

June 23, 2025



Introduction to HPC & HPG

Molly Mitchell, PhD
Bioinformatics Supervisor for FDOH

This resource was made possible through funding provided under the Epidemiology and Laboratory Capacity for Prevention and Control of Emerging Infectious Diseases (ELC) Cooperative Agreement (CK24-0002), Project D: Advanced Molecular Detection to the Florida Department of Health. The conclusions, findings, and opinions expressed by authors do not necessarily reflect the official position of the U.S. Department of Health and Human Services, the Public Health Service, or the Centers for Disease Control and Prevention.

Intros – BRR Team



Molly Mitchell, PhD
Bioinformatics Supervisor
Molly.Mitchell@flhealth.gov



Nikhil Yengala Reddy, MS
Bioinformatician
Nikhil.Yengala@flhealth.gov



Sam Bernhoft, MPH
Domain Lead Bioinformatician
Samantha.Bernhoft@flhealth.gov

For any BRR requests, please send an email to bphl-sebioinformatics@flhealth.gov
Based on the requests one of us from the team will respond as soon as possible.

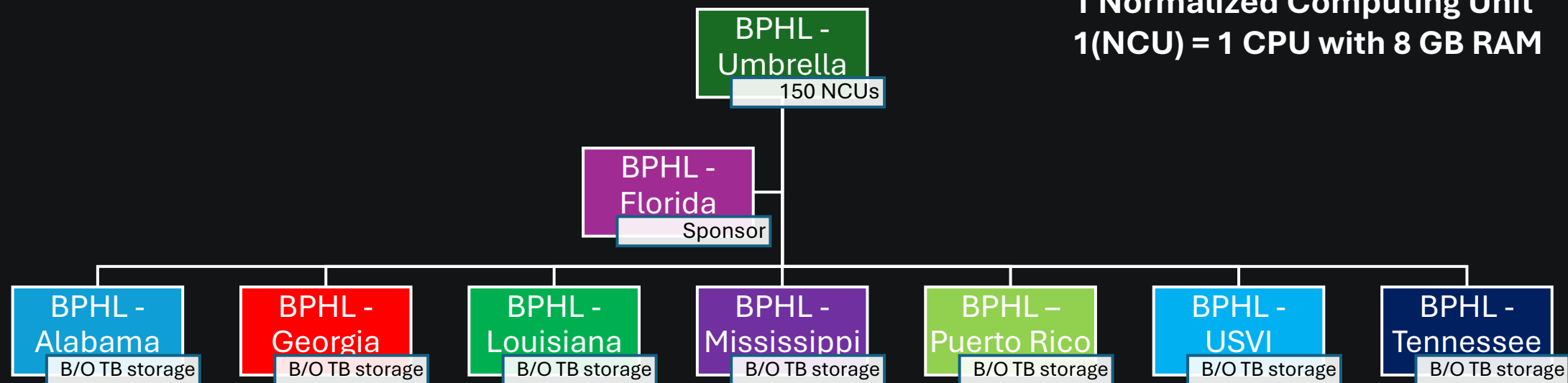


Overview of HiPerGator

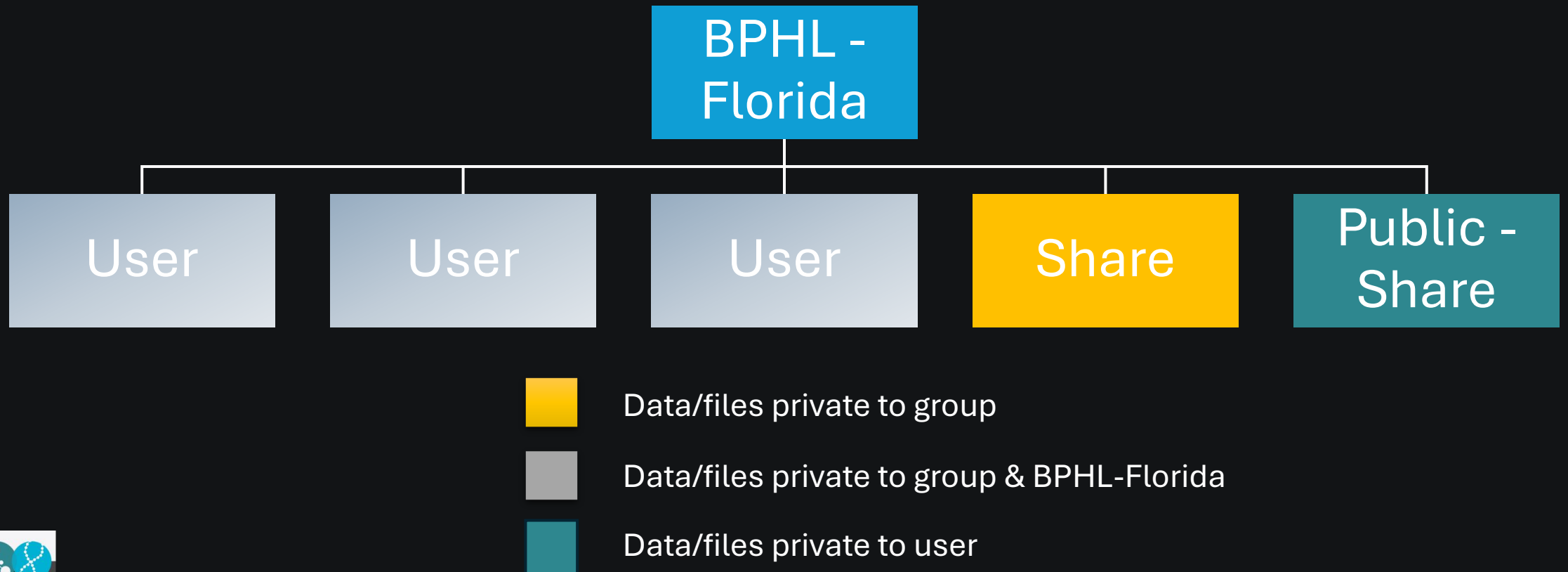
- HiPerGator is a powerful supercomputer by University of Florida which includes the latest generation of processors and offers nodes for memory-intensive computation.
- HiPerGator can be accessed through Linux command line via ssh, or through Open OnDemand
- Work on HiPerGator is submitted to SLURM scheduler to run in the batch system when resources are available.
- Compute capacity is sold in NCUs which is 1 CPU core & 7.8GB RAM
- Storage: **Blue** for high performance file system and **orange** for low performance file system; can be purchased in terabyte quantities.
- **Southeast Region jurisdictions receive computational resources for FREE (supported by BRR funding from CDC ELC)!**

HiPerGator State Groups

1 Normalized Computing Unit
1(NCU) = 1 CPU with 8 GB RAM



HiPerGator State Groups

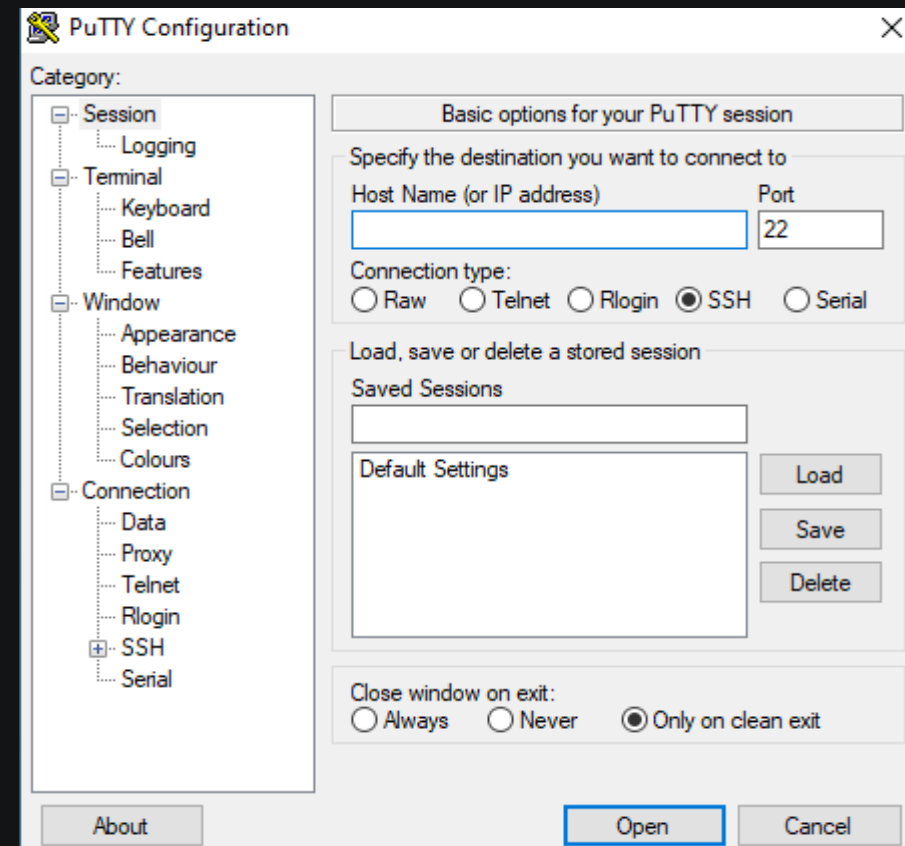


Storage on HPG

- **Home** storage: /home/<user>
 - 40 GB limit
 - scripts, code, compiled applications
- **Blue** storage: /blue/<group>/<user>; short-term, high-performance storage
- **Orange** storage: /orange/<group>/; long term storage
- To check your storage, you can navigate to the command line and enter
blue_quota
orange_quota
- **Orange** and **Blue** storage quotas are at the group level and based on investment.
- Also, if you have data that you are not actively using on HiPerGator, make sure you move the data to orange storage or to separate storage off HiPerGator
- As every lab is different, make sure your laboratory has a long-term data storage plans like **cloud**, **server or hard drive backups** that follow your lab's data retention policies

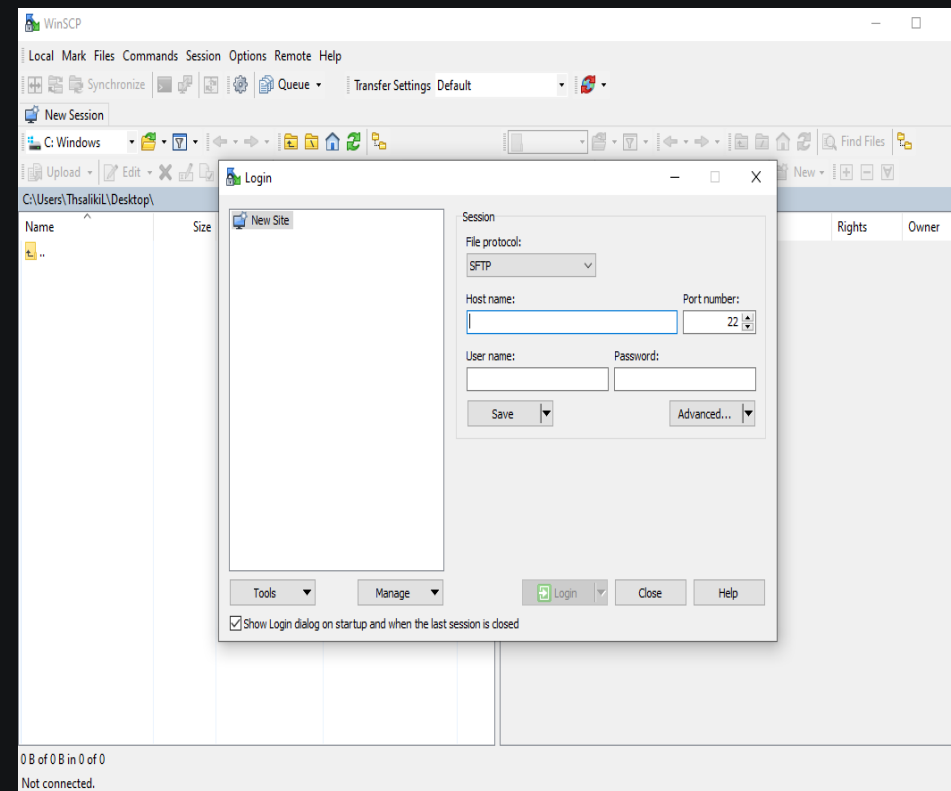
Access to HPG

- Every person should have their own HPG account to perform data analysis
- Detailed instructions on how to request & access your HiPerGator account is available in the attached [link](#)
- There are various software programs to access HPG through your Windows computer which may need to be installed by your IT department to access HPG
 - PuTTY
 - WinSCP



Access to HPG Cont.

- Putty allows you to login to HPG and work in the command-line via a terminal
- WinSCP allows you to transfer files from your local Windows computer to HPG
- Putty & WinSCP are free software that are usually approved for use by most IT departments but there are other programs besides these that work the same
 - You can always discuss those options with your IT department



Interactive Sessions

- For testing out tools in the command line that require more resources, like space or time, you can use the below command

```
srun --qos=bphl-umbrella --account=bphl-umbrella --cpus-per-task=<number of  
cpus> --mem=<memAmount>gb --time=<time limit> --pty bash -i
```

- Interactive work (other than managing jobs) should be done on a login node when the following are true:
 - No more than 16 cores
 - No longer than 10 min (wall time)
 - No more than 64 GB of RAM
- Otherwise, you need to start an interactive session, as described above, which will allow you to work interactively on a compute node

HiPerGator Trainings – by FDOH

- Introduction to HPC & HiPerGator
- Introduction to Linux – Part 1
- BaseSpace Command Line Interface
- Introduction to Linux – Part 2
- Compute environment setup & software install
- Individual pipeline trainings

Please let us know how we could improve these trainings

Updated trainings will be shared on the [StaPH-B/Southeast-region GitHub](#)


Open OnDemand

- Open OnDemand is setup to provide a web interface to start and attach to HiPerGator jobs
 - Supports GUI programs, job status, and other interactions with HiPerGator via seamless connections through web browsers.
- Open OnDemand service is available at the following URL: <https://ondemand.rc.ufl.edu>
- **Note:** IT Department may block from using Open OnDemand on your computer, so please make sure you contact IT people to solve this issue.

Open OnDemand on HPG

1. Connect to <https://ood.rc.ufl.edu/>
2. Launch a terminal
3. Launch many graphical applications like Rstudio, VScode, Matlab, etc.
4. Manage job submission
5. Upload, download, and edit files via your browser

Open OnDemand Dashboard

 Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾ My Interactive Sessions

Help ▾ Logged in as thsalikilakshmi Log Out

NOTICE: The gres request for A100 GPUs has changed. To request an A100 gpu, please use the gpu type "a100". For details, please visit https://help.rc.ufl.edu/doc/GPU_Access#SLURM_Options_for_A100_GPUs.

Welcome to the Gator Nation!

You are accessing a University of Florida information system and agree to abide by the terms and conditions of the UF Acceptable Use Policy.

Celebrating 5 Years

HiPerGator

The University of Florida Supercomputer

Starting an Interactive Application

- Select the application from drop-down menu of “Interactive Applications” to start an interactive application
- A form with the SLURM options will pop up on the screen for that application
- Fill in the fields with the resources you would like to allocate for the application session

Interactive Application

UF

Files ▾

Jobs ▾

Clusters ▾

Interactive Apps ▾

My Interactive Sessions

Help ▾

Logged in as thsalikilakshmi

Log Out

NOTICE: The gres request for A100 GPUs has changed. To request an A100 gpu, please use the gpu type "a100". For details, please visit https://help.rc.ufl.edu/doc/GPU_Access#SLURM_Options_for_A100_GPUs.

Home / My Interactive Sessions / Rstudio

Interactive Apps

Desktops

Hipergator Desktop

GUIs

Artemis

Beast

Console

DSI Studio

Fluent

Gaussian

IGV

ITK-SNAP

MRicron

Matlab

QGIS

Rstudio

SAS

SCIRun

Spyder

VMD

VScode

Visit

HWGUI Apps

FSLeyes

Freeview

MRicroGL

MRview

Servers

Jupyter Notebook

Rstudio version: 5080666

This app will launch Rstudio on the [HPG Cluster](#) using your investment resources. You will be able to interact with the VMD GUI through a VNC session.

Select RStudio version

Rstudio for R/4.2

This defines the version of RStudio you want to load.

Number of MPI Tasks (--ntasks)

1

Number of MPI tasks requested for application, (default = 1).

Number of CPU cores requested per MPI task (--cpus-per-task, -p)

1

Number of cores per MPI task requested for application, (default = 1).

Maximum memory requested for this job in Gigabytes (--mem, -m)

Maximum amount of memory to be used by the job (blank/default = 600M per CPU). If you are using advanced memory options in **Additional SLURM Options** below, then leave this blank.

SLURM Account (--account, -A)

Enter an alternative account if required. (default = primary investor)

QoS (Required if custom Account is set, --qos, -q)

Enter an alternative QoS, **required if alternative account is entered above. Please note, if you use the burst qos (-b), your jobs may take longer to start.** (default = primary investor)

Time Requested for this job in hours (--time, -t).

1

Time in hours requested from SLURM for this job to run. (default = 1)

Cluster partition (--partition, -p)

default

Select a specific cluster partition for job. (default = first available compute partition)

Generic Resource Request (--gres).

This is the Generic resource request string to request GPU resources. See also https://help.rc.ufl.edu/doc/GPU_Access

Additional SLURM Options

Include any further valid srun options per <https://slurm.schedmd.com/srun.html>. Each additional option should be separated by a space.

Launch

* The Rstudio session data for this session can be accessed under the data root directory.

powered by

OPEN OnDemand

OnDemand version: v1.8.20



General Form Fields

- Version – select the version of the application you want to load
- Number of MPI Tasks (--ntasks)
- Number of CPU cores requested per MPI task (--cpus-per-task, -p)
- Maximum memory requested for this job in Gigabytes (--mem, -m)
 - **Note:** If you are using advanced memory options in Additional SLURM Options below, then leave this blank.
- Time Requested for this job in hours (--time, -t)

General Form Fields Cont.


- SLURM Account (--account, -A): bphl-umbrella
- QoS (Required if custom Account is set, --qos, -q): bphl-umbrella
- Cluster partition (--partition, -p)
- Generic Resource Request (--gres)
- Additional SLURM Options

Select launch to submit your job request

Connecting to an Interactive Application

- Once the job has started, connect using the **"My Interactive Applications"** menu
- Select this menu and you will see a connection box for each application you have running and pending
- Once the job starts, a **Launch (App Name)** icon will appear
- Select this link to open a new browser window connecting you to your application

Job Connect Card

 Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾ My Interactive Sessions Help ▾ Logged in as thsalikilakshmi Log Out

NOTICE: The gres request for A100 GPUs has changed. To request an A100 gpu, please use the gpu type "a100". For details, please visit https://help.rc.ufl.edu/doc/GPU_Access#SLURM_Options_for_A100_GPUs.

Session was successfully created.

[Home](#) / [My Interactive Sessions](#)

Interactive Apps

Desktops

Hipergator Desktop

GUIs

Artemis

Beast

Console

DSI Studio

Fluent

Gaussian

Rstudio (62522002) 1 node | 1 core | Running

Host: c0702a-s11.ufhpc

Created at: 2023-04-26 09:36:36 EDT

Time Remaining: 58 minutes

Session ID: [3b90ba9a-b734-4dd7-9897-9882756a7776](#)

Compression

0 (low) to 9 (high)

Image Quality

0 (low) to 9 (high)

Launch Rstudio

View Only (Share-able Link)

Advanced Molecular Detection
Southeast Region Bioinformatics

20

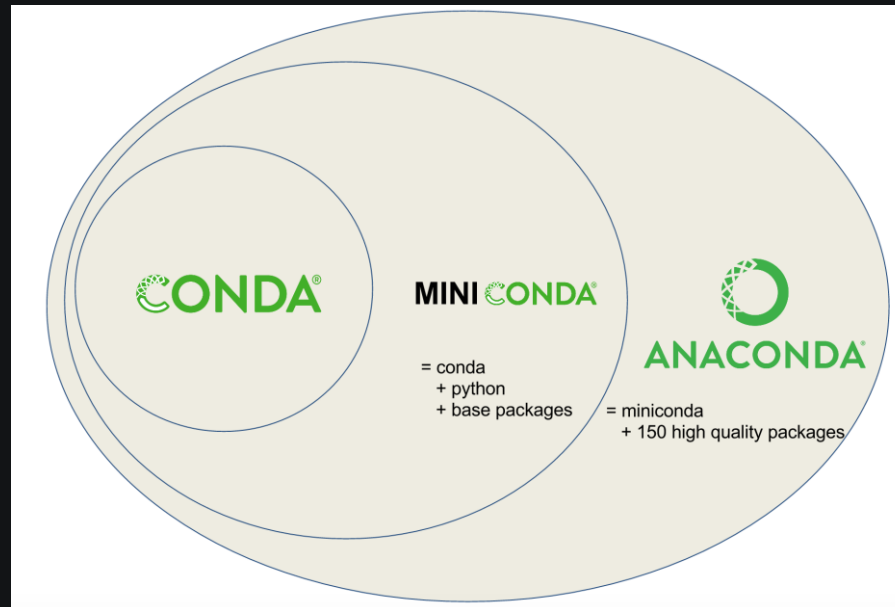
Conda/Anaconda

- Python-based environment & package manager
- Creates reproducible analysis pipelines using crowd-sourced & version-controlled packages
- Conda creates self-contained modules that contain the necessary programs, etc. for completing a particular computing task
- Reduces difficulties with typical tool/package installation
- Works really well with workflow managers – Nextflow!

Conda/Anaconda

Environment vs Package

- Environment is a computing environment which is the collection of programs, language libraries, etc. in which a computer operates
- Package is a collection of software that may contain things such as programs (e.g., Python), programming libraries (e.g., Perl), or other useful tools.
- Conda combines packages to construct environments for doing complex tasks



Graphic from: The Carpentries Incubator

Using HPG Conda

- Instead of installing anaconda, you can use the conda module in HPG

```
$ module load conda
```

- Creating and naming an environment (-yp 'your path' to where you want the conda env installed)

```
$ conda create -yp /blue/bphl-<state>/<user>/<conda_envs>/<env_name>
```

```
$ conda activate /blue/bphl-<state>/<user>/<conda_envs>/<env_name>
```

Updating Conda and Adding Channels

- Make sure to update conda regularly

```
$ conda update conda
```

- Adding additional channels - may already be installed (must do this the first time you use so you can install tools)

```
$ conda config --add channels defaults  
$ conda config --add channels bioconda  
$ conda config --add channels conda-forge
```


Conda Environments

- By default, an environment called "base" is created when installing & initializing Conda upon login. Base contains standard Python packages.
- To activate an environment in Conda, type

```
$ conda activate <environment name>
```

- As the base environment is loaded, to check what programs it contains, type

```
$ conda env list
```

Conda Environments Cont.

- To deactivate the environment, type

```
$ conda deactivate
```

- To create a new environment in conda, deactivate the current environment and then type

```
$ conda create -n <environment name>
```

- To delete an environment

```
$ conda remove --name <environment name> --all
```

- To delete an environment with an invalid path name (HPG conda module)

```
$ conda remove --all --prefix "/blue/bphl-<state>/<user>/conda_env/name_of_env"
```

Mamba

- Reimplementation of conda written in C++
 - Faster version
- Install your own version

```
$ conda install -c conda-forge mamba
```

- Or to use on HPG, load conda

```
$ module load conda
```

- Then, call mamba in place of conda

```
$ mamba create -n <environment name>  
$ mamba activate <environment>  
$ mamba install <dependencies> <package>
```

Example: Shigatyper Environment

- Let's use shigatyper as an example ([Shigatyper :: Anaconda.org](https://anaconda.org/bioconda/shigatyper))
 - “Quick and easy tool designed to determine Shigella serotype using Illumina or ONT reads”
- Create a shigatyper environment and install shigatyper with bioconda and conda-forge repository channel

```
$ conda create -n shigatyper -c conda-forge -c bioconda shigatyper
```

- Or the same environment can be made with the following three commands

```
$ conda create -n shigatyper
```

```
$ conda activate shigatyper
```

```
$ conda install -c conda-forge -c bioconda shigatyper
```

Why Use Containers?

- Containers are an empty room which users can configure and customize
- Containers are a standardized unit of software that packages code and all dependencies
- Containers change the user space into a swappable component which means programs, custom configurations, and environment can be independent
- Consolidating a workflow into a Singularity/Apptainer container simplifies distribution and replication of scientific results



Singularity – now Apptainer

- Singularity is a container platform
- Software application that allows users to have ‘full control’ over their operating system without the need for any ‘super-user’ privileges using the notion of containers or images
- Allows you to create and run containers that package software in a way that is portable & reproducible
- **Note** – on HPG singularity is available as apptainer module (**module load apptainer**)
- Containers are built using Singularity which can run on HPC clusters
- A container is a single file

```
$ singularity --version
```

```
$ singularity help
```

Importance of Apptainer

- Used to run complex applications on HPC clusters in a simple, portable, and reproducible way
- Verifiable reproducibility and security which uses cryptographic signatures, an immutable container image format, and in-memory decryption
- Integration over isolation by default
- Easily make use of GPUs, high speed networks, parallel filesystems on a cluster or server by default
- The single file SIF container format is easy to transport and share.
- A simple, effective security model

Setup your directory for using Singularity images on HPG

```
$ mkdir /blue/bphl-<state>/<user>/singularity/  
$ ln -s /blue/bphl-<state>/<user>/singularity/ ~/.singularity
```

- By default, Singularity will try to store your images in /home/.singularity/. However, these files can be quite large and may exceed your home directory storage quota.
- Use the above commands to make a symbolic link to a hidden directory on the /blue drive.

How to get Singularity Container Images

- A container image is a single executable file that defines the software environment and runs the container
- A single container image can be used to run multiple instances of the same container simultaneously for different jobs
- To get a container image for use on HiPerGator:
 - You can either pull (i.e., download) pre-built container images into one of your directories
 - Or externally build a custom container image from a definition file and then transfer it to one of your directories

Build Containers

- The "build" command accepts a target as input and produces a container as output
- It produces containers in two different formats:
 - A compressed read-only Singularity Image File (SIF) format suitable for production (default)
 - A writable root directory called a sandbox for interactive development (--sandbox option)
- The build command accepts an existing container as a target to create a container in either supported format or by converting existing containers from one format to another

```
$ singularity build
```

Build Containers – StaPH-B Toolkit

- Path for prebuilt StaPH-B tools

```
/apps/staphb-toolkit/containers/
```

- For example, for shigatyper:

```
$ singularity build shigatyper_2.0.3.sif docker://staphb/shigatyper:2.0.3
```

- [GitHub - StaPH-B/docker-builds: Dockerfiles and documentation on tools for public health bioinformatics](#)
- Any tool from this link can be built and then used similarly when applying the command above

Running Singularity Container Images

- Use the command **singularity exec** for executing commands within a container non-interactively

```
singularity exec
```

- If you want to include the singularity commands to run your job scripts when running batch slurm jobs make sure to load singularity **module load apptainer**

Nextflow Overview

- [Nextflow's documentation! — Nextflow 22.10.6 documentation](#)
- “Its fluent DSL simplifies the implementation and deployment of complex parallel and reactive workflows on clouds and clusters”
 - DSL – Domain Specific Language
- Enables scalable and reproducible scientific workflows using containers
 - **Singularity/Apptainer** or Docker
- Adaptable pipelines which are written in the most common scripting languages
- Can be used with conda environments
- There is also cloud support/utility
 - AWS, Google, Azure

Nextflow Processes and Channels

- A Nextflow script is made by joining multiple processes
- These processes can be written in any scripting language and executed on a Linux platform
 - Bash, Perl, Ruby, Python, etc.
- Processes can be run independently and are isolated from one another
 - Each process defines a channel as input and output
 - The string of processes (with inputs and outputs) make up the pipeline
- The pipeline runs as a channel, running the processes (can be simultaneous)

Importance of Nextflow

- We will be using Nextflow (as a dependency) for installing and using PHoeNix
- All our [BPHL-Molecular](#) pipelines are being converted to Nextflow
- Most bioinformatics pipelines are now developed using Nextflow
- Nextflow is also the most common workflow engine and will be the main format for the AMR Platform pipelines

Using Nextflow on HPG

- Load Nextflow module

```
$ module load nextflow
```

- Or create a conda/mamba specific nextflow environment (Note: we will be doing this for **PHoeNix #1**)

```
$ module load conda
```

```
mamba create -yp /blue/bphl-<state>/<user>/<conda_envs>/nextflow
```

```
mamba activate /blue/bphl-<state>/<user>/<conda_envs>/nextflow
```

```
mamba install -c bioconda nextflow=21.10.6
```



Basics of a Nextflow Command

- What's included in a basic Nextflow command
 - Calling Nextflow
 - Options
 - Command argument (run, pull, clone, clean, help, etc.)
 - Script file
 - Profile
 - Entry – if necessary

```
$ nextflow [options] COMMAND [arg...] script_file -profile <singularity/docker/custom> -entry  
[option]
```

Nextflow Training

- [Nextflow: implementing a simple pipeline | Microbiome binfies \(telatin.github.io\)](#)
 - Andrea Telatin's Nextflow Training
- [Nextflow Training Workshop | Seqera Labs](#)
- Use these links to learn more about writing and executing your own Nextflow workflows!
- We will be the beta-testers for NF-Tower this next fiscal year

Pipelines

Sanibel & Sanibel PB

- Nextflow version of the FLAQ_AMR pipeline (FL – BPHL's standard bacterial assembly pipeline with AMR detection) to analyze NGS data in fastq format from bacterial genome
- Compared to FLAQ_AMR, Sanibel significantly reduces runtime and is especially suitable for analysis of large sample sizes
- Sanibel PB pipeline includes clonal complex and serotype of *Neisseria* sp. and *H. influenzae* species as well as the other analyses

Pipelines Cont.

Talbot

- Talbot is an upgraded version of the pipeline FL-cgSNP
- Nextflow pipeline for pan genome analyses including phylogenetic tree

Amelia

- Nextflow pipeline used to analyze targeted NGS of *Mycoplasma genitalium*
- Outputs include QC, amplicon statistics, SNP calling, amino acid variation, etc.

Pipelines Cont.

Pensacola

- Nextflow pipeline to analyze *Candida auris* data from PacBio sequencing

Daytona

- Nextflow version of the Flaq_sc2 pipeline (FL BPHL's SARS-CoV-2 analysis pipeline)
- Human read removal function is added to the pipeline
- Daytona_WNV and Daytona_dengue

Pipelines Cont.

Flisochar (Florida Isolate Characterization)

- Pipeline used to characterize bacterial isolates
- Improves the identification of bacterial isolates using hybrid assembly from short and long-read sequencing data
- **Flisochar_wf** version for processing both ONT and PacBio long reads

Pipelines Cont.

FLAQ-SC2 (FLAQ-SARS-CoV-2)

- Generates SARS-CoV-2 consensus assemblies from ARTIC targeted amplicon sequencing using Illumina (e.g., Nextera XT or DNA Prep) and non-Illumina (e.g., Tailed) library prep
- Outputs variant file and final report with quality metrics (including a PASS/FAIL quality flag based on public repository submission criteria). Automatically generates flags if indels or internal stop codons are present, indicating the need for manual review, based on NCBI's VADR annotation tool
- Individual scripts also are available to prepare and format assemblies for batch submissions to GISAID and NCBI's Genbank, and to remove indels/SNPs that are likely PCR or sequencing artifacts/errors prior to submission)
- **FLAQ-SC2-meta** for wastewater samples that automatically runs the Freyja tool to estimate lineage populations
- **FLAQ-SC2-clearlabs** for Clearlabs data outputs to generate a report with assembly metrics, pangolin lineage, and VADR flags
- **FLAQ-MPX** for Monkeypox analysis pipeline for clinical specimens and Generates consensus assembly from tiled-amplicon data and variant calling

Analysis Request and/or Support

- Email bphl-sebioinformatics@flhealth.gov
- One-on-one Teams sessions to go through each New HPG User Training, including a walk-through to set up your HPG environment, and hands-on experience with each pipeline/script needed
- Custom pipeline/script development as requested
- All pipelines & scripts will be shared via GitHub ([BPHL-Molecular · GitHub](#)) or the public-share directory in HiPerGator via /blue/bphl-<state>/public-share/



Advanced Molecular Detection

Southeast Region Bioinformatics

Questions?

bphl-sebioinformatics@flhealth.gov

Molly Mitchell, PhD

Bioinformatics Supervisor

Molly.Mitchell@flhealth.gov

Nikhil Yengala Reddy, MS

Bioinformatician

Nikhil.Yengala@flhealth.gov