions performed on remote compute nodes as a result of user input or program out.

# Real-time User Interactions



# Interactive HPC Computing

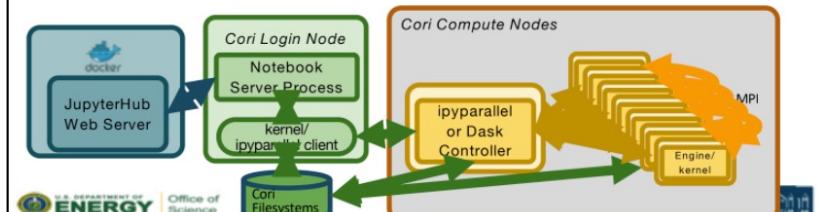
<https://jupyter.org/>



Interactive Distributed Computing with Jupyter (NERSC)

## Jupyter architecture

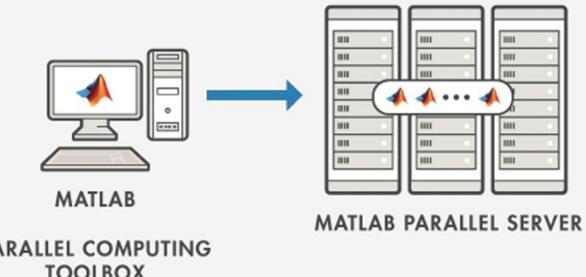
- Allocate nodes on Cori interactive queue and start ipyparallel or Dask cluster
  - Developed %ipcluster magic to setup within notebook
- Compute nodes traditionally do not have external address
  - Required network configuration / policy decisions
- Distributed training communication is via MPI Horovod or Cray ML Plugin



<https://drive.google.com/file/d/1-OFJrk1q3L1d3uakr2xkozrPn2c2VZpZ/view>

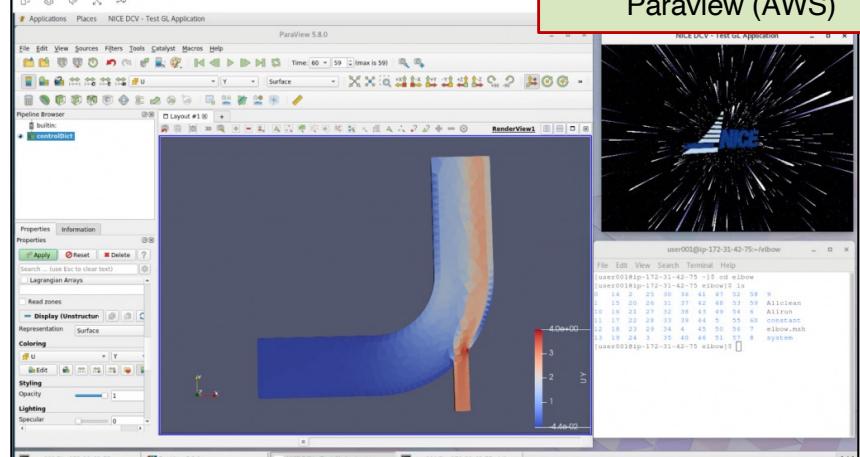
Parallel Matlab (AWS)

```
>> parpool(parcluster('HPC1'),100);  
>> parfor i=1:3000  
>> c(i,:) = eig(rand(1000));  
>> end
```



<https://azuremarketplace.microsoft.com/en-us/marketplace/apps/mathworks-inc.matlab-parallel-server-listing?tab=Overview>

Paraview (AWS)



<https://aws.amazon.com/blogs/compute/how-to-run-3d-interactive-applications-with-nice-dcv-in-aws-batch/>

# Outline

- What is Interactive High-Performance Computing?
- Accessing Interactive HPC Nodes
- Running Interactive Apps
- Running Interactive Apps Using the Expanse Portal
- Notebook Security

# Accessing Interactive Compute Nodes on Expanse

- Connect via terminal using SSH → secure connections
- Use the *srun* command to obtain nodes for ‘live,’ command line interactive access:

|            |  |
|------------|--|
| <b>CPU</b> | <code>srun --partition=debug --pty --account=use300 --nodes=1 --ntasks-per-node=4 --mem=8G -t 00:30:00 --wait=0 --export=ALL /bin/bash</code>                |
| <b>GPU</b> | <code>srun --partition=gpu-debug --pty --account=use300 --ntasks-per-node=10 --nodes=1 --mem=96G --gpus=1 -t 00:30:00 --wait=0 --export=ALL /bin/bash</code> |

(Tested 7/27/22)

# Using An Interactive CPU node

```
[username@login01 openmp]$ module purge
[username@login01 openmp]$ module load slurm
[username@login01 openmp]$ module load cpu
[username@login01 openmp]$ module load gcc/10.2.0
[username@login01 openmp]$ module load openmpi/4.0.4
[username@login01 openmp]$ srun --partition=debug --pty --account=use300--nodes=1 --ntasks-per-node=64 --mem=128G -t 00:30:00 --wait=0 --export=ALL /bin/bash
```

Request an interactive node  
for 30 minutes

```
[mthomas@exp-9-55 openmp]$ export OMP_NUM_THREADS=16
[mthomas@exp-9-55 openmp]$ ./hello_openmp
HELLO FROM THREAD NUMBER = 0
HELLO FROM THREAD NUMBER = 8
HELLO FROM THREAD NUMBER = 9
HELLO FROM THREAD NUMBER = 10
HELLO FROM THREAD NUMBER = 11
HELLO FROM THREAD NUMBER = 12
HELLO FROM THREAD NUMBER = 13
HELLO FROM THREAD NUMBER = 14
HELLO FROM THREAD NUMBER = 15
HELLO FROM THREAD NUMBER = 4
HELLO FROM THREAD NUMBER = 7
HELLO FROM THREAD NUMBER = 6
HELLO FROM THREAD NUMBER = 5
HELLO FROM THREAD NUMBER = 3
HELLO FROM THREAD NUMBER = 2
HELLO FROM THREAD NUMBER = 1
```

- Exit interactive session when your work is done or you will be charged CPU time.
- Beware of oversubscribing your job: don't ask for more cores than you have requested.
- Intel compiler allows this, but your performance will be degraded.

# Using An Interactive GPU node

```
[snip]
Last login: Fri Feb 18 12:58:32 2022 from 76.176.117.51
[username@login02 ~]$
[username@login02 ~]$ srun --partition=gpu-shared --pty --nodes=1 --ntasks-per-node=16 --mem=374 --gpus=1 -t 00:30:00 --wait=0 --export=ALL --account=use300 /bin/bash
srun: job 9794018 queued and waiting for resources
srun: job 9794018 has been allocated resources
[mthomas@exp-14-57 ~]$
[mthomas@exp-14-57 ~]$ nvidia-smi
Fri Feb 18 13:04:19 2022
```

Request an interactive node  
for 30 minutes

Verify you are on a GPU node

| NVIDIA-SMI 460.32.03 |                    |               | Driver Version: 460.32.03 |                 | CUDA Version: 11.2 |            |        |
|----------------------|--------------------|---------------|---------------------------|-----------------|--------------------|------------|--------|
| GPU                  | Name               | Persistence-M | Bus-Id                    | Disp.A          | Volatile           | Uncorr.    | ECC    |
| Fan                  | Temp               | Perf          | Pwr:Usage/Cap             | Memory-Usage    | GPU-Util           | Compute M. | MIG M. |
| 0                    | Tesla V100-SXM2... | On            | 00000000:86:00.0          | Off             | 0%                 | 0          | N/A    |
| N/A                  | 34C                | P0            | 41W / 300W                | 0MiB / 32510MiB | Default            |            |        |

| Processes:                 |    |    |     |      |                  |
|----------------------------|----|----|-----|------|------------------|
| GPU                        | GI | CI | PID | Type | Process name     |
| ID                         | ID |    |     |      | GPU Memory Usage |
| No running processes found |    |    |     |      |                  |

```
[username@login02 ~]$ exit
```

Exit when tasks are done

# Outline

- What is Interactive High-Performance Computing?
- Accessing Interactive HPC Nodes
- Running Interactive Apps
- Running Interactive Apps Using the Expanse Portal
- Notebook Security

# Viewing Data on HPC Cluster (Simple)

- Cat the file contents
- Plot the data: using gnuplot
- Run Matlab
- Run a data viewing app for the data type:
  - NetCDF, HPF, other

# Visualization Apps: Use X11 Forwarding

- On MacOS:
- Install X11 forwarding software:
  - MacOS: Xquartz

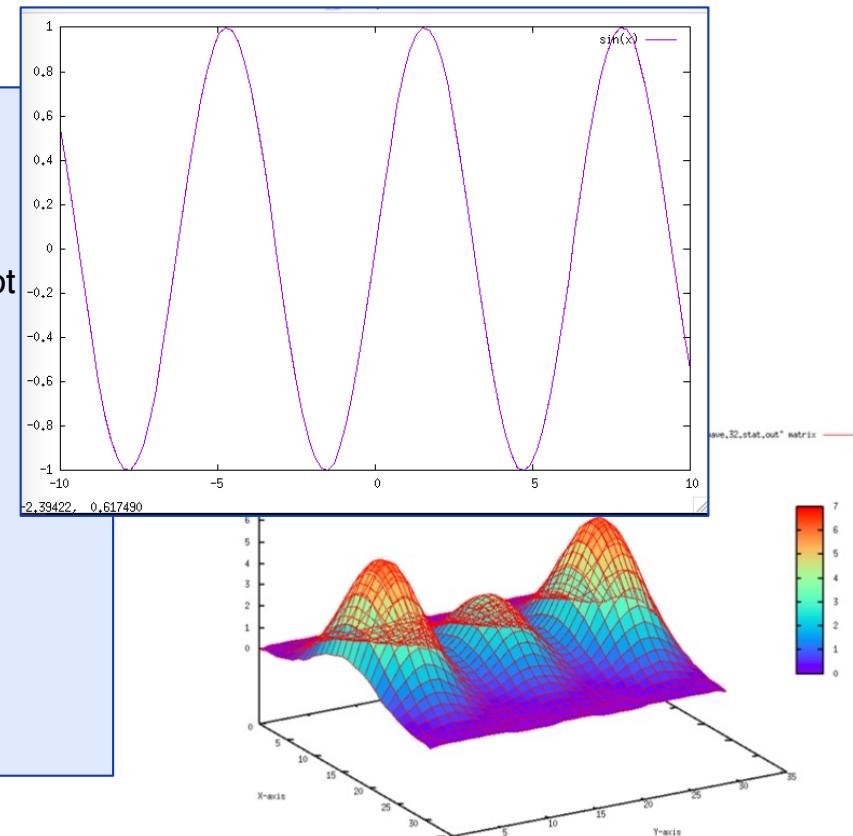
```
[mthomas@home]$ brew install --cask xquartz
Updating Homebrew...
Updated 2 taps (homebrew/core and homebrew/cask).
==> New Formulae
[SNIP]
installer: The upgrade was successful.
🍺 xquartz was successfully installed!
quantum:~ mthomas$ which xquartz
/opt/X11/bin/xquartz
```

- Use the connection command:  
**ssh -Y mthomas@login.expanse.sdsc.edu'**

# gnuplot

- *gnuplot* is a command-driven interactive function plotting program. It can be used to plot functions and data points in both two- and three- dimensional plots in many different formats.
- It is designed primarily for the visual display of scientific data.
- Can be run from the gnuplot app or from within your program so you can save visualizations of results.

```
module load cpu/0.15.4 gcc/10.2.0
[mthomas@login01 ~]$ module load gnuplot/5.2.8
[mthomas@login01 ~]$ which gnuplot
/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen2/gcc-
10.2.0/gnuplot-5.2.8-uwugzxg4dgxaiciheiepgol67cw7m6yg/bin/gnuplot
[mthomas@login01 gnuplot-ex]$ gnuplot
G N U P L O T
Version 5.2 patchlevel 8  last modified 2019-12-01
Copyright (C) 1986-1993, 1998, 2004, 2007-2019
Thomas Williams, Colin Kelley and many others
gnuplot home:  http://www.gnuplot.info
faq, bugs, etc: type "help FAQ"
immediate help: type "help" (plot window: hit 'h')
Terminal type is now 'x11'
gnuplot> plot sin(x)
gnuplot>
```



# Expanse: Matlab from the command line

```
[mthomas@login01 ~]$ srun --partition=debug --pty --account=use300 --nodes=1 --ntasks-per-node=4 --mem=8G -t 00:30:00 --wait=0 --export=ALL /bin/bash
srun: job 14833549 queued and waiting for resources
srun: job 14833549 has been allocated resources
[mthomas@exp-9-55 ~]$ module list
Currently Loaded Modules:
 1) shared    2) slurm/expanse/21.08.8    3) sdsc/1.0    4) DefaultModules    5) cpu/0.15.4    6)
gcc/10.2.0    7) gnuplot/5.2.8

[mthomas@exp-9-55 ~]$ module list
Currently Loaded Modules:
 1) shared    2) slurm/expanse/21.08.8    3) sdsc/1.0    4) DefaultModules    5) cpu/0.15.4    6)
gcc/10.2.0    7) gnuplot/5.2.8
[mthomas@exp-9-55 ~]$ module load matlab/2022a
[mthomas@exp-9-55 ~]$ matlab
MATLAB is selecting SOFTWARE OPENGL
rendering.                                              < M A T L A B (R) >
Copyright 1984-2022 The MathWorks, Inc.
R2022a (9.12.0.1884302) 64-bit (glnxa64)
February 16, 2022
>>a=[1 3 5; 2 4 6; 7 8 10]
a=
 1   3   5
 2   4   6
 7   8  10
>>b=sin(a)
b =
 0.8415  0.1411 -0.9589
 0.9093 -0.7568 -0.2794
 0.6570  0.9894 -0.5440
>>plot(b)
>>
```

# UCSD Triton Shared Computing Cluster (TSCC)

## Interactive Job: Hello-MPI

```
qsub -I -l nodes=2:ppn=10 -l walltime=0:50:00
```

```
[mthomas@tscc-gpu-9-4 hello-mpi]$  
[mthomas@tscc-gpu-9-4 hello-mpi]$ qsub -I -l nodes=1:ppn=16 -l walltime=0:30:00 -q glean  
qsub: waiting for job 26842841.tscc-mgr7.local to start  
qsub: job 26842841.tscc-mgr7.local ready
```

Request interactive node using glean queue node

```
[mthomas@tscc-gpu-9-4 ~]$ pwd  
[mthomas@tscc-gpu-9-4 hello-mpi]$ mpirun -np 8 ./hello-mpi  
node 0 : Hello and Welcome to TSCC!  
node 6 : Hello and Welcome to TSCC!  
node 1 : Hello and Welcome to TSCC!  
node 3 : Hello and Welcome to TSCC!  
node 4 : Hello and Welcome to TSCC!  
node 5 : Hello and Welcome to TSCC!  
node 7 : Hello and Welcome to TSCC!  
node 2 : Hello and Welcome to TSCC!
```

Interactive nodes can be used for running parallel jobs (MPI, OpenMP, CUDA code if GPU), compiling large codes, viz apps.

# TSCC Interactive Jobs: Applications Running Matlab without GUI

```
qsub -l -l nodes=1:ppn=16 walltime=0:30:00 -q glean
```

```
[mthomas@tscc-login1 hello-mpi]$ qsub -I -l nodes=1:ppn=16 -l  
walltime=0:30:00 -q glean
```

```
qsub: waiting for job 26842936.tscc-mgr7.local to start  
qsub: job 26842936.tscc-mgr7.local ready
```

```
[mthomas@tscc-13-12 ~]$ module load matlab  
[mthomas@tscc-13-12 ~]$ matlab -nodisplay  
    < M A T L A B (R) >  
Copyright 1984–2016 The MathWorks, Inc.  
R2016b (9.1.0.441655) 64-bit (glnxa64)  
September 7, 2016
```

```
To get started, type one of these: helpwin, helpdesk, or demo.  
For product information, visit www.mathworks.com.
```

```
>> A = [1 3 0; 2 4 -1; 4 9 -1]  
A =  
    1     3     0  
    2     4    -1  
    4     9    -1  
>> B=A'  
B =  
    1     2     4  
    3     4     9  
    0    -1    -1  
>> A*B  
ans =  
   10    14    31  
   14    21    45  
   31    45    98  
>> exit  
[mthomas@tscc-13-12 ~]$
```

# TSCC Interactive Jobs: Launching Matlab with GUI

To use a Graphical User Interface (GUI) as part of your interactive job, **you will need to set up Xforwarding**. Example below is for using XQuartz on a MAC. For examples for MacOS and Windows, see: <http://systems.eecs.tufts.edu/x11-forwarding/>

**Step 1:** Set X11 forwarding on the computer that you are connecting from:  
Update or install Xquartz and **restart** your system

```
[mthomas@home]$ brew install --cask xquartz
Updating Homebrew...
Updated 2 taps (homebrew/core and homebrew/cask).
==> New Formulae
[SNIP]
installer: The upgrade was successful.
🍺 xquartz was successfully installed!
quantum:~ mthomas$ which xquartz
/opt/X11/bin/xquartz
```

**Step 2:** Log on to TSCC, using **-Y** option (trusted)

```
[mthomas@home]$ ssh -Y tscc-login.sdsc.edu -l mthomas
```

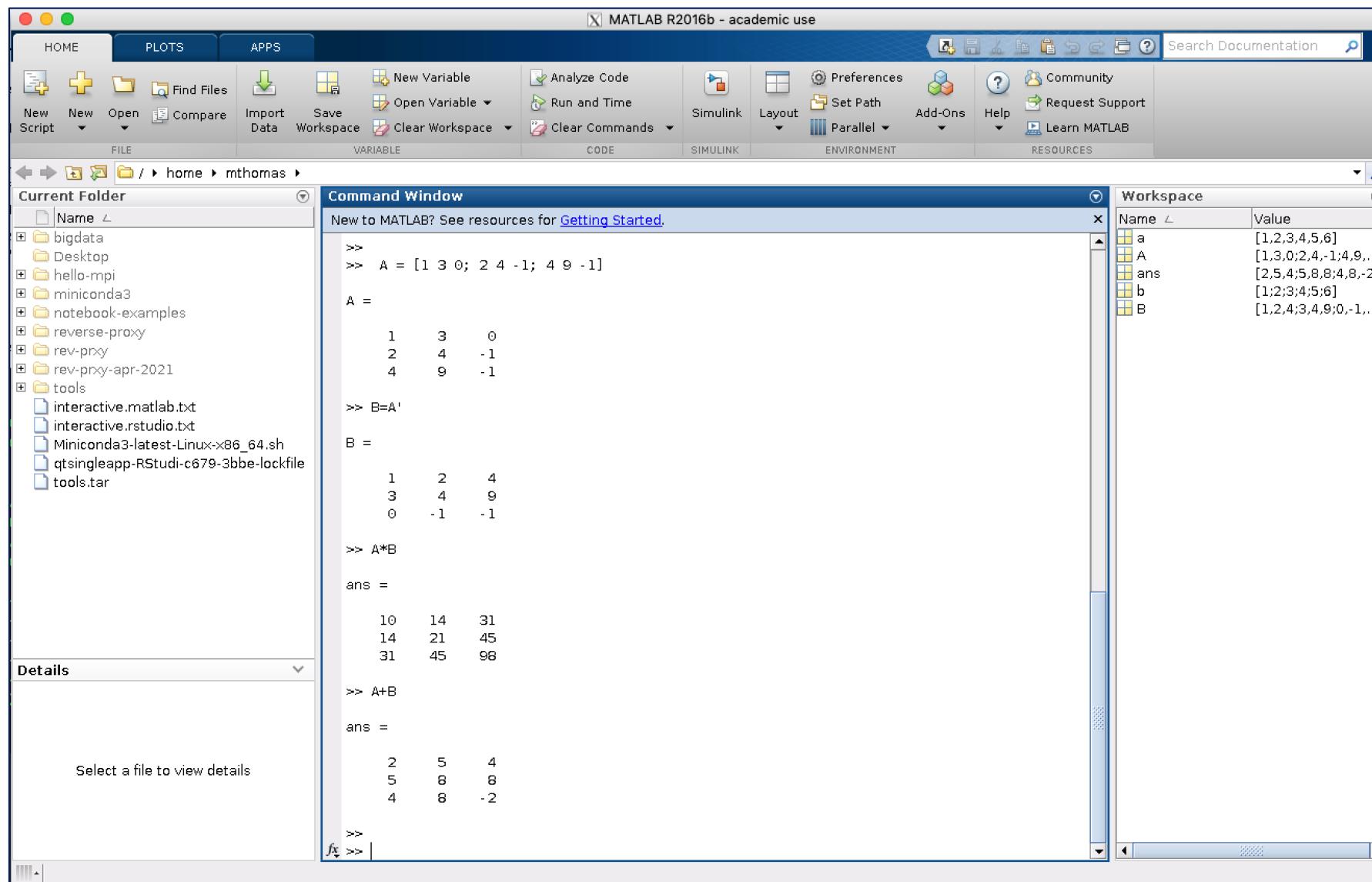
**Step 3:** Request an interactive node, using **-X** option

```
[mthomas@tscc-login1 hello-mpi]$ qsub -I -X -l
nodes=1:ppn=1 -q glean
qsub: waiting for job 26843028.tscc-mgr7.local to start
qsub: job 26843028.tscc-mgr7.local ready
[mthomas@tscc-4-46 ~]$
```

**Step 4:** Setup your module environment and run Matlab

```
[mthomas@tscc-4-46 ~]$ module load matlab
[mthomas@tscc-4-46 ~]$ matlab
MATLAB is selecting SOFTWARE OPENGL rendering.
```

# TSCC Interactive Jobs: Running Matlab with GUI



# TSCC Interactive Jobs: Running R console (no GUI)

**Step 3:** Request an interactive node, using **-X** option

```
[mthomas@tscc-login2 ~]$ qsub -l -q glean -l nodes=1:ppn=1
qsub: waiting for job 26844488.tscc-mgr7.local to start
qsub: job 26844488.tscc-mgr7.local ready
[mthomas@tscc-4-46 ~]$
```

**Step 4:** Setup your module environment and run Matlab

```
[mthomas@tscc-4-46 ~]$ module load R
Unloading compiler-dependent module openmpi_ib/3.1.4
[mthomas@tscc-4-46 ~]$ R
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)
```

R is free software and comes with ABSOLUTELY NO WARRANTY.  
[SNIP]

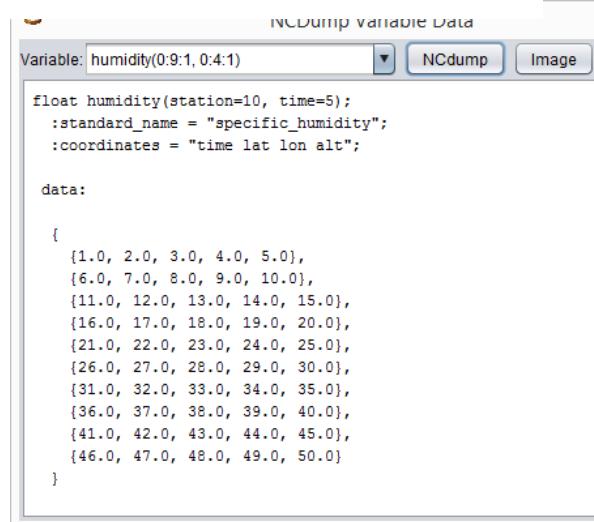
R is a collaborative project with many contributors.

[SNIP]

```
> myString <- "Hello, World!"
> print ( myString )
[1] "Hello, World!"
>
```

# Viewing NetCDF Files

- NetCDF (Network Common Data Form) is a set of software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data
- NetCDF clients (ncview, ncdump) can be used to query and plot data in real-time



INCDUMP VARIABLE Data

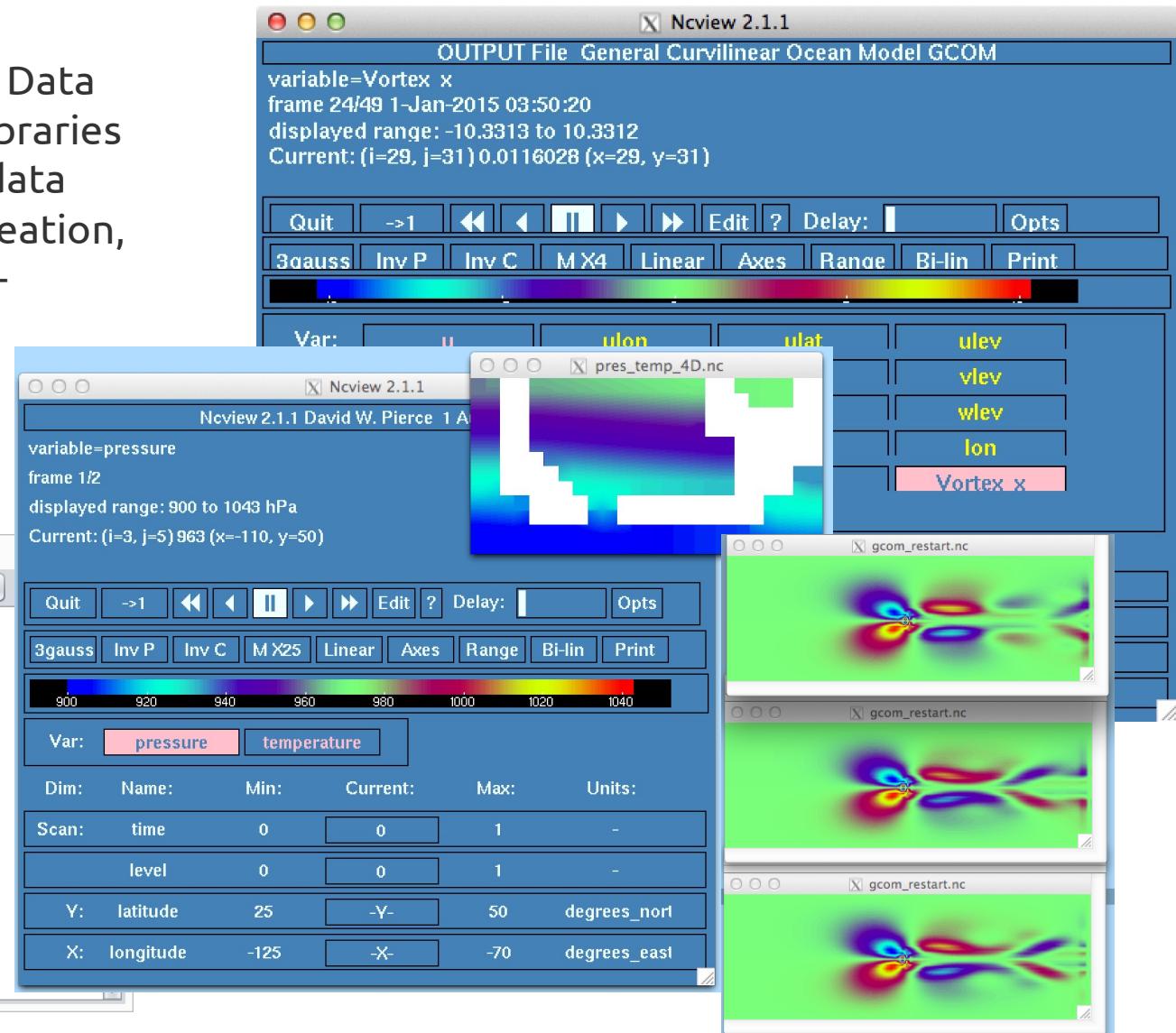
Variable: humidity(0:9:1, 0:4:1)

NCdump Image

```
float humidity(station=10, time=5);
:standard_name = "specific_humidity";
:coordinates = "time lat lon alt";

data:

{
    {1.0, 2.0, 3.0, 4.0, 5.0},
    {6.0, 7.0, 8.0, 9.0, 10.0},
    {11.0, 12.0, 13.0, 14.0, 15.0},
    {16.0, 17.0, 18.0, 19.0, 20.0},
    {21.0, 22.0, 23.0, 24.0, 25.0},
    {26.0, 27.0, 28.0, 29.0, 30.0},
    {31.0, 32.0, 33.0, 34.0, 35.0},
    {36.0, 37.0, 38.0, 39.0, 40.0},
    {41.0, 42.0, 43.0, 44.0, 45.0},
    {46.0, 47.0, 48.0, 49.0, 50.0}
}
```



# Outline

- What is Interactive High-Performance Computing?
- Accessing Interactive HPC Nodes
- Running Interactive Apps
- Running Interactive Apps Using the Expanse Portal
- Notebook Security

# Expanse User Portal

The image shows a screenshot of the Expanse User Portal interface. The main dashboard includes:

- File Browsing, editing, creation, upload, download, etc.**: A file browser window showing a list of files in a directory.
- SDSC**: The portal provides an integrated, and easy to use interface to access Expanse HPC resources.
- Pinned Apps**: A featured subset of all available apps, including Active Jobs, Home Directory, Job Composer, Expanse Shell Access, MATLAB, RSTUDIO, Allocation and Usage Information, and Jupyter.
- OOD 2.0 Features**: A section showing Active Jobs, Allocation and Usage Information, and Job Composer.
- Interactive Services**: A window showing the MATLAB interface with a plot of a function.
- Job Script Editing and Submission**: A window showing the Job Details for a "Simple Sequential Job" named "linear".
- Active Job monitoring and Management**: A table showing a list of active jobs, including columns for Job ID, User, Account, Time, Used CPU, Status, and Cluster.

- <https://portal.expanse.sdsc.edu>; access using XSEDE credentials
- Securely hosts batch job submission & monitoring, and interactive applications
- Portal simplifies launching supported interactive applications → manages software dependencies

# Expanse Portal: File Management

The screenshot illustrates the Expanse Portal's file management interface and its integration with a terminal session.

**File Management Interface:** The top portion shows a list of files and directories in the '/home/mthomas/expanse/' directory. Key features include:

- A toolbar with buttons for "Open in Terminal", "New File", "New Directory", "Upload", "Download", and "Copy/Mov".
- A breadcrumb navigation bar: "/ home / mthomas / expanse /".
- A "Change directory" input field.
- Filtering options: "Show Owner/Mode", "Show Dotfiles", and "Filter: [text input field]."
- A message indicating "Showing 28 of 62 rows".

**Terminal Session:** A red arrow points from the "Open in Terminal" button in the file management interface to a terminal window on the right. The terminal window shows a session on the host "login.expanse.sdsc.edu" with the initial directory "/home/mthomas/expanse". The session output includes a long list of file and directory entries, such as .viminfo, .vimrc, .vnc, wget-hsts, README.txt, classes, comet-files, conda-activate.txt, conda-install-tmp, dev, galileo-examples, galileo-repo, gnuplot-ex, gpuhack22, hpctr-examples, and hpctrain. The terminal prompt is "[mthomas@login01 ~]\$".

# Expanse Portal: Running Matlab

The image shows a multi-step process for launching a Matlab session on the Expanse Portal:

- Step 1: Interactive Apps Selection**  
A screenshot of the "Interactive Apps" section of the portal. Under the "MATLAB" entry, a tooltip provides information: "This app will launch a MATLAB GUI on Expanse. You will be able to interact with the MATLAB GUI through a VNC session. Please email [help@xsede.org](mailto:help@xsede.org) to be added to matlab-groups." Below it, the "Session ID" is listed as `fee7f0fb-48df-46af-8f4`.
- Step 2: Session Configuration**  
The configuration form for the Matlab session:
  - compute**: The selected compute node.
  - Reservation**: An empty text input field.
  - Number of hours**: Set to 1.
  - Account**: Set to `use300`.
  - I would like to receive an email when the session starts**: A checked checkbox.
  - Working directory**: Set to `home`.
  - Number of cores**: Set to 1.
  - Memory (GB)**: Set to 64.
- Step 3: Session Confirmation**  
A confirmation message: "Session was successfully deleted." This likely refers to a previous step where a session was terminated.
- Step 4: Session Details**  
A summary of the Matlab session:
  - Host:** `exp-1-18.expanse.sdsc.edu`
  - Created at:** 2022-08-01 22:48:55 PDT
  - Time Remaining:** 25 minutes
  - Session ID:** `fee7f0fb-48df-46af-8f40-ea7bf62c1bda`
  - Nodes:** 1 node | **Cores:** 128 cores | **Status:** Running
- Step 5: Matlab GUI**  
A screenshot of the Matlab R2020b interface. The Command Window shows:

```
>> a=[1,2,3,4,5]
a=[1,2,3,4,5]
!
```

An error message: "Error: Invalid expression. When calling a function or indexing a variable, use parentheses. Otherwise, check for mismatched delimiters."

```
>> a=[1 2 3 4 5]
a =
1 2 3 4 5
```

```
>> b=sin(a)
b =
0.8415 0.9099 0.1411 -0.7568 -0.9589
```

```
>> plot(b)
Warning: MATLAB has disabled some advanced graphics rendering features by switching to software OpenGL. For more information, click here.
```

# Expanse Portal: Launching Notebooks

The screenshot displays three browser windows illustrating the process of launching a Jupyter notebook session.

- Top Left Window:** A configuration form titled "Jupyter Session". It includes fields for Account (use300), Partition (shared), Time limit (min) (30), Number of cores (1), Memory required per node (GB) (2), and GPUs (optional) (0). Below the form, there's a note about Singularity Image File Location with options for /cm/shared/apps/container or /cm/shared/apps/containers/singularity/pytorch/pytorch-latest.sif.
- Top Right Window:** A file browser showing the contents of a directory. It lists files: numpy\_intro.ipynb, hello\_world\_gpu.ipynb, hello\_world.ipynb (selected), hello.py, and README.md. The hello\_world.ipynb file was modified 24 minutes ago.
- Bottom Window:** A Jupyter Notebook interface. The code cell at [9] contains:

```
llc cqm_mbm_total cqm_mbm_local clzer
o iperf x
lbrv svm_l
ushbyasid
avic v_vm
id overflow
[9]: # Check to see if system is GPU:
!nvidia-smi
```

A message indicates that NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. The code cell at [10] contains:

```
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA
driver. Make sure that the latest NVIDIA driver is installed and runni
ng.
```

```
[10]: # if you see: /bin/bash: nvidia-smi: command not found
# the system is not GPU
```

- PORTAL LIVE DEMO

# Outline

- What is Interactive High-Performance Computing?
- Accessing Interactive HPC Nodes
- Running Interactive Apps
- Running Interactive Apps Using the Expanse Portal
- Notebook Security

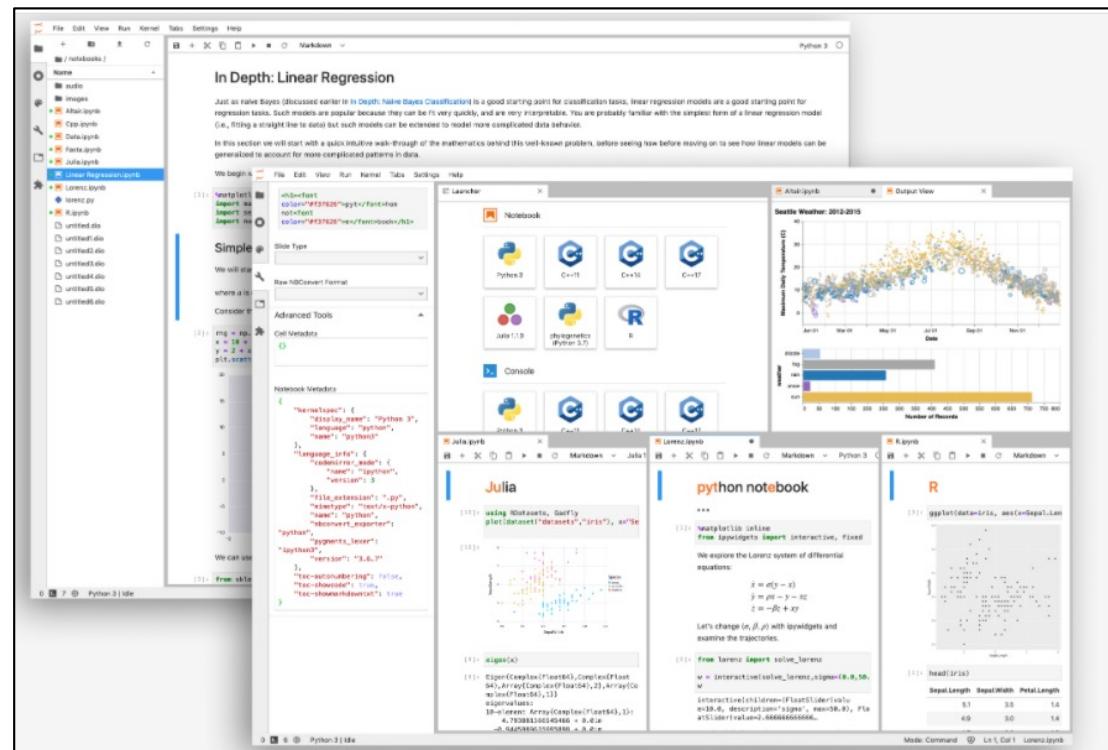
# Jupyter Notebooks

## What is Jupyter?

*Jupyter is a free, open-source, **interactive** web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document. (J. Perkel, <https://www.nature.com/articles/d41586-018-07196-1>)*

## Common Jupyter Services:

- Jupyter Notebooks (single user)
- JupyterLab: advanced version of notebook
- JupyterHub: multiuser Jupyter service



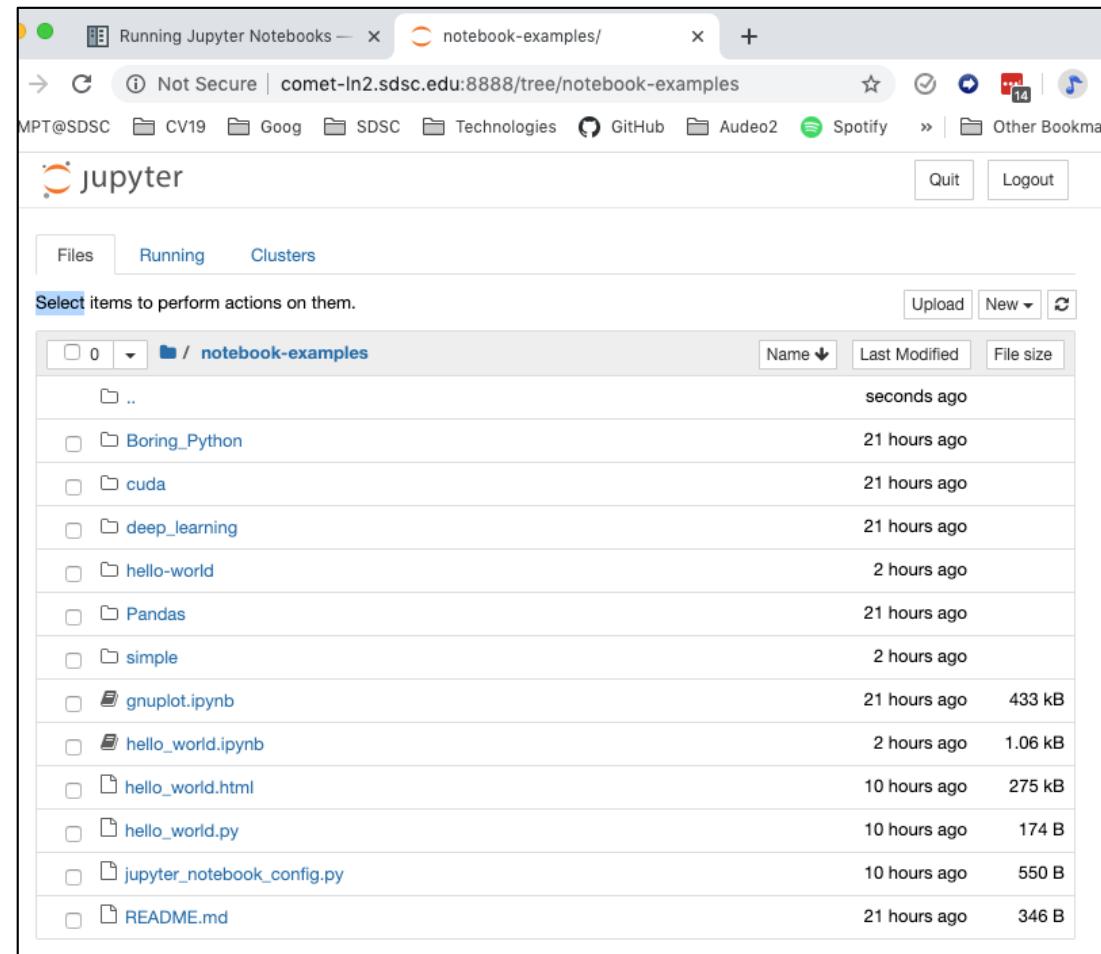
# Interactive Services have a Key Vulnerability: They Provide Access to HPC File Systems, often over HTTP

## SDSC Interactive Services Policy:

- Portals, JupyterHub, and other services cannot be mounted directly to disk (must be on VM or external)
  - Many use root in vulnerable ways
  - If a user launches Jupyter Lab or Notebooks, the jobs will be killed.
- Applications cannot run on login nodes

## SDSC recommendation:

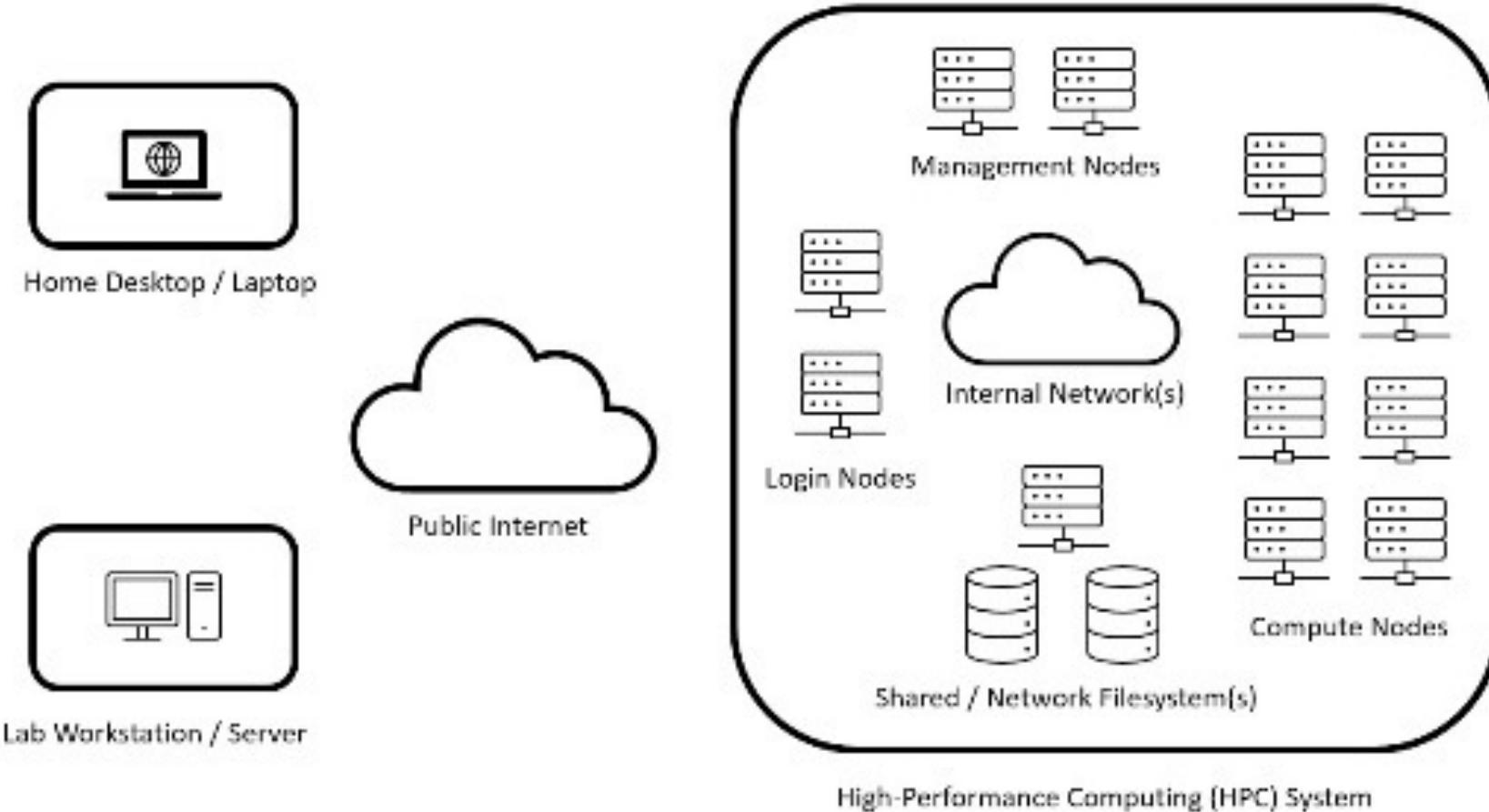
- Use secure connections: when you choose insecure connections your account is vulnerable to hacking
- Use portal.expanse.sdsc.edu



# Accessing and Running Secure Notebooks on HPC Systems

- Install notebook application:
  - Locally: install Anaconda on your laptop
  - Remotely:
    - Install Anaconda/conda on the remote machine (default is HTTP) – **not recommended**
    - Launch securely (HTTPS) using SRPS/*galyleo* -- **recommended**
- Running remotely:
  - Connect over HTTP (default, insecure)
  - Connect over HTTP + SSH tunneling (secure, but inconvenient)
  - **Connect over HTTPS + using the *Satellite Reverse Proxy Service* (SRPS) and *galyleo client* (secure, convenient)**
- You can launch Jupyter services on SDSC:
  - CPU and GPUs
  - Interactive nodes: command line
  - Slurm batch script
- **Treat the Notebook URL like a Password!**

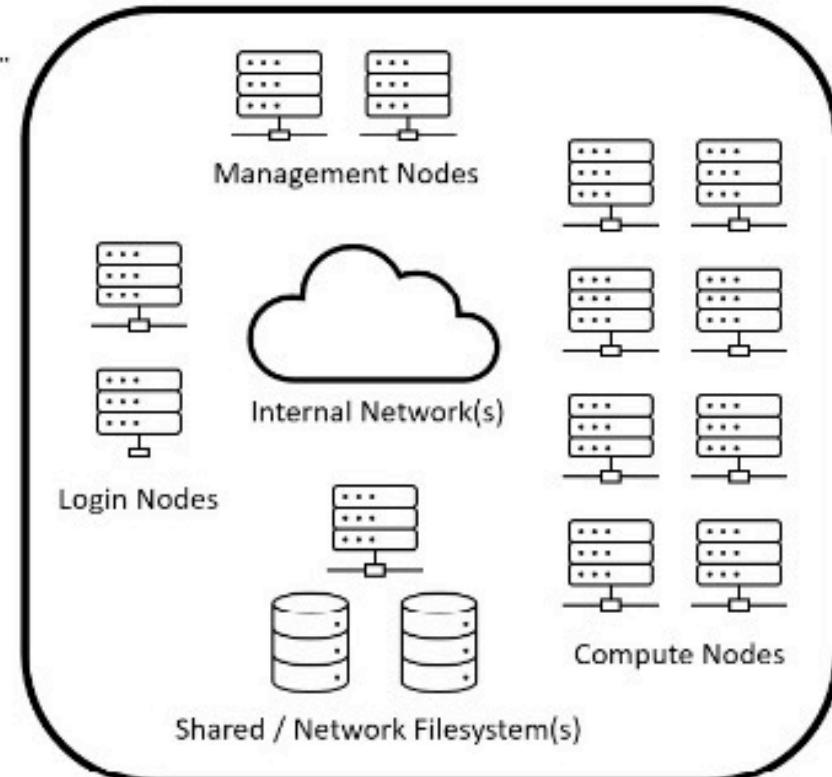
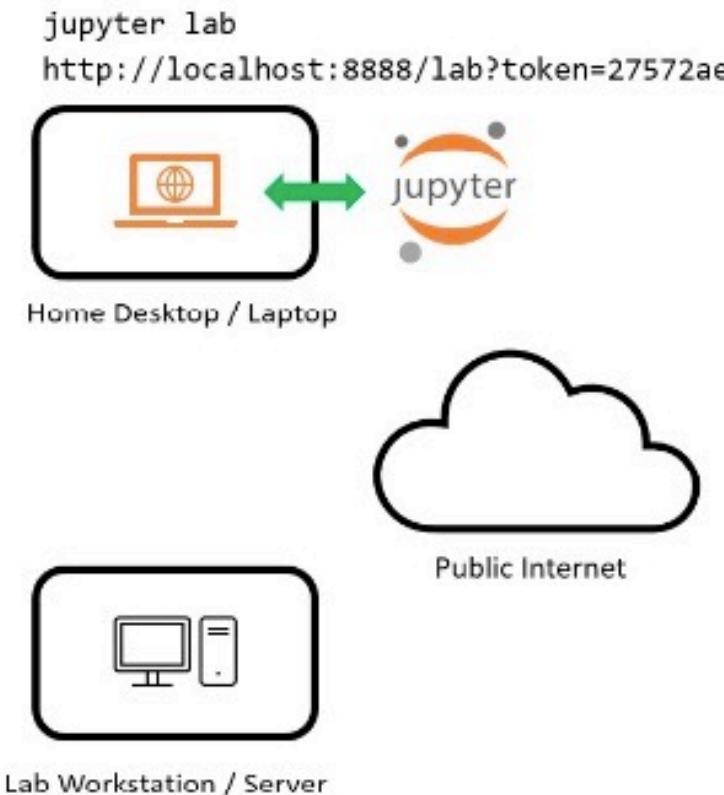
# Standard HPC Ecosystem



Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

# Running Notebooks Locally Over HTTP

Install the Anaconda application and use it to launch your notebooks



High-Performance Computing (HPC) System

Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

# Anaconda on Local Host or Laptop

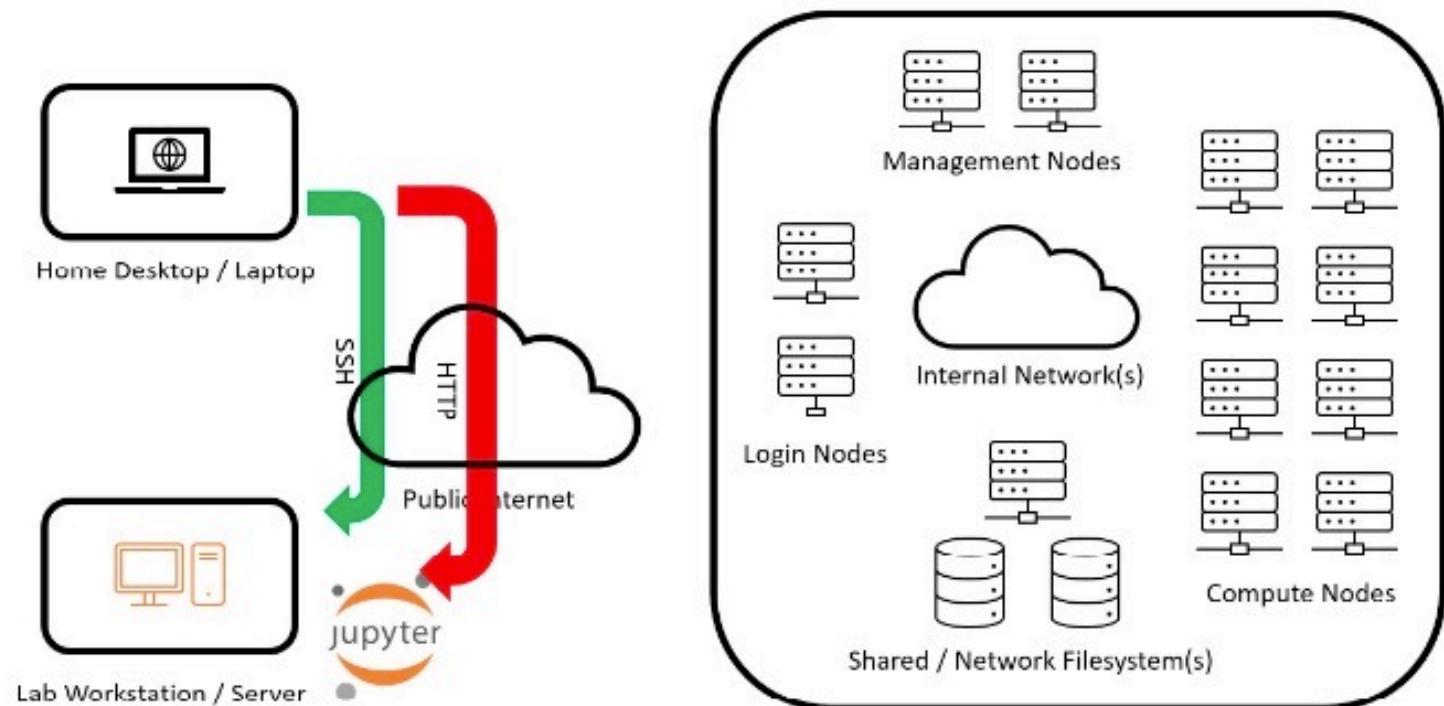
The image shows the Anaconda Navigator interface running on a local host or laptop. The main window displays a file browser on the left and a central area for managing various applications. A red circle highlights the "CONDA NAVIGATOR" title at the top of the central area. Below it, the "Applications on base (root)" section lists several data science tools:

- JupyterLab**: Version 1.0.2. Description: An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture. A red circle highlights the "Launch" button.
- Notebook**: Version 6.0.0. Description: Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis. A red circle highlights the "Launch" button.
- Spyder**: Version 3.3.6. Description: Scientific Python Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features. A red circle highlights the "Launch" button.
- Glueviz**: Version 0.13.3. Description: Multidimensional data visualization across files. Explore relationships within and among related datasets. A green "Install" button is visible.
- Orange 3**: Version 3.19.0. Description: Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox. A green "Install" button is visible.
- RStudio**: Version 1.1.456. Description: A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks. A green "Install" button is visible.

To the right of the application list, there is a file browser window titled "localhost:8807/tree/dev/sdsc...." showing files like "hello.ipynb", "hello\_world.ipynb", "hello.py", and "hello.rb". Below the file browser is a terminal window showing command-line output related to Jupyter and Python environments.

# Running Notebooks Remotely Over HTTP

- Messages over HTTP
- Not encrypted
- Vulnerable to hacking

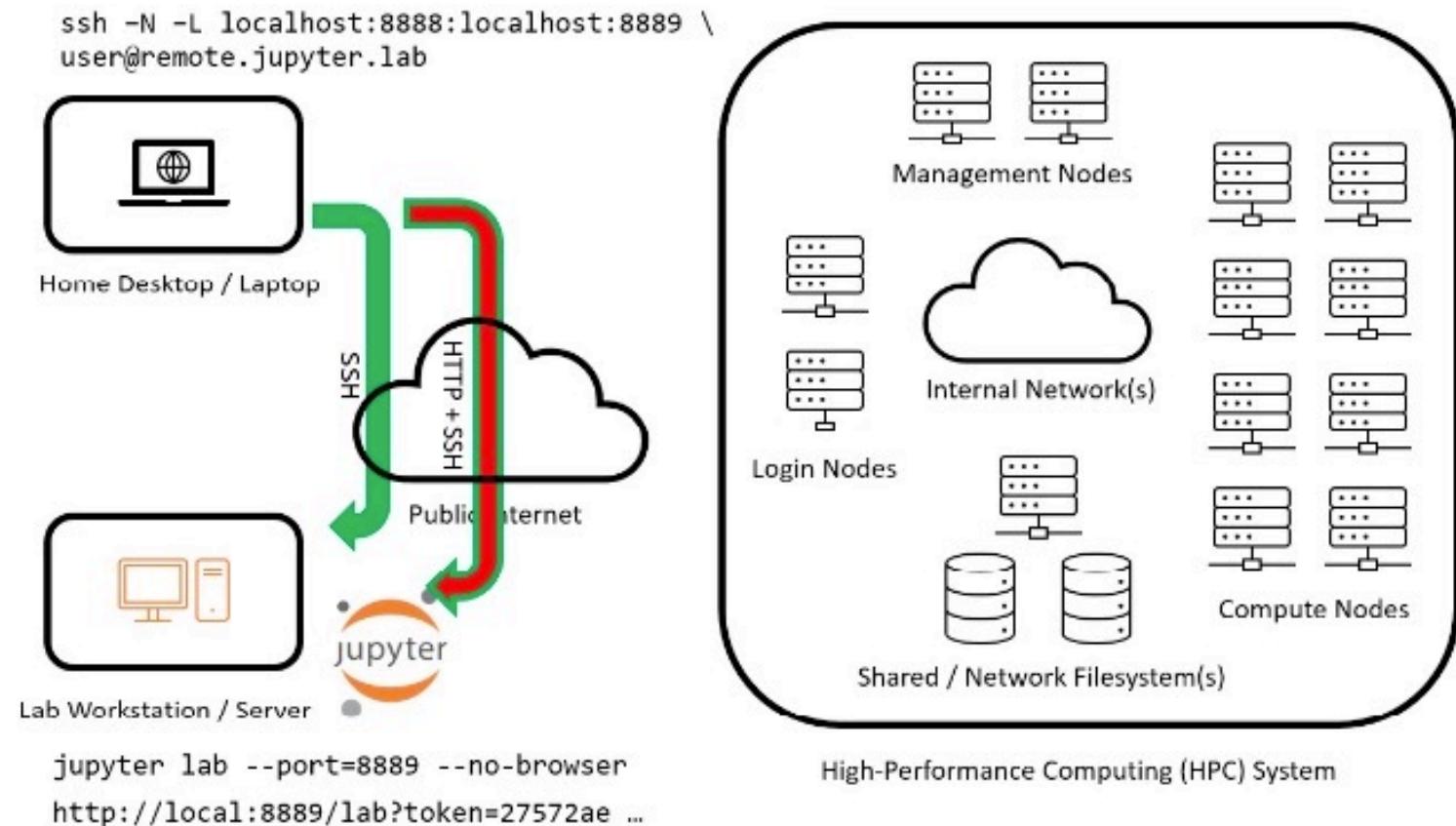


```
jupyter lab --ip="$(hostname)" --no-browser      High-Performance Computing (HPC) System  
http://remote.jupyter.lab:8888/lab?token=27572ae ...
```

Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

# Running Notebooks Remotely Over HTTP Using SSH Tunneling

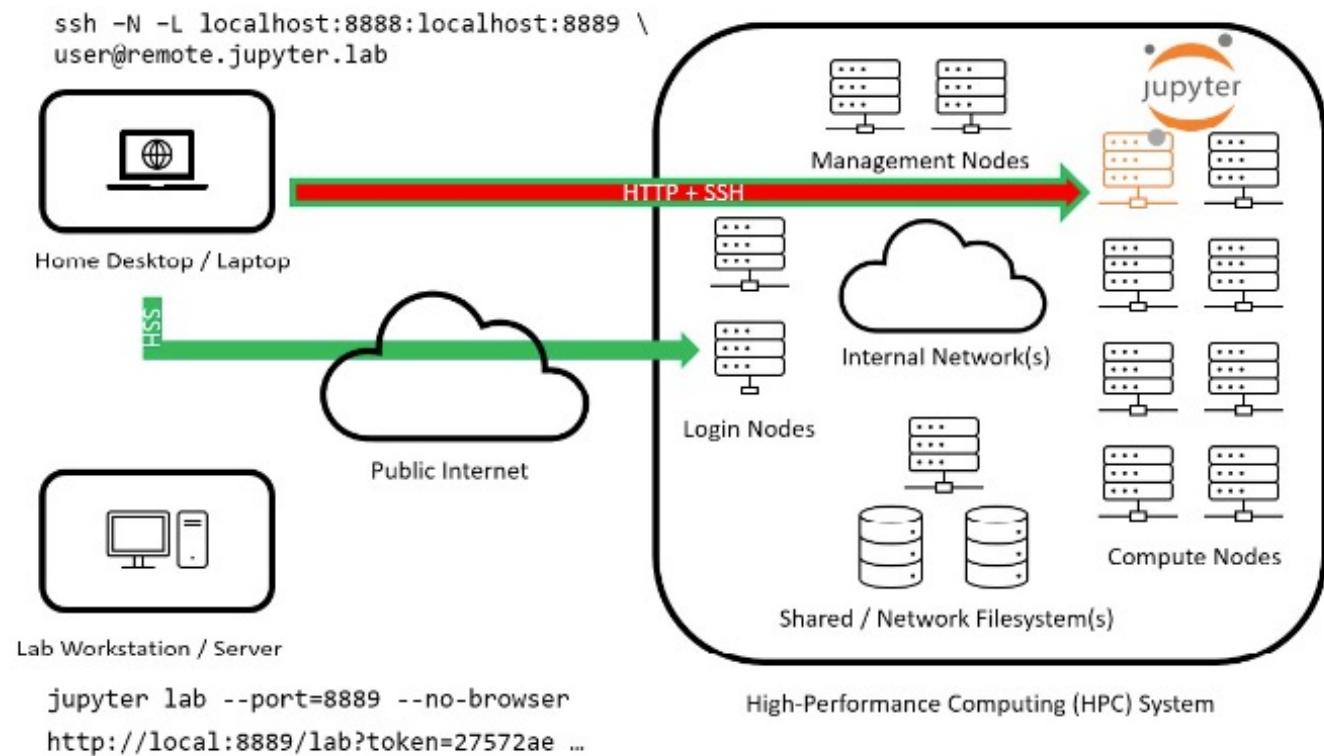
- Notebooks running on Remote host
- Messages over HTTP, but sent via secure tunnel



Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

# Running Notebooks Remotely Over HTTP Using SSH Tunneling on HPC System

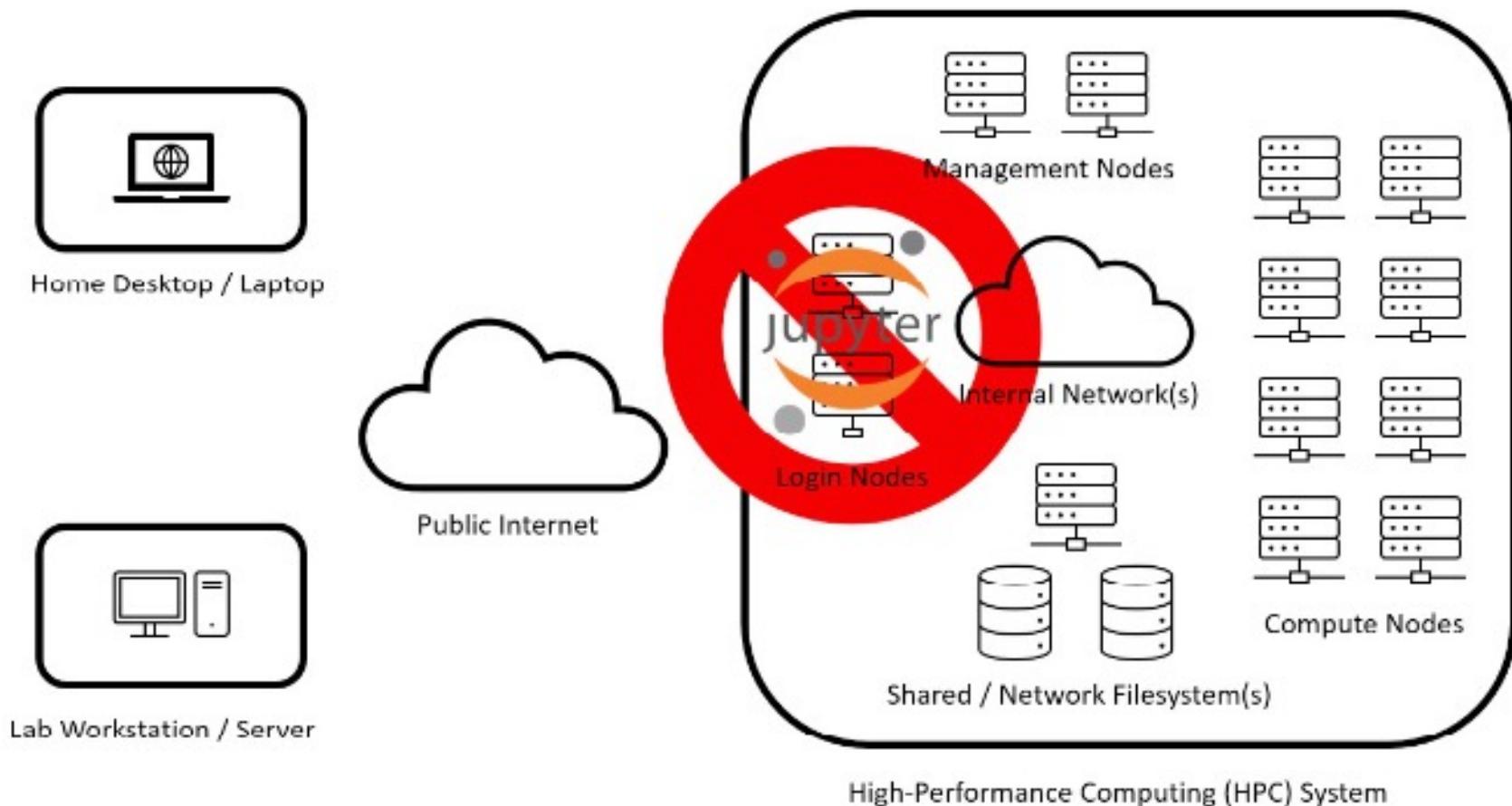
- Notebook running on HPC System
- Messages over HTTP, but passing via secure tunnel
- But do not run on compute nodes!



Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

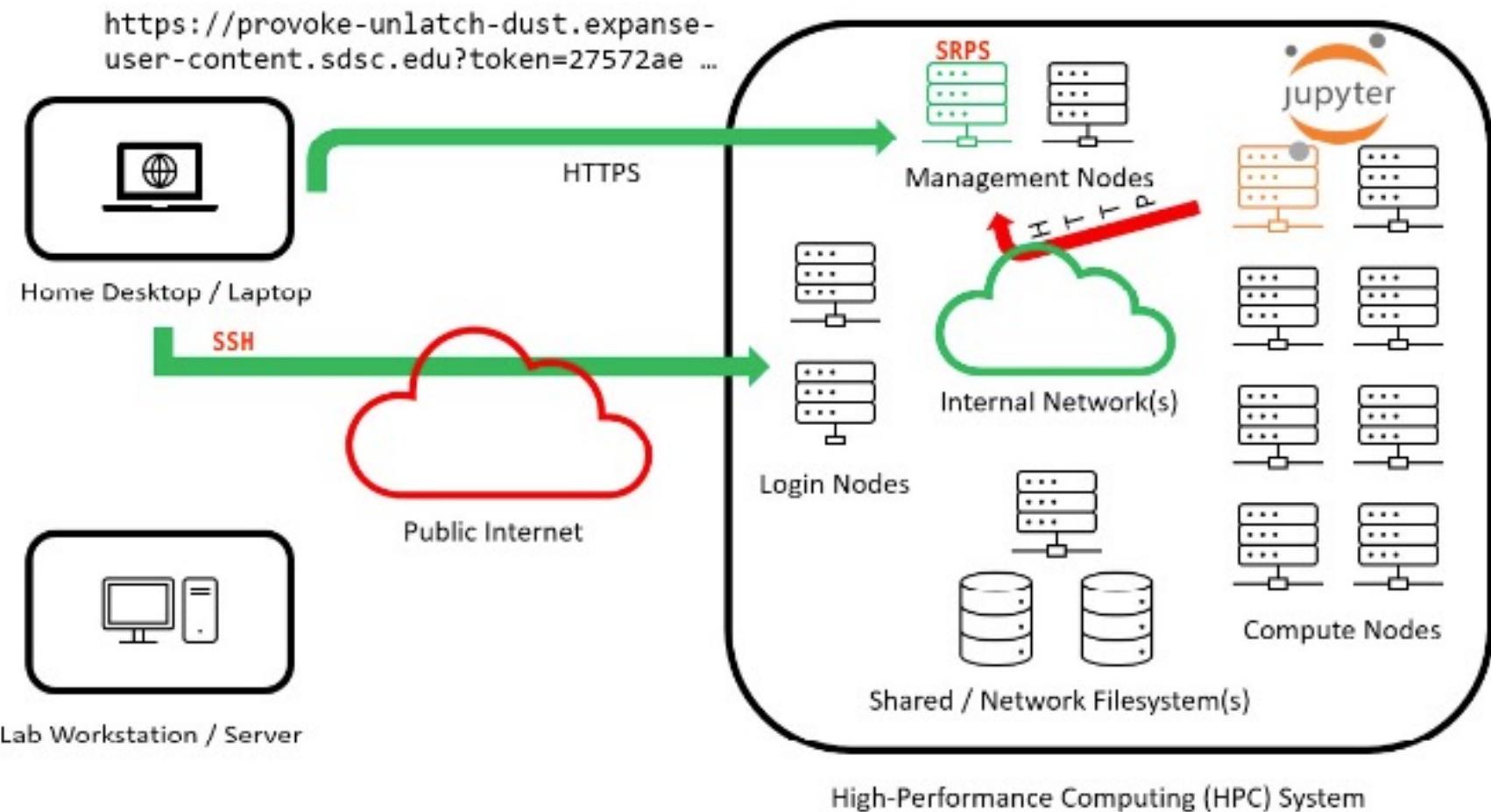
# Warning

**Do not run notebooks/interactive services on the login nodes!**



Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

# Running Notebooks Remotely Using SRPS



Src: M Kandes: [https://education.sdsc.edu/training/interactive/202112\\_running\\_jupyter\\_notebooks\\_on\\_expanse/](https://education.sdsc.edu/training/interactive/202112_running_jupyter_notebooks_on_expanse/)

# SDSC Satellite Reverse Proxy Service

- SRPS: prototype system that allows users to launch secure standard Jupyter Notebooks on any Expanse compute node using a reverse proxy server.
  - Notebooks will be hosted on the internal cluster network as an HTTP service using standard Jupyter commands.
  - Service available to the user outside of the cluster firewall over HTTPS connection between the external users web browser and the reverse proxy server.
- Goal: minimize software changes for users, improve security of user notebooks running on SDSC systems.
- SRPS can run on any HPC system capable of supporting Apache on internal network.

# galyleo

- 2nd generation shell utility developed to orchestrate a user's interaction with both Satellite and Slurm to start a Jupyter session within a batch job.
- Developed while reviewing start-jupyter (prototype client) codebase to sort out how best to support Expanse (OOD) Portal and HPC User Services Group long-term; integrated into an existing SSH tunneling orchestration utility to use Satellite proxy service instead
- Key features in design:
  - No need to install conda environment or update packages
  - Increases flexibility for users to configure software environment; but also try to makes it simpler for them to do this themselves
  - Batch job script is generated completely on-the-fly.
  - Command-line argument driven.
  - Quiet mode for OOD portal

<https://github.com/mkandes/galyleo>

# Satellite Client: galyleo

Key features in design:

- User calls galyleo.sh launch script, which requests token from Satellite, passes token to batch job script and submits the job to Slurm; token redeemed from batch job once it runs
- Increase flexibility for users to configure software environment; but also try to make it simpler for them to do themselves
- Batch job script is generated completely on-the-fly.
- Command-line argument driven.
- Quiet mode for OOD portal

```
[username@login01 ~]$ galyleo.sh --help  
USAGE: galyleo.sh launch [command-line  
option] {value}
```

| command-line option    | : | value |
|------------------------|---|-------|
| -A   --account         | : |       |
| -R   --reservation     | : |       |
| -p   --partition       | : |       |
| -q   --qos             | : |       |
| -N   --nodes           | : |       |
| -n   --ntasks-per-node | : |       |
| -c   --cpus-per-task   | : |       |
| -M   --memory-per-node | : | GB    |
| -m   --memory-per-cpu  | : | GB    |
| -G   --gpus            | : |       |
| --gres                 | : |       |
| -t   --time-limit      | : |       |
| -j   --jupyter         | : |       |
| -d   --notebook-dir    | : |       |
| -r   --reverse-proxy   | : |       |
| -D   --dns-domain      | : |       |
| -s   --sif             | : |       |
| -B   --bind            | : |       |
| --nv                   | : |       |
| -e   --env-modules     | : |       |
| --conda-env            | : |       |
| -Q   --quiet           | : | 1     |

<https://github.com/mkandes/galyleo>

# Launching CPU notebooks using galileo

**Step 1:**  
Login and setup user environment

**Step 2:**  
Run command to launch secure notebook

**Step 3:**  
Copy URL; paste into browser on local system

**Step 4:**  
Monitor Slurm queue;  
wait for job to start

```
[username@login01 ~]$ which galileo.sh  
/usr/bin/which: no galileo.sh in (/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen/gcc-8.3.1  
[SNIP]  
home/username/.local/bin:/home/username/bin)  
[username@login01 ~]$ export PATH="/cm/shared/apps/sdsc/galileo:${PATH}"  
[username@login01 ~]$ which galileo  
/cm/shared/apps/sdsc/galileo/galileo
```

```
[username@login01 ~]$ galileo launch --account use300 --partition shared --cpus 1 --  
memory 2 --time-limit 00:30:00 --env-modules cpu,gcc,anaconda3  
[snip]
```

Submitted Jupyter launch script to Slurm. Your SLURM\_JOB\_ID is 9773665

[snip]

Your Jupyter notebook session will begin once compute resources are allocated to your job by the scheduler.

```
https://carload-spray-koala.expanse-user-content.sdsc.edu/lab
```

```
[username@login01 ~]$ squeue -u username  
JOBID PARTITION NAME USER ST  
9773665 gpu-debug galileo- username PD
```

| TIME | NODES | NODELIST(REAON) |
|------|-------|-----------------|
| 0:00 | 1     | (None)          |

```
[username@login01 ~]$ squeue -u username  
JOBID PARTITION NAME USER ST  
9773665 gpu-debug galileo- username R
```

| TIME | NODES | NODELIST(REAON) |
|------|-------|-----------------|
| 0:20 | 1     | exp-7-59        |

# Satellite Server Pending Page

- Load notebook URL in browser; wait for it to launch
- Monitor pending page
- Run the “squeue” command on the HPC system to check job status
- If the job queue is busy, it may take a while to launch the notebook
- **Treat Jupyter Notebook URL as a password!**

### Satellite Reverse Proxy Service

SDSC Expanse

Job State: Mapped

In Queue  
Job has not yet started.

Running  
Job has started, but has not redeemed Satellite Token.

Mapped  
Job has redeemed Satellite Token, but no proxy entry exists yet.

Proxied  
Proxy entry created, ready to go!

Dead  
Job died or exited, no further progress will occur.

```
[mthomas@login02 ~]$ squeue -u mthomas
      JOBID PARTITION     NAME      USER ST      TIME  NODES NODELIST(REASON)
      9774239  shared galileo-  mthomas R   3:49      1 exp-1-13
      9774274  debug galileo-  mthomas R   0:12      1 exp-9-55
[mthomas@login02 ~]$
```

# Satellite Pending Page

The screenshot shows a Jupyter Notebook interface running on a system with a Satellite Reverse Proxy Service. The notebook displays a 'Hello World' example and a terminal session showing the execution of a Python script.

**Jupyter Notebook Content:**

- Hello World:** A heading followed by the text "FileName: hello\_world\_cpu.ipynb".
- CPU Version:** A section stating "No package dependencies".
- Terminal Session:**
  - [8]: `print('Hello world!!!!')`  
Hello world!!!!
  - [9]: `# Import hello module  
import hello`  
`# Define a local function  
def world2(name):  
 print(name)`
  - [10]: `# Call function  
world2("mary")`  
mary
  - [11]: `hello.greeting("good times")`  
Greetings, good times
  - [12]: `hello.world("World.")`  
Hello, World.

**Satellite Reverse Proxy Service Status:**

The status page shows the job state transitions: In Queue, Running, Mapped, and Proxied. It also includes a detailed description of each state:

- In Queue:** Job has not yet started.
- Running:** Job has started, but has not redeemed Satellite Token.
- Mapped:** Job has redeemed Satellite Token, but no proxy entry exists yet.
- Proxied:** Proxy entry created, ready to go!
- Dead:** Job died or exited, no further progress will occur.

# Launching GPU notebooks using galyleo

- GPU Notebooks run better when using containers. SDSC maintains several containers on Expanse
- See: /cm/shared/apps/containers/singularity

Step 1:  
Login and setup user environment

```
[username@login01 ~]$ which galyleo.sh  
/usr/bin/which: no galyleo.sh in (/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen/gcc-8.3.1  
[SNIP]  
home/username/.local/bin:/home/username/bin)  
[username@login01 ~]$ export PATH="/cm/shared/apps/sdsc/galyleo:${PATH}"  
[username@login01 ~]$ which galyleo  
/cm/shared/apps/sdsc/galyleo/galyleo  
[username@login01 ~]$ galyleo launch --account use300 --partition gpu-debug --cpus  
10 --memory 93 --gpus 1 --time-limit 00:5:00 --env-modules singularitypro --sif  
/cm/shared/apps/containers/singularity/pytorch/pytorch-latest.sif --bind  
/expanses,/scratch --nv
```

Step 2:  
Run command to launch secure notebook

```
[snip]  
Submitted Jupyter launch script to Slurm. Your SLURM_JOB_ID is 9773912  
[snip]  
Your Jupyter notebook session will begin once compute resources are allocated to  
your job by the scheduler.  
https://grief-fantastic-given.expanse-user-content.sdsc.edu?token=5097acb6f32ab82dd51b4524c267d2fd
```

Step 3:  
Copy URL; paste into browser on local system

```
[username@login01 ~]$ squeue -u username  
JOBID PARTITION      NAME      USER ST          TIME NODES NODELIST(REASON)  
9773912 gpu-debug galyleo- username PD          0:00      1 (None)  
[username@login01 ~]$ squeue -u username  
JOBID PARTITION      NAME      USER ST          TIME NODES NODELIST(REASON)  
9773912 gpu-debug galyleo- username R           0:20      1 exp-7-59
```

Step 4:  
Monitor Slurm queue;  
wait for job to start

# Notebook is launched on GPU device

The screenshot shows a Jupyter Notebook interface running in a web browser. The left sidebar displays a file browser with the following contents:

- /notebook-examples / Hello\_World /
- Name      Last Modified
- hello\_worl... 7 months ago
- hello\_worl... 28 minutes ago (selected)
- hello.py 7 months ago
- README.md 7 months ago

The main area shows two tabs: "numpy\_intro.ipynb" and "hello\_world\_gpu.ipynb". The "hello\_world\_gpu.ipynb" tab is active, showing Python 3 code output. The code includes a long string of CPU-related assembly instructions and a command to check for a GPU:

```
sc art arch_perfmon pebs rep_good nopl xtopology nonstop_tsc cpuid aperf fmp perf pni pclm ulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpc pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowp refetch cpuid_fault epb cat_l3 cdp_l3 invpcid_single intel_ppin ssbd mba ibrs ibpb stibp i brs_enhanced tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm cqm mpx rdt_a avx512f avx512dq rdseed adx smap clflushopt clwb intel_pt avx512cd avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mb_m_total cqm_mbm_local dtherm ida arat pln pts pku ospke avx512_vnni md_clear flush_l1d arc_h_capabilities
```

```
[9]: # Check to see if system is GPU:  
!nvidia-smi
```

The output of the command is shown in a terminal-like window, with the NVIDIA-SMI information highlighted by a yellow oval:

```
Fri Feb 18 00:15:50 2022  
+-----+  
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2 |  
+-----+  
| GPU Name Persistence-M| Bus Id Disp.A | Volatile Uncorr. ECC |  
| Fan Temp Perf Pwr:Usage/Cap| Memory-Usage | GPU-Util Compute M. |  
| | | | | | MIG M. |  
+-----+  
| 0 Tesla V100-SXM2... On | 00000000:18:00.0 Off | 0 |  
| N/A 37C P0 68W / 300W | 0MiB / 32510MiB | 0% Default |  
| | | | | | N/A |  
+-----+  
  
+-----+  
| Processes:  
| GPU GI CI PID Type Process name GPU Memory |  
| ID ID | Usage |  
+-----+  
| No running processes found | | |  
+-----+
```

```
[10]: # if you see: /bin/bash: nvidia-smi: command not found  
# the system is not GPU
```

# Notebook Examples

<https://github.com/sdsc-hpc-training-org/notebook-examples>

- Collection of tested, working notebooks tested on Expanse and other SDSC HPC systems
- Includes range of materials from “hello world” to Spark ML notebooks.
- Note: collection changes often, based on testing and contributions

The screenshot shows the GitHub repository page for 'sdsc-hpc-training-org / notebook-examples'. The repository is public, as indicated by the 'Public' badge. The 'Code' tab is selected. The master branch is active, showing 3 branches and 0 tags. The commit history lists 60 commits, all made by 'marypthomas' and dated 'now'. The commits are: 'Update README.md', 'Boring\_Python\_Book', 'CUDA\_GPU\_NVIDIA', 'Data\_Analysis', 'Hello\_World', 'Matplotlib\_Intro', 'Notebook\_Dev\_Basics', 'NumPy\_Intro', 'hpc-notebooks', '.DS\_Store', '.gitignore', and 'README.md'. The 'hpc-notebooks' commit is noted as merging hpc notebooks material. The 'README.md' commit is noted as updating it.

| Commit                       | Message                      | Date           |
|------------------------------|------------------------------|----------------|
| marypthomas Update README.md | 32163d3 now                  | 60 commits     |
| Boring_Python_Book           | updating training material   | 7 months ago   |
| CUDA_GPU_NVIDIA              | updating training material   | 7 months ago   |
| Data_Analysis                | updating training material   | 7 months ago   |
| Hello_World                  | updating training material   | 7 months ago   |
| Matplotlib_Intro             | updating training material   | 7 months ago   |
| Notebook_Dev_Basics          | updating training material   | 7 months ago   |
| NumPy_Intro                  | updating training material   | 7 months ago   |
| hpc-notebooks                | merge hpc notebooks material | 16 minutes ago |
| .DS_Store                    | add hello world              | 10 months ago  |
| .gitignore                   | update material              | 2 years ago    |
| README.md                    | Update README.md             | now            |

# Resources

- Expanse :
  - Landing page: [expanse.sdsc.edu](http://expanse.sdsc.edu)
  - User Guide: [https://expanse.sdsc.edu/support/user\\_guides/expanse.html](https://expanse.sdsc.edu/support/user_guides/expanse.html)
- GitHub Repo for Satellite and Galileo:
  - <https://github.com/sdsc-hpc-training-org/satellite>
  - <https://github.com/mkandes/galileo>
- SDSC Training Resources
  - [https://www.sdsc.edu/education\\_and\\_training/training](https://www.sdsc.edu/education_and_training/training)
  - <https://github.com/sdsc-hpc-training-org/notebook-examples>
- XSEDE Training Resources
  - <https://www.xsede.org/for-users/training>
  - <https://cvw.cac.cornell.edu/expanse/>
- Problems? Contact [help@xsede.org](mailto:help@xsede.org)

# Thank You!

if you have problems, please contact [help@xsede.org](mailto:help@xsede.org)