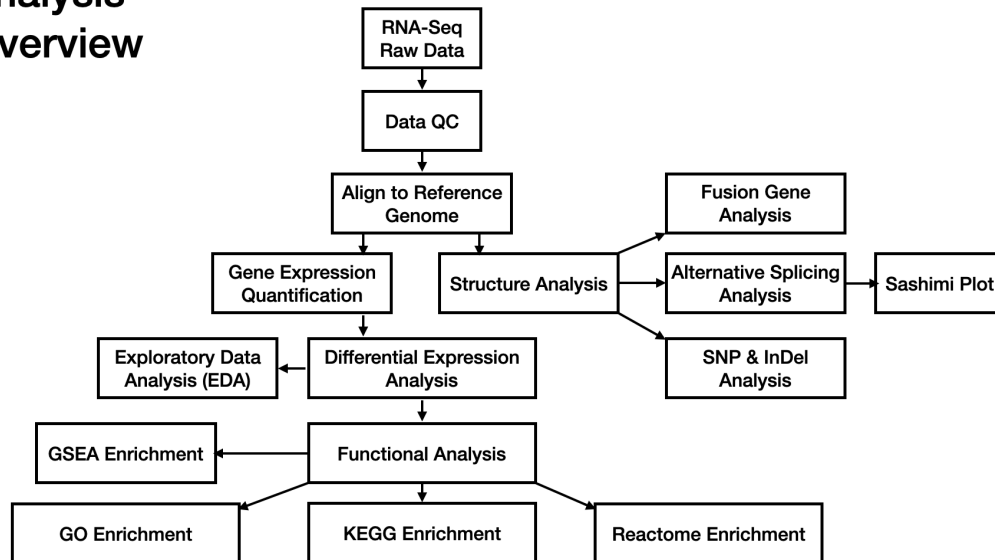
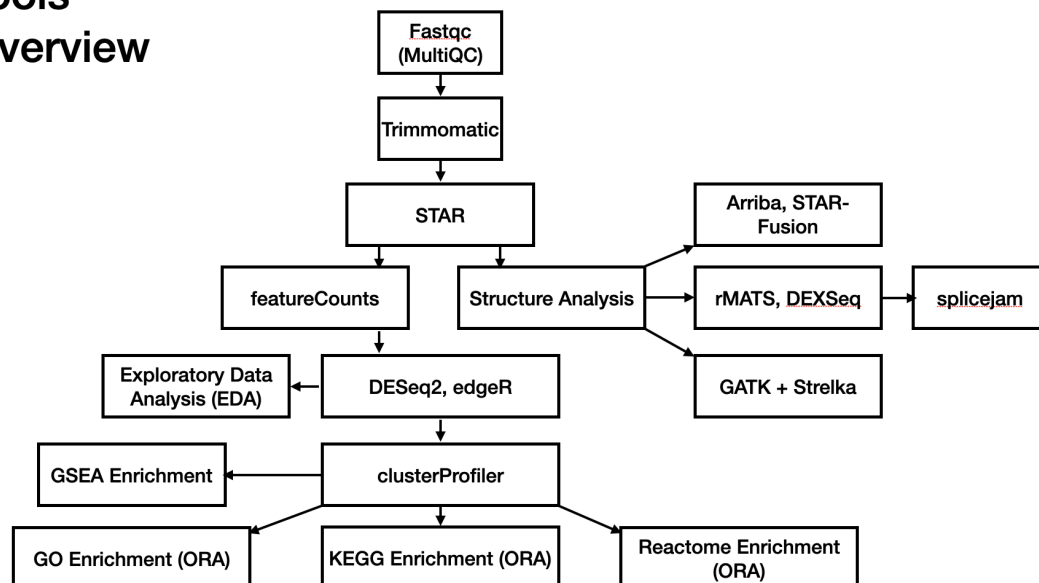


## Analysis Overview



## Tools Overview



### Data analysis workflow:

On Savio (Scratch directory storage limit: Unlimited; Home directory storage limit: 10GB), your computer, or Galaxy (Storage limit: 250GB):

1. Upload RNA-seq Data (Illumina platform) to Savio HPC server - Fastq files
2. Quality Control of Reads (**Software: FastQC, MultiQC**)
3. Trimming and Filtering Reads (**Software and bash code: Trimmomatic, slurm files**)
4. Aligning/Mapping Reads to a Reference Genome (**Software and bash code: STAR, slurm file**)
5. Calculating Expression levels: Quantification (**Software and bash code: featureCounts, slurm file**)
6. Differential Expression (Comparing Gene Expression Between Conditions) (**Software and R code: DESeq2, edgeR, mouse\_dge.R**)
7. Functional Analysis (**Software and R code: clusterProfiler, mouse\_cluster.R**)
8. Alternative Splicing Analysis (**Software and R code: rMATS, DEXSeq, splicejam, ggsashimi; mouse\_rmats.R**)
9. Visualization of Data (**R Code: mouse\_dge.R, mouse\_cluster.R, mouse\_rmats.R, splicejam, ggsashimi**)

## Setting Up Your System

1. Logging On
2. Learning How to Edit Your Files on the Terminal
3. Setting up Conda
4. Creating a Conda Environment and Activating It
5. Uploading Files
6. Learning About Bash
7. Learning About SLURM files

### 1. Logging On

#### HPC computing on Savio:

To obtain a Savio account, you need access to at least one project. A user account will be created for you automatically when you are granted access to a project. It usually takes a few days to get your account.

#### Instructions for obtaining HPC Savio account:

<https://docs-research-it.berkeley.edu/services/high-performance-computing/getting-account/>

#### A. Quick Instructions for Logging On

("\$" is the terminal prompt; don't type \$)

#### Log into Savio from local machine terminal window using a bash command:

Bash:

```
$ ssh username@hpc.brc.berkeley.edu (password is your pin+authenticator digits)
```

#### To logout of Savio using a bash command:

```
$ logout
```

For information regarding logging into Savio, see:

<https://docs-research-it.berkeley.edu/services/high-performance-computing/user-guide/logging-brc-clusters/>

**After you've logged onto Savio, move to Scratch:**

```
$ cd /global/scratch/users/your-username
```

## **B. Step-By-Step Instructions for Logging On**

Logging onto Savio (Berkeley's HPC cluster) from a terminal on a local computer:

```
$ ssh username@hpc.brc.berkeley.edu
```

Password: Your-pin + Google Authenticator (phone app) #

Example: Google Authenticator says: "105 149" and your pin is "9876", so your password is: 9876105149

If successful, you'll see:

```
Last login: Tue Jun 10 16:27:25 2025 from "IP address"  
[your-username@ln003 ~]$
```

The "~" means that you are located in your home directory. Switch immediately to your "scratch" directory:

```
$ cd /global/scratch/users/your-username
```

You'll see:

```
[your-username@ln003 your-username]$
```

On Savio, the scratch directory is called "your-username" - that's why you now see "your-username" vs "~".

## **2. Learning How to Edit Your Files on the Terminal**

### **Creating and Editing Files on the Terminal: The Nano Editor**

To create/open a file for editing:

```
$ nano .condrc (the cursor is the "$"; the file's name is ".condarc")
```

To edit:

Just start typing and use up/down/sideways arrows to move around

To save the file:

Press "control" and o

To close the file:

Press "control" and x

## **Creating and Editing Files on the terminal: The Vim Editor**

To create/open a file for editing:

\$ vi .condarc (the cursor is the "\$"; the file's name is ".condarc")

To edit:

Type "i": Enters editing mode (from the left). Now you can type your script.

Hit "esc": Exits editing mode. You can use up/down/sideways arrows to move around.

Now, type "dd": Deletes an entire row wherever the cursor is.

Type "x": Deletes a single character where the cursor is.

Type "i" again (Enter editing mode from the left) or type "a" (Enter editing mode on the right).

Hit "esc": Exits editing mode.

Type "ZZ": Saves and closes the file.

## **3. Setting up Conda**

**Move to your home directory:**

\$ cd

**Obtain Miniconda:**

\$ wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh

**Install Miniconda:**

\$ bash Miniconda3-latest-Linux-x86\_64.sh

**Activate Miniconda:**

\$ source ~/.bashrc

**Verify that Miniconda works:**

\$ conda --version

**Update and install Conda channels:**

\$ conda update -n base -c defaults conda

\$ conda config --add channels defaults

\$ conda config --add channels bioconda

\$ conda config --add channels conda-forge

**Set-up your .condarc file** (this directs your installations to scratch first, then to home if needed):

**Step 1.** You need to first make some directories because you are installing software in your scratch directory not your home directory (remember to replace “your-username” with your actual username):

```
$ mkdir -p /global/scratch/users/your-username/conda/pkgs/  
$ mkdir -p /global/scratch/users/your-username/conda/envs/
```

Next, move to your home directory unless you are already there:

```
$ cd /global/home/users/your-username/
```

**Step 2.** Use nano or vim to create/edit your .condarc file (I’m using vim) in your terminal (see above for nano and vim description):

```
$ vi .condarc (if you use nano, the command is: $ nano .condarc; remember “$” is the prompt)
```

**Here are the contents of the .condarc file:**

channels:

- <https://software.repos.intel.com/python/conda>
- conda-forge
- bioconda
- defaults

pkgs\_dirs:

- /global/scratch/users/your-username/conda/pkgs/
- /global/home/users/your-username/miniconda3/pkgs/

envs\_dirs:

- /global/scratch/users/your-username/conda/envs/
- /global/home/users/your-username/miniconda3/envs

auto\_activate\_base: false

solver: libmamba

## 4. Creating a Conda Virtual Environment and Activating It

**Check your current Python and R versions:**

```
$ python --version (gives “Python 3.12.2”, for example)
```

```
$ R --version (gives “R 4.4.3”, for example)
```

**Create a Conda virtual environment with a specific Python and R version:**

```
$ conda create -n mouse_env python=3.12.2 r-base=4.4.* -c conda-forge -c bioconda
```

(“\*” means the latest of the 4.4 versions)

Or, you can type (“\$” and “>” are prompts; be sure to type the “\” when shown):

```
$ conda create -n mouse_env \  
> python=3.12.2 \  
> r-base=4.4.* \
```

```
> -c conda-forge -c bioconda
```

**Activate environment:**

```
$ conda activate mouse_env
```

**To deactivate environment:**

```
$ conda deactivate
```

**Installing software in your virtual environment with Conda:**

Example: Install the software “samtools” into mouse\_env:

```
$ conda activate mouse_env
```

```
$ conda install -c bioconda samtools (You only need to install once)
```

**Other ways of installing software in your virtual environment:****1. Try using pip:**

```
$ pip install samtools
```

**2. Create a new conda environment for the specific software:**

```
$ conda create -n samtools_env -c conda-forge -c bioconda samtools
```

(This creates a conda environment with the samtools software package)

**5. Uploading/Downloading Files onto/from Savio****Method A: Use Open OnDemand:**

Use Open OnDemand (upload/download files from local computer to Savio cluster) by clicking on the following website and then logging into your Open OnDemand account using your UC Berkeley credentials:

<https://ood.brc.berkeley.edu>

**Method B: Use Globus:**

<https://docs-research-it.berkeley.edu/services/high-performance-computing/user-guide/data/transferring-data/using-globus-connect-savio/>

**Method C: Downloading files from online using wget:****Download file from a sequencing center’s website:**

Substitute actual portal for “novogene\_data\_portal”

```
wget https://novogene_data_portal.novogene.com/myfile.fastq.gz
```

(“gz” means gunzipped file)

**Download file from NCBI:**

Example:

```
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR102/004/SRR10207204/SRR10207204_1.fastq.gz
```

## 6. Learning About Bash: Common Bash Commands for the Terminal

Change to the Scratch directory from the Home directory, from a limited space (backed up) to an “unlimited” space (not backed up):

```
$ cd /global/scratch/users/your-username
```

Show your location:

```
$ pwd
```

Move backwards to previous directory:

```
$ cd .. (yes, you type two dots)
```

Move forwards to a directory within current directory (nested directories):

```
$ cd directory-name
```

Example:

```
$ pwd (gives us: toy_data/star_results/bam_files)
```

```
$ cd .. (moves us back to toy_data/star_results)
```

```
$ cd .. (moves us back to toy_data)
```

Then,

```
$ cd star_results (moves us forward to star_results folder), or
```

```
$ cd star_results/bam_files (moves us to bam_files)
```

Make a folder/directory:

```
$ mkdir folder-name
```

Copy a file:

```
$ cp original-file new-file
```

Remove a file:

```
$ rm file-name
```

Remove a directory:

```
$ rmdir directory-name
```

If folder is not empty, we need to empty it:

```
$ cd directory-name
```

```
$ rm *
```

```
$ cd ..
```

```
$ rmdir directory-name
```

Move a file to a new location/directory:

```
$ mv file-name new-location
```

List your files in a directory:

```
$ ls
```

or

```
$ ls -la
```

List the contents of a file:  
cat file-name

Example:  
cat hello.txt

## 7. Learning About SLURM files

### Run a SLURM job:

Typically, you can run the SLURM file in your scratch base directory (in case, you are using relative paths):

```
$ cd /global/scratch/users/your-username
```

```
$ sbatch my-file.sh (using the file “my-file.sh”; .sh means it is a bash or SLURM file)
```

Savio replies with your job number

### Check to see if your job finished:

```
$ squeue -u your-user-name
```

Checking on the status of your job:

```
$ ls <— you’ll see *.err and *.out files
```

```
$ cat *.err (or, cat *.out) or (if multiple .err and .out files),
```

```
$ cat job-name-job-number.err
```

```
$ cat job-name-job-number.out
```

These commands give information on what the computer has accomplished so far or any errors.

The difference between sbatch and srun:

Use sbatch for SLURM files you already wrote, software like trimmomatic or rMATS with many files, and srun when debugging code:

```
sbatch trim.sh (using a bash code file and the trimmomatic software)
```

```
srun test.R (debugging a file)
```

To use srun:

```
$ srun --pty -A Savio-project-you-joined -p savio2_htc (or another) -t 01:00:00 (time=1 hour)  
bash -i
```