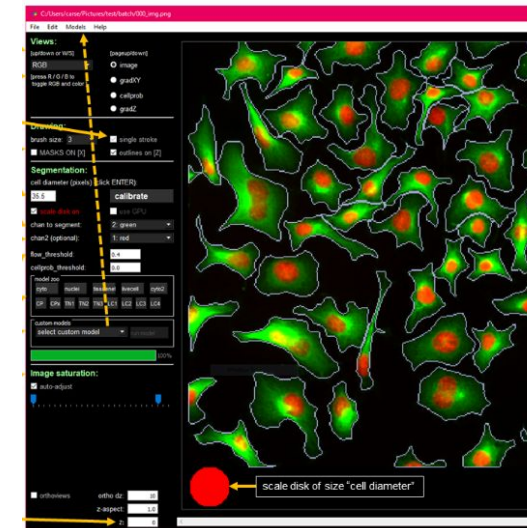
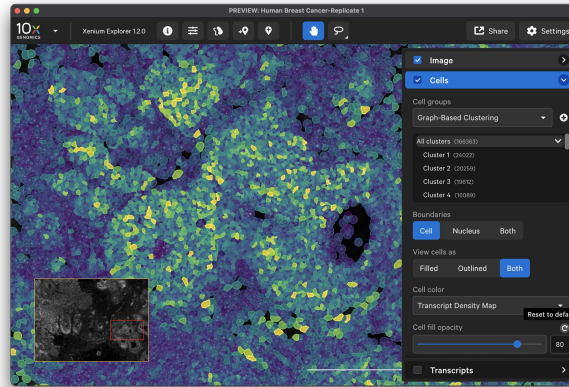
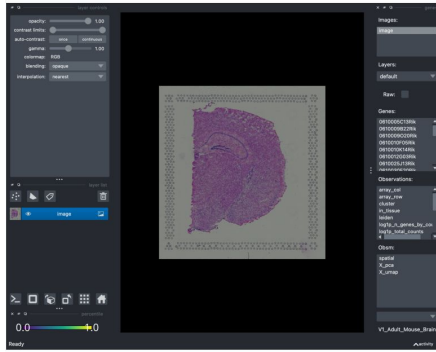


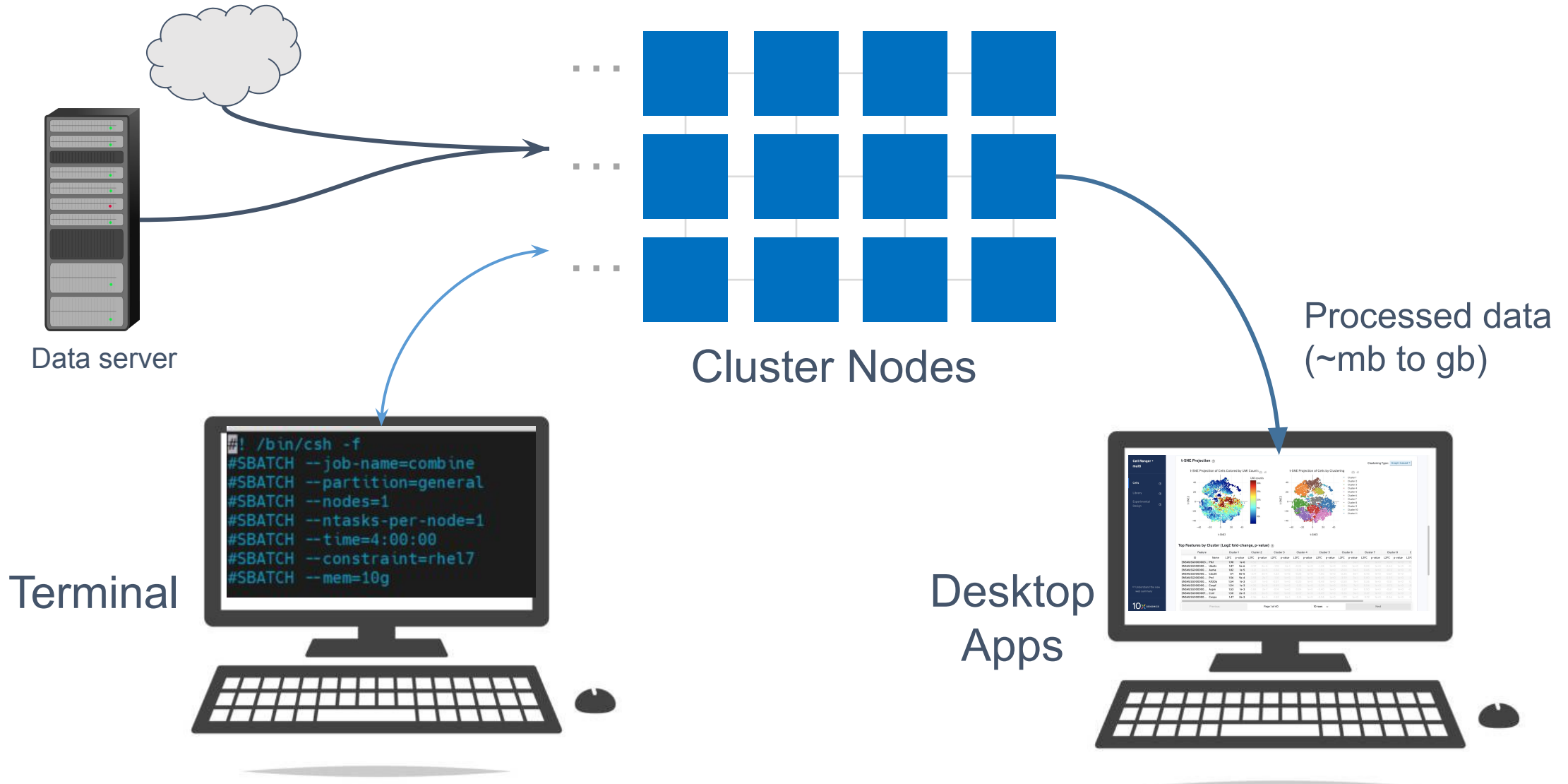
Interactive Computing on the HPC3 for Spatial-omics



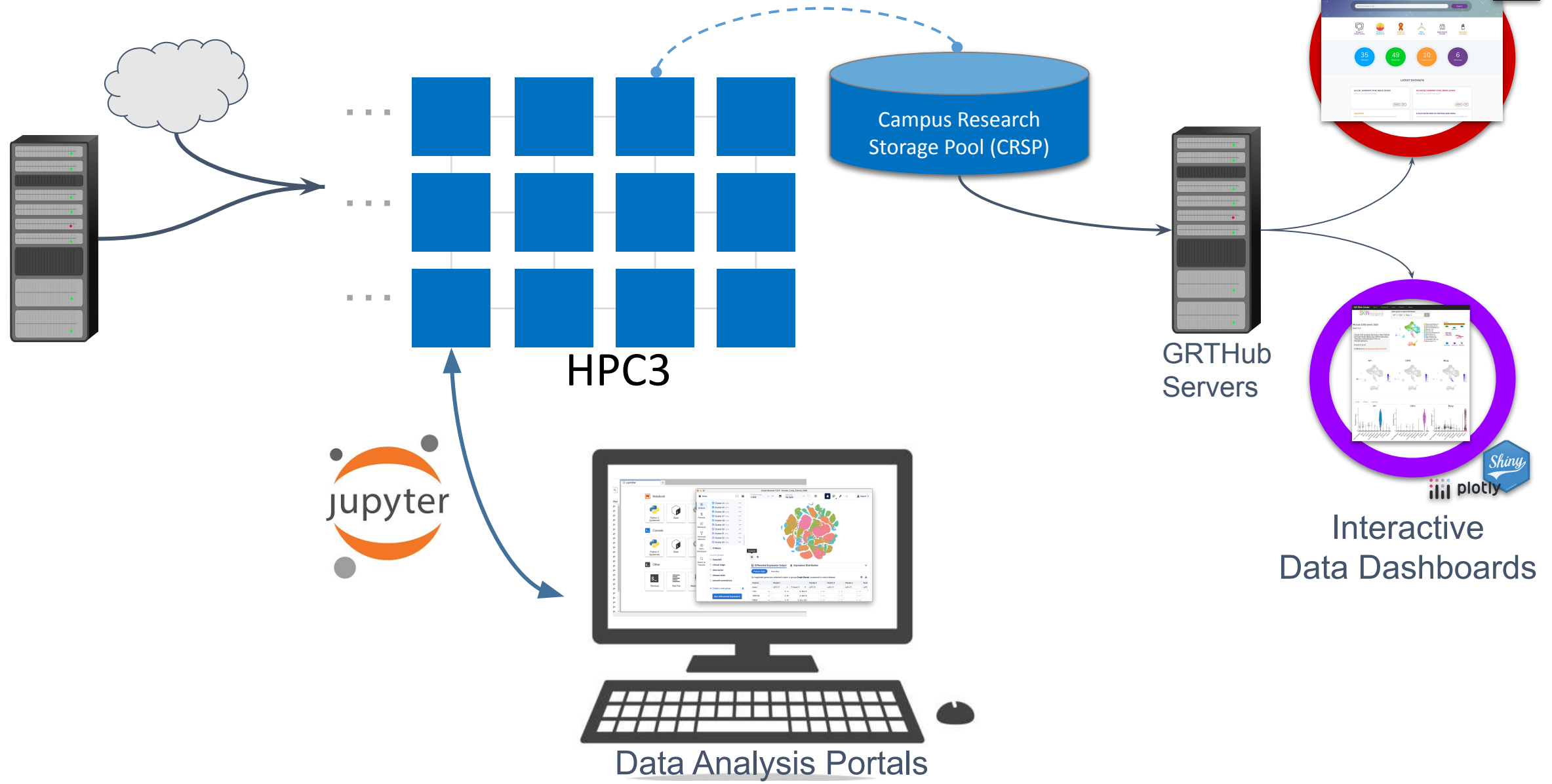
Ivan Chang

UCI GRTHub and UCI RCIC
iychang@uci.edu

Traditional HPC Workflow



Interactive Computing & Data Portals



Workshop Agenda

- **Overview of HPC3**

- Connecting to HPC3 & loading software modules
- Navigating file systems and managing storage
- Slurm account management and sample job submission

- Interactive Computing on HPC3

- SSH x11 forwarding/tunneling of software
- Containers on HPC3 and Jupyter Eco System
- Jupyter notebook and JupyterLab
- RStudios and RShiny Apps
- XPRA x11 to HTML5 via Jupyterhub

- Spatial-omics Apps (Live Demo)

- Fiji & CellProfiler
- Loupe Browser & Xenium Explorer
- CellPose & SquidPy

Overview of HPC3

The rest of the slide deck are for your reference only. We will not be going through these directly for this workshop.

Key UCI HPC3 Resource Specs



HPC3 - Rocky 8.8 Linux Distro

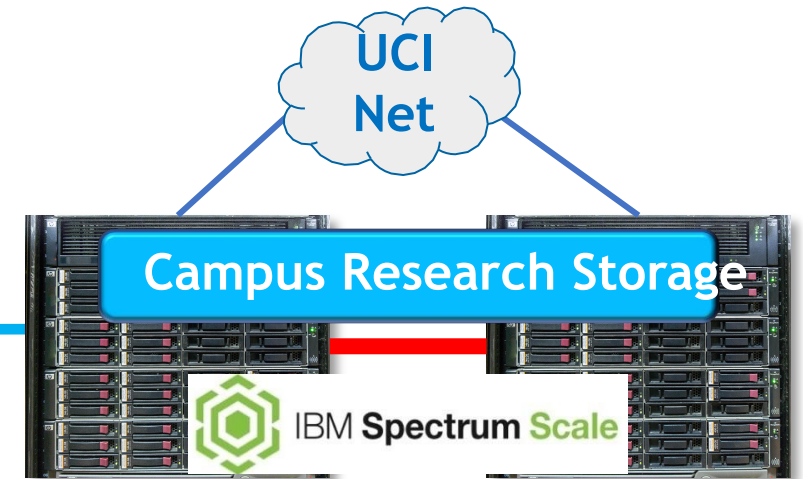
- ~9400 Cores/216 Hosts (compute nodes)
- 57,846 GB Aggregated memory
- 52 Nvidia V100 16Gb GPUs
- 56 Nvidia A30 24GB GPUs
- 8 Nvidia A100 80GB GPUs
- EDR (100Gbps) Infiniband
- 10GbE Ethernet
- Minimum
 - 4GB memory/core
 - AVX2 instruction set (Epyc/Intel CPUs)



Eight Parallel File Systems

DFS3, DFS4, DFS5, ...

- 7PB usable storage
- ~6GB/sec bandwidth/System
- Single Copy/No Snapshots



OIT
DC

ICS
DC

CRSP - Campus Research Storage Pool

- 1 PB usable storage
- Available anywhere on UCI Network
- Dual Copy of All Data
- Snapshots
- Highly available

<https://rcic.uci.edu/hpc3/specs.html>

Research Cyberinfrastructure Center

Philip Papadopoulos Ph.D. Office: 217 MSTB, x45343

Director, Research Cyberinfrastructure Center. Cluster and Scalable Filesystem design and implementation. High-performance networking.

Ivan Chang Ph.D. Office: Sprague Hall x46023

Project Specialist, Bioinformatics, Containerized computing, System administration.

Francisco Lopez Office: 238 MSTB, x48818

System administrator and Storage expertise. Unix system administration, secure computing, cloud computing.

Nick Santucci Office: 225 MSTB, x40084

Systems Administration and Software integration. Unix system administration, cluster computing, high performance storage.

Joulien Tatar, Ph.D. Office: 223 MSTB, x48402, Slack

NSF Campus Cyber-Infrastructure Engineer. High performance networking, data acquisition, physical sciences computing, archival storage, cloud computing.

Imam Toufique Office: 211 MSTB, x43239

CRSP Lead and HPC* System Developer. Unix system administration, cluster computing, high performance storage, high-availability.

Nadya Williams M.S. Office: 219 MSTB, x42829

Software Integration and provisioning Specialist. Unix system administration, Software Development, Direct Engagement.

HPC3 Quick Start Guides

(<https://rcic.uci.edu/about/intro.html>)

Getting an account

send email to hpc-support@uci.edu

Logging in

[Connecting to HPC3](#)

Submitting your first job

[SLURM tutorial](#)

Available software

[Environment modules tutorial](#)

Purchasing Hardware/Core Hours

[Beyond baseline allocation](#)

All about accounting

[Free and Accounted jobs](#)

Storage

[Home Area, Parallel File Systems, and CRSP](#)

Getting Help

[Ask for help or software install](#)

Basics of being a good citizen on a cluster

1. Cluster is a shared resource, **it is NOT your personal machine**
2. What you do affects all the other users, so think before you hit that *Enter* key
 - Do not run interactive jobs on login nodes
 - Do not transfer data on login nodes
3. Secured from mischief and disasters.
 - We restrict users' ability (permissions) to install and run unwanted software applications
 - It is your responsibility to act secure
 - **Be careful when bringing applications from unknown sources. DO NOT ask for sudo access**
4. For your jobs: use resources you need, don't ask for more
Study this Slurm guide <https://rcic.uci.edu/slurm/slurm.html>
5. Be mindful how you submit tickets
[Getting help!](#)

High-level View of what things cost

No Cost Allocations

Role	HPC3 Core Hours	GPU Hours	Home Area Storage	DFS Storage	CRSP Storage
Faculty	200K hours/year ¹	By Request ~2K hours/year ¹	50GB	1TB in Pub	1 TB
Student	1000 hours	---	50GB	1TB in Pub	---

Cloud-like Costs

	HPC3 Core Hours	GPU Hours	Home Area Storage	DFS Storage	CRSP Storage
Faculty	\$.01/core hour	\$0.32/GPU hour	Not expandable	\$100/TB/5 years	\$60/TB/year
AWS Equivalent	C5n.large \$.063	P3.2xlarge \$1.95	---	---	S3 ² Standard \$242/TB/year

¹ Exact amounts dependent on # requests/available hardware

² Comparison difficult - S3 has higher durability, CRSP has no networking fee.

HPC3 Policies for CPU and memory scheduling

Partition	Default memory/core	Max memory/core	Default / Max runtime	Cost	Jobs preemption
CPU Partitions					
standard	3 GB	6 GB	2 day / 14 day	1 / core-hr	No
free	3 GB	18 GB	1 day / 3 day	0	Yes
debug	3 GB	18 GB	15 min / 30 min	1 / core-hr	No
highmem	6 GB	10 GB	2 day / 14 day	1 / core-hr	No
hugemem	18 GB	18 GB	2 day / 14 day	1 / core-hr	No
maxmem	1.5 TB/node	1.5 TB/node	1 day / 7 day	40 / node-hr	No
GPU Partitions					
gpu3	3 GB	9 GB	2 day / 14 day	1 / core-hr, 32 / GPU-hr	No
free-gpu	3 GB	9 GB	1 day / 3 day	0	Yes
gpu-debug	3 GB	9 GB	15 min / 30 min	1 / core-hr, 32 / GPU-hr	No

RCIC Recommended Online Tutorials

The Missing Semester of Your CS Education Many topics as separate lectures, including Shell Tools and Scripting, Editors (Vim), Command-line Environment, Data Wrangling, Git, security and more.

The Software Carpentry teaches basic skills via workshops and lessons, here are direct links:

- **The UNIX Shell** The Unix shell fundamentals
- **Introduction to Python** Learn the basics of Python programming language.
- **Introduction to R** Learn the basics of R programming language.

Connecting to HPC3 and loading
software modules

Logging onto HPC3

Step 1 Connect to UCI campus VPN, see instructions [UCI campus VPN](#)

Step 2 Open your Terminal application and start ssh session. Alternatively, you could use the jupyterhub interface at <https://hpc3.rcic.uci.edu/biojhub4/> directly in your browser

Step 3 Either in ssh session or jupyterhub interface, you will need to use your regular UCI credentials (UCINetID and password) to connect to an HPC3 login node hpc3.rcic.uci.edu

SSH login Example

- Logging in is via ssh with your UCInetID
`ssh hpc3.rcic.uci.edu -l panteater`
or
`ssh panteater@hpc3.rcic.uci.edu`
- Use passphrase and ssh public key authentication **Do not use empty ssh passphrase!!!** <https://www.ssh.com/ssh/public-key-authentication>
- If you plan to run interactive graphics programs
`ssh -X -Y hpc3.rcic.uci.edu -l panteater`

But X11 used by Linux for graphics is high bandwidth and can be sensitive to network latency, some people prefer **x2go**
<https://wiki.x2go.org/doku.php>

Duo authentication

Password:
Duo two-factor login for <UCINETID>

Enter a passcode or select one of the following options:

1. Duo Push to XXX-XXX-XXXX

Passcode or option (1-1):

After a successful login you will see a screen similar to the following:

Last login: Thu Jul 15 15:25:59 2021 from 10.240.58.4

+	-----																+
			_____	/	_____	()	_____			()		_____					
			/	\	/	\						\					
			()		(										)		
			\	/	\	,									/		
						/											
+	-----																+

```
Distro: Rocky 8.8 Core
Virtual: NO
```

```
CPUs:      40
RAM:       191.9GB
BUILT:     2023-03-02 13:32
```

ACCEPTABLE USE: <https://rcic.uci.edu/documents/RCIC-Acceptable-Use-Policy.pdf>
login-i15 2001%

Loading software modules

- **Environment module** is a user interface to the Modules package which provides for the dynamic modification of the user's environment via modulefiles.
- Each **modulefile** contains all the info needed to configure the shell to use a specific application.
- Command **module load** interprets the modulefiles and
 - Sets aliases
 - Sets environment variables
 - Loads depended modules
- Command **module avail** lists all installed software and their versions

General info for Linux <https://modules.readthedocs.io/en/latest/>
Read User guide for HPC3 <https://rcic.uci.edu/software/modules.html>

Environment modules update your environment

Case 1: usage of multiple versions of software

login-i16 which R

/usr/bin/which: no R in (/usr/local/bin:/usr/bin:/usr/sbin:/data/homezvol0/npw/bin)

login-i16 module avail R

----- /opt/rcic/Modules/modulefiles/LANGUAGES -----

R/3.6.2 R/4.0.2

login-i16 module load R/4.0.2

login-i16 which R

/opt/apps/R/4.0.2/bin/R

login-i16 module list

Currently Loaded Modulefiles:

1) OpenBLAS/0.3.6 2) java/1.8.0 3) icu/65.1

login-i16 module unload R/4.0.2

login-i16 module list

No Modulefiles Currently Loaded.

login-i16 module load R/3.6.2

login-i16 which R

/opt/apps/R/3.6.2/bin/R

Case 2: load/unload different software modules

login-i16 module load gcc/8.4.0

login-i16 module list

Currently Loaded Modulefiles:

1) gcc/8.4.0

login-i16 module load hdf5/1.10.5/gcc.8.4.0

login-i16 module list

Currently Loaded Modulefiles:

1) gcc/8.4.0 2) java/1.8.0 3) hdf5/1.10.5/gcc.8.4.0

login-i16 module unload hdf5/1.10.5/gcc.8.4.0

login-i16 module list

Currently Loaded Modulefiles:

1) gcc/8.4.0

Always unload module in reverse order: FILO!

Environment module commands summary

search	\$ module avail	shows all installed software environment modules
	\$ module avail R	show R modules
	\$ module keyword salmon	check all modules for a keyword
	salmon/1.1.0 : Name salmon salmon/1.1.0 :	
info	\$ module display R	shows environment modification + description
	\$ module help R	show module specific help (description)
use	\$ module load R	loads R at whatever latest version not ideal
	\$ module load R/4.0.2	loads R at specified version preferred method
	\$ module list	lists currently loaded modules
	\$ module unload R/4.0.2	unloads specified module (in reverse order if many)
	\$ module purge	removes all loaded modules

Navigating file systems and managing storage

The filesystem storage is generally in 3 areas. Please see the links below for detailed information about each filesystem.

HOME **The HOME area** has a 50GB quota for each user. In addition, there is a space for snapshots. Total for home and snapshots is 100GB. Each user HOME is in /data/homezvolX/<account>

DFS **The BeeGFS Parallel storage File System (DFS)** access remains the same. All users have /pub/<account> area. Depending on a lab affiliation, users may have space in /dfs3, /dfs4, /dfs5, /dfs6, /dfs7, /dfs8, /dfs9 (9/19/2023)

CRSP **The Campus Research Storage Pool (CRSP)** is available in /share/crsp. Depending on a lab affiliation, users may have space in /share/crsp/lab/<labname./<account>

Check \$HOME quota

To see your current quota usage do:

```
[user@login-x:~]$ df -h ~
```

Filesystem	Size	Used	Avail	Use%	Mounted on
10.240.58.6:/homezvol0/panteater	50G	3.5G	47G	7%	/data/homezvol0/panteater

The ~ stands for your \$HOME. The output above shows that user panteater used 3.5Gb of its 50Gb allocation.

If you want to see the usage by files and directories in \$HOME

```
[user@login-x:~]$ cd
[user@login-x:~]$ ls
bin          examples    local      perl5
biohub3_dir  info       mat.yaml   R
classify-image.py keras-nn.py modulefiles sbank-out
```

change to your \$HOME directory

list contents of \$HOME

```
[user@login-x:~]$ du -s -h *
7.0Mbin
166Mbiohub3 dir
8.5Kclassify-image.py
647Kexamples
91K info
4.5Kkeras-nn.py
126Mlocal
4.5Kmat.yaml
60K modulefiles
512 perl5
1.2GR
25K sbank-out
```

find disk usage for each file and directory in \$HOME. The output shows disk usage in kilobytes (K), megabytes (M) or gigabytes (G). For directories, all contents inside are included. For example, a directory **R** uses 1.2Gb of disk space.

To see the quotas for user panteater on DFS pool /dfs6

```
[user@login-x:~]$ dfsquotas panteater dfs6
```

==== [Group Quotas on dfs6]

Quota information for storage pool Default (ID: 1):

user/group		size		chunk files	
name	id	used	hard	used	hard
-----	-----	-----	-----	-----	-----
panteater_lab	012345	26.25 TiB	50.00 TiB	1310459	18500000
alpha users	158537	0 Byte	1 Byte	0	1
panteater	000865	755.59 GiB	1024.00 GiB	258856	unlimited

Data Transfer to HPC3

Often users need to bring data from other servers and laptops. To transfer data one needs to use `scp` (secure copy) or `rsync` (file copying tool). Please see [detailed data transfer examples](#).

Alternatively, one can use graphical tools (Filezilla, MountainDuck, or WinSCP) to transfer files between a local laptop and the cluster. Follow each program instructions how to do this.

In all of the transfer application you will need to use hpc3.rcic.uci.edu to indicate a remote server (where you want to transfer your files) and use your UCNetID credentials for your user name and password.

Slurm account management and sample job submission

HPC3 SLURM

Slurm is an open-source workload manager for Linux clusters and provides:

1. access to resources (computer nodes) to users so they can run their applications.
2. framework to start, execute, and monitor work on a set of allocated nodes.
3. management of a queue for pending work.

Helpful UCI HPC3 specific slurm guide:

<https://rcic.uci.edu/hpc3/slurm.html>

Simple code of conduct for running applications on HPC3

1. **All jobs, batch or interactive must be submitted to the scheduler**
2. **Do not run computational jobs on login nodes** this adversely affects many users. Login nodes are meant for light editing or compilation and for submitting jobs. Any job that runs for more than an hour or is using significant memory and CPU within an hour should be submitted to Slurm either as interactive or batch job.
3. **Ssh access to the compute nodes is turned off** to prevent users from starting jobs bypassing Slurm. See [attaching to running job](#) below.
4. **Do not run Slurm jobs in your \$HOME.**
5. **Make sure you stay within your disk quota.** File system limits are generally the first ones that will negatively affect your job. See [storage guides](#)

Cluster Partitions

HPC3 has different kinds of hardware, memory footprints, and nodes with GPUs. All nodes (servers) all are separated into groups according to their resources. Slurm uses the term partition to signify a queue of resources. We have a few separate partitions, most users will need to use *standard* and *free* partitions:

- **standard partition** is for jobs that should not be interrupted. Usage is charged against the user's Slurm bank account. Each user gets FREE one time allocation of 1000 core hours to run jobs here. **Users are NOT CHARGED ANY \$**. If all allocation is used, users can run jobs only if they are associated with labs that have core hours in their lab banks. Usually, lab bank is a PI lab account.
- **free partition** is for jobs that can be preempted (killed) by standard jobs. Users can run jobs in this partition even if they have only 1 core-hour left. There are no charges for this partition.

HPC3 Policies for CPU and memory scheduling

Partition	Default memory/core	Max memory/core	Default / Max runtime	Cost	Jobs preemption
CPU Partitions					
standard	3GB	6GB	2day / 14day	1 / core-hr	No
free	3GB	18GB	1day / 3day	0	Yes
debug	3GB	18GB	15min / 30min	1/core-hr	No
highmem	6GB	10GB	2day / 14day	1/core-hr	No
hugemem	18GB	18GB	2day / 14day	1/core-hr	No
GPU Partitions					
gpu	3GB	9GB	2day / 14day	1/core-hr, 32/GPU-hr	No
free-gpu	3GB	9GB	1day / 3day	0	Yes
gpu-debug	3GB	9GB	15min / 30min	1/core-hr, 32/GPU-hr	No

Checking your allocations

`sbank` is short for "Slurm Bank". `Sbank` is used to display HPC3 user account information. In order to run jobs on HPC3, a user must have available CPU hours. To check how many CPU hours are available in your personal account, run the command with your account name:

```
[user@login-x:~]$ sbank balance statement -a panteater
```

User	Usage	Account	Usage	Account	Limit	Available (CPU hrs)
panteater*	58	PANTEATER	58		1,000	942

To check how many CPU hours are available in all accounts that you have access to and how much you used:

```
[user@login-x:~]$ sbank balance statement -u panteater
```

User	Usage	Account	Usage	Account	Limit	Available (CPU hrs)
panteater*	58	PANTEATER	58		1,000	942
panteater*	6,898	PI_LAB	6,898		100,000	93,102

Slurm Batch Job

simplejob.sub

```
#!/bin/bash

#SBATCH --job-name=test      ## Name of the job.
#SBATCH -A panteater_lab    ## account to charge
#SBATCH -p standard         ## partition/queue name
#SBATCH --nodes=1           ## (-N) number of nodes to use
#SBATCH --ntasks=1          ## (-n) number of tasks to launch
#SBATCH --cpus-per-task=1    ## number of cores the job needs
#SBATCH --error=slurm-%J.err ## error log file

# Run command hostname and save output to the file out.txt
srun hostname > out.txt
```

To submit the job do:

```
[user@login-x:~]$ sbatch simplejob.sub
Submitted batch job 362
```

Please look through <https://rcic.uci.edu/hpc3/examples.html> for different job examples

Job status

To check the status of your job in the queue:

```
[user@login-x:~]$ squeue -u panteater
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST (REASON)
362	standard	test	panteater	R	0:03	1	hpc3-17-11

To get detailed info about the job:

```
[user@login-x:~]$ scontrol show job 362
```

The output will contain a list of key=value pairs that provide job information.

To cancel a specific job:

```
[user@login-x:~]$ scancel <jobid>
```

Job history

We have a cluster-specific tool to print a ledger of jobs based on specified arguments. Default is to print jobs of the current user for the last 30 days:

```
[user@login-x:~]$ /pub/hpc3/zotledger -u panteater
```

DATE	USER	ACCOUNT	PARTITION	JOBID	JOBNAME	ARRAYLEN	CPUS	WALLHOURS	SUs
2021-07-21	panteater	panteater	standard	1740043	srun	-	1	0.00	0.00
2021-07-21	panteater	panteater	standard	1740054	bash	-	1	0.00	0.00
2021-08-03	panteater	lab021	standard	1406123	srun	-	1	0.05	0.05
2021-08-03	panteater	lab021	standard	1406130	srun	-	4	0.01	0.02
2021-08-03	panteater	lab021	standard	1406131	srun	-	4	0.01	0.02
TOTALS	-	-	-	-	-	-	-	0.07	0.09

To find all available arguments use:

```
[user@login-x:~]$ /pub/hpc3/zotledger -h
```

Job info

`sacct` can be used to see accounting data for all jobs and job steps. An example below shows how to use job ID for the command:

```
[user@login-x:~]$ sacct -j 43223
```

JobID	JobName	Partition	Account	AllocCPUS	State	ExitCode
36811_374	array	standard	panteater_l+	1	COMPLETED	0:0

The above command uses a default output format. A more useful example will set a specific format for `sacct` that provides extra information:

```
[user@login-x:~]$ export
```

```
SACCT_FORMAT="JobID%20,JobName,Partition,Elapsed,State,MaxRSS,AllocTRES%32"
```

```
[user@login-x:~]$ sacct -j 600
```

JobID	JobName	Partition	Elapsed	State	MaxRSS	AllocTRES
600	all1	free-gpu	03:14:42	COMPLETED		billing=2,cpu=2,gres/gpu=1,mem=+
600.batch	batch		03:14:42	COMPLETED	553856K	cpu=2,mem=6000M,node=1
600.extern	extern		03:14:42	COMPLETED	0	billing=2,cpu=2,gres/gpu=1,mem=+

Interactive Computing

Slurm interactive jobs

To request an interactive job, use the `srun` command. Suppose you are enabled to charge to the `panteater_lab` account then, to start an interactive session you can use one of 3 methods :

```
[user@login-x:~]$ srun --pty /bin/bash -i (1)
[user@login-x:~]$ srun --pty -p free /bin/bash -i (2)
[user@login-x:~]$ srun -A panteater_lab --pty /bin/bash -i (3)
```

After you are done use `logout` command to logout:

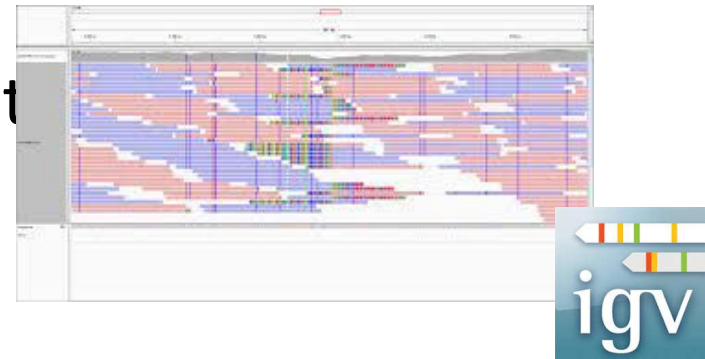
```
[user@hpc3-118-04:~]$ logout
```

- 1) you will be put on an available node in standard partition using your default Slurm bank account
- 2) you will be put on an available node in free partition using your default Slurm bank account
- 3) you will be put on an available node in standard partition using `panteater_lab` account

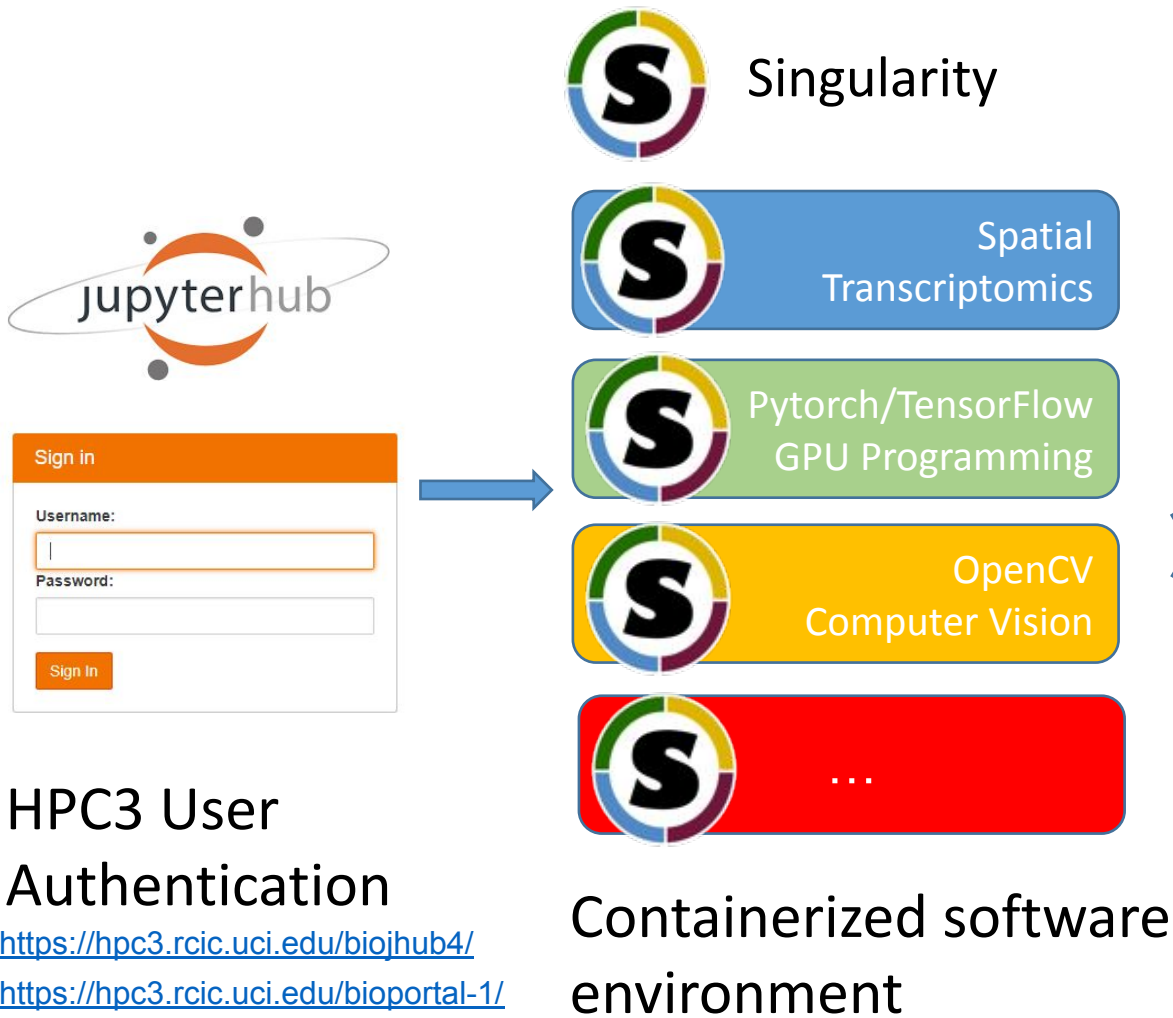
See 2.4. Interactive Job of <https://rcic.uci.edu/hpc3/slurm.html> for more options

Running Native X11 GUI apps

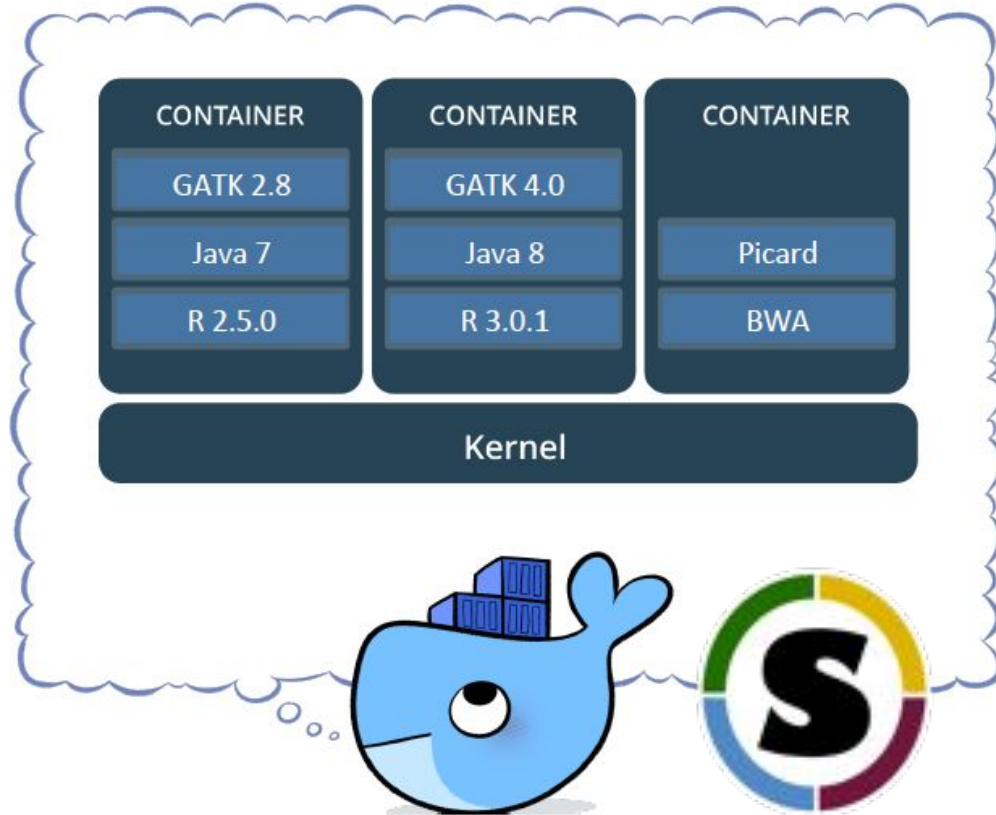
- Enable X11 forwarding in your SSH client
- For example: `ssh hpc3.rcic.uci.edu -X`
- Request an interactive session:
 - `srun -A account_lab -p standard --ntasks=1 --cpus-per-task=4 --nodes=1 --x11 --pty /bin/bash -i`
- Load application module (e.g. IGV Viewer)
 - `module load igv`
 - `igv`
- X11 Application forwarded to your computer



Jupyter Ecosystem - Running Containers Interactively on HPC3

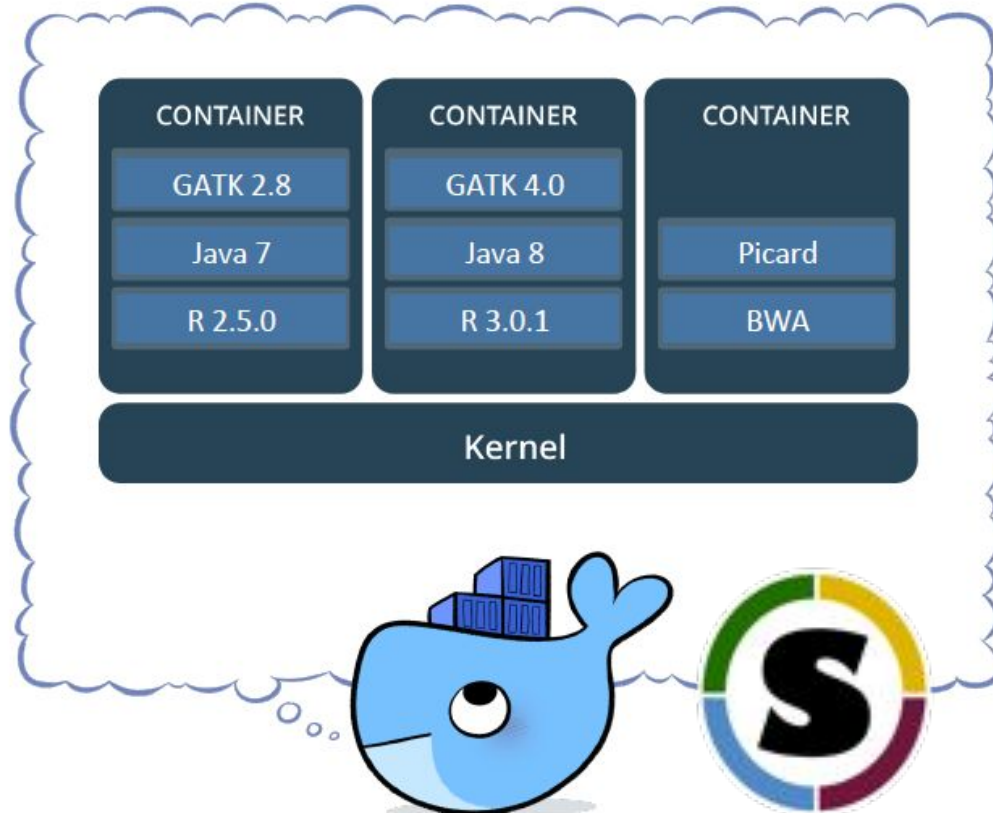


Container Technology Overview



- Containers are isolated, but share OS and bins/libraries
- Provide highly customized software environment apart from the host system

Container Benefits



Modified from <https://www.docker.com/what-container>

- **Portability:** containers can be published and shared via cloud-based container hubs (<https://hub.docker.com/>, <https://dockstore.org/>, <https://www.singularity-hub.org/>) or transferred directly as image files
- **Versioning:** container build files can be stored in git/github repositories
- **Reproducibility:** published containers are immutable, and provides a snapshot of the computing environment used to run analysis

Using your favorite browser go to: <https://hpc3.rcic.uci.edu/biojhub4/hub/login> You will see the following screen where you will Use your usual HPC3 credentials to sign in:

Sign in

Username:

Password:

Sign In

After authentication you will see a screen with server options as in the figure below:

Server Options

Select Partition/Reservation to Use

Standard

Select Account to Charge

iychang

Specify number of CPU cores (max 16)

2

memory per CPU core (max 6Gb per core)

4

Select a Containerized Notebook Image

[Rocky8] R4.1.2, Python 3.10.2, Rstudio, Rshiny, Seurat, Slurm Support

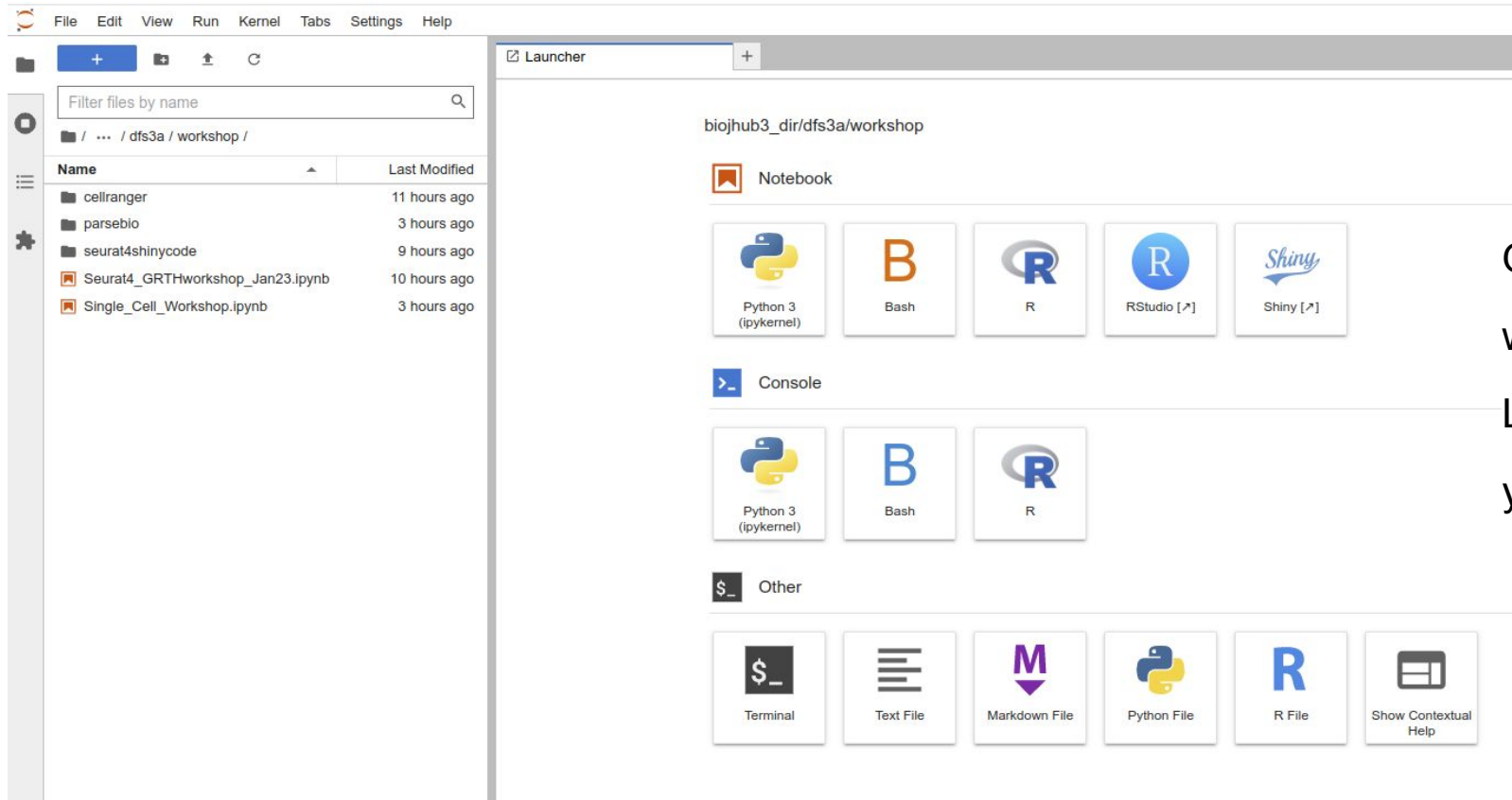
Resume last session if available



Start

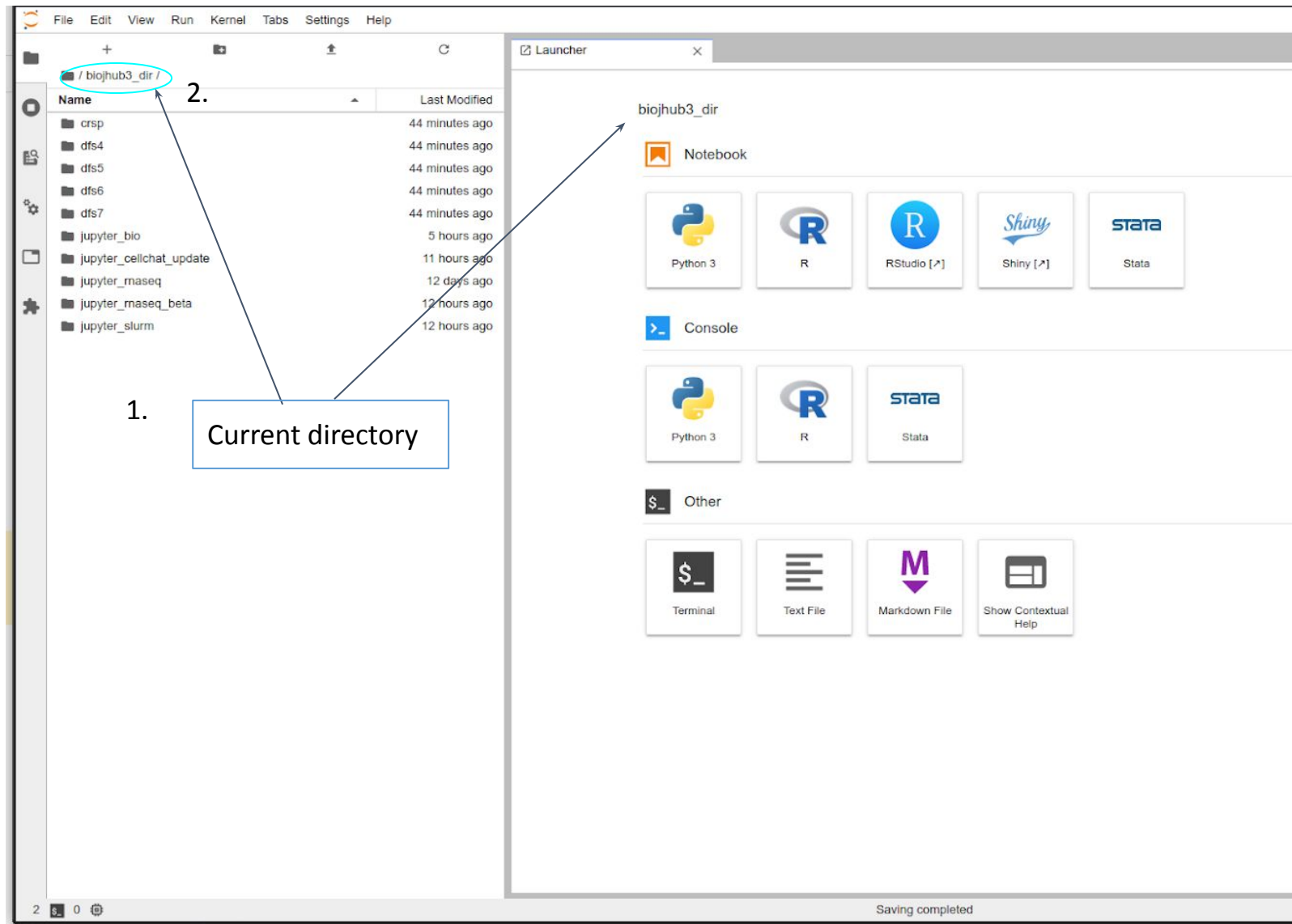
For this workshop, modify the *Select Account to Charge* to be one of your Slurm accounts, change number of CPUs to **6** and the memory per CPU core to **6**, select the “[Rocky8][Beta]R4.3.3/Python 3.10.2 **ML Workshop**)” container and press **Start**.

Main Jupyter Interface



Once the notebook is done spawning, you will get a file browser on the left, and a Launcher screen with a number GUI apps you can use on the right.

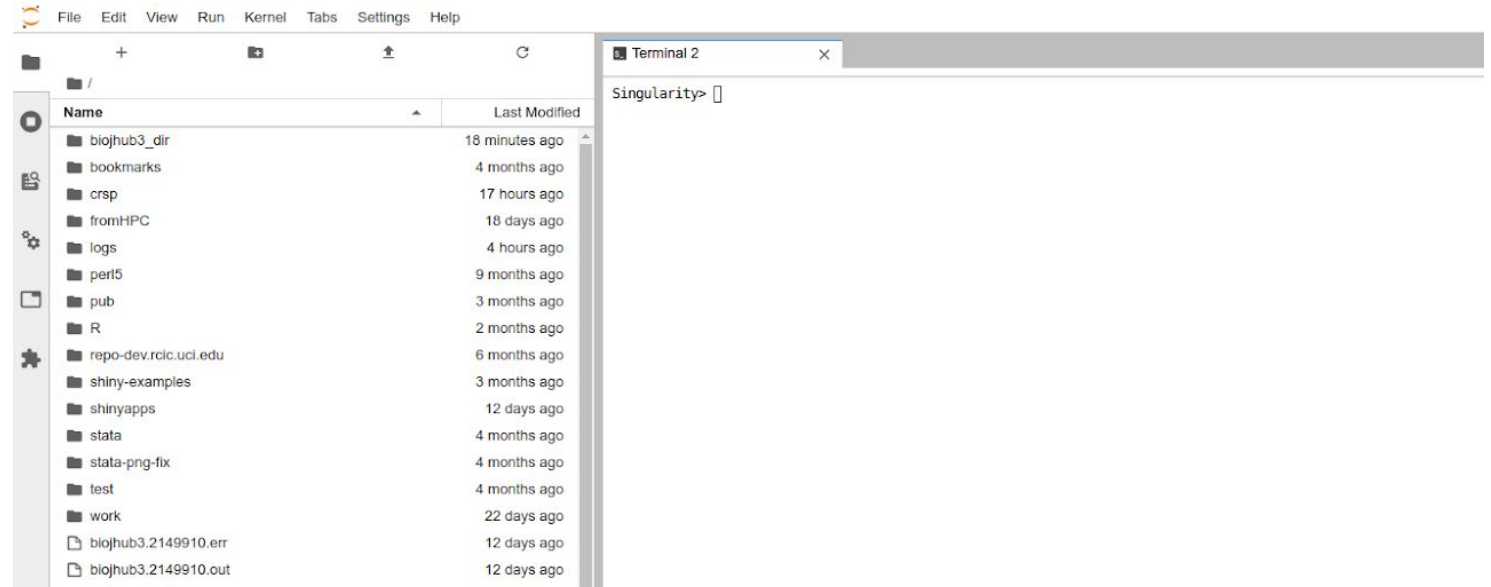
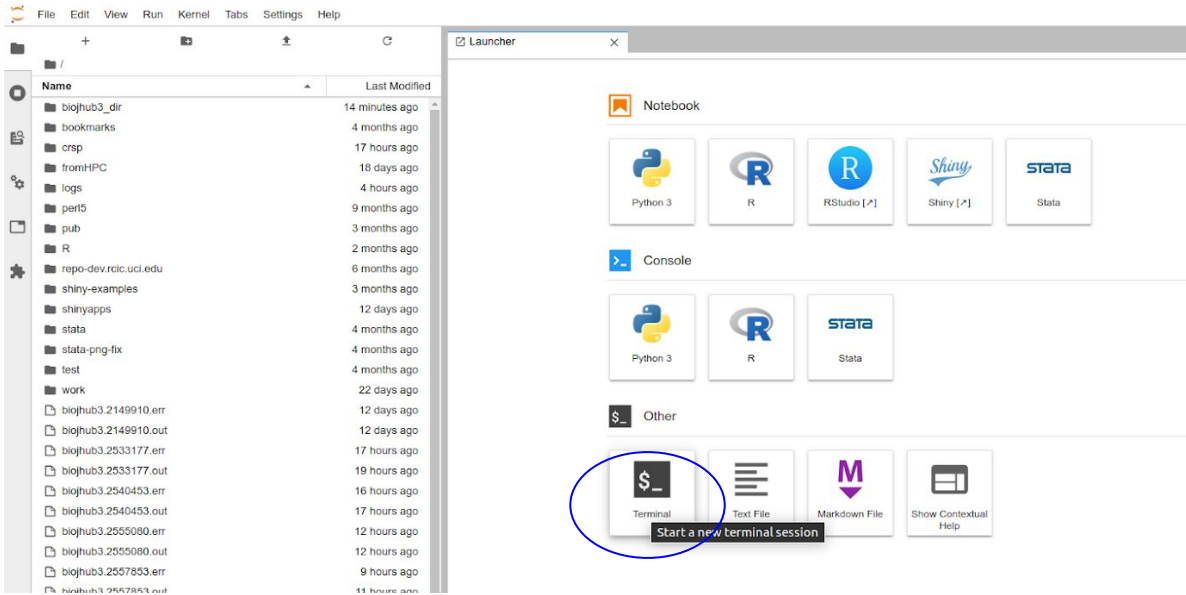
Navigating File Systems in Jupyter



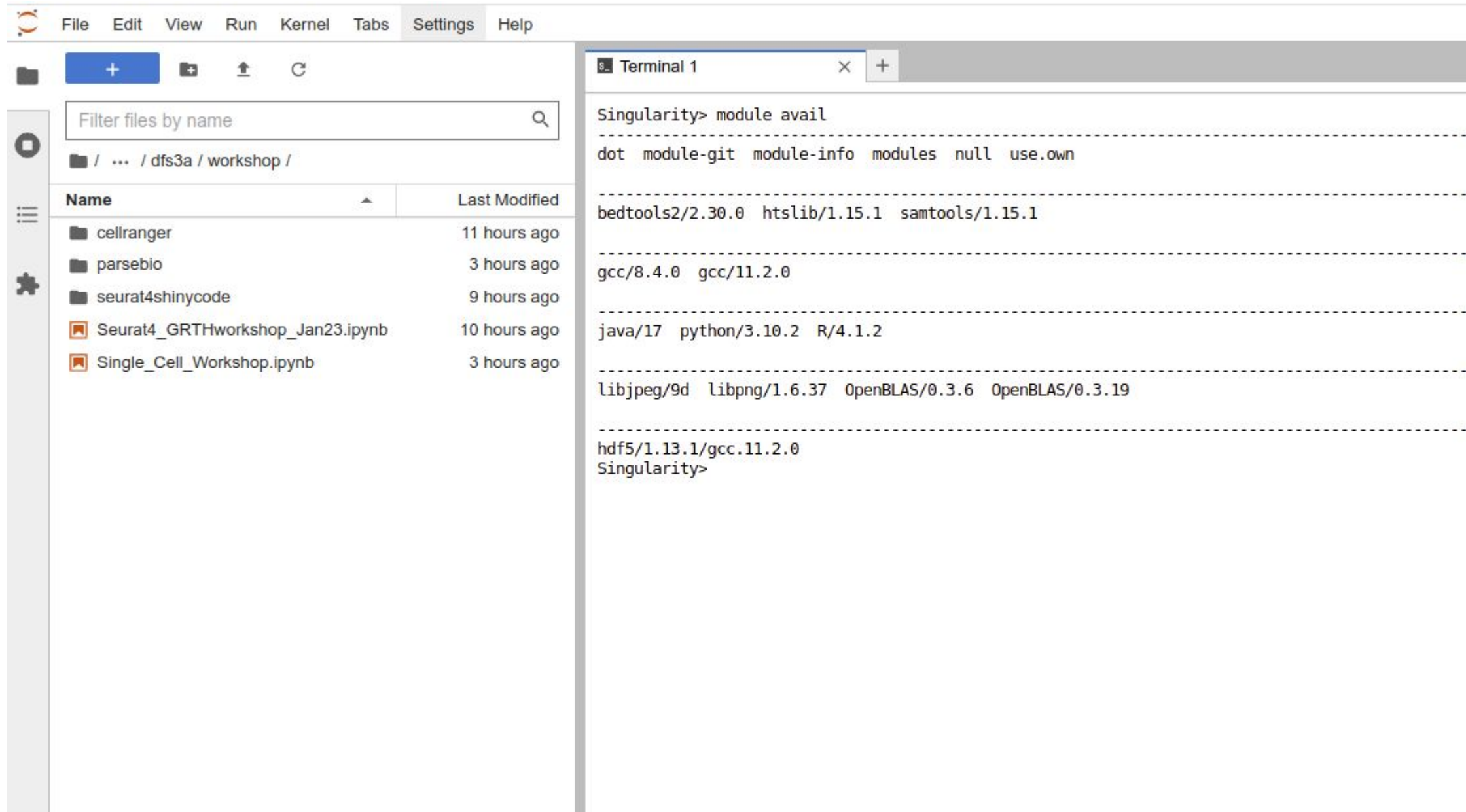
1. File browser GUI
2. Icon quick link
3. File menu -> *Open from Path* will allow you to directly enter the path.

*Due to preventative measures to avoid accidental deposit of files in sensitive areas of HPC3 file system, the root directory of the file browser is set to a fictitious home directory. To quickly traverse to other file systems, several shortcuts are automatically created, serving as symbolic links to the various file systems on HPC3.

Starting a New App (Terminal)



Running Container Modules from Terminal



The screenshot shows a JupyterLab environment. On the left is a file browser with a search bar and a list of files and folders. On the right is a terminal window titled 'Terminal 1' showing the output of the 'module avail' command.

File Browser:

- Search: Filter files by name
- Path: / ... / dfs3a / workshop /
- Table:

Name	Last Modified
cellranger	11 hours ago
parsebio	3 hours ago
seurat4shinycode	9 hours ago
Seurat4_GRTHworkshop_Jan23.ipynb	10 hours ago
Single_Cell_Workshop.ipynb	3 hours ago

Terminal 1:

```
Singularity> module avail
-----
dot module-git module-info modules null use.own
-----
bedtools2/2.30.0 htlib/1.15.1 samtools/1.15.1
-----
gcc/8.4.0 gcc/11.2.0
-----
java/17 python/3.10.2 R/4.1.2
-----
libjpeg/9d libpng/1.6.37 OpenBLAS/0.3.6 OpenBLAS/0.3.19
-----
hdf5/1.13.1/gcc.11.2.0
Singularity>
```

SSH to HPC3 login node for full software stack

File Edit View Run Kernel Tabs Settings Help
+

Filter files by name

/ ... / dfs3a / workshop /

Name	Last Modified
cellranger	11 hours ago
parsebio	3 hours ago
seurat4shinycode	9 hours ago
Seurat4_GRTHworkshop_Jan23.ipynb	10 hours ago
Single_Cell_Workshop.ipynb	3 hours ago

iychang@login-i16:~

Singularity> ssh hpc3.rcic.uci.edu

```

+-----+
|               |
|   login-i16   |
|               |
+-----+

Distro: Rocky 8.6 Green Obsidian
Virtual: NO

CPUs:      40
RAM:       191.8GB
BUILT:     2022-08-30 14:54

ACCEPTABLE USE: https://rcic.uci.edu/documents/RCIC-Acceptable-Use-Policy.pdf
Last login: Tue Jan 31 00:29:36 2023 from 10.240.2.34
[iychang@login-i16:~]$ module avail

-----

dot module-git module-info modules null use.own

-----

mpi/openmpi-x86_64 pmi/pmix-x86_64

-----

pytorch/1.5.1 pytorch/1.11.0 tensorflow/2.0.0 tensorflow/2.8.0 tensorRT/8.4.2.4

-----

bamtools/2.5.1 bedtools2/2.30.0 bowtie2/2.4.5 cutadapt/4.0 fastp/0.20.0 gatk/4.2.6.1
bamtools/2.5.2 bowtie/1.2.3 bustools/0.40.0 deeptools/3.5.0 fastp/0.23.2 hmmer/3.3
bcftools/1.10.2 bowtie2-python2/2.4.1 bwa/0.7.8 deeptools/3.5.1 fastqc/0.11.9 htlib/1.10.2
bcftools/1.15.1 bowtie2/2.4.1 bwa/0.7.17 edirect/2020 freebayes/1.3.2 htlib/1.15.1
bedtools2/2.29.2 bowtie2/2.4.4 cutadapt/2.10 edirect/2022 gatk/4.1.9.0 isal/2.30.0

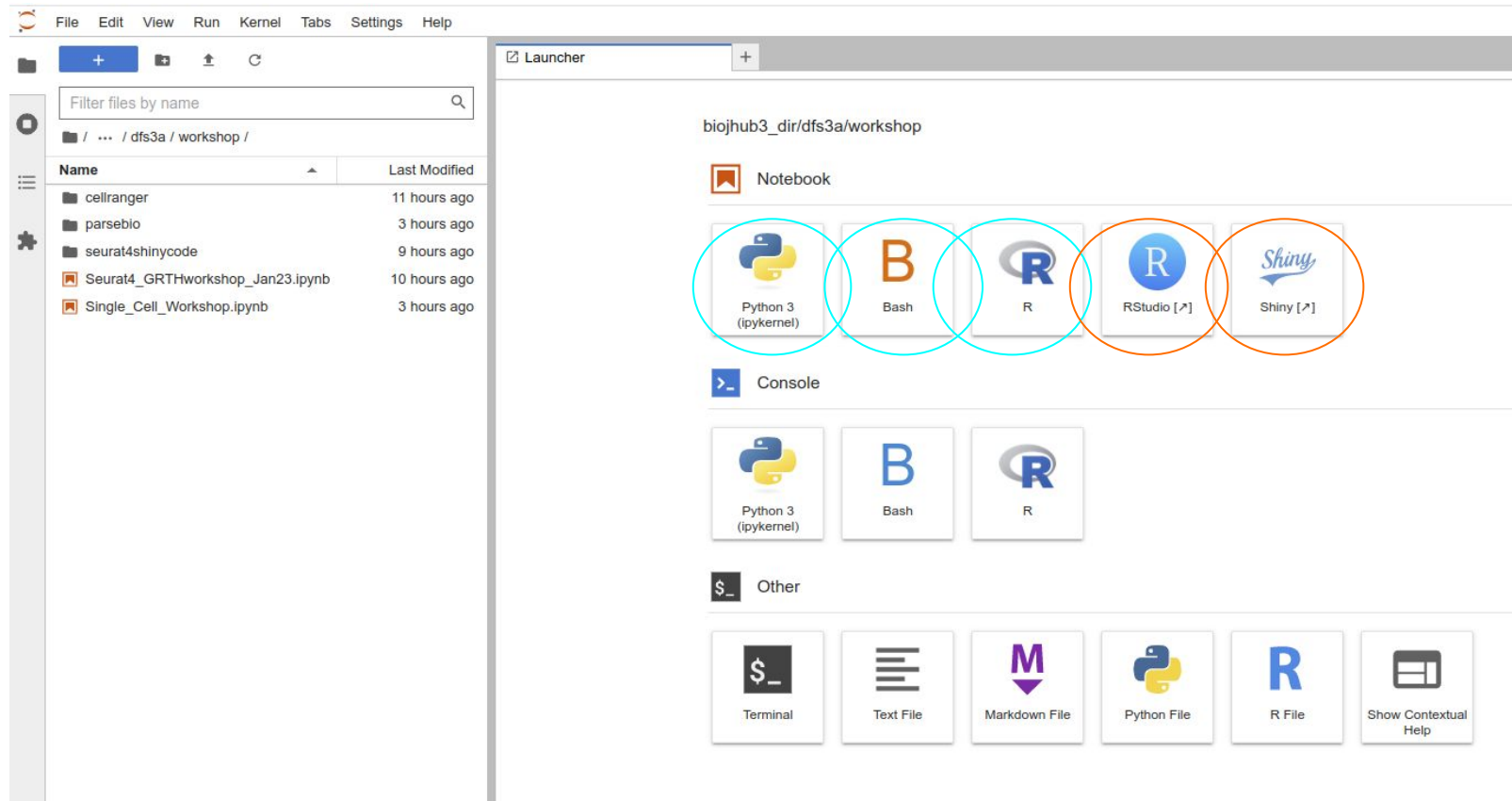
-----

amber/19.11/gcc.8.4.0 gromacs/2020.4/gcc.8.4.0-cuda.10.1.243.openmpi.4.0.3
amber/21.12/gcc.11.2.0 gromacs/2020.4/gcc.8.4.0-openmpi.4.0.3
gromacs/2020.4/gcc.8.4.0 gromacs/2021.2/gcc.8.4.0
gromacs/2020.4/gcc.8.4.0-cuda.10.1.243 gromacs/2021.2/gcc.8.4.0-cuda.10.1.243
gromacs/2020.4/gcc.8.4.0-cuda.10.1.243.openmpi.4.0.3 gromacs/2021.2/gcc.8.4.0-cuda.10.1.243.openmpi.4.0.3

-----

```

Jupyter Notebook & other web apps

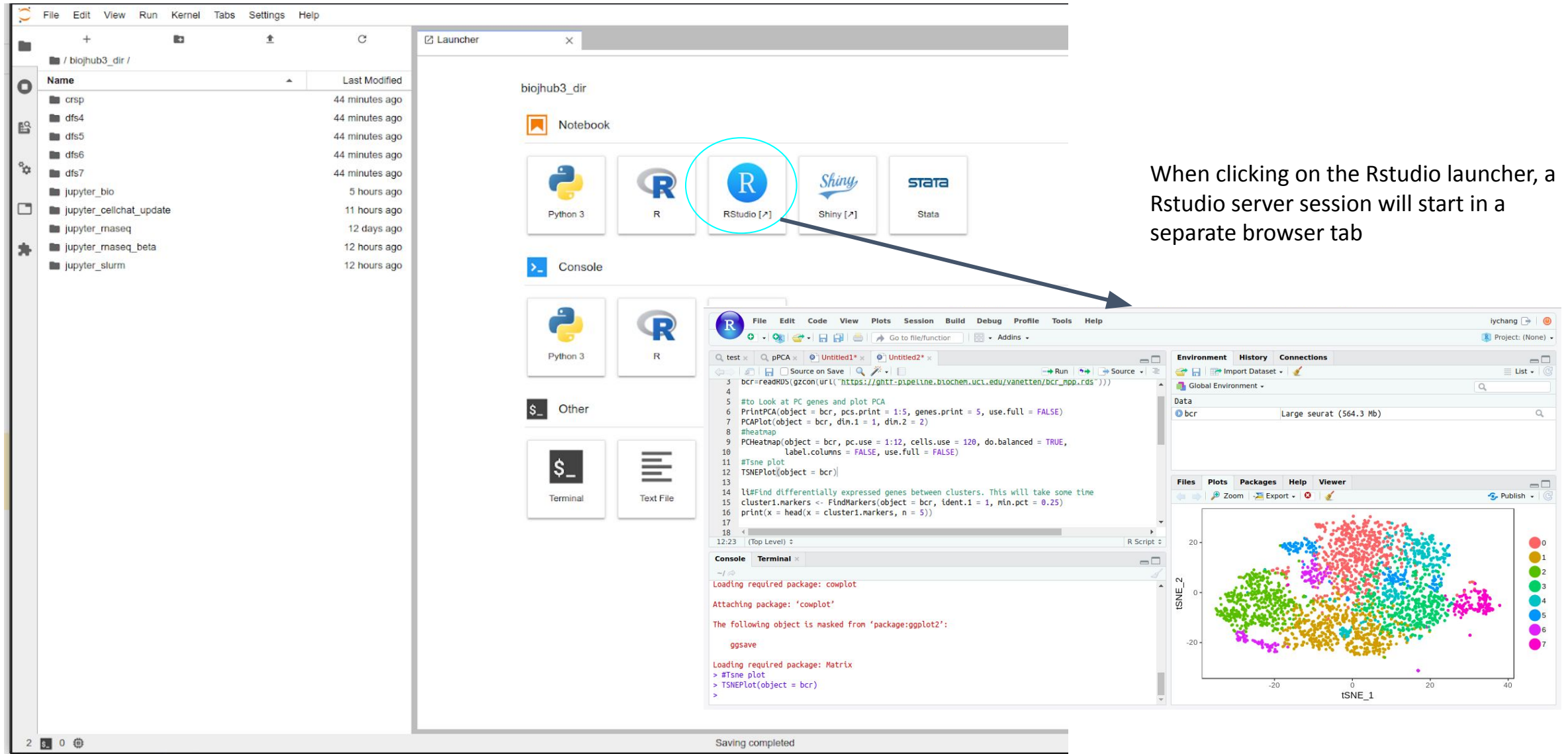


In this container, the user can open either a python, R, or Bash Jupyter computational notebook that will connect to the respective backend kernel and programming language environment.

Support for Matlab, Mathematica, and Julia are also available.

In addition, alternative web based development platform such as Rstudio and R Shiny are also available. When you click on these icons, a new browser tab will appear instead of a Jupyter notebook tab.

RStudio



When clicking on the Rstudio launcher, a Rstudio server session will start in a separate browser tab

The screenshot displays the RStudio Launcher interface. The top section shows the file explorer for the `biohub3_dir` project. The middle section contains the Launcher, which lists various applications: Python 3, R, RStudio (circled in blue), Shiny, and Stata. Below the Launcher is the Console, which shows the R script being executed. The script includes commands for loading data from a URL, performing PCA, and generating a t-SNE plot. The bottom right panel shows the Environment pane with the `bcr` object (Large seurat (564.3 Mb)) and the Plots pane displaying a t-SNE plot of the data.

```
3 bcr=readRDS(gzcon(url("https://ghnti-pipeline.biochem.uct.edu/vanetten/bcr_mpp.rds")))
4
5 #to Look at PC genes and plot PCA
6 PrintPCA(object = bcr, pcs.print = 1:5, genes.print = 5, use.full = FALSE)
7 PCAPlot(object = bcr, dim.1 = 1, dim.2 = 2)
8 #heatmap
9 PHeatmap(object = bcr, pc.use = 1:12, cells.use = 120, do.balanced = TRUE,
10          label.columns = FALSE, use.full = FALSE)
11 #Tsne plot
12 TSNEPlot(object = bcr)|
13
14 li#Find differentially expressed genes between clusters. This will take some time
15 cluster1.markers <- FindMarkers(object = bcr, ident.1 = 1, min.pct = 0.25)
16 print(x = head(x = cluster1.markers, n = 5))
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Console Output:

```
Loading required package: cowplot
Attaching package: 'cowplot'
The following object is masked from 'package:ggplot2':
  ggsave
Loading required package: Matrix
> #Tsne plot
> TSNEPlot(object = bcr)
>
```

Environment:

- bcr: Large seurat (564.3 Mb)

Plots:

- TSNE_1
- TSNE_2

Files:

- Untitled1*
- Untitled2*

Help:

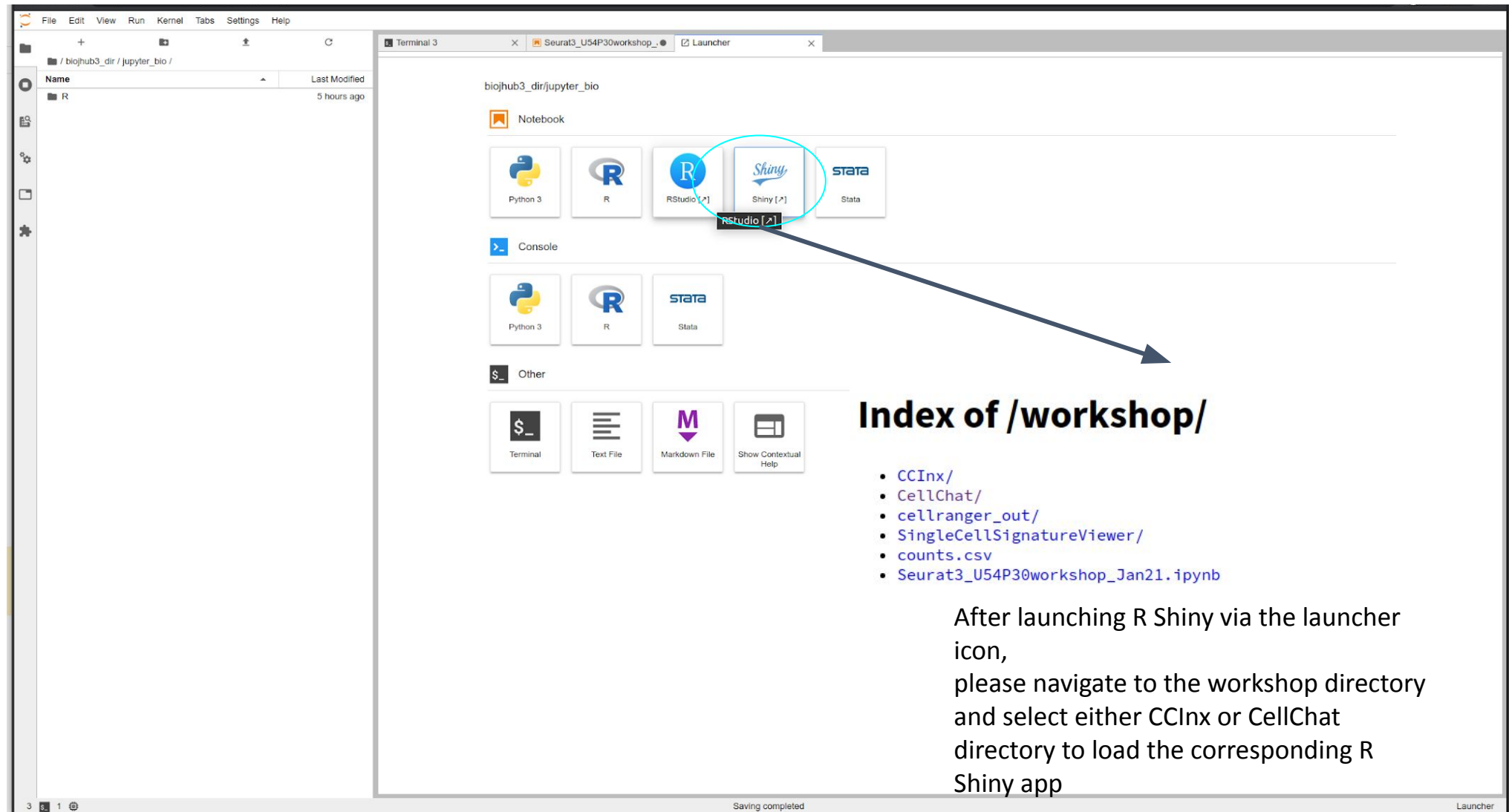
- Go to file/function
- Addins

Viewer:

- Zoom
- Export
- Publish

Saving completed

R Shiny Apps



biohub3_dir/jupyter_bio /

Terminal 3 x Seurat3_U54P30workshop_... Launcher x

biohub3_dir/jupyter_bio

Notebook

Python 3 R RStudio [1] Shiny [1] Stata

Console

Python 3 R Stata

Other

Terminal Text File Markdown File Show Contextual Help

Index of /workshop/

- [CCInx/](#)
- [CellChat/](#)
- [cellranger_out/](#)
- [SingleCellSignatureViewer/](#)
- [counts.csv](#)
- [Seurat3_U54P30workshop_Jan21.ipynb](#)

After launching R Shiny via the launcher icon,
please navigate to the workshop directory
and select either CCInx or CellChat
directory to load the corresponding R
Shiny app

3 1 2

Saving completed

Launcher

Jupyter Server Proxy

- Allows piggy-backing of other web services through Jupyterhub
- For example:
 - Rstudio server
 - Rshiny Dashboards
 - Xpra X11-to-HTML

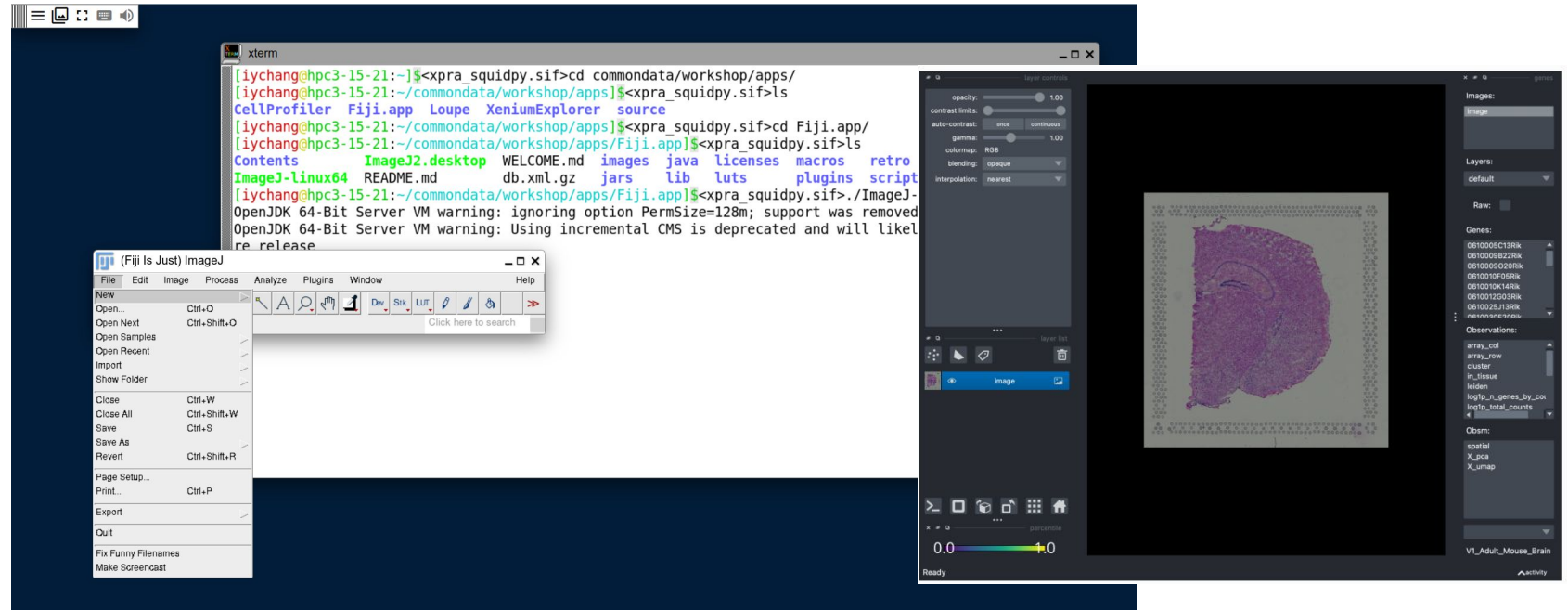


Linux Wine

Enable running windows applications in linux



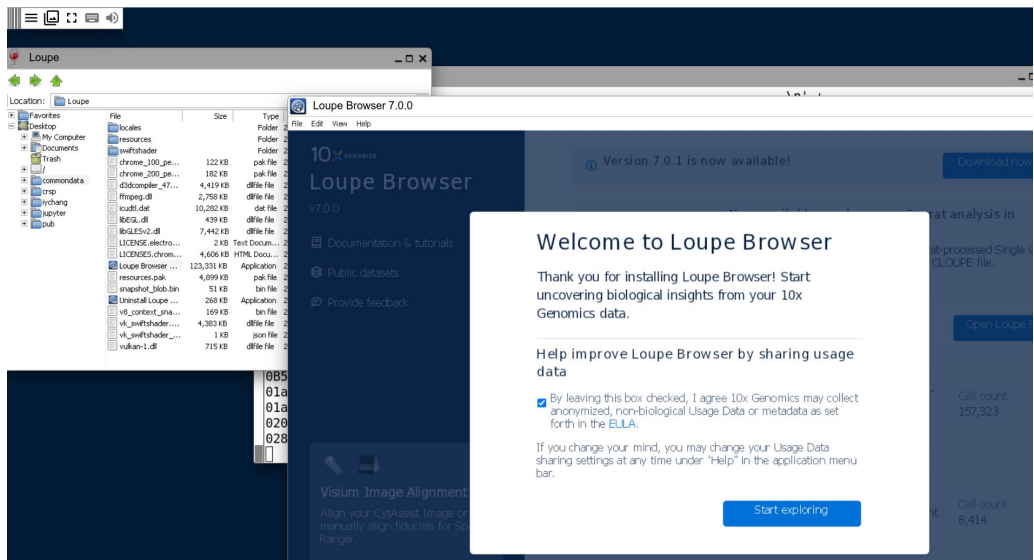
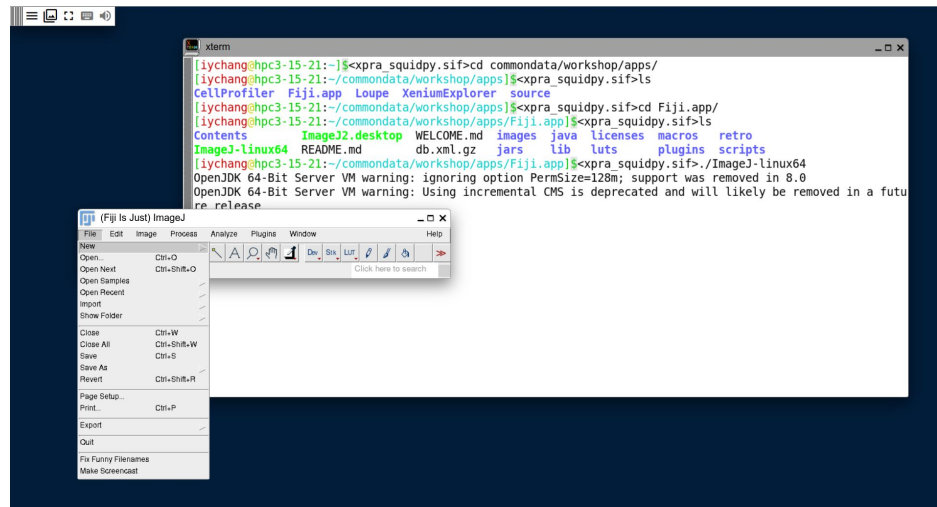
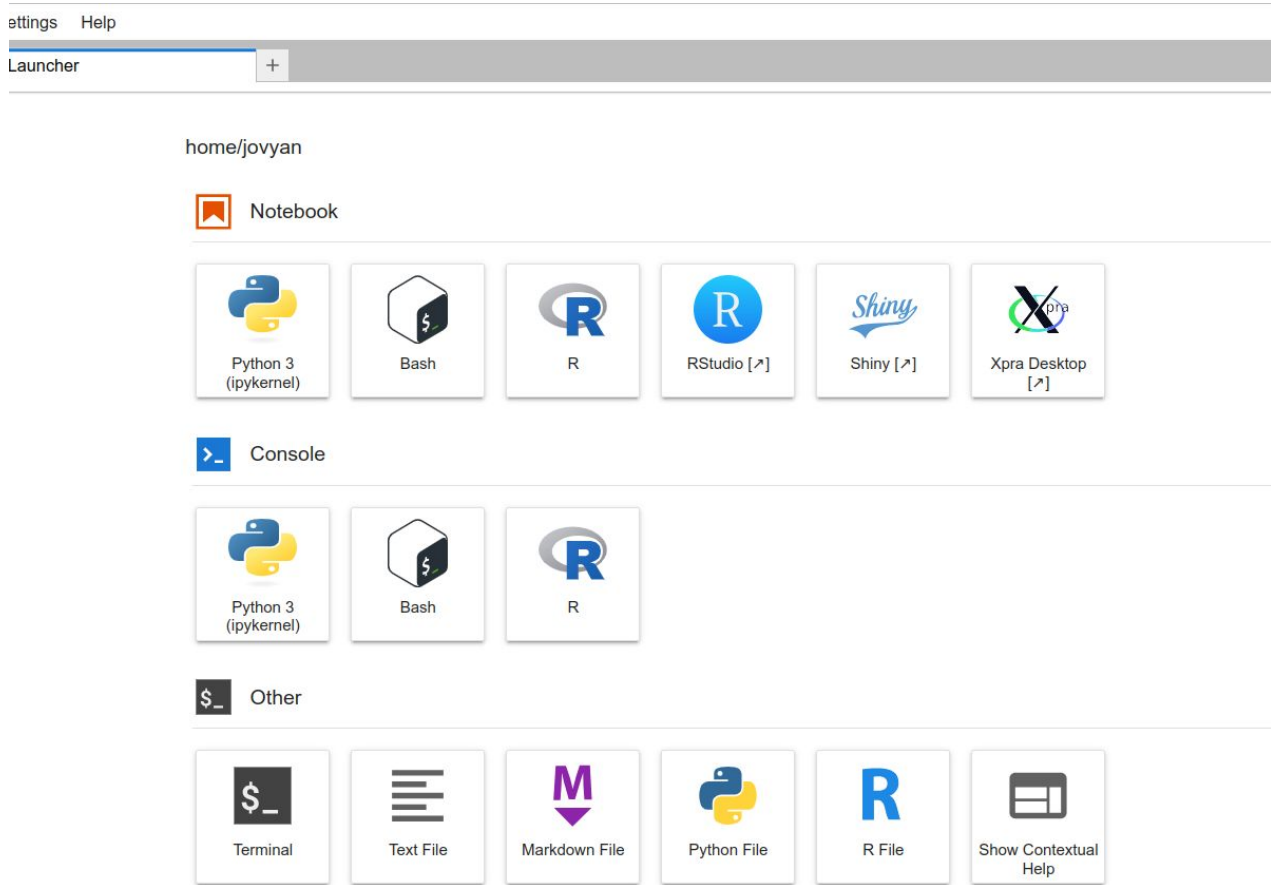
Synchronous Graphical User Interface: Xpra



X Persistent Remote Applications: run X clients on a remote host and direct the display to the local machine without the X clients closing or losing any state in case the network connection between the local machine and the remote host is lost.

Xpra X11-to-HTML: HTML5 based Xpra client that can be run via browser

Xpra/Wine via Jupyterhub



Be sure to stop your Jupyterhub notebook server after you are done. From the **File** menu choose **Hub Control Panel** and you will be forwarded to a screen similar where you can press on **Stop My Server** to shut down the server:



Research Cyberinfrastructure Center

[Home](#)[Token](#)[Admin](#)

npw

[Logout](#)

Stop My Server

My Server

Named Servers

In addition to your default server, you may have additional 3 server(s) with names. This allows you to have more than one server running at the same time.

Server name	URL	Last activity	Actions
Name your server	Add New Server		

Spatial-omics Live Demo

- Fiji & CellProfiler
- Loupe Browser & Xenium Explorer
- CellPose & SquidPy

<https://hpc3.rcic.uci.edu/biojhub4/>

[/dfs8/commondata/workshop/Seuratv5_VisiumHDworkshop_Nov24.ipynb](https://hpc3.rcic.uci.edu/biojhub4/dfs8/commondata/workshop/Seuratv5_VisiumHDworkshop_Nov24.ipynb)

Location of XPRA enabled Spatial-omics Apps

<i>/dfs8/commondata/workshop/apps/CellProfiler</i>	(Windows)
<i>/dfs8/commondata/workshop/apps/Fiji.app</i>	(Linux - X11)
<i>/dfs8/commondata/workshop/apps/Loupe</i>	(Windows)
<i>/dfs8/commondata/workshop/apps/XeniumExplorer</i>	(Windows)
<i>/dfs8/commondata/workshop/apps/ImageStudio</i>	(Windows)

For windows apps, please access via the command: *wine explorer*

Location of sample data:

/dfs8/commondata/workshop/Spatial_demo

XPRA - CellPose Container Specific Instruction

From the xterm in Xpra Desktop, enter:

```
python -m cellpose
```

or

```
cellpose
```

Acknowledgement

GRTHub: Suzanne Sandmeyer, Melanie Oakes, Jenny Wu, Christina Lin

RCIC: Phil Papadopoulos, Imam Toufique, Francisco Lopez, Nick Santucci,
Joulien Tatar, Nadya Williams