# Analysis of RNA-Seq Data – Protocol

Biomedical Data Analysis – WS 23/24

Mariam Zaidi
Matriculation Number: 22300141
E-mail: mariam.zaidi@stud.th-deg.de
Deggendorf Institute of Technology

# Introduction

*Biomedical Data Analysis*

Biomedical data analysis encompasses a wide range of techniques in order to transform large datasets of biological and medical data into actionable knowledge. The field leverages **statistical** and **computational** methods to manage, analyze, and extract insights from datasets and plays a key role in advancing medical research and improving healthcare.

*Reproducible Research*

Reproducible research refers to the practice of publishing scientific claims, data analyses, and their corresponding data and software code. Since it allows other researchers to assess the existing findings and build upon them, reproducibility is becoming increasingly significant pertaining to the growing complexity of data analyses, larger datasets, and highly sophisticated computations. It allows the focus to remain on the actual content of an analysis rather than on superficial details reported in a written summary (Peng, 2011). Moreover, reproducible research elevates the impact of research by making it more accessible, credible, and operative, ultimately benefiting the society.

To further elaborate, reproducibility, which refers to the reproduction of results from an experiment or study using the **same methods, data, and analysis steps**, differs from the practice of replication. In replication, a study or experiment is repeated using the same methods on **different subjects or samples** to confirm or refute the results of the initial study. Replication provides stronger evidence for validating the robustness of an experiment's findings while reproducibility ensures that the findings can be independently verified and built upon (Peng, 2011).

Several technical challenges (Peng, 2011) need to be addressed to enable reproducible research, including:

- Lack of standardization or consistency in data formats, measurement protocols, and statistical techniques,

- Unavailability of tools for managing and sharing data,
- Restricted raw data due to privacy concerns or storage limitations,
- Poorly or undocumented code,
- Lack of incentives for researchers to share their data.

Managing these challenges requires a multifaceted approach involving institutions, funding agencies, researchers, and software developers. Moreover, reproducible research can be further expedited by encouraging open communication and collaboration in the culture of scientific research and through open science initiatives such as data repositories and reproducibility checklists.

*NCBI SRA*

NCBI (National Center for Biotechnology Information) SRA (Sequence Read Archive) is a public database consisting of archived and freely distributable next-generation sequence data and metadata from human, animal, plant, and environmental samples. The SRA provides researchers with access to a vast repository of biological data, such as alignment information and raw sequencing data which facilitates reproducibility by enabling access and analyzation of the same data used in other studies (*The Sequence Read Archive (SRA): Getting Started*, n.d.; OpenAI, 2024).

*Conda/Bioconda*

To appropriately create and manage reproducible computation environments, a package manager and environment management system called Conda/Bioconda is utilized by researchers. This open-source system is specific for scientific computing and can install, run, and update software packages and dependencies across distinct operating systems. Bioconda, a channel for the Conda package manager, provides tools and software specific to bioinformatics (Grüning et al., 2018). Conda/Bioconda is significant in advancing reproducible research since it permits researchers to create and share easily reproducible computational environments due to their consistency across different platforms.

*Jupyterlab*

Jupyterlab, on the other hand, is an open-source web-based interactive development environment supporting code, text, and media, that allows researchers to create and share reproducible research notebooks. These Jupyter notebooks can contain live code, equations, visualizations, and narrative text providing a modern, flexible, and powerful environment to create clear and comprehensive records of research processes (Beg et al., 2021). Jupyterlab's dynamic nature enables researchers to share their work in an easily accessible format simultaneously making it easier to reproduce and validate scientific findings.

NCBI SRA, Conda/Bioconda, and Jupyterlab are all essential tools for promoting reproducibility in life sciences research. These tools empower researchers to verify, extend, and improvise the work of others by providing access to raw data, overseeing software dependencies, and promoting code sharing.

## Technical Setup

Connect to eduroam network on campus or initiate a VPN connection to DIT (Cisco AnyConnect client). This step is always required prior to establishing a connection to the Virtual Machine (VM) on campus.

Open the search bar using Command + Space Bar keys. Search for "Terminal," the command-line interface for macOS, and open it using the Enter key.

Download the SSH key from the email into the downloads folder. From the downloads folder, copy the path to the SSH key by right-clicking on the 'key_vm_36.txt' file, pressing and holding the Option key, and selecting 'Copy "key_vm_36.txt" as Pathname'.

**Connect to remote Linux machine via SSH:**

Type in the following command specifying the IP address and the path to the SSH in the terminal and press Enter:

```
ssh ubuntu@10.70.1.105 -i /Users/mariam/Downloads/key_vm_36.txt
```

```
Mariams-MacBook-Air:~ mariam$ ssh ubuntu@10.70.1.105 -i /Users/mariam/Downloads/key_vm_36.txt
```

Connecting to the server for the first time will return a warning message stating,

'Are you sure you want to continue connecting (yes/no/[fingerprint])?'

Type "yes" and press Enter.

```
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-86-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Oct 24 12:06:14 PM UTC 2023

  System load:  0.0                Processes:              172
  Usage of /:   29.2% of 23.40GB   Users logged in:        0
  Memory usage: 6%                 IPv4 address for ens33: 10.70.1.105
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Oct  8 11:19:42 2023 from 172.26.1.4
```

Connection to remote Linux machine is successful.

```
ubuntu@lsi:~$
```

SSH session is active. Commands can be executed after the prompt.

To exit a session, type exit in the terminal window.

**Installing conda:**

For quick command line installation of Miniconda on the VM, the code specific for Linux machines is:

```
mkdir -p ~/miniconda3

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh -O
~/miniconda3/miniconda.sh

bash ~/miniconda3/miniconda.sh -b -u -p ~/miniconda3

rm -rf ~/miniconda3/miniconda.sh
```

After installation, the following command initializes the newly installed miniconda:

```
~/miniconda3/bin/conda init bash
```

The command conda info can be used to check if conda was installed properly. It displays information about the current conda installation, such as the version of conda and the location of the user config file.

This command is used in a terminal as follows:

```
(base) ubuntu@lsi:~$ conda info

     active environment : base
    active env location : /home/ubuntu/miniconda3
            shell level : 1
       user config file : /home/ubuntu/.condarc
 populated config files :
          conda version : 23.9.0
    conda-build version : not installed
         python version : 3.11.5.final.0
       virtual packages : __archspec=1=x86_64
                          __glibc=2.35=0
                          __linux=5.15.0=0
                          __unix=0=0
       base environment : /home/ubuntu/miniconda3  (writable)
      conda av data dir : /home/ubuntu/miniconda3/etc/conda
  conda av metadata url : None
           channel URLs : https://repo.anaconda.com/pkgs/main/linux-64
                          https://repo.anaconda.com/pkgs/main/noarch
                          https://repo.anaconda.com/pkgs/r/linux-64
                          https://repo.anaconda.com/pkgs/r/noarch
          package cache : /home/ubuntu/miniconda3/pkgs
                          /home/ubuntu/.conda/pkgs
       envs directories : /home/ubuntu/miniconda3/envs
                          /home/ubuntu/.conda/envs
                platform : linux-64
              user-agent : conda/23.9.0 requests/2.31.0 CPython/3.11.5 Linux/5.15.0-86-generic ubuntu/22.04.3 glibc/2.35
                 UID:GID : 1000:1000
              netrc file : None
            offline mode : False
```

All the details regarding the current Conda installation are displayed.

*Conda channels*

During package installation in Conda, packages are downloaded and extracted from a directory using the directory's URL. These directories containing packages are called channels. Channels serve as the base for hosting and managing packages. They are the locations where packages are stored remotely and can be downloaded from.

**Installing Bioconda and Conda-Forge:**

For instance, the channel Bioconda provides biomedical research-related software packages using the conda package manager. Conda-Forge, an open-source repository for scientific software packages, offers pre-built Conda packages that can be used for scientific computing.

Setting up Bioconda is done using the commands:

```
conda config --add channels defaults
conda config --add channels bioconda
conda config --add channels conda-forge
conda config --set channel_priority strict
```

**Creating an environment named 'py39' with python version 3.9:**

A new environment named "py39" is created using the following code. The Python version of 3.9 is specified by mentioning python=3.9.

```
conda create --name py39 python=3.9
```

The command is executed in the terminal with the displayed result:

```
(base) ubuntu@lsi:~$ conda create --name py39 python=3.9
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/ubuntu/miniconda3/envs/py39

  added / updated specs:
    - python=3.9
```

To activate the conda environment:

```
conda activate py39
```

To deactivate the conda environment:

```
conda deactivate
```

**Understanding the context of the Linux shell PATH variable:**

The PATH environment variable in Linux shell specifies the directories where the executable files are located. The base environment is the default environment created when conda is installed. It consists of the conda package manager and additional basic packages.

Upon installation of conda, the path to the base environment is added to the PATH variable which allows the execution of conda commands from anywhere in the terminal.

To view the path of the current working environment, the command is:

```
echo $PATH
```

```
(py39) ubuntu@lsi:~$ echo $PATH
/home/ubuntu/miniconda3/envs/py39/bin:/home/ubuntu/miniconda3/condabin:/usr/local/sbin:
/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```

This is the path of the 'py39' environment.

To view the path of the base environment, use the conda deactivate command and run echo $PATH again.

```
(base) ubuntu@lsi:~$ echo $PATH
/home/ubuntu/miniconda3/bin:/home/ubuntu/miniconda3/condabin:/usr/local/sbin:/usr/local
/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```

To easily switch between environments and run conda commands from anywhere in the terminal, the base environment in the PATH variable is important. When the new environment is created, conda adds the path to the new environment to the PATH variable as well. The newly generated environment 'py39' is a clean environment with no packages unless newly installed.

The difference observed in their respective paths are highlighted.

/home/ubuntu/miniconda3/envs/py39/bin: precedes the base path in 'py39' environment ensuring that the commands and packages specific to 'py39' will be prioritized when the environment is active. This occurs due to its **path precedence**. Moreover, this isolation prevents conflicts and compatibility problems by establishing a separate space for dependencies and packages for projects using 'py39'.

**Installing fastqc via condo to py39:**

Fastqc is installed to the py39 environment using the following command:

```
conda install -c bioconda fastqc
```

It is executed in the terminal as displayed:

```
(py39) ubuntu@lsi:~$ conda install -c bioconda fastqc
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/ubuntu/miniconda3/envs/py39

  added / updated specs:
    - fastqc


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    expat-2.5.0                |       hcb278e6_1         134 KB  conda-forge
    fastqc-0.12.1              |       hdfd78af_0        11.1 MB  bioconda
    font-ttf-dejavu-sans-mono-2.37|       hab24e00_0         388 KB  conda-forge
    fontconfig-2.14.2          |       h14ed4e7_0         266 KB  conda-forge
    freetype-2.12.1            |       h267a509_2         620 KB  conda-forge
    libexpat-2.5.0             |       hcb278e6_1          76 KB  conda-forge
    libpng-1.6.39              |       h753d276_0         276 KB  conda-forge
    openjdk-8.0.382            |       hd590300_0        88.4 MB  conda-forge
    perl-5.32.1                | 4_hd590300_perl5        12.7 MB  conda-forge
    ------------------------------------------------------------
                                           Total:       114.0 MB

The following NEW packages will be INSTALLED:

  expat              conda-forge/linux-64::expat-2.5.0-hcb278e6_1
  fastqc             bioconda/noarch::fastqc-0.12.1-hdfd78af_0
  font-ttf-dejavu-s~ conda-forge/noarch::font-ttf-dejavu-sans-mono-2.37-hab24e00_0
  fontconfig         conda-forge/linux-64::fontconfig-2.14.2-h14ed4e7_0
  freetype           conda-forge/linux-64::freetype-2.12.1-h267a509_2
  libexpat           conda-forge/linux-64::libexpat-2.5.0-hcb278e6_1
  libpng             conda-forge/linux-64::libpng-1.6.39-h753d276_0
  openjdk            conda-forge/linux-64::openjdk-8.0.382-hd590300_0
  perl               conda-forge/linux-64::perl-5.32.1-4_hd590300_perl5


Proceed ([y]/n)? y
```

The fastqc program can be started by running the following command in the terminal:

```
fastqc – h
```

Running the command displays an overview of the fastqc tool, its usage, and the options available.

```
(py39) ubuntu@lsi:~$ fastqc -h

            FastQC - A high throughput sequence QC analysis tool

SYNOPSIS

        fastqc seqfile1 seqfile2 .. seqfileN

    fastqc [-o output dir] [--(no)extract] [-f fastq|bam|sam]
           [-c contaminant file] seqfile1 .. seqfileN

DESCRIPTION

    FastQC reads a set of sequence files and produces from each one a quality
    control report consisting of a number of different modules, each one of
    which will help to identify a different potential type of problem in your
    data.

    If no files to process are specified on the command line then the program
    will start as an interactive graphical application.  If files are provided
    on the command line then the program will run with no user interaction
    required.  In this mode it is suitable for inclusion into a standardised
    analysis pipeline.

    The options for the program as as follows:

    -h --help       Print this help file and exit

    -v --version    Print the version of the program and exit
```

**Viewing environments:**

To view a list of all the environments, present in conda, including their paths, use the conda env list command. The * indicates the current working environment.

```
(py39) ubuntu@lsi:~$ conda env list
# conda environments:
#
base                     /home/ubuntu/miniconda3
py39                  *  /home/ubuntu/miniconda3/envs/py39
```

The packages available in the conda environment can be listed with the command conda list.

```
(py39) ubuntu@lsi:~$ conda list
# packages in environment at /home/ubuntu/miniconda3/envs/py39:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                 conda_forge    conda-forge
_openmp_mutex             4.5                       2_gnu    conda-forge
bzip2                     1.0.8                h7f98852_4    conda-forge
ca-certificates           2023.7.22            hbcca054_0    conda-forge
expat                     2.5.0                hcb278e6_1    conda-forge
fastqc                    0.12.1               hdfd78af_0    bioconda
font-ttf-dejavu-sans-mono 2.37                 hab24e00_0    conda-forge
fontconfig                2.14.2               h14ed4e7_0    conda-forge
freetype                  2.12.1               h267a509_2    conda-forge
ld_impl_linux-64          2.40                 h41732ed_0    conda-forge
libexpat                  2.5.0                hcb278e6_1    conda-forge
libffi                    3.4.2                h7f98852_5    conda-forge
```

To list packages installed in a specific environment, include the -n option followed by the name of the environment.

```
(base) ubuntu@lsi:~$ conda list -n py39
# packages in environment at /home/ubuntu/miniconda3/envs/py39:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                 conda_forge    conda-forge
_openmp_mutex             4.5                       2_gnu    conda-forge
anyio                     4.0.0              pyhd8ed1ab_0    conda-forge
argon2-cffi               23.1.0             pyhd8ed1ab_0    conda-forge
argon2-cffi-bindings      21.2.0           py39hd1e30aa_4    conda-forge
arrow                     1.3.0              pyhd8ed1ab_0    conda-forge
asttokens                 2.4.1              pyhd8ed1ab_0    conda-forge
async-lru                 2.0.4              pyhd8ed1ab_0    conda-forge
attrs                     23.1.0             pyh71513ae_1    conda-forge
babel                     2.13.1             pyhd8ed1ab_0    conda-forge
backcall                  0.2.0              pyh9f0ad1d_0    conda-forge
backports                 1.0                pyhd8ed1ab_3    conda-forge
backports.functools_lru_cache 1.6.5              pyhd8ed1ab_0    conda-forge
bash                      5.2.15              h7f99829_1    conda-forge
bash_kernel               0.9.0              pyh1a96a4e_0    conda-forge
beautifulsoup4            4.12.2             pyha770c72_0    conda-forge
bleach                    6.1.0              pyhd8ed1ab_0    conda-forge
brotli-python             1.1.0            py39h3d6467e_1    conda-forge
bzip2                     1.0.8                h7f98852_4    conda-forge
ca-certificates           2023.7.22            hbcca054_0    conda-forge
cached-property           1.5.2                hd8ed1ab_1    conda-forge
cached_property           1.5.2              pyha770c72_1    conda-forge
certifi                   2023.7.22          pyhd8ed1ab_0    conda-forge
cffi                      1.16.0           py39h7a31438_0    conda-forge
charset-normalizer        3.3.1              pyhd8ed1ab_0    conda-forge
comm                      0.1.4              pyhd8ed1ab_0    conda-forge
debugpy                   1.8.0            py39h3d6467e_1    conda-forge
decorator                 5.1.1              pyhd8ed1ab_0    conda-forge
defusedxml                0.7.1              pyhd8ed1ab_0    conda-forge
```

**Installing jupyterlab, bioconductor-deseq2 and bash_kernel via conda to py39:**

Installation of the packages bash_kernel, jupyterlab, and bioconductor-deseq2 is done using the following commands:

bash_kernel:

```
conda install -c conda-forge bash_kernel
```

jupyterlab:

```
conda install conda-forge jupyterlab
```

bioconductor-deseq2:

```
conda install bioconda bioconductor-deseq2
```

**Note:** If the packages do not install or consume an unusual amount of time during the installation process due to occurring conflicts, create a new environment and install the new packages there. Conflicts could arise due to incompatible dependencies among the packages.

Since the bioconductor-deseq2 package did not install in 'py39' due to conflicts, a new environment named 'bio' is created.

To confirm the installation of packages, run conda list in the respective environment.

**Note:** the conda --help command can be used alongside commands in the command line to acquire help for that command. It also displays a list of supported conda commands.

```
(bio) ubuntu@lsi:~$ conda list
# packages in environment at /home/ubuntu/miniconda3/envs/bio:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                 conda_forge    conda-forge
_openmp_mutex             4.5                       2_gnu    conda-forge
_r-mutex                  1.0.1               anacondar_1    conda-forge
aioeasywebdav             2.4.0            py38h578d9bd_1001    conda-forge
aiohttp                   3.9.1            py38h01eb140_0    conda-forge
aiosignal                 1.3.1              pyhd8ed1ab_0    conda-forge
amply                     0.1.6              pyhd8ed1ab_0    conda-forge
anyio                     4.0.0              pyhd8ed1ab_0    conda-forge
appdirs                   1.4.4              pyh9f0ad1d_0    conda-forge
argcomplete               3.1.4              pyhd8ed1ab_0    conda-forge
argon2-cffi               23.1.0             pyhd8ed1ab_0    conda-forge
argon2-cffi-bindings      21.2.0           py38h01eb140_4    conda-forge
arrow                     1.3.0              pyhd8ed1ab_0    conda-forge
asttokens                 2.4.1              pyhd8ed1ab_0    conda-forge
async-lru                 2.0.4              pyhd8ed1ab_0    conda-forge
async-timeout             4.0.3              pyhd8ed1ab_0    conda-forge
attmap                    0.13.2             pyhd8ed1ab_0    conda-forge
attrs                     23.1.0             pyh71513ae_1    conda-forge
babel                     2.13.1             pyhd8ed1ab_0    conda-forge
backcall                  0.2.0              pyh9f0ad1d_0    conda-forge
backports                 1.0                pyhd8ed1ab_3    conda-forge
backports.functools_lru_cache 1.6.5                pyhd8ed1ab_0    conda-forge
bash                      5.2.15               h7f99829_1    conda-forge
bash_kernel               0.9.0              pyh1a96a4e_0    conda-forge
bcrypt                    4.1.2            py38h0cc4f7c_0    conda-forge
beautifulsoup4            4.12.2             pyha770c72_0    conda-forge
binutils_impl_linux-64    2.40                 hf600244_0    conda-forge
bioconductor-biobase      2.60.0            r43ha9d7317_0    bioconda
bioconductor-biocgenerics 0.46.0            r43hdfd78af_0    bioconda
bioconductor-biocparallel 1.34.2            r43hf17093f_0    bioconda
bioconductor-data-packages 20230718              hdfd78af_1    bioconda
bioconductor-delayedarray 0.26.6            r43ha9d7317_0    bioconda
bioconductor-deseq2       1.40.2            r43hf17093f_0    bioconda
bioconductor-genomeinfodb 1.36.1            r43hdfd78af_0    bioconda
bioconductor-genomeinfodbdata 1.2.10              r43hdfd78af_0    bioconda
```

```
json5                     0.9.14             pyhd8ed1ab_0    conda-forge
jsonpointer               2.4              py39hf3d152e_3    conda-forge
jsonschema                4.19.2             pyhd8ed1ab_0    conda-forge
jsonschema-specifications 2023.7.1           pyhd8ed1ab_0    conda-forge
jsonschema-with-format-nongpl 4.19.2                pyhd8ed1ab_0    conda-forge
jupyter-lsp               2.2.0              pyhd8ed1ab_0    conda-forge
jupyter_client            8.5.0              pyhd8ed1ab_0    conda-forge
jupyter_core              5.5.0            py39hf3d152e_0    conda-forge
jupyter_events            0.8.0              pyhd8ed1ab_0    conda-forge
jupyter_server            2.9.1              pyhd8ed1ab_0    conda-forge
jupyter_server_terminals  0.4.4              pyhd8ed1ab_1    conda-forge
jupyterlab                4.0.7              pyhd8ed1ab_0    conda-forge
jupyterlab_pygments       0.2.2              pyhd8ed1ab_0    conda-forge
jupyterlab_server         2.25.0             pyhd8ed1ab_0    conda-forge
ld_impl_linux-64          2.40                 h41732ed_0    conda-forge
libexpat                  2.5.0                hcb278e6_1    conda-forge
libffi                    3.4.2                h7f98852_5    conda-forge
```

**Environments, Virtual Machines, and Containers**

*Environments*

- Tools for separating distinct projects or applications in software development.
- Different versions of a program, including its dependencies, can be installed on the same operating system.
- Allows the working of more than one project that contains different dependencies or requires distinct versions of the same package.
- Creating separate environments helps avoid conflicts between packages.

*Virtual Machines (VM)*

- A software program emulating a physical computer hardware setup.
- Several operating systems can be operated on a single physical machine using VMs.
- Hardware resources are allocated to each VM using hypervisor, a software program.
- Essential when applications requiring different operating systems need to be run.

*Containers*

- Packaging format that bundles all the components an application needs to ease the process of sharing and distributing applications whilst maintaining isolation from each other.
- Containers, unlike VMs, exist within a single operating system and share its kernel and resources.
- An entire operating system environment is encapsulated. This enables the processing of applications tailored to specific operating systems without the need for a complete VM setup.

**Connecting R and Jupyter:**

Ensure that the environment is activated and type the following command on the command line to open R:

```
R
```

Next, to install IRkernel, use the command:

```
install.packages('IRkernel')
```

```
(bio) ubuntu@lsi:~$ R

R version 4.3.2 (2023-10-31) -- "Eye Holes"
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: x86_64-conda-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages('IRkernel')
--- Please select a CRAN mirror for use in this session ---
Secure CRAN mirrors

 1: 0-Cloud [https]
 2: Australia (Canberra) [https]
 3: Australia (Melbourne 1) [https]
 4: Australia (Melbourne 2) [https]
 5: Australia (Perth) [https]
 6: Austria [https]
 7: Belgium (Brussels) [https]
 8: Brazil (PR) [https]
 9: Brazil (RJ) [https]
10: Brazil (SP 1) [https]
11: Brazil (SP 2) [https]
12: Bulgaria [https]
13: Canada (MB) [https]
14: Canada (ON) [https]
15: Chile (Santiago) [https]
16: China (Beijing 2) [https]
17: China (Beijing 3) [https]
18: China (Hefei) [https]
19: China (Hong Kong) [https]
20: China (Guangzhou) [https]
21: China (Jinan) [https]
22: China (Lanzhou) [https]
23: China (Nanjing) [https]
```

From the list of CRAN mirrors, select the appropriate option based on the location and input the respective number.

```
Selection: 41
trying URL 'https://packages.othr.de/cran/src/contrib/IRkernel_1.3.2.tar.gz'
Content type 'text/plain' length 45172 bytes (44 KB)
==================================================
downloaded 44 KB

* installing *source* package 'IRkernel' ...
** package 'IRkernel' successfully unpacked and MD5 sums checked
** using staged installation
** R
** inst
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded from temporary location
** testing if installed package can be loaded from final location
** testing if installed package keeps a record of temporary installation path
* DONE (IRkernel)

The downloaded source packages are in
        '/tmp/RtmplOUIDe/downloaded_packages'
Updating HTML index of packages in '.Library'
Making 'packages.html' ... done
> q()
Save workspace image? [y/n/c]: n
```

Exit from R using:

```
q()
```

Input the following command on the command line to use R from a Jupyter notebook:

```
R -e "IRkernel::installspec()"
```

**Starting Jupyter server:**

To configure the Jupyter server, the command used is:

```
jupyter server --generate-config
```

The terminal window should display the following.

```
(bio) ubuntu@lsi:~$ jupyter server --generate-config
Overwrite /home/ubuntu/.jupyter/jupyter_server_config.py with default config? [y/N]y
Writing default config to: /home/ubuntu/.jupyter/jupyter_server_config.py
```

Enter the command:

```
jupyter server password
```

The following message to enter the password is displayed.

```
(bio) ubuntu@lsi:~$ jupyter server password
Enter password:
Verify password:
[JupyterPasswordApp] Wrote hashed password to /home/ubuntu/.jupyter/jupyter_server_config
.json
```

Enter password and confirm it.

Use the following command to start the server:

```
jupyter lab --no-browser --ip "*"
```

A range of print outs, like the following image, will be displayed. This indicates that the server is active, and the prompt will not appear in this window again.

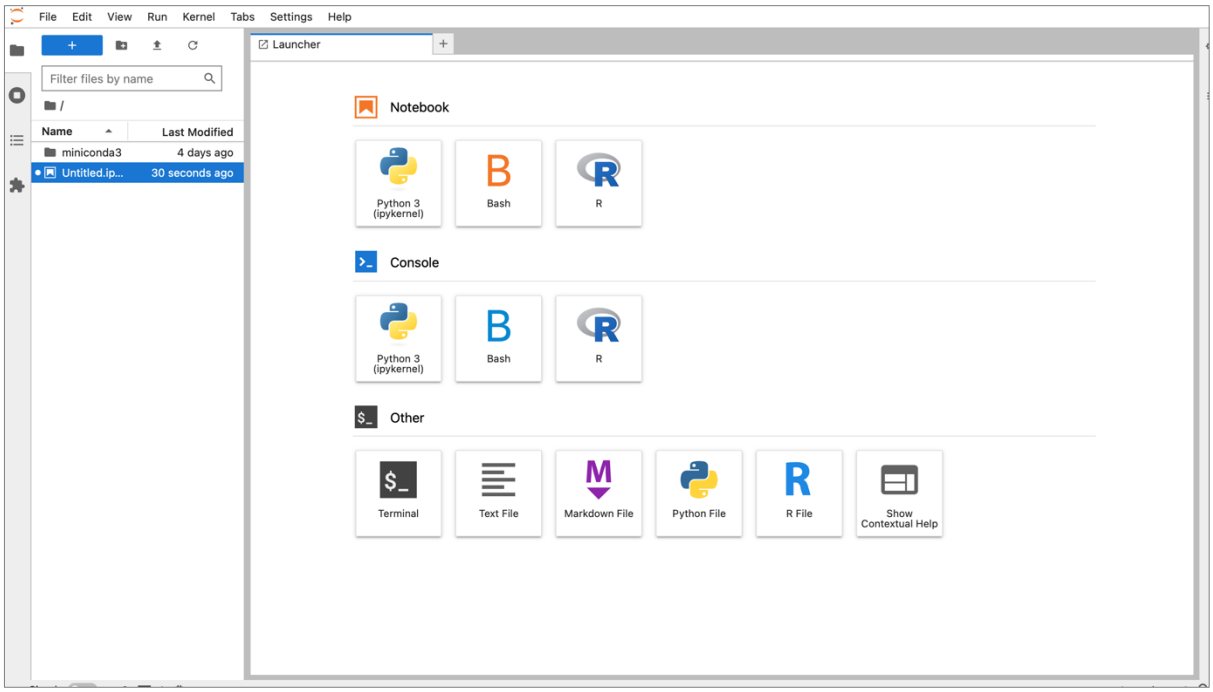
```
(bio) ubuntu@lsi:~$ jupyter lab --no-browser --ip "*"
[I 2023-11-07 11:48:47.134 ServerApp] Package jupyterlab took 0.0000s to import
[I 2023-11-07 11:48:47.155 ServerApp] Package jupyter_lsp took 0.0200s to import
[W 2023-11-07 11:48:47.155 ServerApp] A `_jupyter_server_extension_points` function was not found in jupyter_lsp.
 Instead, a `_jupyter_server_extension_paths` function was found and will be used for now. This function name wil
l be deprecated in future releases of Jupyter Server.
[I 2023-11-07 11:48:47.160 ServerApp] Package jupyter_server_terminals took 0.0054s to import
[I 2023-11-07 11:48:47.161 ServerApp] Package notebook_shim took 0.0000s to import
[W 2023-11-07 11:48:47.161 ServerApp] A `_jupyter_server_extension_points` function was not found in notebook_shi
m. Instead, a `_jupyter_server_extension_paths` function was found and will be used for now. This function name w
ill be deprecated in future releases of Jupyter Server.
[I 2023-11-07 11:48:47.161 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2023-11-07 11:48:47.164 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2023-11-07 11:48:47.168 ServerApp] jupyterlab | extension was successfully linked.
[I 2023-11-07 11:48:47.293 ServerApp] notebook_shim | extension was successfully linked.
[W 2023-11-07 11:48:47.392 ServerApp] WARNING: The Jupyter server is listening on all IP addresses and not using
encryption. This is not recommended.
[I 2023-11-07 11:48:47.393 ServerApp] notebook_shim | extension was successfully loaded.
[I 2023-11-07 11:48:47.394 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2023-11-07 11:48:47.395 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2023-11-07 11:48:47.397 LabApp] JupyterLab extension loaded from /home/ubuntu/miniconda3/envs/bio/lib/python3.
8/site-packages/jupyterlab
[I 2023-11-07 11:48:47.397 LabApp] JupyterLab application directory is /home/ubuntu/miniconda3/envs/bio/share/jup
yter/lab
[I 2023-11-07 11:48:47.397 LabApp] Extension Manager is 'pypi'.
[I 2023-11-07 11:48:47.400 ServerApp] jupyterlab | extension was successfully loaded.
[I 2023-11-07 11:48:47.400 ServerApp] The port 8888 is already in use, trying another port.
[I 2023-11-07 11:48:47.400 ServerApp] Serving notebooks from local directory: /home/ubuntu
[I 2023-11-07 11:48:47.400 ServerApp] Jupyter Server 2.9.1 is running at:
[I 2023-11-07 11:48:47.400 ServerApp] http://localhost:8889/lab
[I 2023-11-07 11:48:47.400 ServerApp]     http://127.0.0.1:8889/lab
[I 2023-11-07 11:48:47.400 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip
confirmation).
[I 2023-11-07 11:48:47.524 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-
server-nodejs, javascript-typescript-langserver, jedi-language-server, julia-language-server, pyright, python-lan
guage-server, python-lsp-server, r-languageserver, sql-language-server, texlab, typescript-language-server, unifi
ed-language-server, vscode-css-languageserver-bin, vscode-html-languageserver-bin, vscode-json-languageserver-bin
, yaml-language-server
```

To test if the Jupyter server is working, open a new terminal window. Establish another ssh connection using a statement similar to the normal ssh connection but with an additional -L option:

```
ssh ubuntu@10.70.1.105 -i /Users/mariam/Downloads/key_vm_36.txt -L
8888:localhost:8888
```

The additional request to local port 8888 is redirected, through a secure connection, to the IP i.e., the specified remote machine.

Open the browser and type “localhost:8888” and click enter. The browser connects to the Jupyter server and displays the lab as follows:




Since the Jupyter server runs in a terminal, the server will be stopped once the terminal is closed or the ssh connection is interrupted.

To test this, press CTRL+C to stop the Jupyter server. The Jupyter server is disconnected, and the lab no longer appears on the browser.

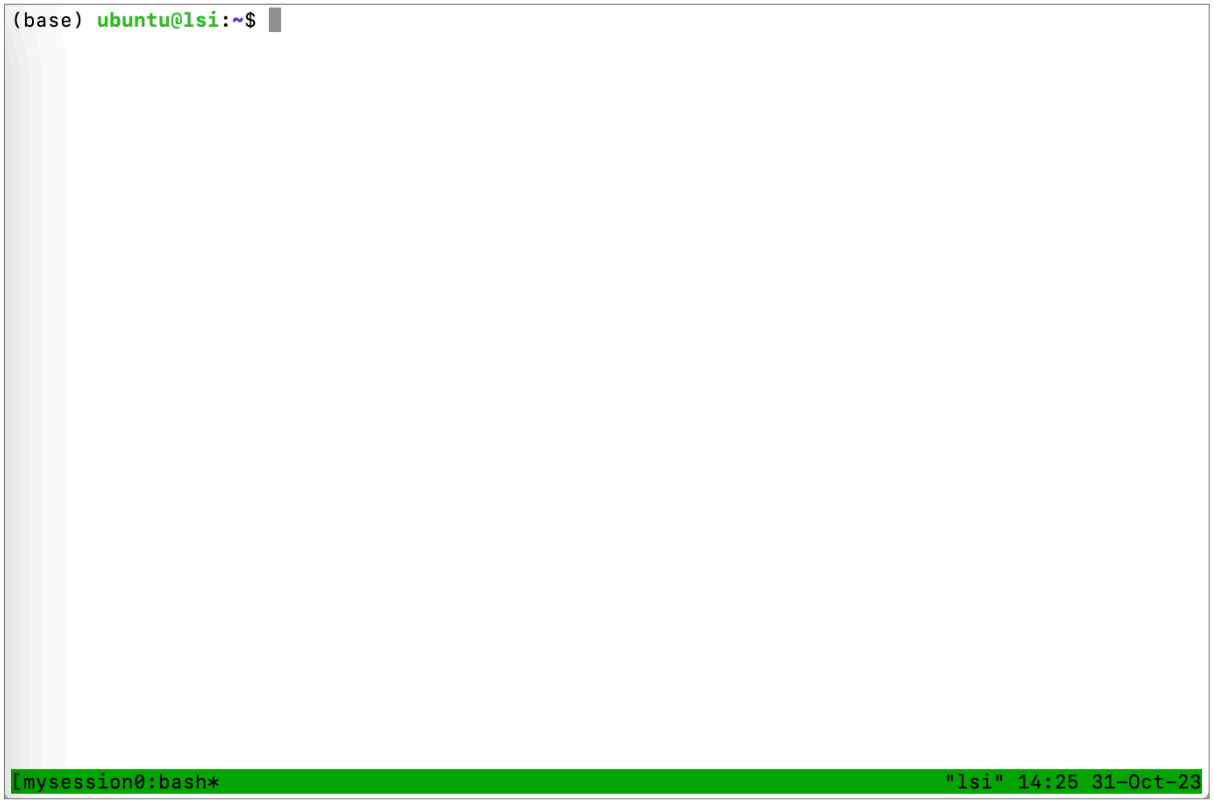
**tmux**

For long-running processes, such as the Jupyter server, tmux is a beneficial solution.

tmux is an open-source terminal multiplexer for Unix-like operating systems. It allows multiple terminal sessions to be accessed simultaneously in a single window. It is useful for running more than one command-line program simultaneously, or for working on different tasks in the same terminal window.

The following command creates a new session with the name ‘mysession’:

```
tmux new -s mysession
```




To detach from a session:

```
CTRL-B + D
```

To reattach to a session:

```
tmux attach -t [session-name]
```

To create a new or multiple windows:

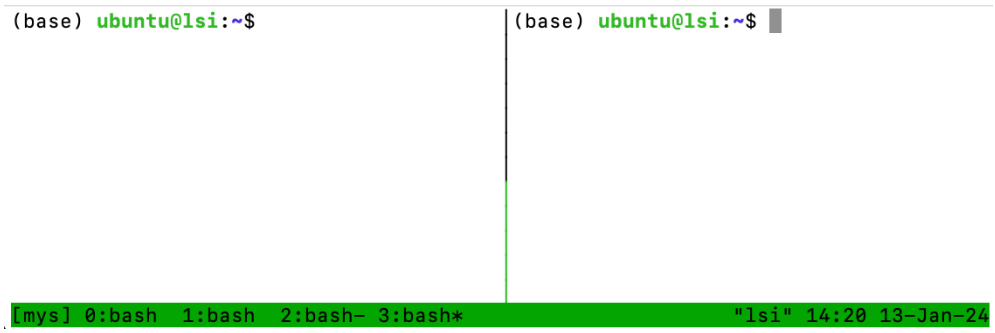
```
CTRL-B + C
```


To switch between windows:

```
CTRL-B + W
```

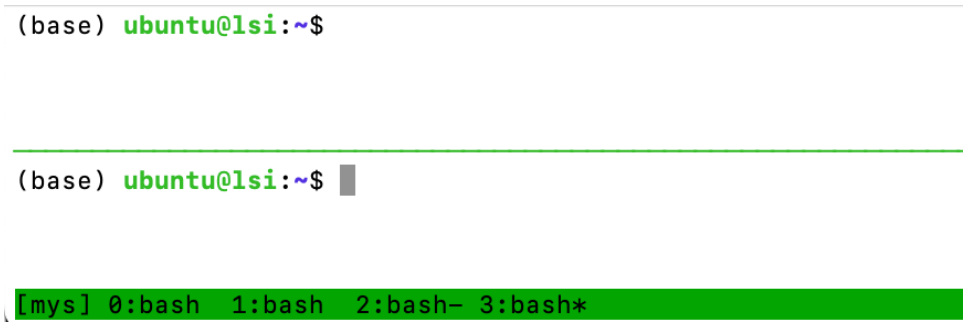
To split the window into horizontal panes:

```
CTRL-B + %
```




To split the window into vertical panes:

```
CTRL-B + “
```



To switch between panes:

```
CTRL-B + O
```

*Session*

In tmux, a session refers to a collection of independent windows sharing a common configuration that can be saved and restored. It permits the grouping of tasks together and each session has a unique name.

*Window*

A window in tmux refers to a separate virtual terminal within a session. Several programs can be run together, and work can be organized into independent workspaces. Each window has its own process space and title.

*Pane*

Pane represents a sub-division of a window. Panes in tmux compartmentalize workspaces as it divides the window into several smaller terminals which allows programs to be run simultaneously within the same window.

In general, a session represents a container for one or more windows. A window contains one or more panes.

To ensure that the Jupyter server runs constantly, create a new tmux session using:

```
tmux new -s jserver
```

Activate the ‘bio’ environment in the tmux session:

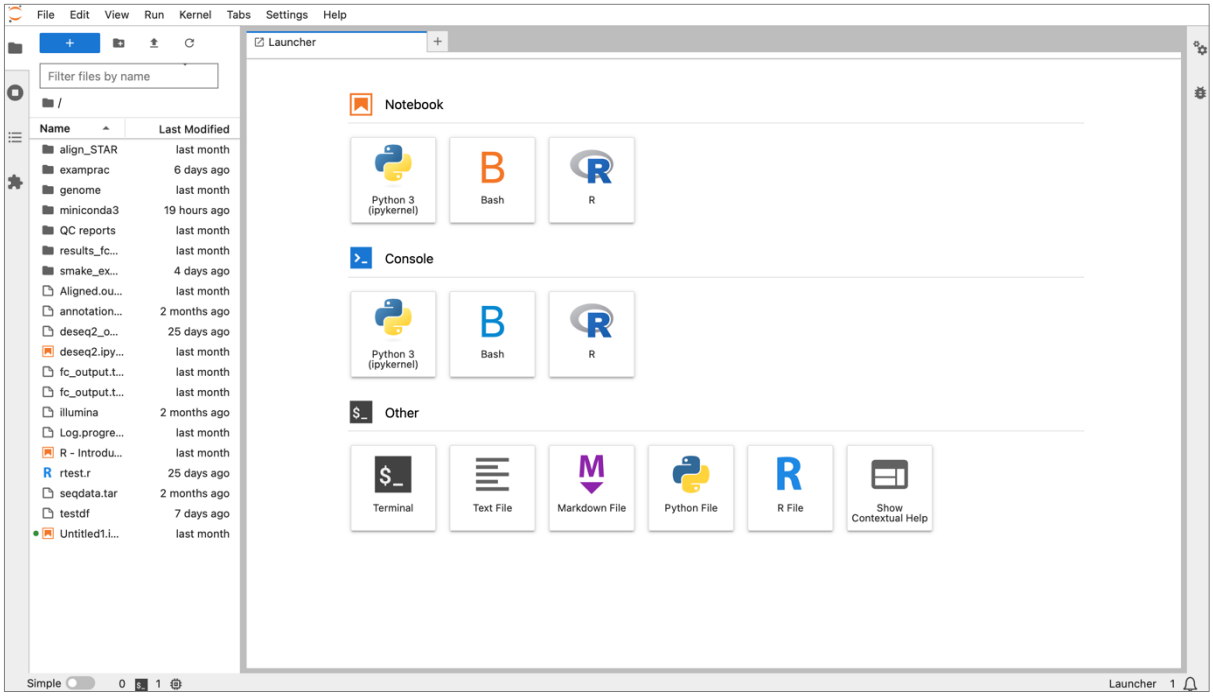
```
conda activate bio
```

With the command used previously, start the Jupyter server in the tmux session:

```
jupyter lab --no-browser --ip "*"
```

Detach from the session. Open the browser and connect to the Jupyter server.

From the screen displayed, create a new notebook with any kernel (Python, Bash, or R) by clicking on the options under ‘Notebook’.




To change the first cell to a markdown cell, click on the Code option on the toolbar of the notebook and switch to Markdown.

Different levels of headlines can be displayed using the following symbols:

| MARKDOWN | RENDERED OUTPUT |
| --- | --- |
| # Heading level 1 | Heading level 1 |
| ## Heading level 2 | Heading level 2 |
| ### Heading level 3 | Heading level 3 |
| #### Heading level 4 | Heading level 4 |
| ##### Heading level 5 | Heading level 5 |
| ###### Heading level 6 | Heading level 6 |

To make text bold, italic, or bold and italic:

| MARKDOWN | RENDERED OUTPUT |
| --- | --- |
| This text is **bold**. | This text is **bold**. |
| This text is __bold__. | This text is **bold**. |
| This text is *italicized*. | This text is *italicized*. |

| | |
|---|---|
| This text is _italicized_. | This text is *italicized*. |
| This text is ***bold and italicized***. | This text is ***bold and italicized***. |
| This text is __*bold and italicized*__. | This text is ***bold and italicized***. |

To make numbered lists, add line items with numbers followed by periods. To make unnumbered lists, add dashes (-), plus signs (+), or asterisks (*) before the line items:

| MARKDOWN | RENDERED OUTPUT |
|---|---|
| 1. This is the first item.<br>2. This is the second item.<br>3. This is the third item. | 1. This is the first item.<br>2. This is the second item.<br>3. This is the third item. |
| **1. This is the first item.**<br>**1. This is an intended item.**<br>2. This is the second item. | 1. This is the first item.<br>A. This is an intended item.<br>2. This is the second item. |
| *** This is the first item.**<br>*** This is the second item.**<br>* This is the third item. | • This is the first item.<br>• This is the second item.<br>• This is the third item. |
| **+ This is the first item.**<br>**+ This is the second item.**<br>+ This is the third item. | • This is the first item.<br>• This is the second item.<br>• This is the third item. |

To create blockquote(s), add > before the start of a paragraph, for instance:

---

**Markdown:**

```
> I am a student at the Deggendorf Institute of Technology.
```

**Rendered Output:**

> I am a student at the Deggendorf Institute of Technology.

---

**Markdown:**

```
> I am a student at the Deggendorf Institute of Technology.
>
> My major is Life Science Informatics.
```

**Rendered Output:**

> I am a student at the Deggendorf Institute of Technology.
>
> My major is Life Science Informatics.

---

**Markdown:**

```
> I am a student at the Deggendorf Institute of Technology.
>
>> My major is Life Science Informatics.
```

**Rendered Output:**

> I am a student at the Deggendorf Institute of Technology.
>
> > My major is Life Science Informatics.

---

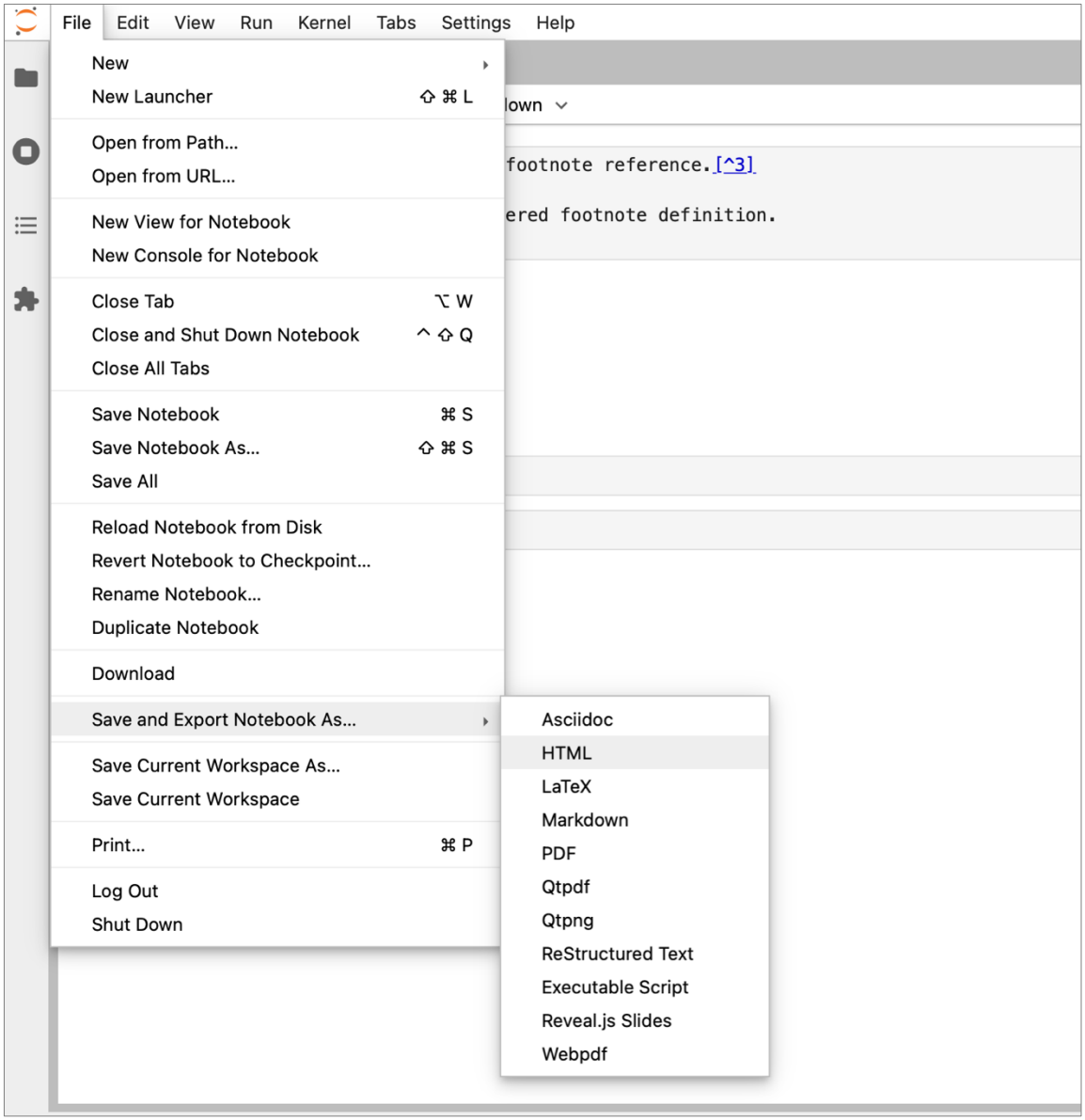
*Extended Markdown Syntax*

Jupyterlab can render tables, create footnotes, and display highlighted code.

| | MARKDOWN | RENDERED OUTPUT |
|---|---|---|
| **TABLE** | `\| Header \| Content \|`<br>`\| ---------- \| -------- \|`<br>`\| Title \| Text \|`<br>`\| Title \| Text \|` | Header Content<br>Title Text<br>Title Text |
| **FOOTNOTE** | - This is a manually-numered footnote reference.[^3]<br><br>[^3]: This is a manually-numbered footnote definition. | • This is a manually-numbered footnote reference.[3]<br>[3] This is a manually-numbered footnote definition. |
| **SYNTAX HIGHLIGHTED CODE** | ```` ```python<br>{<br>"firstName": "Mariam",<br>"lastName": "Zaidi",<br>"age": 23<br>}<br>``` ```` | `{`<br>`"firstName": "Mariam",`<br>`"lastName": "Zaidi",`<br>`"age": 23`<br>`}` |

To export a Jupyter Notebook to an HTML file:

- Open the notebook in Jupyterlab.
- Click File button in the toolbar.
- Click ‘Save and Export Notebook As…’ and choose HTML from the Format menu.


File
Edit
View
Run
Kernel
Tabs
Settings
Help
New
New Launcher ⇧ ⌘ L
Open from Path...
Open from URL...
New View for Notebook
New Console for Notebook
Close Tab ⌥ W
Close and Shut Down Notebook ^ ⇧ Q
Close All Tabs
Save Notebook ⌘ S
Save Notebook As... ⇧ ⌘ S
Save All
Reload Notebook from Disk
Revert Notebook to Checkpoint...
Rename Notebook...
Duplicate Notebook
Download
Save and Export Notebook As...
Save Current Workspace As...
Save Current Workspace
Print... ⌘ P
Log Out
Shut Down
Asciidoc
HTML
LaTeX
Markdown
PDF
Qtpdf
Qtpng
ReStructured Text
Executable Script
Reveal.js Slides
Webpdf
lown
footnote reference.[^3]
ered footnote definition.

## RNA-Seq Analysis

RNA-seq analysis uses next-generation sequencing technique to measure the abundance of different RNA molecules in a biological sample. Researchers can acquire information corresponding to gene expression patterns and regulatory mechanisms by sequencing RNA transcripts. RNA-seq analysis is an indispensable tool in several biological fields such as, disease biology, drug discovery, and gene expression profiling.

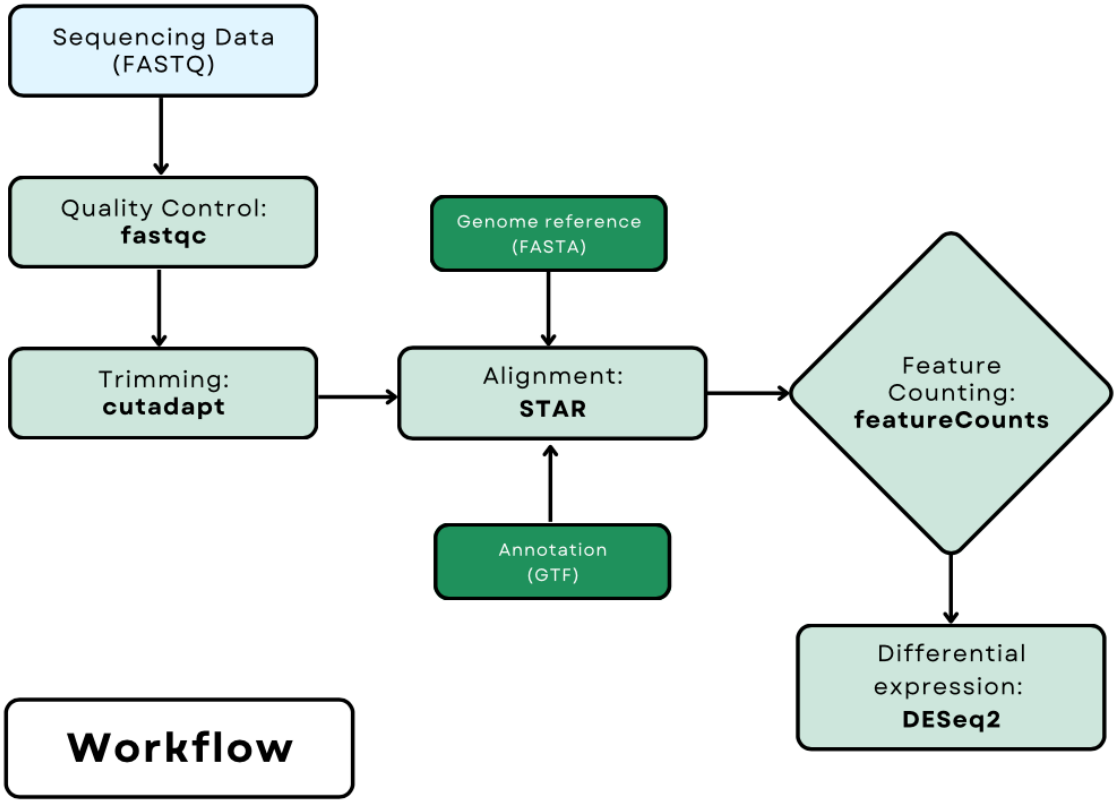
Some sequencing technologies include:

- Illumina
- Oxford Nanopore
- PacBio
- Thermo Fischer Scientific
- Roche

Some next-generation sequencing methods include:

- Whole-genome sequencing (WGS)
- Exome sequencing (Exom-seq)
- Methylation sequencing (Methyl-seq)
- RNA sequencing (RNA-seq)
- ChIP-seq

A summarized version of the protocol followed to process and analyze NGS data in this project is illustrated below.




**File Formats**

*FASTA File Format*

The FASTA file format stores biological sequences such as nucleotides or proteins. It is text-based and indicates each sequence record with a “>” symbol followed by a sequence identifier and optional comments.

An example:

```
>my_sequence
ATGCGGCTTAGCTACGTCGATGCTAAGCTACTTA
>my_protein
MALYLAKGLNALAGLYRDCQEGHIKLTWYAKGMA
>another_sequence
AAUAGGUUUAUCAUGUAUACAGCGGACUAUCAUA
```

*FASTQ File Format*

Similar to FASTA format but usually displays DNA sequences. It contains the nucleotide sequence obtained in addition to the quality score for each nucleotide. A FASTQ file contains a sequence header followed by four entries for each sequence record:

1. Indicates sequence identifier and additional information led by the "@" symbol.
2. Nucleotide sequence.
3. Separator line containing a "+" sign.
4. Quality scores for each base call encoded as a Phred score in ASCII format. Indicates how precisely each nucleotide was measured.

Phred scores are a logarithmic scale representing the probability of a base call being incorrect. A lower score indicates a higher probability of error.

*GTF (Gene Transfer Format) File Format*

It a file format based on General Feature Format (GFF) describing features on genomic sequences. The tab-delimited text format consists of some addition conventions specific to gene information. It consists of one line per feature, each containing nine columns of data. An interesting feature of the GTF is that it allows the generation of a set of non-overlapping annotations. It can store and represent characteristics in a genomic sequence and is commonly used for storing genomic annotations (*Genome Browser FAQ*, n.d., OpenAI, 2024).

**Obtaining data:**

Wget is a command-line application that allows files to be downloaded from the web. Files of several types (images, software, web pages etc.) or multiple files can be downloaded simultaneously.

Download the files onto the remote machine using the wget command as shown below:

```
wget https://nextcloud.th-deg.de/s/BZdNP4BQqRLae7L/download/genome.fa
wget https://nextcloud.th-deg.de/s/DFEkkW8aArwFPWN/download/annotation.gtf
wget https://nextcloud.th-deg.de/s/zrjyWijeAjekRr7/download/illumina_adapter.fa
```

```
wget https://nextcloud.th-deg.de/s/jsbPzAmNfYRBgpk/download/seqdata.tar
```

Once downloaded successfully, the terminal window should look like this for each link:

```
(bio) ubuntu@lsi:~$ wget https://nextcloud.th-deg.de/s/DFEkkW8aArwFPWN/download/annotation.
gtf
--2023-11-07 14:47:07--  https://nextcloud.th-deg.de/s/DFEkkW8aArwFPWN/download/annotation.
gtf
Resolving nextcloud.th-deg.de (nextcloud.th-deg.de)... 10.3.3.32
Connecting to nextcloud.th-deg.de (nextcloud.th-deg.de)|10.3.3.32|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 30712117 (29M) [application/octet-stream]
Saving to: 'annotation.gtf'

annotation.gtf      100%[===================>]  29.29M  --.-KB/s    in 0.08s

2023-11-07 14:47:07 (374 MB/s) - 'annotation.gtf' saved [30712117/30712117]

(bio) ubuntu@lsi:~$ wget https://nextcloud.th-deg.de/s/jsbPzAmNfYRBgpk/download/seqdata.tar
--2023-11-07 14:47:38--  https://nextcloud.th-deg.de/s/jsbPzAmNfYRBgpk/download/seqdata.tar
Resolving nextcloud.th-deg.de (nextcloud.th-deg.de)... 10.3.3.32
Connecting to nextcloud.th-deg.de (nextcloud.th-deg.de)|10.3.3.32|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 116601344 (111M) [application/x-tar]
Saving to: 'seqdata.tar'

seqdata.tar         100%[===================>] 111.20M   213MB/s    in 0.5s

2023-11-07 14:47:38 (213 MB/s) - 'seqdata.tar' saved [116601344/116601344]
```

Make a new directory using mkdir and name it ‘new’. Move ‘seqdata.tar’ into the directory using mv as shown below. Change working directory to the ‘new’ directory.

```
(bio) ubuntu@lsi:~$ mkdir new
(bio) ubuntu@lsi:~$ ls
Untitled.ipynb   annotation.gtf  illumina    new
Untitled1.ipynb  genome.fa       miniconda3  seqdata.tar
(bio) ubuntu@lsi:~$ mv seqdata.tar new
(bio) ubuntu@lsi:~$ ls new
seqdata.tar
(bio) ubuntu@lsi:~$ cd new/
```

The Tar tool is utilized in the creation and management of archives. It can be used to create, extract, and list archive files.

To extract files from seqdata.tar to the current directory, use the following command:

```
tar -xvf seqdata.tar
```

-x extracts files from the archive
-v prints verbose output during extraction process
-f specifies the archive file to extract from

The output should display the 12 files ending with fastq.gz as follows:

```
(bio) ubuntu@lsi:~/new$ tar xvf seqdata.tar
G1_rep1_read1.fastq.gz
G1_rep1_read2.fastq.gz
G1_rep2_read1.fastq.gz
G1_rep2_read2.fastq.gz
G1_rep3_read1.fastq.gz
G1_rep3_read2.fastq.gz
G2_rep1_read1.fastq.gz
G2_rep1_read2.fastq.gz
G2_rep2_read1.fastq.gz
G2_rep2_read2.fastq.gz
G2_rep3_read1.fastq.gz
G2_rep3_read2.fastq.gz
```

**Quality control using fastqc:**

Create a new directory called 'QC reports'.

Run fastqc on each fastq.gz file using the command combined with the '*' wildcard:

```
fastqc *.gz
```

```
(base) ubuntu@lsi:~/QC reports$ fastqc *.gz
Started analysis of G1_rep1_read1.fastq.gz
Approx 5% complete for G1_rep1_read1.fastq.gz
Approx 10% complete for G1_rep1_read1.fastq.gz
Approx 15% complete for G1_rep1_read1.fastq.gz
Approx 20% complete for G1_rep1_read1.fastq.gz
Approx 25% complete for G1_rep1_read1.fastq.gz
Approx 30% complete for G1_rep1_read1.fastq.gz
Approx 35% complete for G1_rep1_read1.fastq.gz
Approx 40% complete for G1_rep1_read1.fastq.gz
Approx 45% complete for G1_rep1_read1.fastq.gz
Approx 50% complete for G1_rep1_read1.fastq.gz
Approx 55% complete for G1_rep1_read1.fastq.gz
Approx 60% complete for G1_rep1_read1.fastq.gz
Approx 65% complete for G1_rep1_read1.fastq.gz
Approx 70% complete for G1_rep1_read1.fastq.gz
Approx 75% complete for G1_rep1_read1.fastq.gz
Approx 80% complete for G1_rep1_read1.fastq.gz
Approx 85% complete for G1_rep1_read1.fastq.gz
Approx 90% complete for G1_rep1_read1.fastq.gz
Approx 95% complete for G1_rep1_read1.fastq.gz
Analysis complete for G1_rep1_read1.fastq.gz
Started analysis of G1_rep1_read2.fastq.gz
```

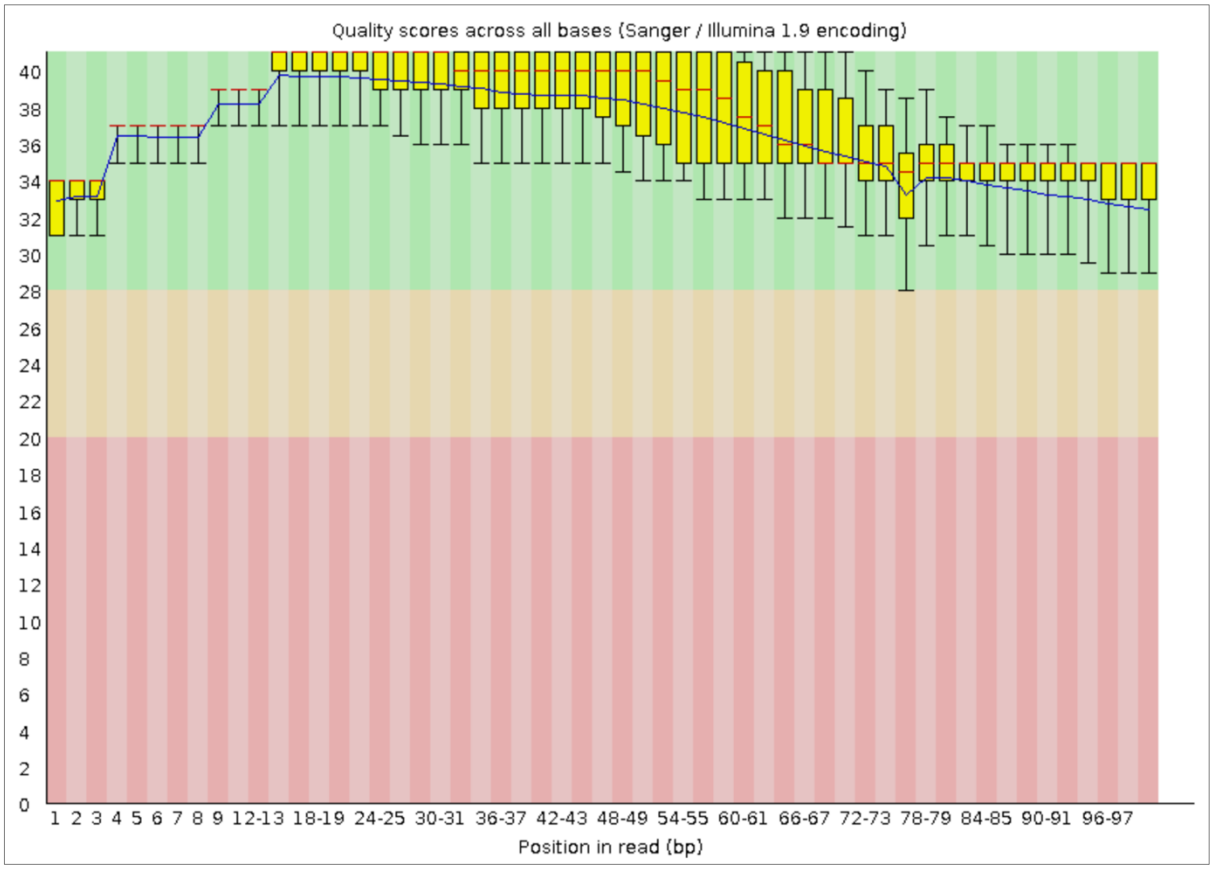
**Note**: The zcat command can be used to uncompress files on the command line. Zcat takes the compressed files and concatenates it.

```
zcat filename.fastq.gz
```

Quality control (QC) on raw sequencing data is critical to ensure that the data quality remains high and reliable. It helps identify and discard low-quality reads to elevate the accuracy of downstream analyses. QC processes can also remove PCR duplicates, adapter contaminations, and alignment errors.

The **Per Base Sequence Quality** section in the FastQC report displays information regarding the distribution of base quality scores (Phred scores) of each base in the sequence reads. This section establishes a quality filter to check if all the nucleotides are reliable or if any filtration is required.

The scores are plotted as a boxplot illustrating median, interquartile range, 1st and 3rd quartiles, and minimum and maximum values as depicted in the example below:



**Adapter trimming and removing low quality reads:**

The cutadapt tool analyzes each read and removes sections that are not required. For instance, it can be used to trim adapters or primers from sequencing reads. It is also used to trim reads to a desired length and remove low-quality bases.

Install cutadapt via conda using the command:

```
conda install -y fastqc cutadapt
```

Using the following commands, remove the first 10 nucleotides, keep reads with a length of >= 25 nucleotides, and remove nucleotides from 3' end of the reads with a Q-value < 30 for each read:

```
cutadapt --cut 10 --minimum-length 25 -q 30 -o G1_rep1_read1_trimmed.fastq.gz -p
G1_rep1_read2_trimmed.fastq.gz G1_rep1_read1.fastq.gz G1_rep1_read2.fastq.gz

cutadapt --cut 10 --minimum-length 25 -q 30 -o G1_rep2_read1_trimmed.fastq.gz -p
G1_rep2_read2_trimmed.fastq.gz G1_rep2_read1.fastq.gz G1_rep2_read2.fastq.gz

cutadapt --cut 10 --minimum-length 25 -q 30 -o G1_rep3_read1_trimmed.fastq.gz -p
G1_rep3_read2_trimmed.fastq.gz G1_rep3_read1.fastq.gz G1_rep3_read2.fastq.gz

cutadapt --cut 10 --minimum-length 25 -q 30 -o G2_rep1_read1_trimmed.fastq.gz -p
G2_rep1_read2_trimmed.fastq.gz G2_rep1_read1.fastq.gz G2_rep1_read2.fastq.gz

cutadapt --cut 10 --minimum-length 25 -q 30 -o G2_rep2_read1_trimmed.fastq.gz -p
G2_rep2_read2_trimmed.fastq.gz G2_rep2_read1.fastq.gz G2_rep2_read2.fastq.gz

cutadapt --cut 10 --minimum-length 25 -q 30 -o G2_rep3_read1_trimmed.fastq.gz -p
G2_rep3_read2_trimmed.fastq.gz G2_rep3_read1.fastq.gz G2_rep3_read2.fastq.gz
```

Run fastqc on the trimmed reads using the command:

```
fastqc *trimmed.fastq.gz
```

Create a new directory named 'trimmed' and move all the trimmed files using the mv command or directly through Jupyterlab to ease the process.

The filtered reports are displayed as shown below in the 'trimmed' directory:

```
(base) ubuntu@lsi:~/QC reports$ ls trimmed
G1_rep1_read1_trimmed.fastq.gz      G2_rep1_read1_trimmed.fastq.gz
G1_rep1_read1_trimmed_fastqc.html   G2_rep1_read1_trimmed_fastqc.html
G1_rep1_read1_trimmed_fastqc.zip    G2_rep1_read1_trimmed_fastqc.zip
G1_rep1_read2_trimmed.fastq.gz      G2_rep1_read2_trimmed.fastq.gz
G1_rep1_read2_trimmed_fastqc.html   G2_rep1_read2_trimmed_fastqc.html
G1_rep1_read2_trimmed_fastqc.zip    G2_rep1_read2_trimmed_fastqc.zip
G1_rep2_read1_trimmed.fastq.gz      G2_rep2_read1_trimmed.fastq.gz
G1_rep2_read1_trimmed_fastqc.html   G2_rep2_read1_trimmed_fastqc.html
G1_rep2_read1_trimmed_fastqc.zip    G2_rep2_read1_trimmed_fastqc.zip
G1_rep2_read2_trimmed.fastq.gz      G2_rep2_read2_trimmed.fastq.gz
G1_rep2_read2_trimmed_fastqc.html   G2_rep2_read2_trimmed_fastqc.html
G1_rep2_read2_trimmed_fastqc.zip    G2_rep2_read2_trimmed_fastqc.zip
G1_rep3_read1_trimmed.fastq.gz      G2_rep3_read1_trimmed.fastq.gz
G1_rep3_read1_trimmed_fastqc.html   G2_rep3_read1_trimmed_fastqc.html
G1_rep3_read1_trimmed_fastqc.zip    G2_rep3_read1_trimmed_fastqc.zip
G1_rep3_read2_trimmed.fastq.gz      G2_rep3_read2_trimmed.fastq.gz
G1_rep3_read2_trimmed_fastqc.html   G2_rep3_read2_trimmed_fastqc.html
G1_rep3_read2_trimmed_fastqc.zip    G2_rep3_read2_trimmed_fastqc.zip
```

After applying cutadapt, the filtered reads are shorter than the original reads. Additionally, the percentage of bases with a quality score of at least 30 is higher comparatively and the percentage of ambiguous or unknown bases is lower. The median quality score is also higher following trimming.

**Aligning reads using STAR:**

The STAR (Spliced Transcripts Alignment to a Reference) program maps the trimmed and quality-controlled reads to the human genome (transcriptome).

Download the current STAR manual:

(https://github.com/alexdobin/STAR/blob/master/doc/STARmanual.pdf)

Install STAR in the 'bio' environment through:

```
conda install star
```

Section 1.2 onwards, in the manual, a detailed description of the workflow is provided, including the STAR command line format and options, to solve the upcoming tasks.

First, create a directory 'genome' and move the 'genome.fa' file into the new directory.

Use the --genomeSAindexNbases11 option with the specified options to create an index for STAR using the fasta file and the corresponding 'annotation.gtf'.

To map the paired-read ends for each sample, use the options --outFileNamePrefix /output/path/prefix and --readFilesCommand zcat.

The final command is:

```
STAR --runThreadN 2 --runMode genomeGenerate --genomeDir genome --genomeFastaFiles
genome/genome.fa --sjdbGTFfile annotation.gtf --genomeSAindexNbases 11
```

Once installation is complete, map paired end reads for the samples using the following commands:

```
STAR --runThreadN 2 --genomeDir genome --readFilesIn /home/ubuntu/'QC
reports'/trimmed/G1_rep1_read1_trimmed.fastq.gz /home/ubuntu/'QC
reports'/trimmed/G1_rep1_read2_trimmed.fastq.gz --outFileNamePrefix output/G1_rep1
--readFilesCommand zcat
```

```
STAR --runThreadN 2 --genomeDir genome --readFilesIn /home/ubuntu/'QC
reports'/trimmed/G1_rep2_read1_trimmed.fastq.gz /home/ubuntu/'QC
reports'/trimmed/G1_rep2_read2_trimmed.fastq.gz --outFileNamePrefix output/G1_rep2
--readFilesCommand zcat
```

```
STAR --runThreadN 2 --genomeDir genome --readFilesIn /home/ubuntu/'QC
reports'/trimmed/G1_rep3_read1_trimmed.fastq.gz /home/ubuntu/'QC
reports'/trimmed/G1_rep3_read2_trimmed.fastq.gz --outFileNamePrefix output/G1_rep3
--readFilesCommand zcat
```

```
STAR --runThreadN 2 --genomeDir genome --readFilesIn /home/ubuntu/'QC
reports'/trimmed/G2_rep1_read1_trimmed.fastq.gz /home/ubuntu/'QC
reports'/trimmed/G2_rep1_read2_trimmed.fastq.gz --outFileNamePrefix output/G2_rep1
--readFilesCommand zcat
```

```
STAR --runThreadN 2 --genomeDir genome --readFilesIn /home/ubuntu/'QC
reports'/trimmed/G2_rep2_read1_trimmed.fastq.gz /home/ubuntu/'QC
reports'/trimmed/G2_rep2_read2_trimmed.fastq.gz --outFileNamePrefix output/G2_rep2
--readFilesCommand zcat
```

```
STAR --runThreadN 2 --genomeDir genome --readFilesIn /home/ubuntu/'QC
reports'/trimmed/G2_rep3_read1_trimmed.fastq.gz /home/ubuntu/'QC
reports'/trimmed/G2_rep3_read2_trimmed.fastq.gz --outFileNamePrefix output/G2_rep3
--readFilesCommand zcat
```

The result is as shown below:

```
(base) ubuntu@lsi:~$ ls align_STAR/
G1_rep1Aligned.out.sam   G1_rep3Aligned.out.sam   G2_rep2Aligned.out.sam
G1_rep1Log.final.out     G1_rep3Log.final.out     G2_rep2Log.final.out
G1_rep1Log.out           G1_rep3Log.out           G2_rep2Log.out
G1_rep1Log.progress.out  G1_rep3Log.progress.out  G2_rep2Log.progress.out
G1_rep1SJ.out.tab        G1_rep3SJ.out.tab        G2_rep2SJ.out.tab
G1_rep2Aligned.out.sam   G2_rep1Aligned.out.sam   G2_rep3Aligned.out.sam
G1_rep2Log.final.out     G2_rep1Log.final.out     G2_rep3Log.final.out
G1_rep2Log.out           G2_rep1Log.out           G2_rep3Log.out
G1_rep2Log.progress.out  G2_rep1Log.progress.out  G2_rep3Log.progress.out
G1_rep2SJ.out.tab        G2_rep1SJ.out.tab        G2_rep3SJ.out.tab
```

**Feature counting:**

The featureCounts tool can be used to count the number of mapped reads per gene. The tool is a part of the subread package.

Install the subread package through conda using:

```
conda install -y fastqc subread
```

Type featureCounts into the command line to verify its installation. It should display the help page as shown:

```
(bio2) ubuntu@lsi:~$ featureCounts

Version 2.0.6

Usage: featureCounts [options] -a <annotation_file> -o <output_file> input_file1 [input_file
2] ...

## Mandatory arguments:

  -a <string>         Name of an annotation file. GTF/GFF format by default. See
                      -F option for more format information. Inbuilt annotations
                      (SAF format) is available in 'annotation' directory of the
                      package. Gzipped file is also accepted.

  -o <string>         Name of output file including read counts. A separate file
                      including summary statistics of counting results is also
                      included in the output ('<string>.summary'). Both files
                      are in tab delimited format.

  input_file1 [input_file2] ...   A list of SAM or BAM format files. They can be
                      either name or location sorted. If no files provided,
                      <stdin> input is expected. Location-sorted paired-end reads
                      are automatically sorted by read names.
```

To count reads per gene, apply the featureCounts tool to the STAR alignment output files (out.sam) and ensure the following:

```
featureCounts -p --countReadPairs -a annotation.gtf -g gene_name -s 2 -o
fc_output.txt align_STAR/G1_rep1Aligned.out.sam align_STAR/G1_rep2Aligned.out.sam
align_STAR/G1_rep3Aligned.out.sam align_STAR/G2_rep1Aligned.out.sam
align_STAR/G2_rep2Aligned.out.sam align_STAR/G2_rep3Aligned.out.sam
```

-p option specifies that the data are paired-end reads,
-g option specifies that 'gene_name' from attributes list of the annotation.gtf should be used,
-s 2 specifies that the dataset is reversely stranded.

The process should run as shown:

```
(bio2) ubuntu@lsi:~$ featureCounts -p --countReadPairs -a annotation.gtf -g gene_name -s 2 -o fc_output.txt
 align_STAR/G1_rep1Aligned.out.sam align_STAR/G1_rep2Aligned.out.sam align_STAR/G1_rep3Aligned.out.sam alig
n_STAR/G2_rep1Aligned.out.sam align_STAR/G2_rep2Aligned.out.sam align_STAR/G2_rep3Aligned.out.sam

        ==========     _____ _    _ ____  _____  ______          _____
        =====         / ____| |  | |  _ \|  __ \|  ____|   /\   |  __ \
          =====      | (___ | |  | | |_) | |__) | |__     /  \  | |  | |
            ====      \___ \| |  | |  _ <|  _  /|  __|   / /\ \ | |  | |
                ====  ____) | |__| | |_) | | \ \| |____ / ____ \| |__| |
        ==========   |_____/ \____/|____/|_|  \_\______/_/    \_\_____/
          v2.0.6

//========================== featureCounts setting ===========================\\
||                                                                            ||
||             Input files : 6 SAM files                                      ||
||                                                                            ||
||                           G1_rep1Aligned.out.sam                           ||
||                           G1_rep2Aligned.out.sam                           ||
||                           G1_rep3Aligned.out.sam                           ||
||                           G2_rep1Aligned.out.sam                           ||
||                           G2_rep2Aligned.out.sam                           ||
||                           G2_rep3Aligned.out.sam                           ||
||                                                                            ||
||             Output file : fc_output.txt                                    ||
||                 Summary : fc_output.txt.summary                            ||
||              Paired-end : yes                                              ||
||         Count read pairs : yes                                             ||
||               Annotation : annotation.gtf (GTF)                            ||
||       Dir for temp files : ./                                              ||
||                                                                            ||
||                  Threads : 1                                               ||
||                    Level : meta-feature level                              ||
||       Multimapping reads : not counted                                     ||
|| Multi-overlapping reads : not counted                                      ||
||    Min overlapping bases : 1                                               ||
||                                                                            ||
\\============================================================================//
```

**Differential expression analysis:**

R is a free and open-source programming language and environment popularly utilized for statistical computing, data analysis, and visualization. It provides a variety of techniques and can be extended easily through packages. The combination of Python's interpretability and versatility coupled with the analytical capability of R, enhances the resourcefulness of data science.

The R package, DESeq2, will be used to analyze high-throughput data and detect significant differentially expressed features within the reads.

Run R using the command:

```
R
```

Create rtest.r file which contains the following commands. The file takes the featureCounts output and deseq2_output.txt as input. This is followed by the DESeq2 analysis, and the result is saved to the output file.

```
 GNU nano 6.2                                          rtest.r
library(DESeq2)

args = commandArgs(trailingOnly=T)
print(args)
countdata <- read.table(args[1], header=T, stringsAsFactors=F)
genenames <- countdata$Geneid
countdata <- countdata[, 7:ncol(countdata)]
colnames(countdata) <- c(paste("G1_rep_",1:3,sep=""), paste("G2_rep_",1:3,sep="")) #rename columns
countdata <- as.matrix(countdata)
rownames(countdata) <- genenames

#Get a boolean vector to remove all ERCC entries
sel <- sapply(rownames(countdata), function(x){ if(substr(x, 1,5)=="ERCC-"){return(FALSE)}else{return(TRUE)} })
countdata <- countdata[sel, ]

coldata <- data.frame("condition"=as.factor(c(rep("cancer", 3), rep("ref", 3))), row.names=colnames(countdata))

dds <- DESeqDataSetFromMatrix(countData = countdata,
                              colData = coldata,
                              design = ~ condition)

dds <- DESeq(dds)
res <- results(dds)
write.table(res, file=args[2], sep="\t", row.names=T, quote=F)
```

The functions commandArgs() and write.table() read the provided command line arguments and write the matrix to deseq2_output.txt, respectively.

Execute rtest.r via:

```
Rscript rtest.r /home/ubuntu/results_fcounts/featureCounts_output.txt /home/ubuntu/
deseq2_output.txt
```

## Workflow Manager

Workflow managers serve as software solutions designed to automate, streamline, and enhance the efficiency of diverse processes and tasks in an organization. In response to growing volume and intricacy of data, workflow managers address the need for scalability and reproducibility in analyses. Their primary functions include simplifying pipeline development, optimizing resource utilization, managing software installation and versions, and facilitating operation on diverse compute platforms (Wratten et al., 2021).

Snakemake, an extension of Python, is popularly used to create and execute data analysis pipelines. It simplifies the process of specifying rules, handling input and output files, and managing the flow of data between computational tasks (Snakemake — Snakemake 6.6.0 Documentation, n.d.; OpenAI, 2024)

To install Snakemake, use the following command in the 'bio' environment:

```
conda install bioconda::snakemake
```

Successful installation should appear as shown below:

```
(bio) ubuntu@lsi:~$ conda install bioconda::snakemake
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 23.9.0
  latest version: 23.11.0

Please update conda by running

    $ conda update -n base -c conda-forge conda

Or to minimize the number of packages updated during conda update use

     conda install conda=23.11.0



# All requested packages already installed.
```

Build a Snakemake workflow based on the commands used throughout the course with the outcome being the featureCounts output.

Open a new file named ‘seq’ in the editor using the command:

```
nano seq
```

The ‘seq’ file will contain the Snakemake workflow as depicted below:

```
SAMPLES = ["G1_rep1", "G1_rep2", "G1_rep3", "G2_rep1", "G2_rep2", "G2_rep3"]

rule all:
        input:
                "../fcount_output.txt"

rule quality_check:
        input:
                "{sample}_read1.fastq.gz"
        output:
                "../qualitycheck/{sample}_read1_fastqc.html"
        shell:
                "fastqc -o ../qualitycheck {input}"

rule adapter_trimming:
        input:
                fileA="{sample}_read1.fastq.gz",
                fileB="{sample}_read2.fastq.gz"
        output:
                outA="../trimmedresults/{sample}_read1_trimmed.fastq.gz",
                outB="../trimmedresults/{sample}_read2_trimmed.fastq.gz"
        shell:
                "cutadapt --cut 10 --minimum-length 25 -q 30 -o {output.outA} -p {output.outB} {input.fileA}
                {input.fileB}"

rule star_index:
        input:
                g="../genome/genome.fa",
                a="../annotation.gtf"
        output:
                directory("../indexstar")
        shell:
                "STAR --runThreadN 2 --runMode genomeGenerate --genomeDir {output} --genomeFastaFiles
                {input.g} --sjdbGTFfile {input.a} --genomeSAindexNbases 11"

rule star_alignment:
        input:
                fA="../trimmedresults/{sample}_read1_trimmed.fastq.gz",
                fB="../trimmedresults/{sample}_read2_trimmed.fastq.gz"
        output:
                "../starop/{sample}Aligned.out.sam"
        params:
                prefix="../starop/{sample}_"
        shell:
                "STAR --runThreadN 2 --genomeDir ../indexstar --readFilesIn {input.fA} {input.fB}
                --outFileNamePrefix {params.prefix} --outSAMtype SAM --readFilesCommand zcat"

rule feature_count:
        input:
                files=expand("../starop/{sample}Aligned.out.sam", sample=SAMPLES)
        output:
                "../fcount_output.txt"
        shell:
                "featureCounts -p --countReadPairs -a ../annotation.gtf -g gene_name -s 2 -o {output}
                {input.files}"
```

The rule ‘all’ indicates the target that Snakemake will create, in this case the featureCounts output is the final desired output. Rules ‘quality_check,’ ‘adapter_trimming,’ ‘star_index’, ‘star_alignment’,

'feature_count' perform quality control on raw sequencing data using fastqc, perform adapter trimming using cutadapt, align reads to the reference using STAR in addition to building an index and performing read alignment, and get read counts for genomic features by using featureCounts, respectively.

To run the workflow, use the command:

```
snakemake –cores 1 –s snm0
```

A successful result is displayed as follows:

```
(bio) ubuntu@lsi:~/QC reports$ snakemake --cores 1 -s seq
Building DAG of jobs...
Using shell: /home/ubuntu/miniconda3/envs/bio/bin/bash
Provided cores: 1 (use --cores to define parallelism)
Rules claiming more threads will be scaled down.
Job stats:
job              count
-------------  -------
all                  1
feature_count        1
total                2

Select jobs to execute...
```

```
        v2.0.6

//========================== featureCounts setting ===========================\\
||                                                                            ||
||             Input files : 6 SAM files                                      ||
||                                                                            ||
||                           G1_rep1Aligned.out.sam                           ||
||                           G1_rep2Aligned.out.sam                           ||
||                           G1_rep3Aligned.out.sam                           ||
||                           G2_rep1Aligned.out.sam                           ||
||                           G2_rep2Aligned.out.sam                           ||
||                           G2_rep3Aligned.out.sam                           ||
||                                                                            ||
||             Output file : fcount_output.txt                                ||
||                 Summary : fcount_output.txt.summary                        ||
||              Paired-end : yes                                              ||
||         Count read pairs : yes                                             ||
||              Annotation : annotation.gtf (GTF)                             ||
||      Dir for temp files : ..                                               ||
||                                                                            ||
||                 Threads : 1                                                ||
||                   Level : meta-feature level                               ||
||      Multimapping reads : not counted                                      ||
|| Multi-overlapping reads : not counted                                      ||
||   Min overlapping bases : 1                                                ||
||                                                                            ||
\\============================================================================//
```

```
//================================= Running ==================================\\
||                                                                            ||
|| Load annotation file annotation.gtf ...                                    ||
||    Features : 26155                                                        ||
||    Meta-features : 1378                                                    ||
||    Chromosomes/contigs : 93                                                ||
||                                                                            ||
|| Process SAM file G1_rep1Aligned.out.sam...                                 ||
||    Strand specific : reversely stranded                                    ||
||    Paired-end reads are included.                                          ||
||    Total alignments : 112591                                               ||
||    Successfully assigned alignments : 90160 (80.1%)                        ||
||    Running time : 0.00 minutes                                             ||
||                                                                            ||
|| Process SAM file G1_rep2Aligned.out.sam...                                 ||
||    Strand specific : reversely stranded                                    ||
||    Paired-end reads are included.                                          ||
||    Total alignments : 136051                                               ||
||    Successfully assigned alignments : 109606 (80.6%)                       ||
||    Running time : 0.01 minutes                                             ||
||                                                                            ||
|| Process SAM file G1_rep3Aligned.out.sam...                                 ||
||    Strand specific : reversely stranded                                    ||
||    Paired-end reads are included.                                          ||
||    Total alignments : 122736                                               ||
||    Successfully assigned alignments : 98128 (80.0%)                        ||
||    Running time : 0.01 minutes                                             ||
||                                                                            ||
|| Process SAM file G2_rep1Aligned.out.sam...                                 ||
||    Strand specific : reversely stranded                                    ||
||    Paired-end reads are included.                                          ||
||    Total alignments : 219593                                               ||
||    Successfully assigned alignments : 173866 (79.2%)                       ||
||    Running time : 0.01 minutes                                             ||
||                                                                            ||
|| Process SAM file G2_rep2Aligned.out.sam...                                 ||
||    Strand specific : reversely stranded                                    ||
||    Paired-end reads are included.                                          ||
||    Total alignments : 153353                                               ||
||    Successfully assigned alignments : 112283 (73.2%)                       ||
||    Running time : 0.01 minutes                                             ||
||                                                                            ||
|| Process SAM file G2_rep3Aligned.out.sam...                                 ||
||    Strand specific : reversely stranded                                    ||
||    Paired-end reads are included.                                          ||
||    Total alignments : 175856                                               ||
||    Successfully assigned alignments : 139719 (79.5%)                       ||
||    Running time : 0.01 minutes                                             ||
||                                                                            ||
|| Write the final count table.                                               ||
|| Write the read assignment summary.                                         ||
||                                                                            ||
|| Summary of counting results can be found in file "../fcount_output.txt.su  ||
|| mmary"                                                                     ||
||                                                                            ||
\\============================================================================//
```

## References


1. Beg, M., Taka, J., Kluyver, T., Konovalov, A., Ragan-Kelley, M., Thiery, N. M., & Fangohr, H. (2021). Using Jupyter for Reproducible Scientific Workflows. *Computing in Science & Engineering*, *23*(2), 36–46. https://doi.org/10.1109/mcse.2021.3052101
2. *Genome Browser FAQ*. (n.d.). Genome.ucsc.edu. Retrieved February 2, 2024, from https://genome.ucsc.edu/FAQ/FAQformat.html#format4
3. Grüning, B., Chilton, J., Köster, J., Dale, R., Soranzo, N., van den Beek, M., Goecks, J., Backofen, R., Nekrutenko, A., & Taylor, J. (2018). Practical Computational Reproducibility in the Life Sciences. *Cell Systems*, *6*(6), 631–635. https://doi.org/10.1016/j.cels.2018.03.014
4. Peng, R. D. (2011). Reproducible Research in Computational Science. *Science*, *334*(6060), 1226–1227. https://doi.org/10.1126/science.1213847
5. *Snakemake — Snakemake 6.6.0 documentation*. (n.d.). Snakemake.readthedocs.io. https://snakemake.readthedocs.io/en/stable/
6. *The Sequence Read Archive (SRA): Getting Started*. (n.d.). Www.ncbi.nlm.nih.gov. https://www.ncbi.nlm.nih.gov/sra/docs/
7. Wratten, L., Wilm, A., & Göke, J. (2021). Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers. *Nature Methods*. https://doi.org/10.1038/s41592-021-01254-9