PM4NGS Documentation

Release 0.15

Roberto Vera Alvarez

Oct 06, 2021

Contents

1 Installation **3**
- 1.1 Requirements . . . 3
- 1.2 PM4NGS with Conda/BioConda (Recommended) . . . 3
  - 1.2.1 Conda installation . . . 3
  - 1.2.2 PM4NGS conda installation . . . 3
  - 1.2.3 PM4NGS conda env activation . . . 3
- 1.3 PM4NGS with Python virtual environment . . . 4
  - 1.3.1 PM4NGS python installation . . . 4
  - 1.3.2 PM4NGS python env activation . . . 4

2 PM4NGS on Ubuntu **5**
- 2.1 Installing PM4NGS . . . 5
- 2.2 Using PM4NGS . . . 5
- 2.3 Running the ChIP-exo demo . . . 6
- 2.4 Running the Jupyter Server . . . 7

3 PM4NGS on Windows Subsystem for Linux **9**
- 3.1 Activating WSL . . . 9
- 3.2 Installing Docker . . . 10
- 3.3 Installing Ubuntu on WSL . . . 13
- 3.4 Installing PM4NGS . . . 14
- 3.5 Using PM4NGS on WSL . . . 14
- 3.6 Running the ChIP-exo demo . . . 15
- 3.7 Running the Jupyter Server . . . 17

4 PM4NGS on GCP instance with Ubuntu **19**
- 4.1 Creating a GCP instance with Ubuntu 20.02 LTS . . . 19
- 4.2 Installing PM4NGS on the GCP instance with Ubuntu . . . 22
- 4.3 Installing PM4NGS . . . 22
- 4.4 Using PM4NGS . . . 23
- 4.5 Running the ChIP-exo demo . . . 23
- 4.6 Running the Jupyter Server . . . 26
- 4.7 Open a firewall rule for accessing the Jupyter Server . . . 27

5 Sample sheet **45**
- 5.1 Single-end example . . . 45
- 5.2 Paired-end example . . . 45

5.3 Processing data from the NCBI SRA . . . 46
5.4 Sample sheet column names and description . . . 46

6 Differential Gene expression and GO enrichment from RNA-Seq data 47
6.1 Input requirements . . . 48
6.2 Pipeline command line . . . 49
6.3 Creating the DGA and GO enrichment from RNA-Seq data project . . . 49
6.4 Jupyter server . . . 52
6.5 Data processing . . . 53
6.6 CWL workflows . . . 53
6.6.1 SRA download and QC workflow . . . 53
6.6.2 Samples Trimming . . . 55
6.6.3 STAR based alignment and sorting . . . 55
6.6.4 RNA-Seq quantification and QC workflow using TPMCalculator . . . 56
6.6.5 Differential Gene Expression analysis from RNA-Seq data . . . 58
6.6.6 GO enrichment from RNA-Seq data . . . 61
6.7 Demo . . . 62

7 Differential Binding detection from ChIP-Seq data 65
7.1 Input requirements . . . 66
7.2 Pipeline command line . . . 66
7.3 Creating the Differential Binding detection from ChIP-Seq data project . . . 67
7.4 Jupyter server . . . 70
7.5 Data processing . . . 70
7.6 CWL workflows . . . 71
7.6.1 SRA download and QC workflow . . . 71
7.6.2 Samples Trimming . . . 72
7.6.3 BWA based alignment and quality control workflow . . . 72
7.6.4 Peak caller workflow using MACS2 . . . 74
7.6.5 MEME Motif detection workflow . . . 76
7.6.6 MEME databases . . . 77
7.6.7 Differential binding analysis with Diffbind . . . 78
7.7 Demo . . . 79

8 Detection of binding events from ChIP-exo data 81
8.1 Input requirements . . . 82
8.2 Pipeline command line . . . 82
8.3 Creating the Detection of binding events from ChIP-exo data project . . . 83
8.4 Jupyter server . . . 86
8.5 Data processing . . . 86
8.6 CWL workflows . . . 87
8.6.1 SRA download and QC workflow . . . 87
8.6.2 Samples Trimming . . . 88
8.6.3 BWA based alignment and quality control workflow . . . 88
8.6.4 Peak caller workflow using MACE . . . 90
8.6.5 MEME Motif detection workflow . . . 91
8.6.6 MEME databases . . . 92
8.7 Demo . . . 93

9 Transcriptome Annotation pipeline for non-model organisms 95
9.1 Annotation Workflow . . . 95
9.2 GCP configuration . . . 96
9.3 Input requirements . . . 96
9.4 Pipeline command line . . . 96
9.5 Creating the annotation project . . . 96

9.6 Jupyter server . . . 98
9.7 Data processing . . . 98
9.8 Demo . . . 98

10 Create a new PM4NGS pipeline **101**
10.1 PM4NGS based pipeline folder structure . . . 101
10.2 CWL tools and workflows specifications . . . 103

11 Utils **105**
11.1 STAR Genomic Indexes . . . 105
11.2 BWA Genomic Indexes . . . 106
11.3 Create BED from GTF . . . 106

12 PM4NGS available genomes **107**

13 Introduction **109**

14 Features **111**

15 Citation **113**

16 Help and Support **115**

17 Public Domain notice **117**


ATTGCATTG
CWL

CHAPTER 1

Installation

1.1 Requirements

1. Poppler (https://poppler.freedesktop.org/)

1.2 PM4NGS with Conda/BioConda (Recommended)

1.2.1 Conda installation

Conda should be already installed and configured using these commands:

```
localhost:~> wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.
↪sh
localhost:~> sh Miniconda3-latest-Linux-x86_64.sh
localhost:~> conda config --add channels defaults
localhost:~> conda config --add channels bioconda
localhost:~> conda config --add channels conda-forge
```

1.2.2 PM4NGS conda installation

PM4NGS should be installed in a Conda virtual environment named *pm4ngs*:

```
localhost:~> conda create -n pm4ngs pm4ngs
```

1.2.3 PM4NGS conda env activation

For activating the conda env:

```
localhost:~> conda activate pm4ngs
localhost:~> pm4ngs-create -v
PM4NGS version: 0.1.dev0+d20200819
```

1.3 PM4NGS with Python virtual environment

1.3.1 PM4NGS python installation

Python 3.6 or above should be installed.

```
localhost:~> python3 -m venv pm4ngs_venv
localhost:~> source pm4ngs_venv/bin/activate
(pm4ngs_venv) localhost:~> pip install wheel
(pm4ngs_venv) localhost:~> pip install pm4ngs
```

1.3.2 PM4NGS python env activation

For activating the virtual env:

```
localhost:~> source pm4ngs_venv/bin/activate
(pm4ngs_venv) localhost:~> pm4ngs-create -v
PM4NGS version: 0.0.4
```

CHAPTER 2

PM4NGS on Ubuntu

Runs these commands on a terminal to to prepare the instance to run PM4NGS

```
veraalva@perseo:~$ sudo apt-get update
veraalva@perseo:~$ sudo apt-get install docker.io python3 python3-pip python3-venv␣
↪python3-dev poppler-utils gcc nodejs tree
veraalva@perseo:~$ sudo usermod -aG docker $USER
```

Close and reopen the terminal to set the docker group in the user.

2.1 Installing PM4NGS

Creates a Python virtual environment named: **pm4ngs_venv** for installing PM4NGS

```
veraalva@perseo:~$ python3 -m venv pm4ngs_venv
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pip install wheel
(pm4ngs_venv) veraalva@perseo:~$ pip install pm4ngs
```

2.2 Using PM4NGS

Open a terminal and activate the **pm4ngs_venv** virtual environment

```
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pm4ngs-chipexo --version
PM4NGS version: 0.0.4
(pm4ngs_venv) veraalva@perseo:~$
```

2.3 Running the ChIP-exo demo

Open a terminal and activate the **pm4ngs_venv** virtual environment

```
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pm4ngs-chipexo-demo
Generating demo for ChIP-exo data analysis project
Downloading file: pm4ngs_chipexo_demo_config.yaml
Downloading file: pm4ngs_chipexo_demo_sample_data.csv
Using config file: pm4ngs_chipexo_demo_config.yaml
{
    "author_name": "Roberto Vera Alvarez",
    "user_email": "veraalva@ncbi.nlm.nih.gov",
    "project_name": "pm4ngs-chipexo",
    "dataset_name": "PRJNA338159",
    "is_data_in_SRA": "y",
    "sequencing_technology": "single-end",
    "create_demo": "n",
    "number_spots": "1000000",
    "organism": "Escherichia coli",
    "genome_name": "NC_000913.3",
    "genome_dir": "{{ cookiecutter.genome_name}}",
    "aligner_index_dir": "{{ cookiecutter.genome_dir}}/BWA/",
    "genome_fasta": "{{ cookiecutter.genome_dir}}/NC_000913.3.fa",
    "genome_gtf": "{{ cookiecutter.genome_dir}}/NC_000913.3.gtf",
    "genome_chromsizes": "{{ cookiecutter.genome_dir}}/NC_000913.3.sizes",
    "use_docker": "y",
    "max_number_threads": "32"
}
Cloning Git repo: https://github.com/ncbi/cwl-ngs-workflows-cbb to /home/veraalva/
↪pm4ngs-chipexo/bin/cwl
Updating CWLs dockerPull and SoftwareRequirement from: /home/veraalva/pm4ngs-chipexo/
↪requirements/conda-env-dependencies.yaml
bamscale with version 0.0.3 update image to: quay.io/biocontainers/bamscale:0.0.3--
↪ha85820d_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bamscale/bamscale-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bamscale:0.0.5--h18f8b1d_1
bedtools with version 2.29.2 update image to: quay.io/biocontainers/bedtools:2.29.2--
↪hc088bd4_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bedtools/bedtools-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bedtools:2.28.0--hdf88d34_0
bioconductor-diffbind with version 2.16.0 update image to: quay.io/biocontainers/
↪bioconductor-diffbind:2.16.0--r40h5f743cb_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-pca.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/macs-cutoff.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/dga_heatmaps.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/diffbind.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/edgeR-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/volcano_plot.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/readQC.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
```

(continues on next page)

(continued from previous page)

```
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
bwa with version 0.7.17 update image to: quay.io/biocontainers/bwa:0.7.17--hed695b0_7
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bwa/bwa-docker.yml with old image␣
↪replaced: quay.io/biocontainers/bwa:0.7.17--h84994c4_5
There is not biocontainer image for gffread version 0.12.1
homer with version 4.11 update image to: quay.io/biocontainers/homer:4.11--
↪pl526h9a982cc_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/homer/homer-docker.yml with old␣
↪image replaced: quay.io/biocontainers/homer:4.11--pl526h2bce143_2
mace with version 1.2 update image to: quay.io/biocontainers/mace:1.2--py27h99da42f_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/mace/mace-docker.yml with old image␣
↪replaced: quay.io/biocontainers/mace:1.2--py27h99da42f_1
meme with version 5.1.1 update image to: quay.io/biocontainers/meme:5.1.1--
↪py37pl526h072abfd_3
    /Users/veraalva/my_ngs_project/bin/cwl/tools/meme/meme-docker.yml with old image␣
↪replaced: quay.io/biocontainers/meme:5.1.1--py27pl526h53063a7_3
Copying file /Users/veraalva/Work/Developer/Python/pm4ngs/pm4ngs-chipexo/example/
↪pm4ngs_chipexo_demo_sample_data.csv  to /Users/veraalva/my_ngs_project/data/my_
↪dataset_name/sample_table.csv
6 files loaded
Using table:
  sample_name file                      condition  replicate
0  SRR4011416        Exp_O2_growth_no_rifampicin          1
1  SRR4011417        Exp_O2_growth_no_rifampicin          2
2  SRR4011421           Exp_O2_growth_rifampicin          1
3  SRR4011425           Exp_O2_growth_rifampicin          2
4  SRR4011418       Stat_O2_growth_no_rifampicin          1
5  SRR4011419       Stat_O2_growth_no_rifampicin          2
 Done
```

2.4 Running the Jupyter Server

Open a terminal and activate the **pm4ngs_venv** virtual environment

```
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ jupyter notebook --no-browser
[I 17:04:45.633 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 17:04:45.633 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 17:04:45.634 NotebookApp] http://localhost:8888/?
↪token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
[I 17:04:45.634 NotebookApp]  or http://127.0.0.1:8888/?
↪token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
[I 17:04:45.634 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmation).
[C 17:04:45.637 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-522-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
     or http://127.0.0.1:8888/?token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
```

Copy the URL with localhost in a browser.

CHAPTER 3

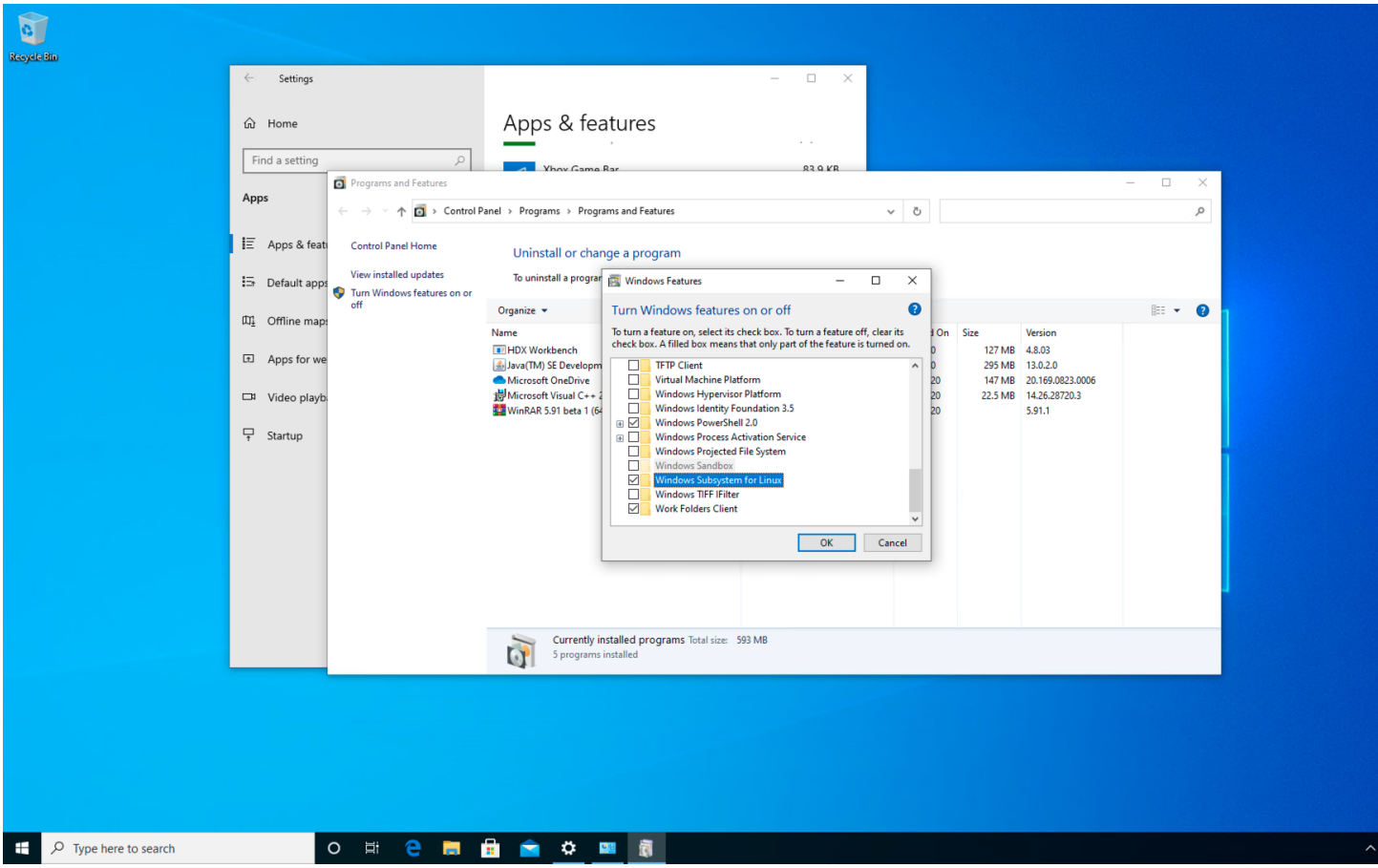
PM4NGS on Windows Subsystem for Linux

PM4NGS can be installed on Microsoft Windows 10 using the Windows Subsystem for Linux (WSL) and Docker.

3.1 Activating WSL

Follow the steps for activating WSL

1. Open the Settings app
2. Go to Apps -> Apps & Features
3. Scroll down to the Programs and Feature link
4. Click the link. The Programs and Features dialog will be opened.
5. On the left, click the link Turn Windows Features on or off.
6. The dialog Windows Features will appear on the screen. Scroll down to the option named Windows Subsystem for Linux and enable it
7. Click OK to apply the changes you made. Windows will install WSL.



8. Reboot the operating system when prompted.

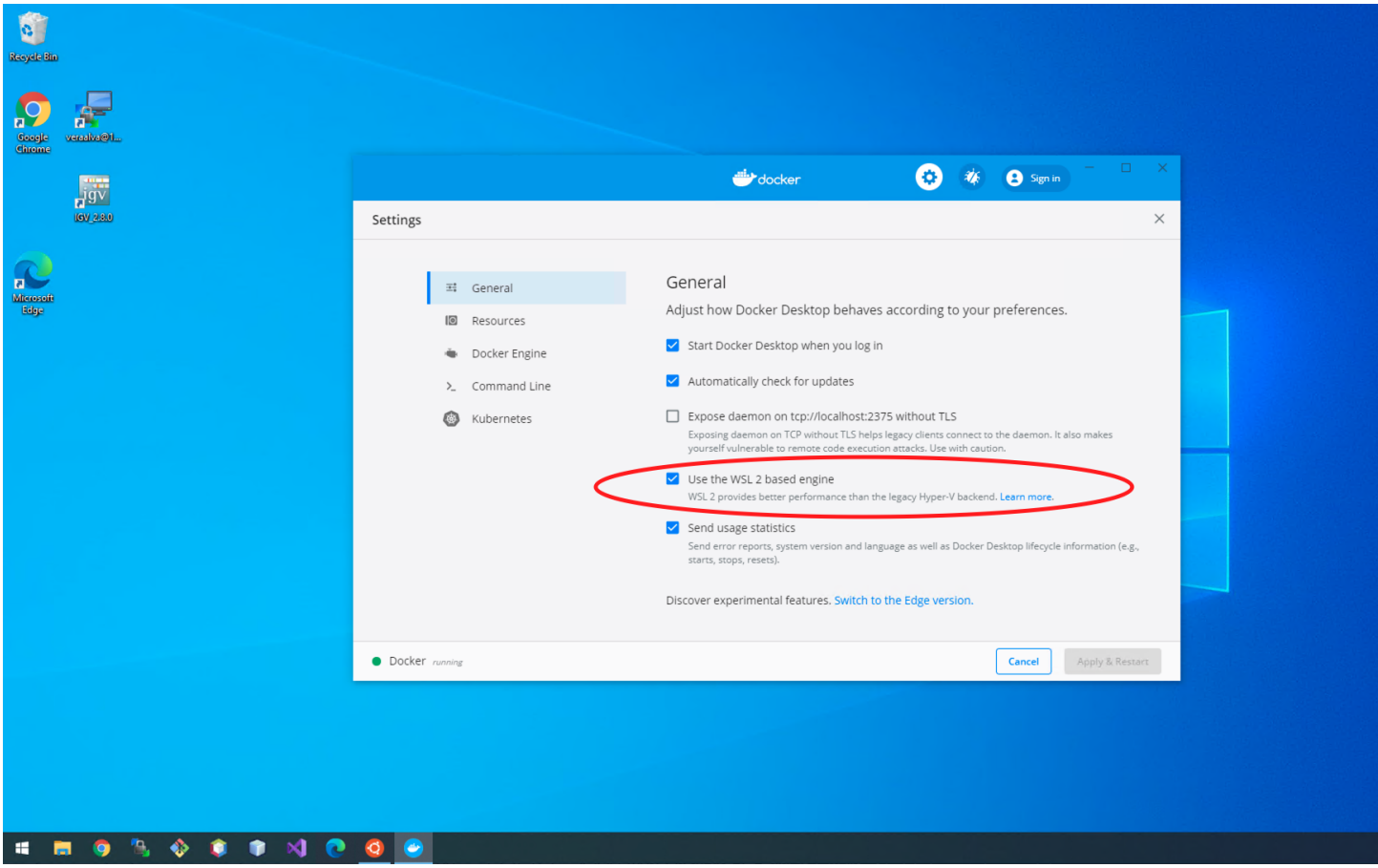
3.2 Installing Docker

Before you install the Docker Desktop WSL 2 backend, you must complete the following steps:

1. Install Windows 10, version 2004 or higher. The Docker Desktop Edge release also supports Windows 10, version 1903 or higher.
2. Enable WSL 2 feature on Windows. For detailed instructions, refer to the Microsoft documentation.

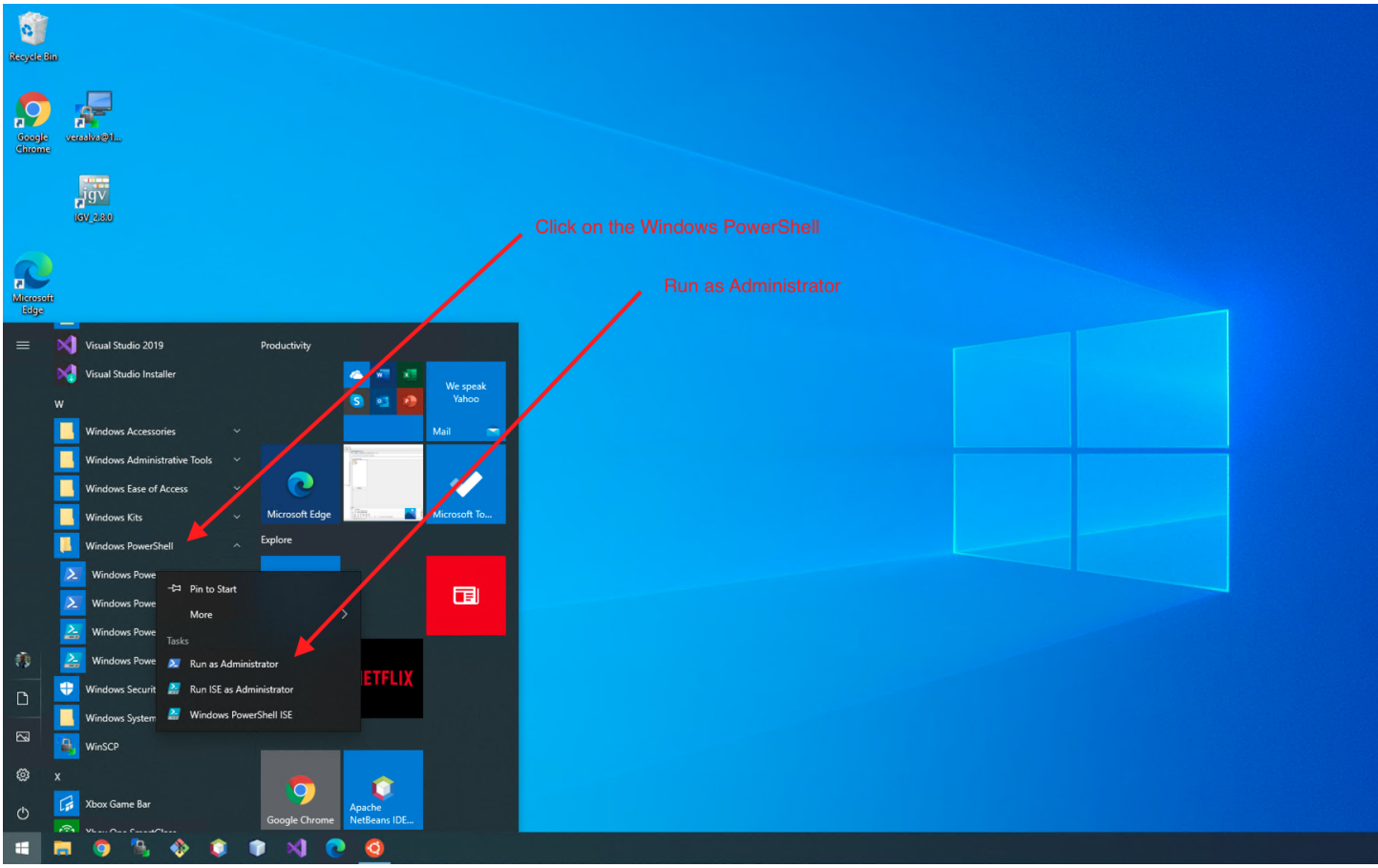
Download and install the Linux kernel update package.

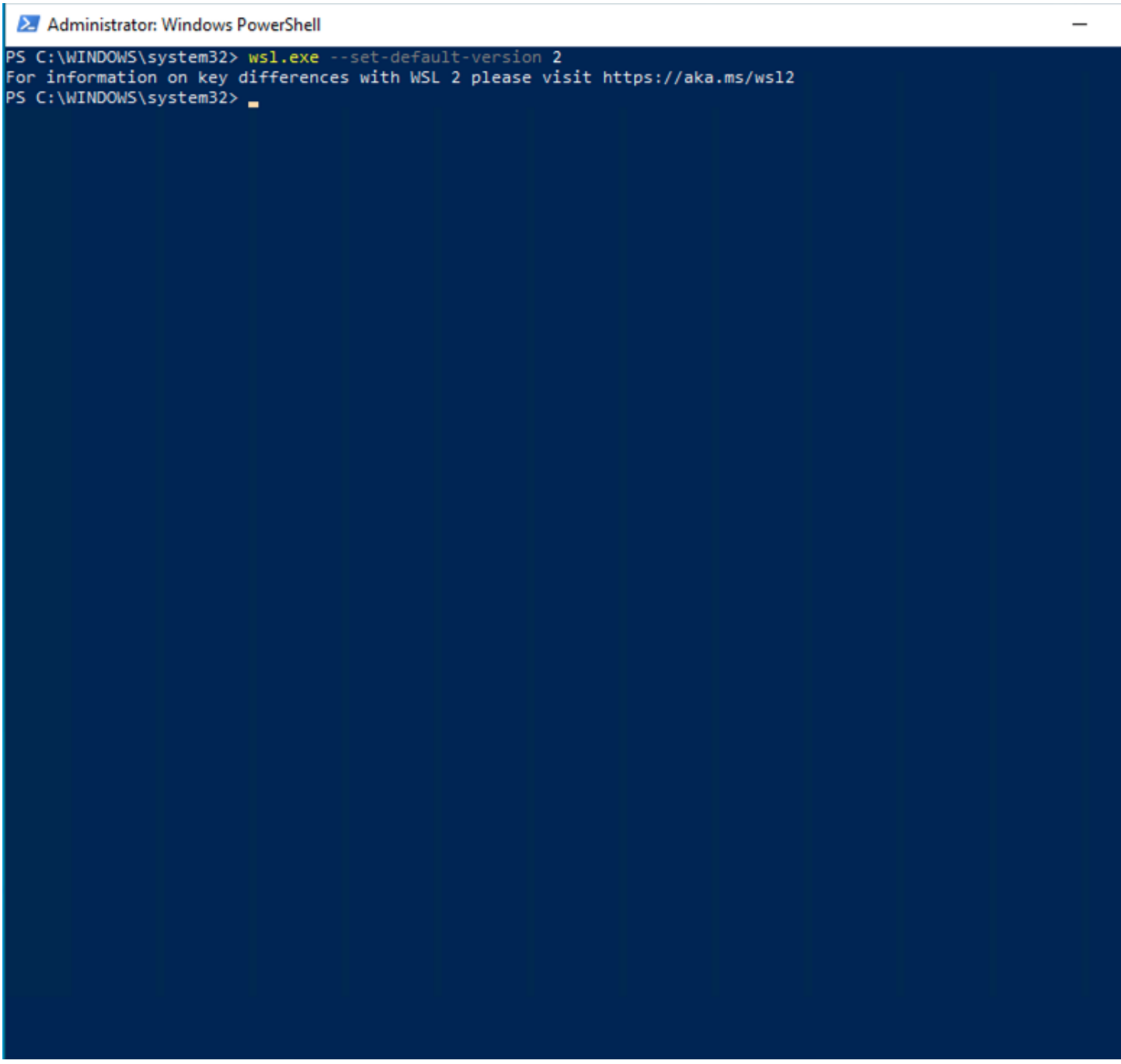
Download Docker Desktop Stable 2.3.0.2 or a later release. After installed, open the Docker Setting and activate the WSL 2. Then, click on Apply and Restart.



Open a PowerShell as Administrator for activating WSL 2 as default option runnign the command

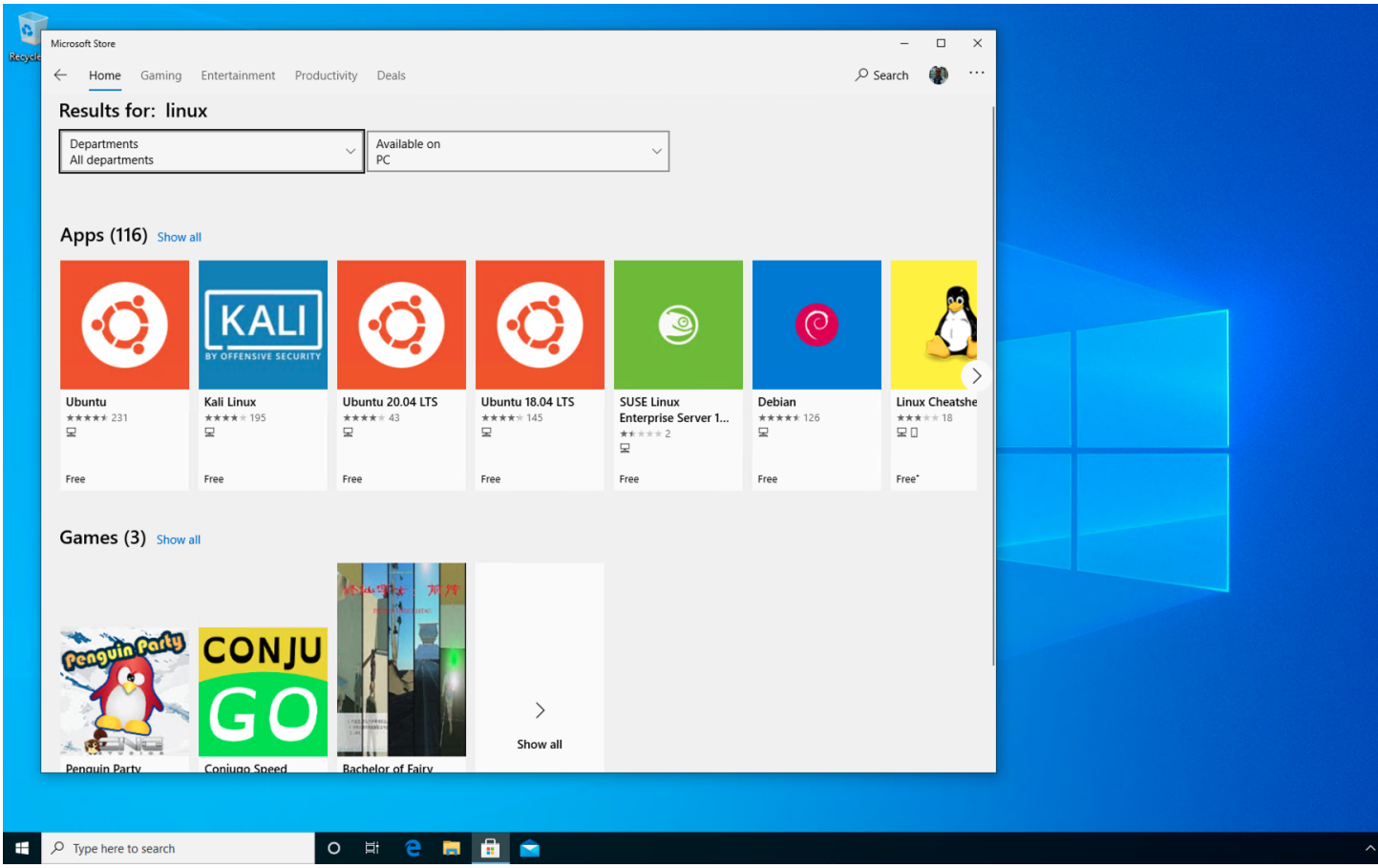
```
wsl.exe --set-default-version 2
```


Click on the Windows PowerShell
Run as Administrator
Recycle Bin
Google Chrome
IGV 2.8.0
Microsoft Edge
Visual Studio 2019
Visual Studio Installer
Productivity
Windows Accessories
Windows Administrative Tools
Windows Ease of Access
Windows Kits
Windows PowerShell
Windows Powe
Windows Securit
Windows System
WinSCP
Xbox Game Bar
Pin to Start
More
Tasks
Run as Administrator
Run ISE as Administrator
Windows PowerShell ISE
We speak Yahoo
Mail
Microsoft To...
Explore
Google Chrome
Apache NetBeans IDE...



3.3 Installing Ubuntu on WSL

Open Microsoft Store and search for Linux, then install the Linux distribution that you like. We recommend Ubuntu 20.04 LTS.



Lunch Ubuntu

3.4 Installing PM4NGS

Runs these commands on the Ubuntu terminal to to prepare the instance to run PM4NGS

```
veraalva@perseo:~$ sudo apt-get update
veraalva@perseo:~$ sudo apt-get install python3 python3-pip python3-venv python3-dev␣
↪poppler-utils gcc nodejs docker.io
veraalva@perseo:~$ sudo usermod -aG docker $USER
```

Close and reopen the terminal to set the docker group in the user.

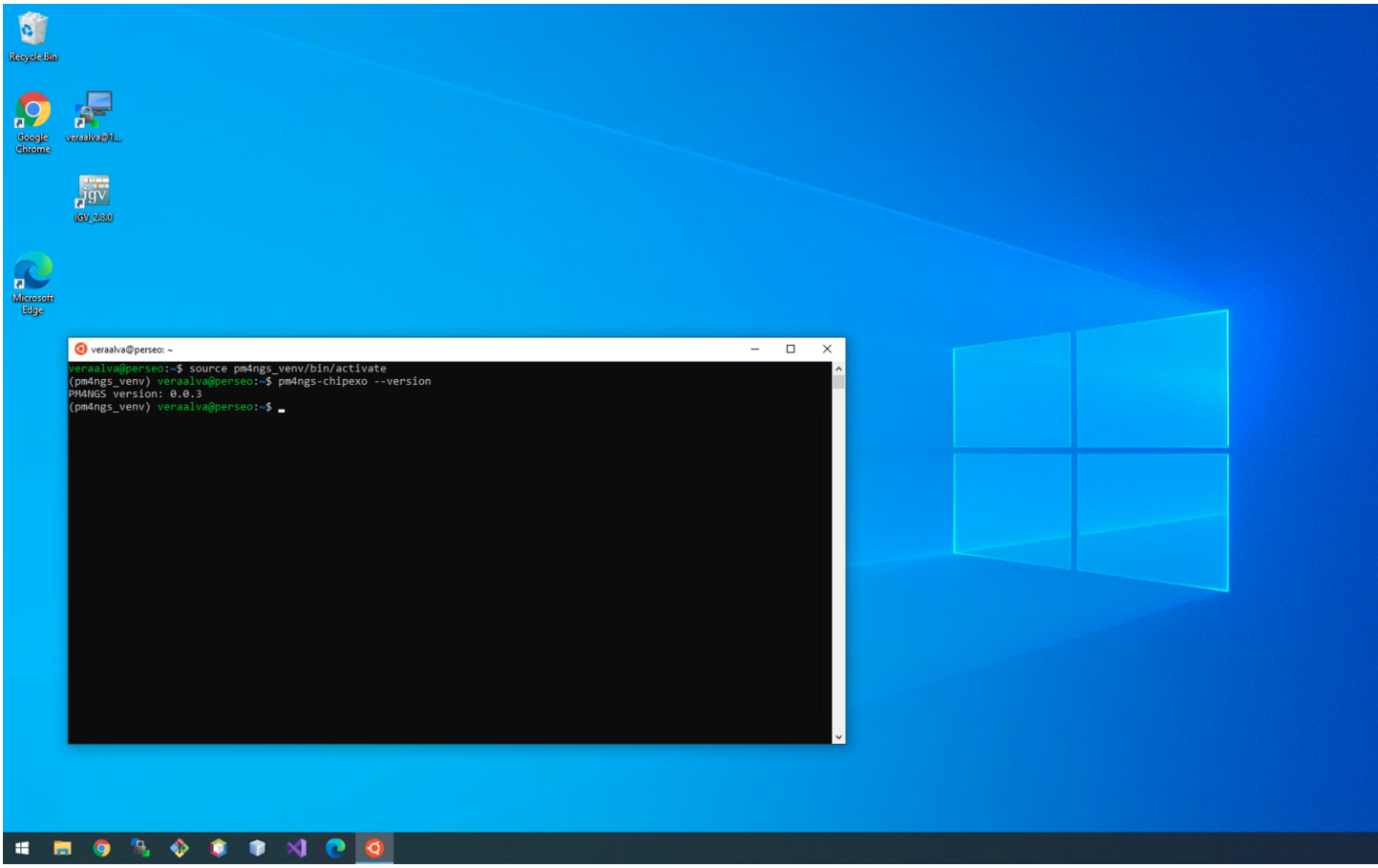
Creates a Python virtual environment named: **pm4ngs_venv** for installing PM4NGS

```
veraalva@perseo:~$ python3 -m venv pm4ngs_venv
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pip install wheel
(pm4ngs_venv) veraalva@perseo:~$ pip install pm4ngs
```

3.5 Using PM4NGS on WSL

Open the Ubuntu app in Windows and activate the **pm4ngs_venv** virtual environment

```
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pm4ngs-chipexo --version
PM4NGS version: 0.0.4
(pm4ngs_venv) veraalva@perseo:~$
```



3.6 Running the ChIP-exo demo

Open the Ubuntu app in Windows and activate the **pm4ngs_venv** virtual environment

```
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pm4ngs-chipexo-demo
Generating demo for ChIP-exo data analysis project
Downloading file: pm4ngs_chipexo_demo_config.yaml
Downloading file: pm4ngs_chipexo_demo_sample_data.csv
Using config file: pm4ngs_chipexo_demo_config.yaml
{
    "author_name": "Roberto Vera Alvarez",
    "user_email": "veraalva@ncbi.nlm.nih.gov",
    "project_name": "pm4ngs-chipexo",
    "dataset_name": "PRJNA338159",
    "is_data_in_SRA": "y",
    "sequencing_technology": "single-end",
    "create_demo": "n",
```

(continues on next page)

(continued from previous page)

```
    "number_spots": "1000000",
    "organism": "Escherichia coli",
    "genome_name": "NC_000913.3",
    "genome_dir": "{{ cookiecutter.genome_name}}",
    "aligner_index_dir": "{{ cookiecutter.genome_dir}}/BWA/",
    "genome_fasta": "{{ cookiecutter.genome_dir}}/NC_000913.3.fa",
    "genome_gtf": "{{ cookiecutter.genome_dir}}/NC_000913.3.gtf",
    "genome_chromsizes": "{{ cookiecutter.genome_dir}}/NC_000913.3.sizes",
    "use_docker": "y",
    "max_number_threads": "32"
}
Cloning Git repo: https://github.com/ncbi/cwl-ngs-workflows-cbb to /home/veraalva/
↪pm4ngs-chipexo/bin/cwl
Updating CWLs dockerPull and SoftwareRequirement from: /home/veraalva/pm4ngs-chipexo/
↪requirements/conda-env-dependencies.yaml
bamscale with version 0.0.3 update image to: quay.io/biocontainers/bamscale:0.0.3--
↪ha85820d_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bamscale/bamscale-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bamscale:0.0.5--h18f8b1d_1
bedtools with version 2.29.2 update image to: quay.io/biocontainers/bedtools:2.29.2--
↪hc088bd4_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bedtools/bedtools-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bedtools:2.28.0--hdf88d34_0
bioconductor-diffbind with version 2.16.0 update image to: quay.io/biocontainers/
↪bioconductor-diffbind:2.16.0--r40h5f743cb_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-pca.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/macs-cutoff.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/dga_heatmaps.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/diffbind.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/edgeR-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/volcano_plot.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/readQC.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
bwa with version 0.7.17 update image to: quay.io/biocontainers/bwa:0.7.17--hed695b0_7
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bwa/bwa-docker.yml with old image␣
↪replaced: quay.io/biocontainers/bwa:0.7.17--h84994c4_5
There is not biocontainer image for gffread version 0.12.1
homer with version 4.11 update image to: quay.io/biocontainers/homer:4.11--
↪pl526h9a982cc_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/homer/homer-docker.yml with old␣
↪image replaced: quay.io/biocontainers/homer:4.11--pl526h2bce143_2
mace with version 1.2 update image to: quay.io/biocontainers/mace:1.2--py27h99da42f_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/mace/mace-docker.yml with old image␣
↪replaced: quay.io/biocontainers/mace:1.2--py27h99da42f_1
meme with version 5.1.1 update image to: quay.io/biocontainers/meme:5.1.1--
↪py37pl526h072abfd_3
    /Users/veraalva/my_ngs_project/bin/cwl/tools/meme/meme-docker.yml with old image␣
↪replaced: quay.io/biocontainers/meme:5.1.1--py27pl526h53063a7_3
Copying file /Users/veraalva/Work/Developer/Python/pm4ngs/pm4ngs-chipexo/example/
↪pm4ngs_chipexo_demo_sample_data.csv  to /Users/veraalva/my_ngs_project/data/my_
↪dataset_name/sample_table.csv
```

(continues on next page)

(continued from previous page)

```
6 files loaded
Using table:
  sample_name file                      condition  replicate
0  SRR4011416        Exp_O2_growth_no_rifampicin          1
1  SRR4011417        Exp_O2_growth_no_rifampicin          2
2  SRR4011421           Exp_O2_growth_rifampicin          1
3  SRR4011425           Exp_O2_growth_rifampicin          2
4  SRR4011418       Stat_O2_growth_no_rifampicin          1
5  SRR4011419       Stat_O2_growth_no_rifampicin          2
 Done
```

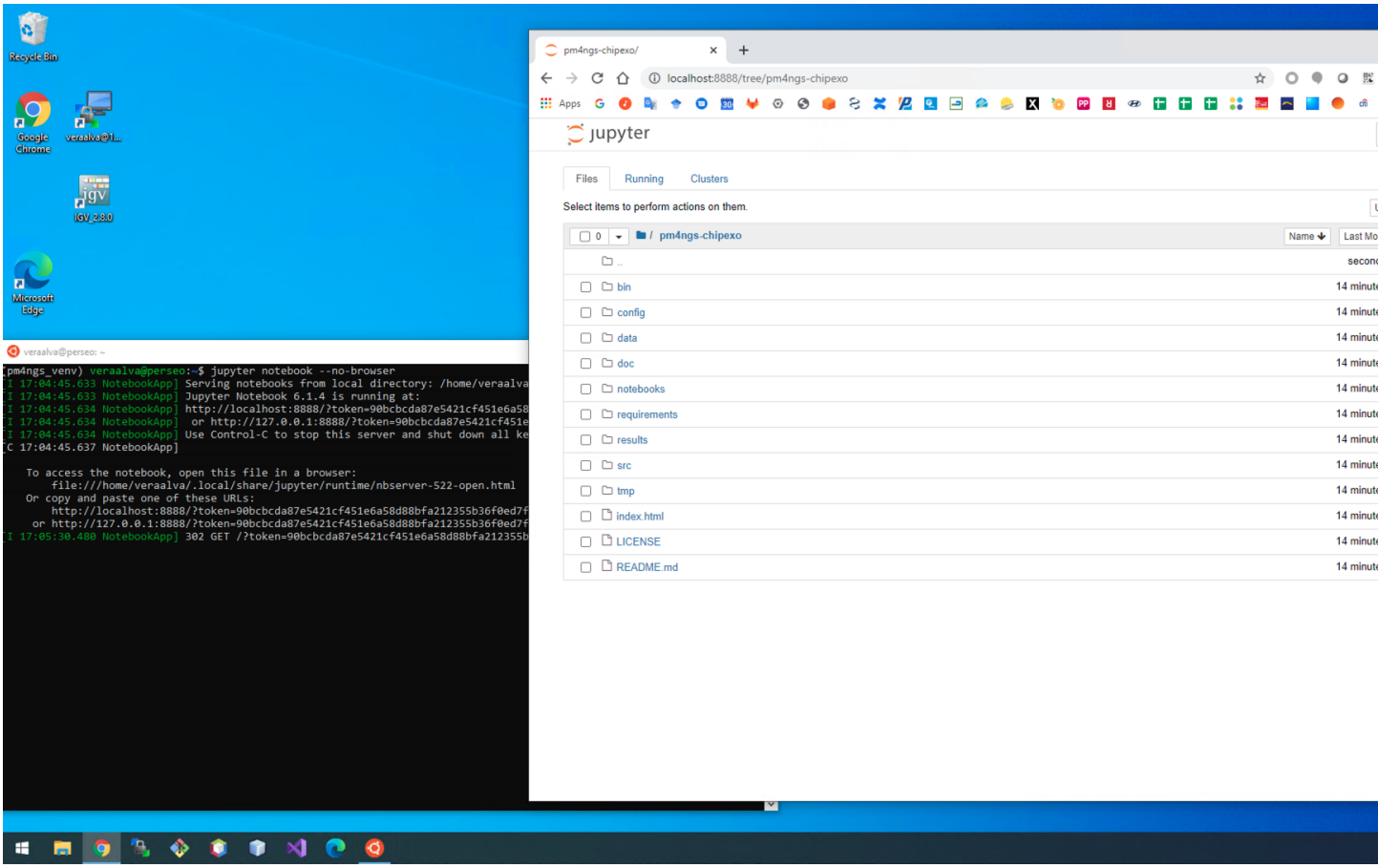
3.7 Running the Jupyter Server

Open the Ubuntu app in Windows and activate the **pm4ngs_venv** virtual environment

```
veraalva@perseo:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ jupyter notebook --no-browser
[I 17:04:45.633 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 17:04:45.633 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 17:04:45.634 NotebookApp] http://localhost:8888/?
↪token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
[I 17:04:45.634 NotebookApp]  or http://127.0.0.1:8888/?
↪token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
[I 17:04:45.634 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmation).
[C 17:04:45.637 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-522-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
     or http://127.0.0.1:8888/?token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f1a
```

Copy the URL with localhost in a browser:

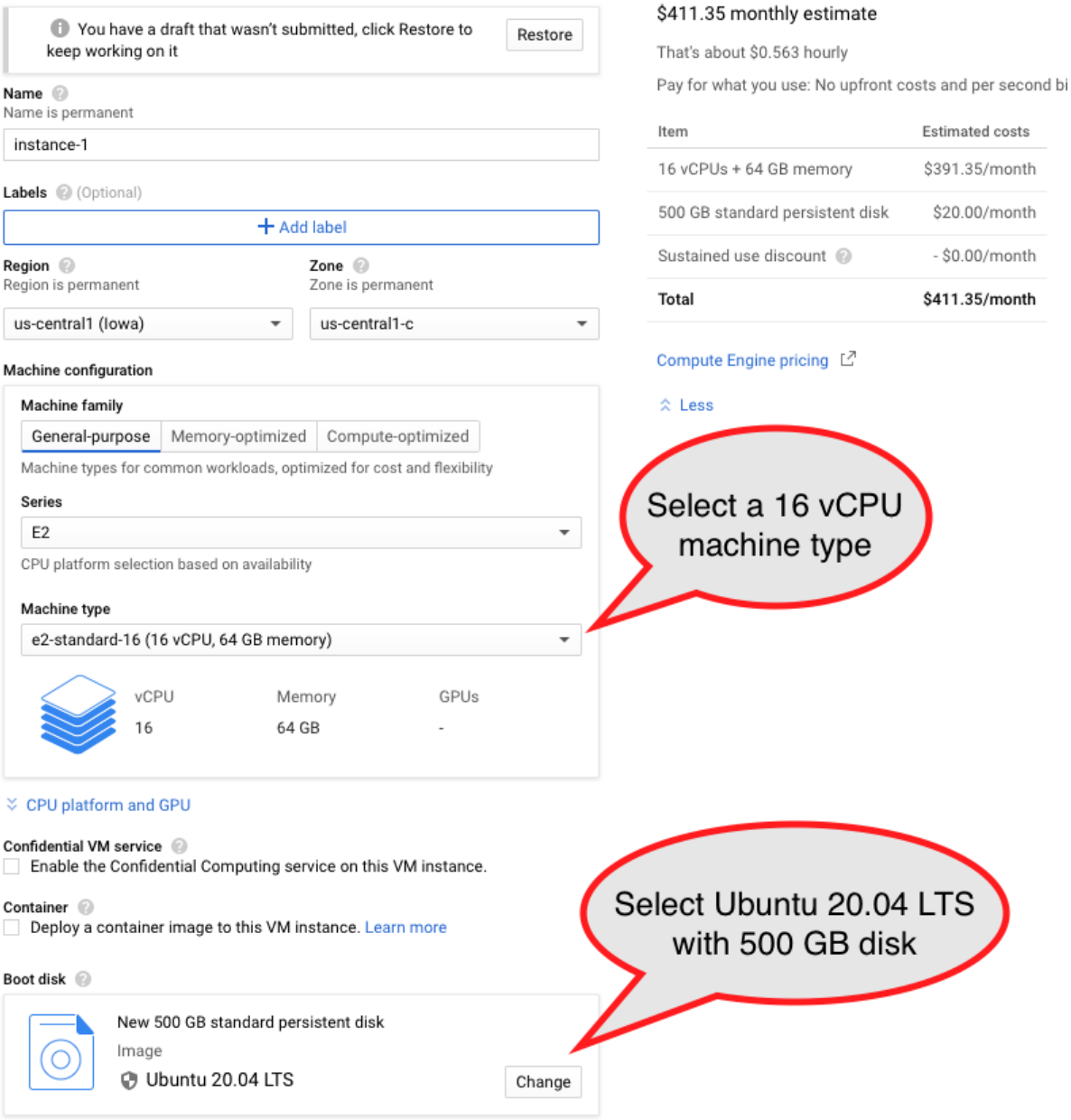

Recycle Bin
Google Chrome
veraalva@1...
IGV 2.8.0
Microsoft Edge
veraalva@perseo: ~
(pm4ngs_venv) veraalva@perseo:~$ jupyter notebook --no-browser
[I 17:04:45.633 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 17:04:45.633 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 17:04:45.634 NotebookApp] http://localhost:8888/?token=90bcbcda87e5421cf451e6a58
[I 17:04:45.634 NotebookApp] or http://127.0.0.1:8888/?token=90bcbcda87e5421cf451e
[I 17:04:45.634 NotebookApp] Use Control-C to stop this server and shut down all ke
[C 17:04:45.637 NotebookApp]
To access the notebook, open this file in a browser:
file:///home/veraalva/.local/share/jupyter/runtime/nbserver-522-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f
or http://127.0.0.1:8888/?token=90bcbcda87e5421cf451e6a58d88bfa212355b36f0ed7f
[I 17:05:30.480 NotebookApp] 302 GET /?token=90bcbcda87e5421cf451e6a58d88bfa212355b
pm4ngs-chipexo/
localhost:8888/tree/pm4ngs-chipexo
Apps
jupyter
Files
Running
Clusters
Select items to perform actions on them.
pm4ngs-chipexo
Name
Last Mo
bin
config
data
doc
notebooks
requirements
results
src
tmp
index.html
LICENSE
README.md
14 minute

CHAPTER 4

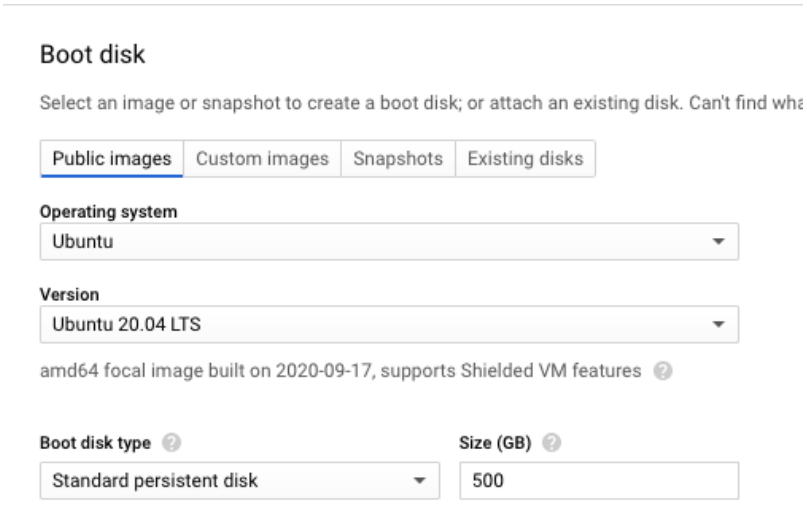
PM4NGS on GCP instance with Ubuntu

4.1 Creating a GCP instance with Ubuntu 20.02 LTS

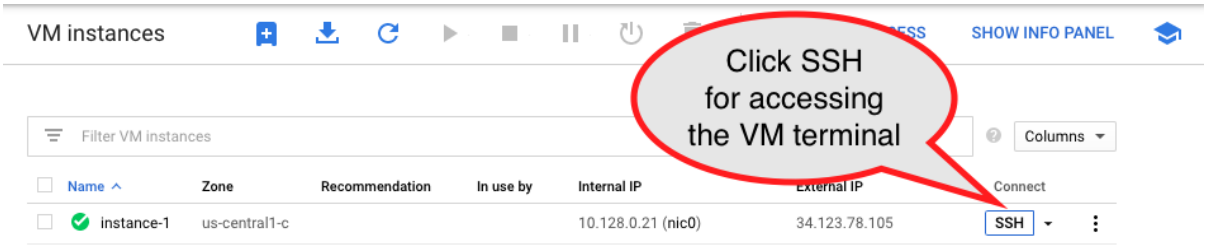
Creates a VM instance running Ubuntu



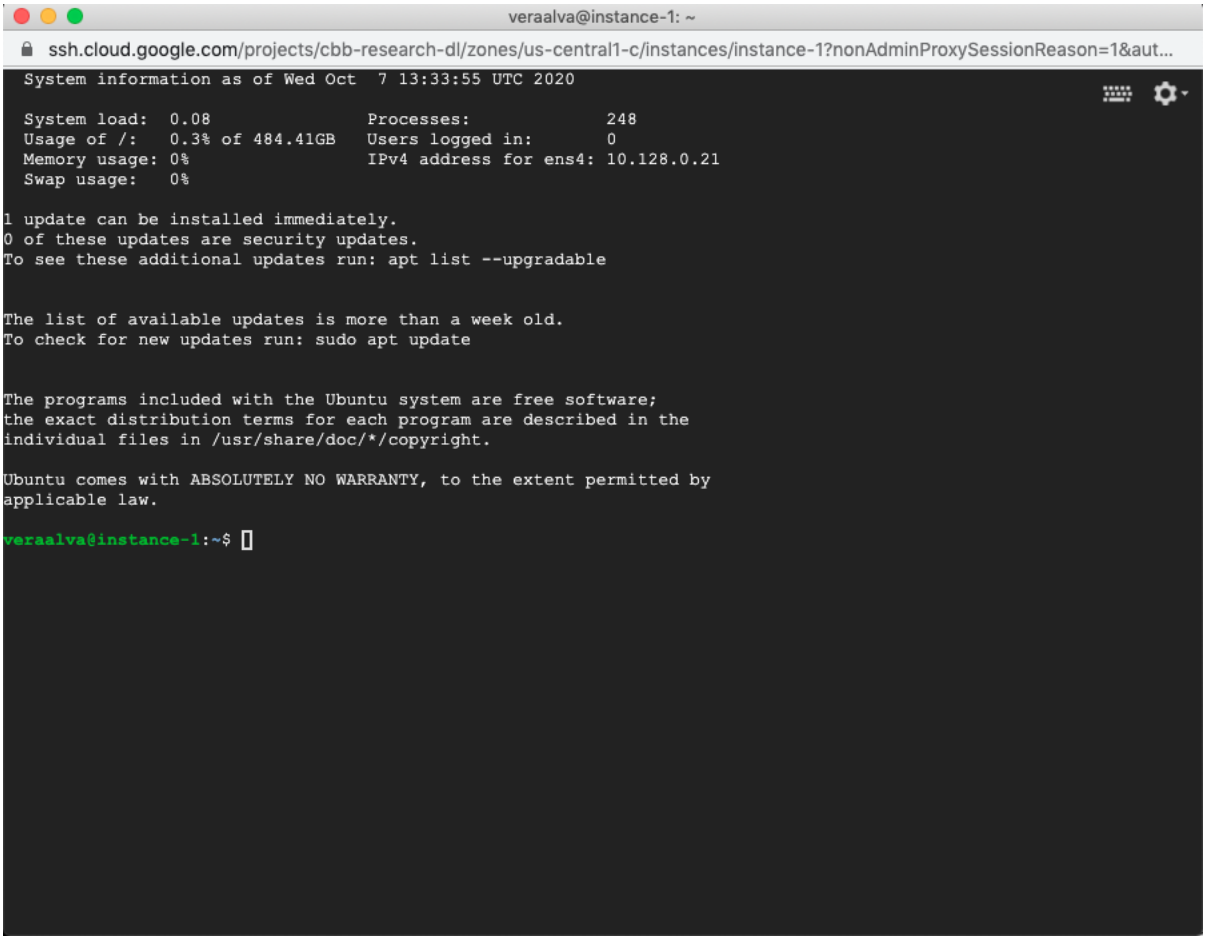
Select boot disk with Ubuntu 20.04 LTS with 500 GB of standard persistent disk.



Click on the **SSH** button for accessing the instance



A terminal is available after accessing through SSH



4.2 Installing PM4NGS on the GCP instance with Ubuntu

Runs these commands on a terminal to prepare the instance to run PM4NGS

```
veraalva@instance-1:~$ sudo apt-get update
veraalva@instance-1:~$ sudo apt-get install docker.io python3 python3-pip python3-
↪venv python3-dev poppler-utils gcc nodejs tree
veraalva@instance-1:~$ sudo usermod -aG docker $USER
veraalva@instance-1:~$ logout
```

Close and reopen the terminal to set the docker group in the user. Then, click on the SSH button again to re-launch the terminal.

4.3 Installing PM4NGS

Creates a Python virtual environment named: **pm4ngs_venv** for installing PM4NGS

```
veraalva@instance-1:~$ python3 -m venv pm4ngs_venv
veraalva@instance-1:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@instance-1:~$ pip install wheel
(pm4ngs_venv) veraalva@instance-1:~$ pip install pm4ngs
```

4.4 Using PM4NGS

Open a terminal and activate the **pm4ngs_venv** virtual environment

```
veraalva@instance-1:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@instance-1:~$ pm4ngs-chipexo --version
PM4NGS version: 0.0.4
(pm4ngs_venv) veraalva@instance-1:~$
```

4.5 Running the ChIP-exo demo

Open a terminal and activate the **pm4ngs_venv** virtual environment

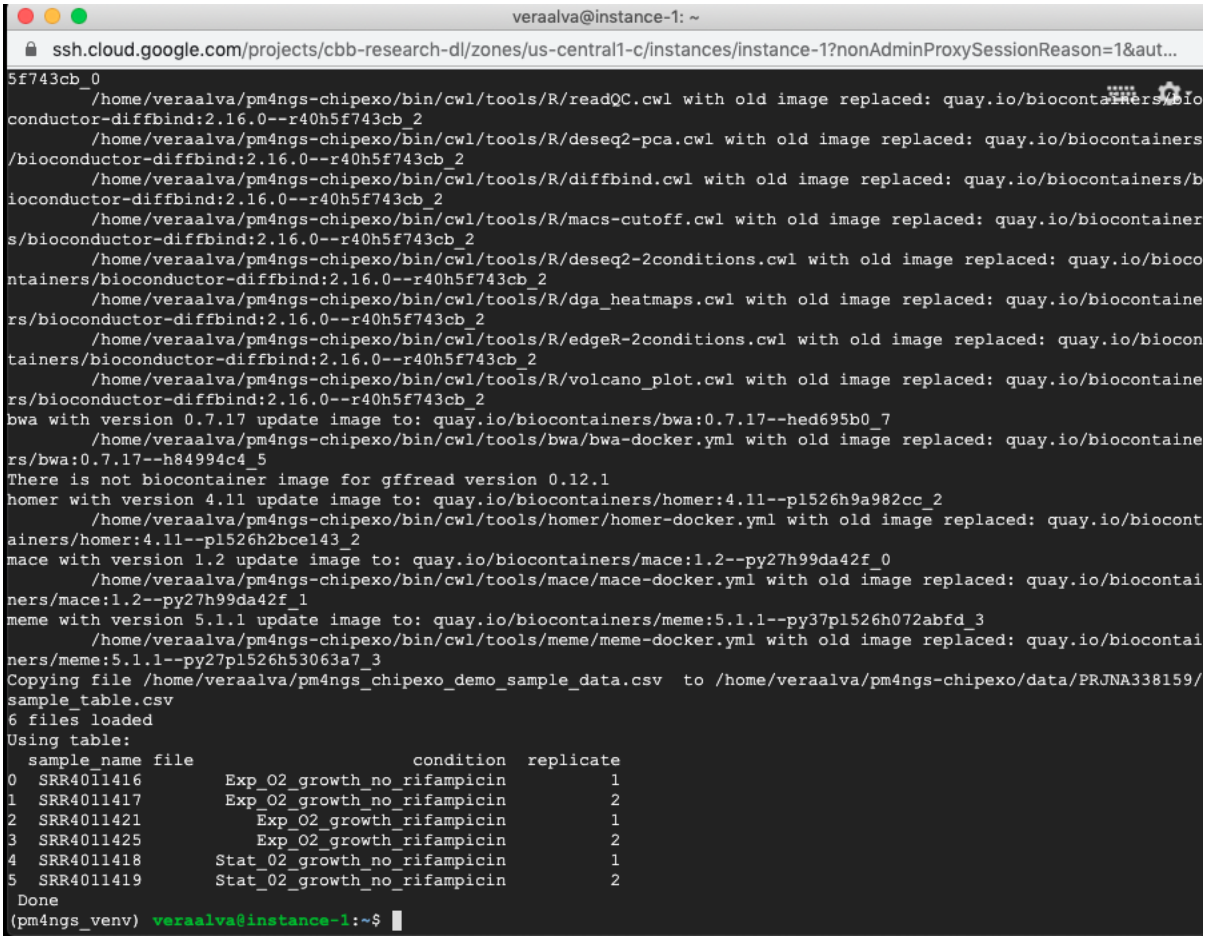
```
veraalva@instance-1:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@perseo:~$ pm4ngs-chipexo-demo
Generating demo for ChIP-exo data analysis project
Downloading file: pm4ngs_chipexo_demo_config.yaml
Downloading file: pm4ngs_chipexo_demo_sample_data.csv
Using config file: pm4ngs_chipexo_demo_config.yaml
{
    "author_name": "Roberto Vera Alvarez",
    "user_email": "veraalva@ncbi.nlm.nih.gov",
    "project_name": "pm4ngs-chipexo",
    "dataset_name": "PRJNA338159",
    "is_data_in_SRA": "y",
    "sequencing_technology": "single-end",
    "create_demo": "n",
    "number_spots": "1000000",
    "organism": "Escherichia coli",
    "genome_name": "NC_000913.3",
    "genome_dir": "{{ cookiecutter.genome_name}}",
    "aligner_index_dir": "{{ cookiecutter.genome_dir}}/BWA/",
    "genome_fasta": "{{ cookiecutter.genome_dir}}/NC_000913.3.fa",
    "genome_gtf": "{{ cookiecutter.genome_dir}}/NC_000913.3.gtf",
    "genome_chromsizes": "{{ cookiecutter.genome_dir}}/NC_000913.3.sizes",
    "use_docker": "y",
    "max_number_threads": "32"
}
Cloning Git repo: https://github.com/ncbi/cwl-ngs-workflows-cbb to /home/veraalva/
↪pm4ngs-chipexo/bin/cwl
Updating CWLs dockerPull and SoftwareRequirement from: /home/veraalva/pm4ngs-chipexo/
↪requirements/conda-env-dependencies.yaml
bamscale with version 0.0.3 update image to: quay.io/biocontainers/bamscale:0.0.3--
↪ha85820d_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bamscale/bamscale-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bamscale:0.0.5--h18f8b1d_1
bedtools with version 2.29.2 update image to: quay.io/biocontainers/bedtools:2.29.2--
↪hc088bd4_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bedtools/bedtools-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bedtools:2.28.0--hdf88d34_0
bioconductor-diffbind with version 2.16.0 update image to: quay.io/biocontainers/
↪bioconductor-diffbind:2.16.0--r40h5f743cb_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-pca.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/macs-cutoff.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
```

(continues on next page)

(continued from previous page)

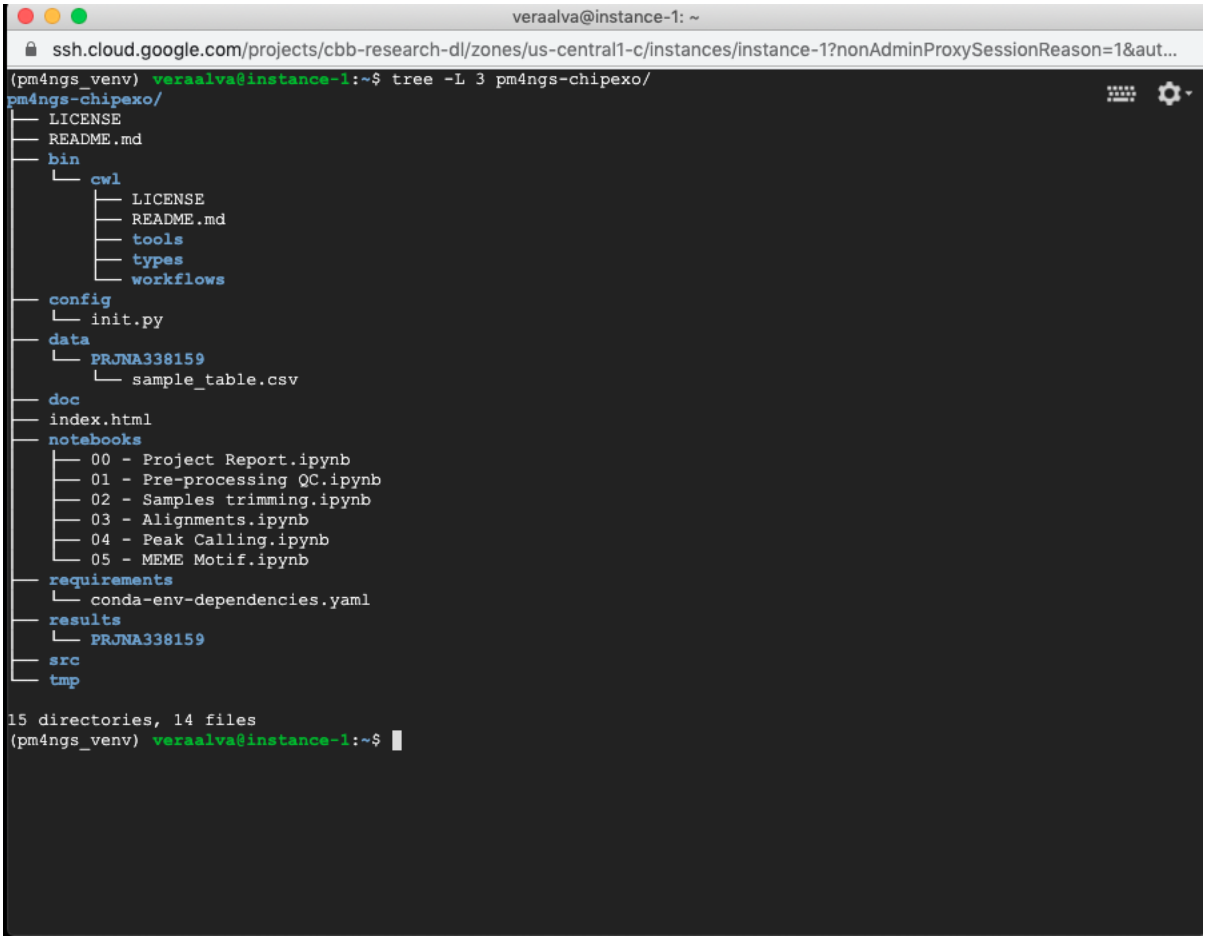
```
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/dga_heatmaps.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/diffbind.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/edgeR-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/volcano_plot.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/readQC.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
bwa with version 0.7.17 update image to: quay.io/biocontainers/bwa:0.7.17--hed695b0_7
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bwa/bwa-docker.yml with old image␣
↪replaced: quay.io/biocontainers/bwa:0.7.17--h84994c4_5
There is not biocontainer image for gffread version 0.12.1
homer with version 4.11 update image to: quay.io/biocontainers/homer:4.11--
↪pl526h9a982cc_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/homer/homer-docker.yml with old␣
↪image replaced: quay.io/biocontainers/homer:4.11--pl526h2bce143_2
mace with version 1.2 update image to: quay.io/biocontainers/mace:1.2--py27h99da42f_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/mace/mace-docker.yml with old image␣
↪replaced: quay.io/biocontainers/mace:1.2--py27h99da42f_1
meme with version 5.1.1 update image to: quay.io/biocontainers/meme:5.1.1--
↪py37pl526h072abfd_3
    /Users/veraalva/my_ngs_project/bin/cwl/tools/meme/meme-docker.yml with old image␣
↪replaced: quay.io/biocontainers/meme:5.1.1--py27pl526h53063a7_3
Copying file /Users/veraalva/Work/Developer/Python/pm4ngs/pm4ngs-chipexo/example/
↪pm4ngs_chipexo_demo_sample_data.csv  to /Users/veraalva/my_ngs_project/data/my_
↪dataset_name/sample_table.csv
6 files loaded
Using table:
  sample_name file                      condition  replicate
0  SRR4011416        Exp_O2_growth_no_rifampicin          1
1  SRR4011417        Exp_O2_growth_no_rifampicin          2
2  SRR4011421           Exp_O2_growth_rifampicin          1
3  SRR4011425           Exp_O2_growth_rifampicin          2
4  SRR4011418       Stat_O2_growth_no_rifampicin          1
5  SRR4011419       Stat_O2_growth_no_rifampicin          2
 Done
```

The terminal will looks like the next image.



Running the command tree to show the project structure

```
(pm4ngs_venv) veraalva@instance-1:~$ tree -L 3 pm4ngs-chipexo/
```

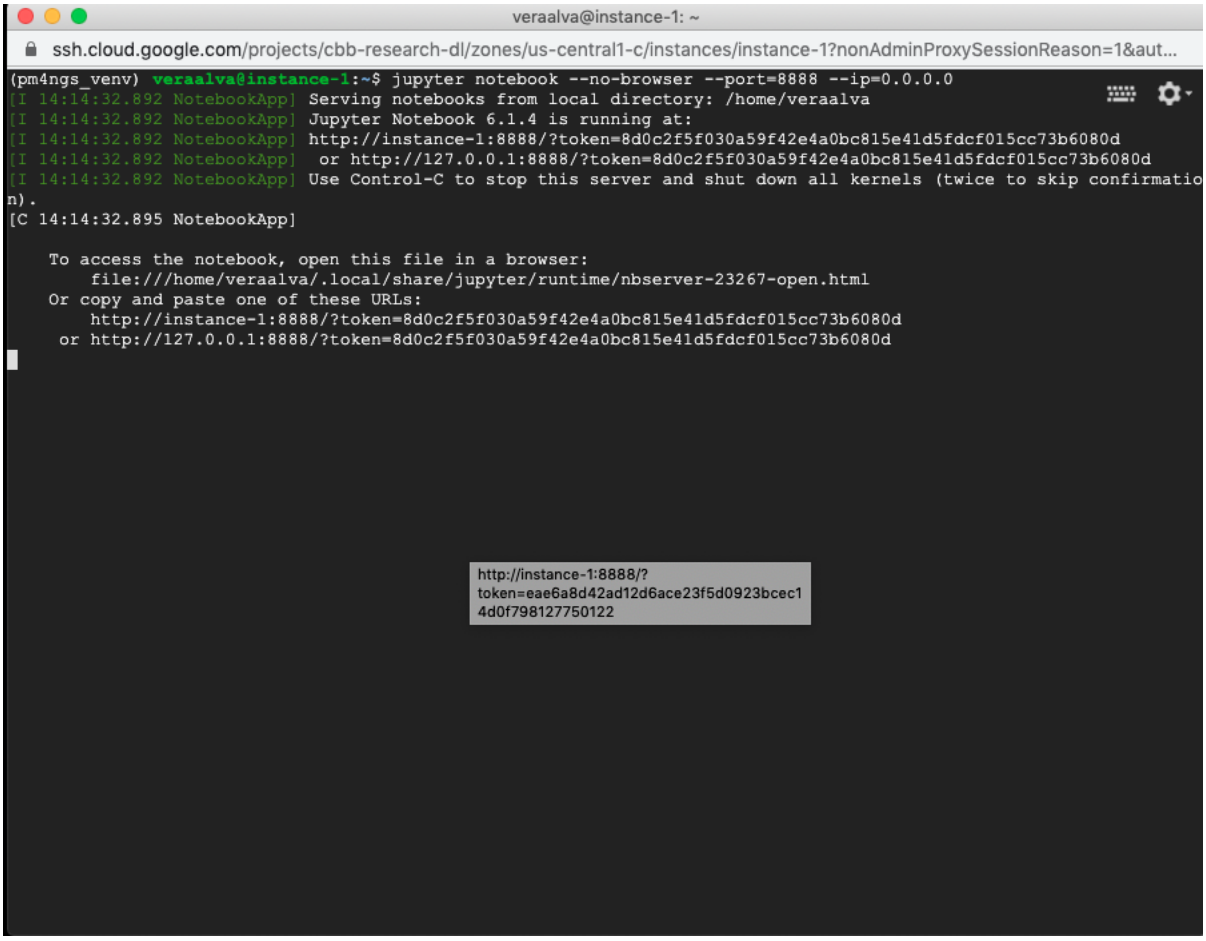


4.6 Running the Jupyter Server

Open a terminal and activate the **pm4ngs_venv** virtual environment and run the jupyter server. As the GCP instance is a remote computer, we need to run the jupyter server with the **–port** and **–ip** options.

```
(pm4ngs_venv) veraalva@instance-1:~$ jupyter notebook --no-browser --port=8888 --ip=0.
↪0.0.0
[I 14:12:52.956 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 14:12:52.956 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 14:12:52.956 NotebookApp] http://instance-1:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp]  or http://127.0.0.1:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmatio
n).
[C 14:12:52.959 NotebookApp]

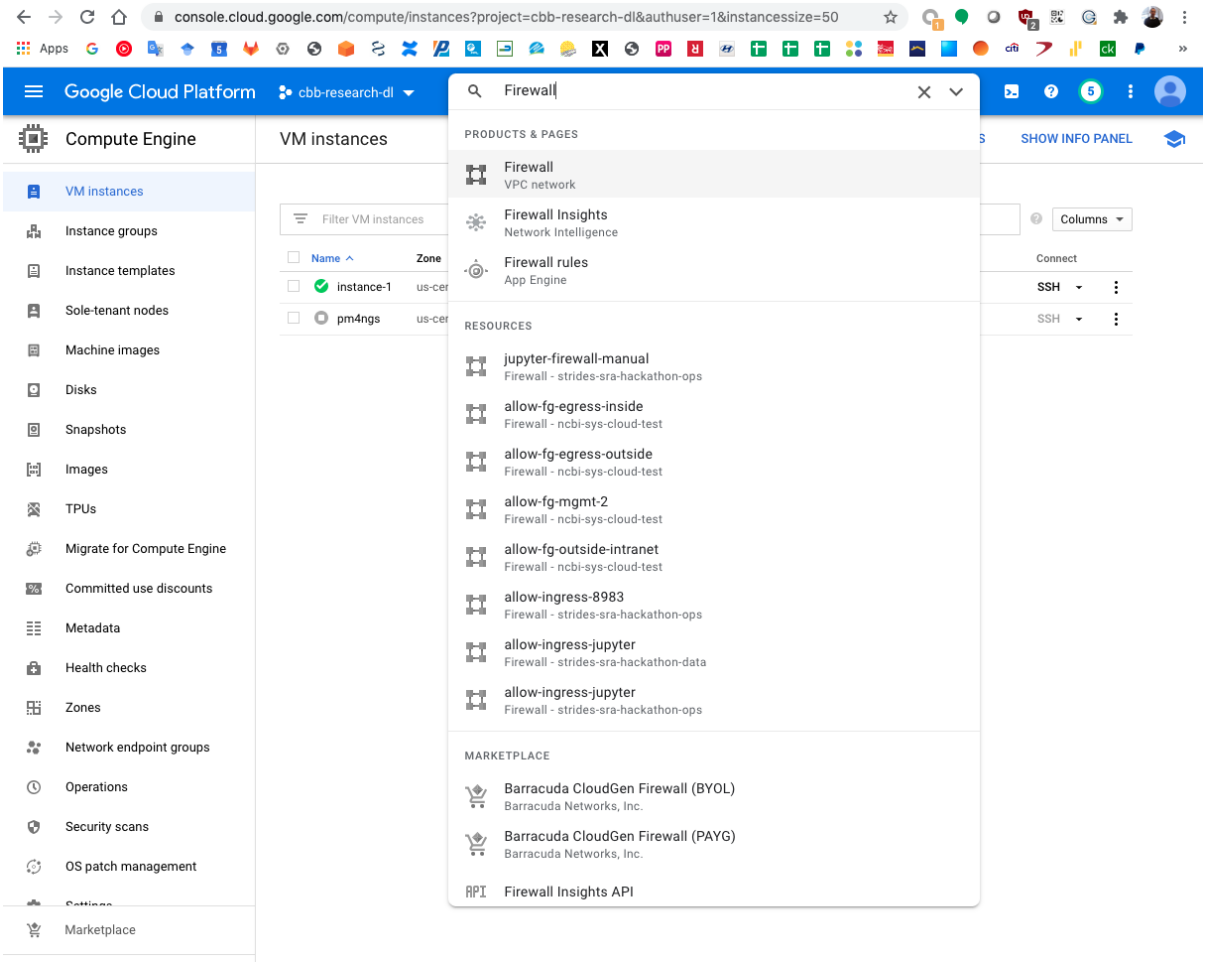
    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-23251-open.html
    Or copy and paste one of these URLs:
        http://instance-1:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
     or http://127.0.0.1:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
```



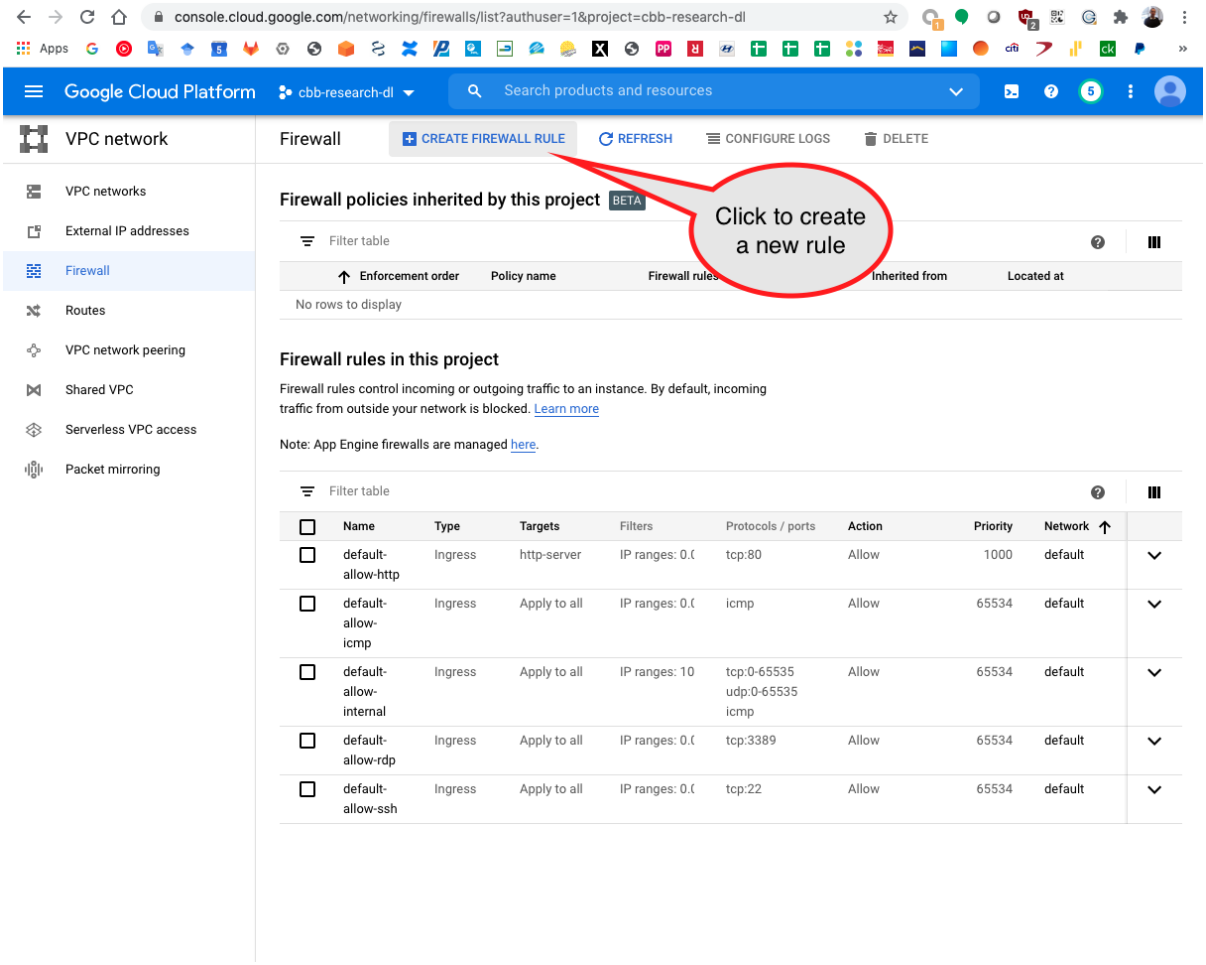
4.7 Open a firewall rule for accessing the Jupyter Server

A GCP firewall rule should be created to access the Jupyter server remotely. From your desktop computer.

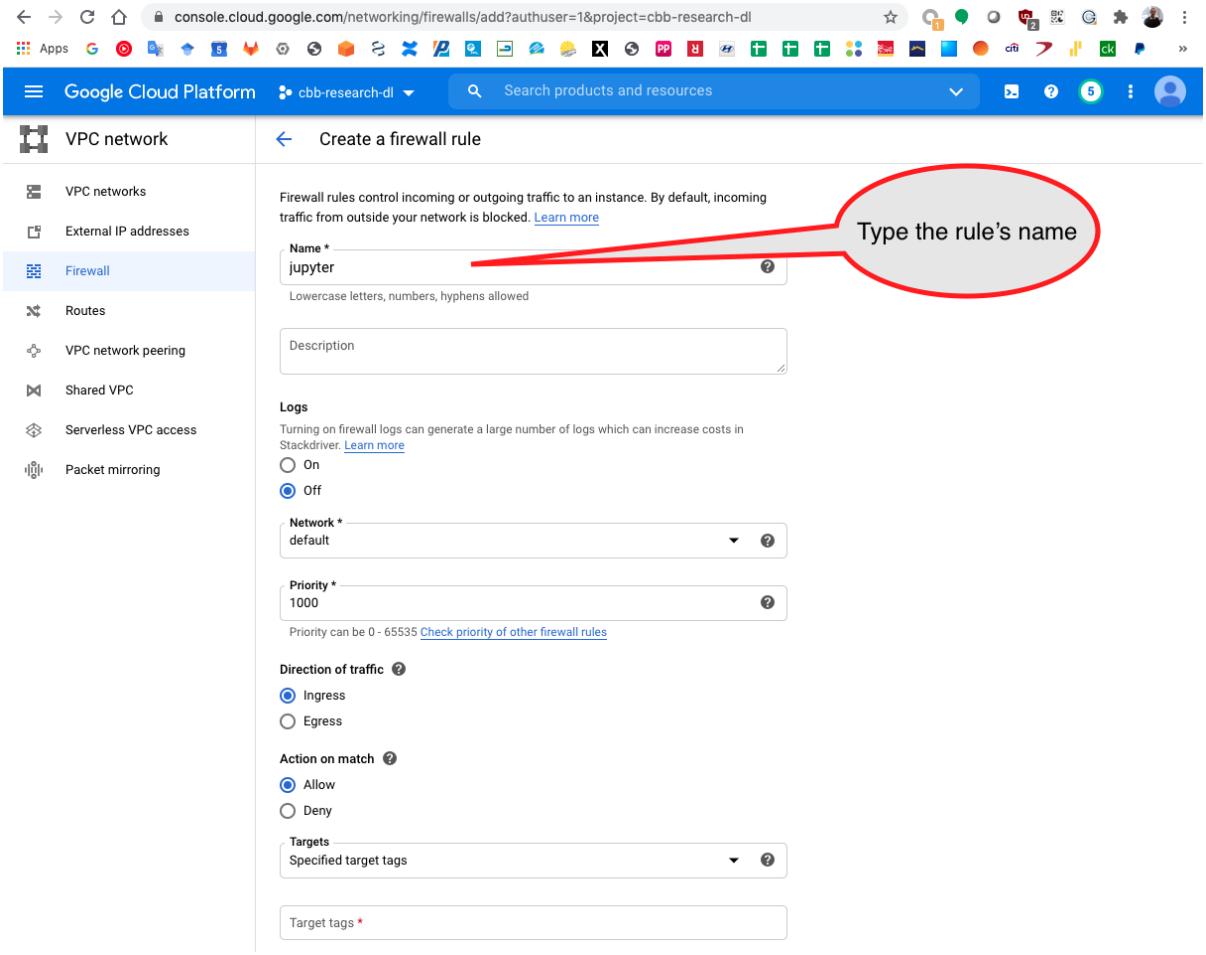
Search for **Firewall** in the GCP search bar.



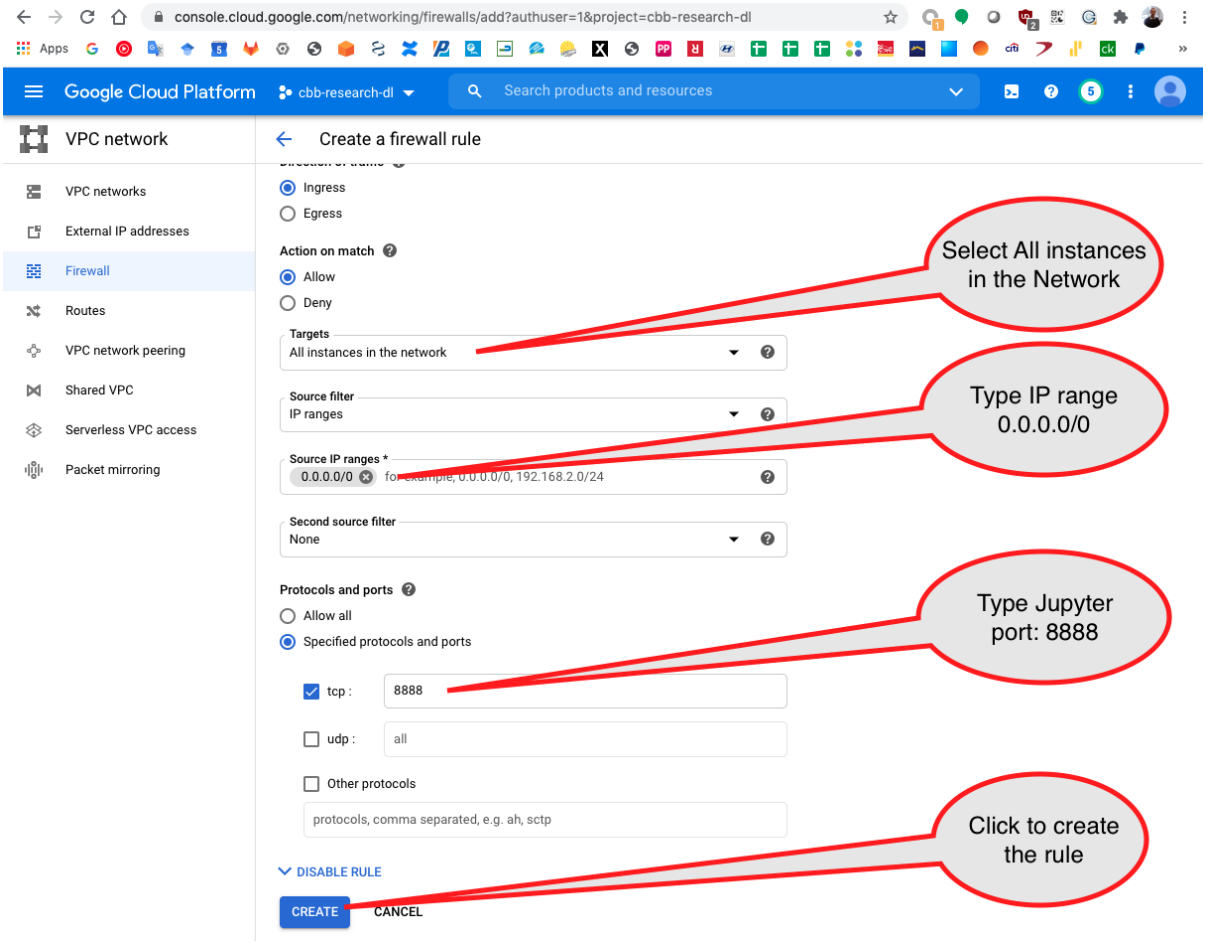
Click on the **Create Firewall Rule** button.



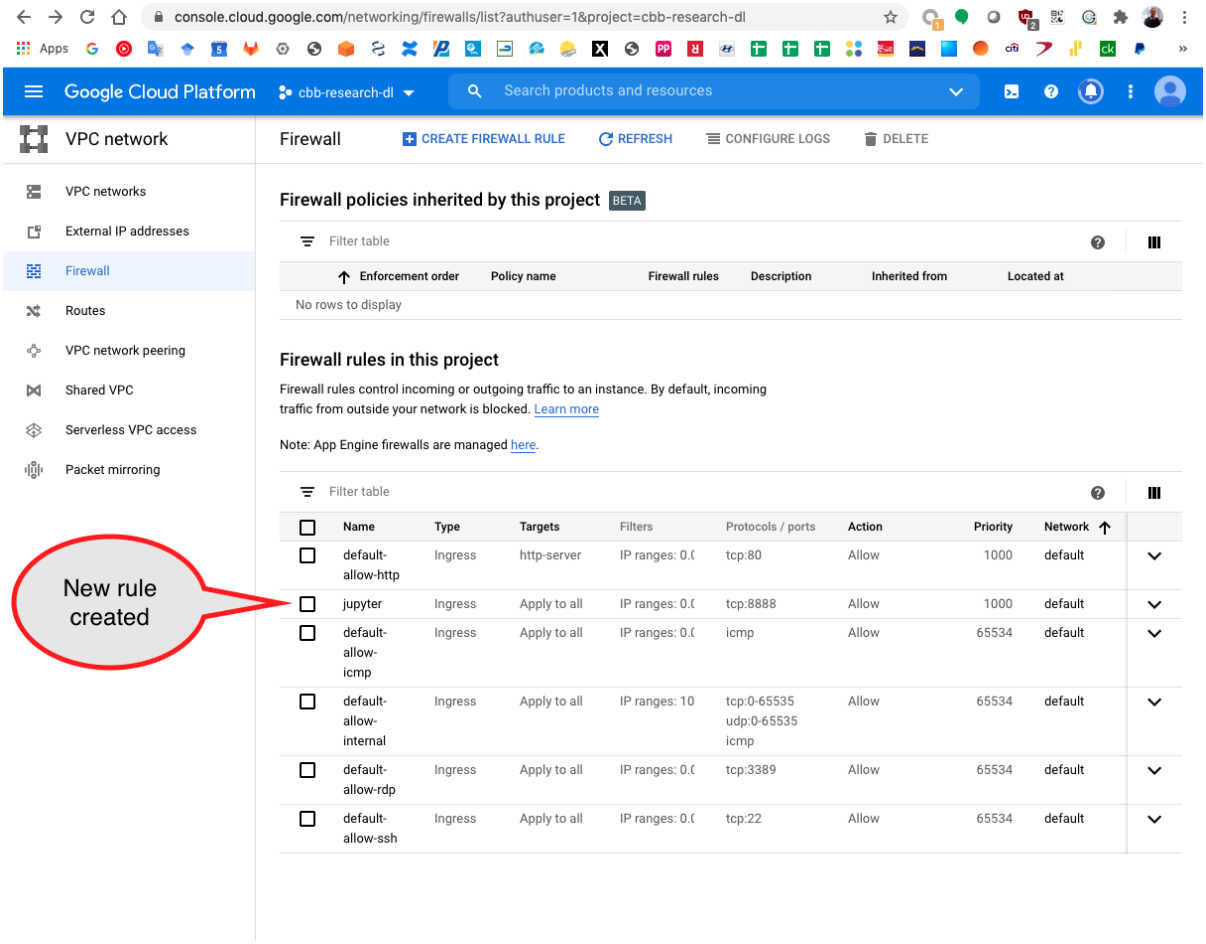
Add the rules name



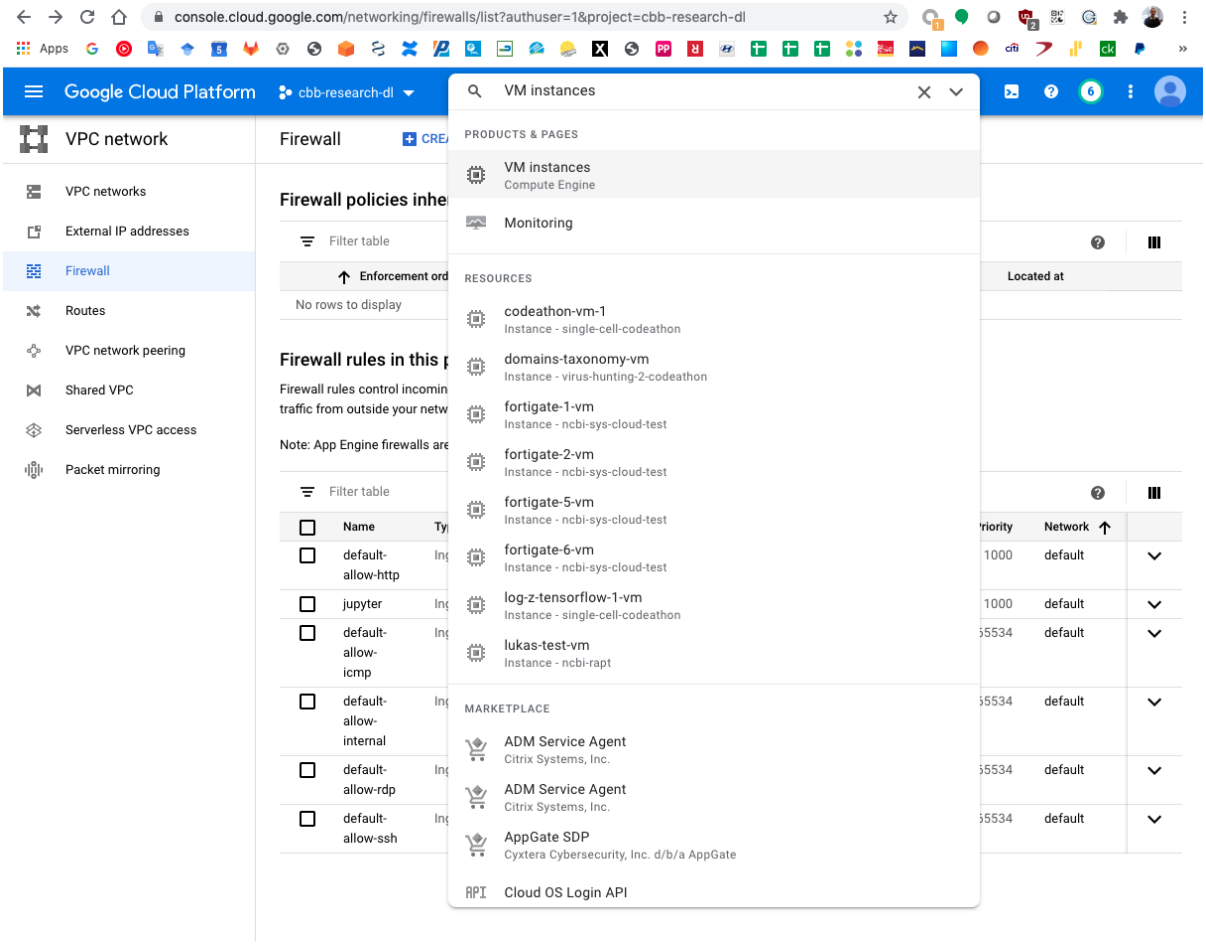
Add the Jupyter port used that is **8888** and click on create.



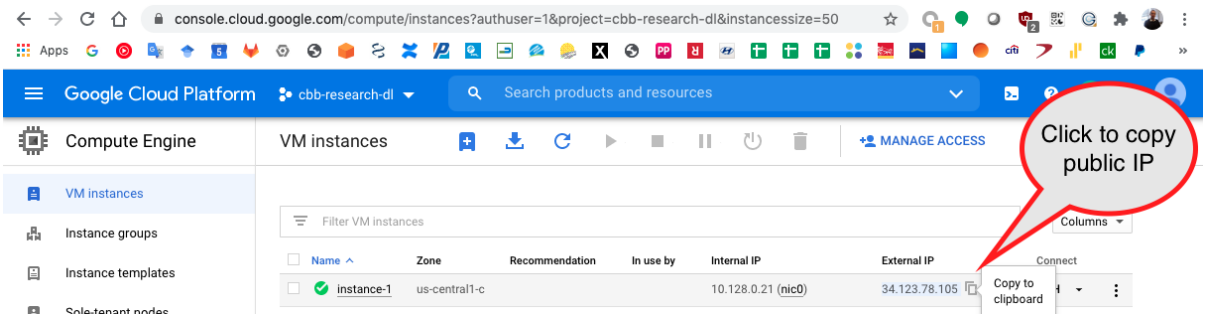
The new rule is created and available



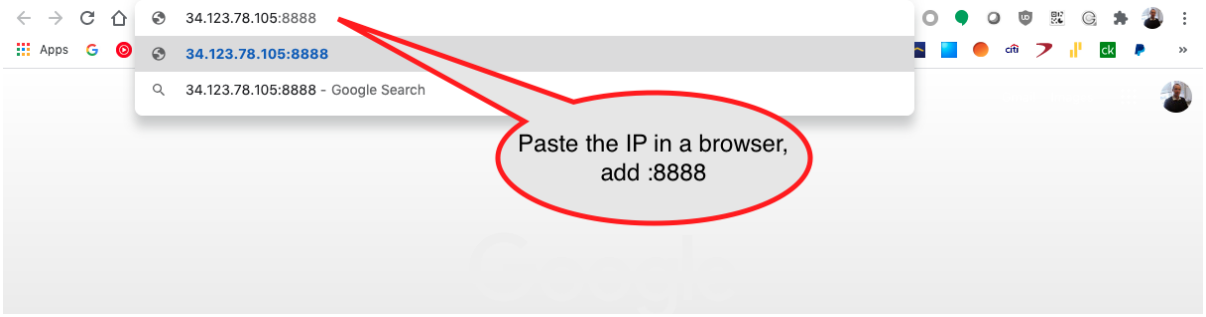
Go back to the VM instances to copy the instance public IP



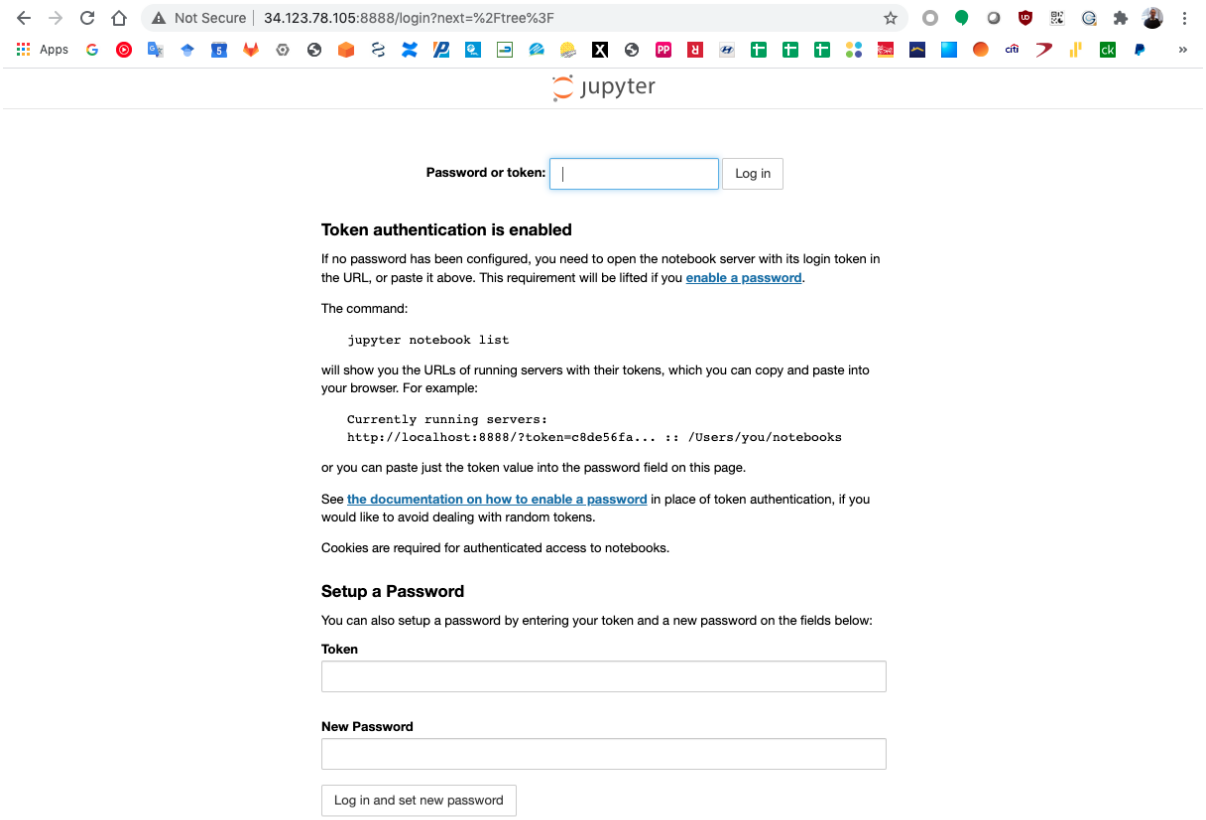
Copy the instance public IP to the clipboard



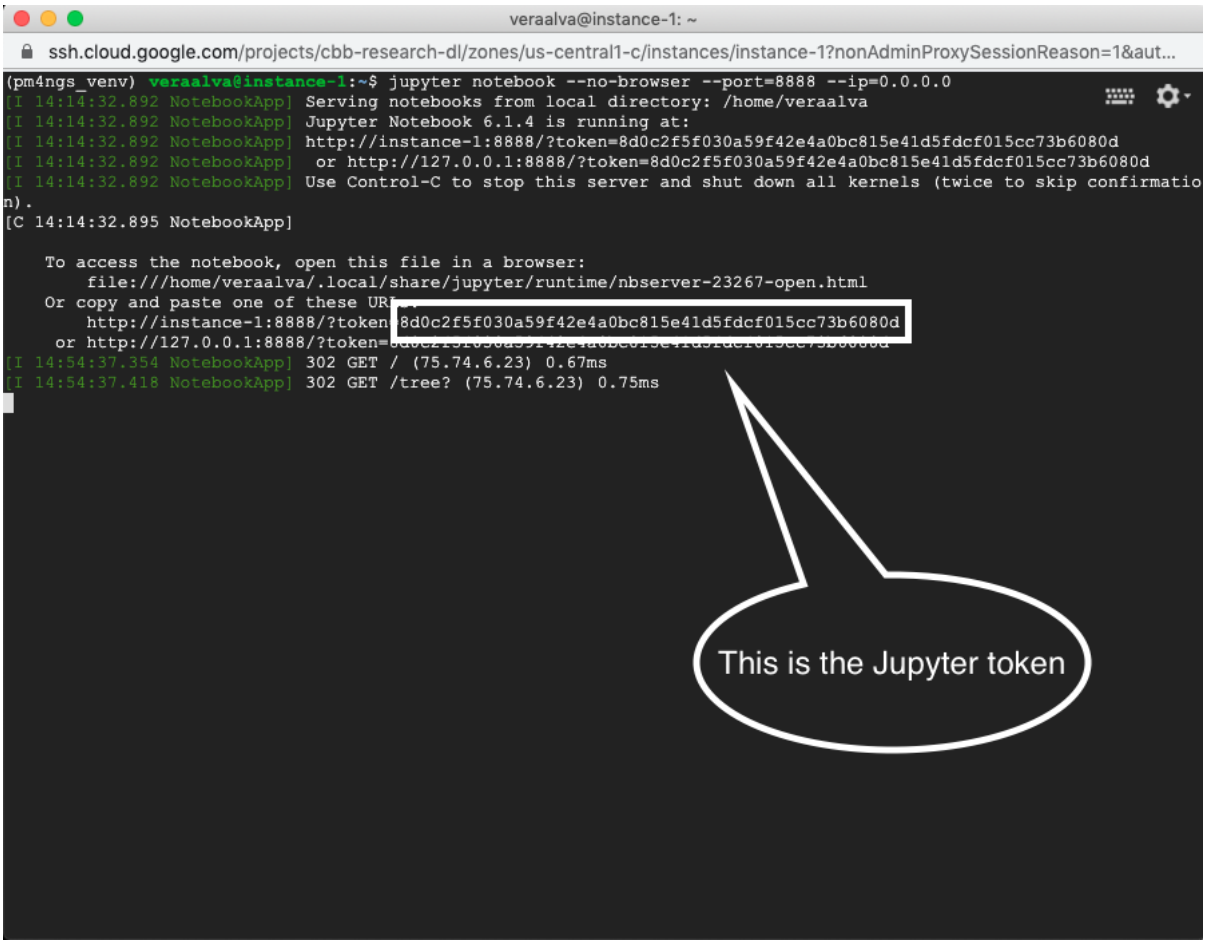
Copy the URL with localhost in a browser adding **:8888** which is the Jupyter server port.



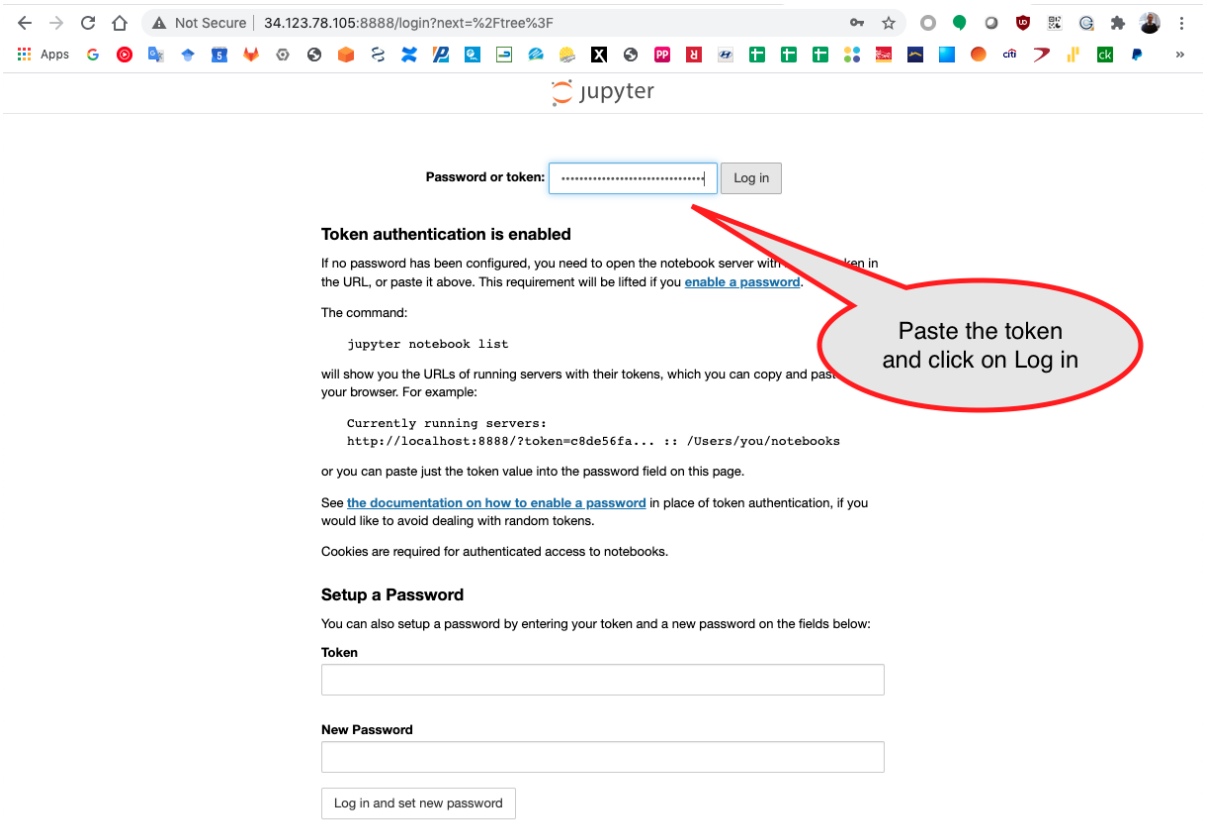
The Jupyter server uses a securoty token to secure the access to your notebooks.



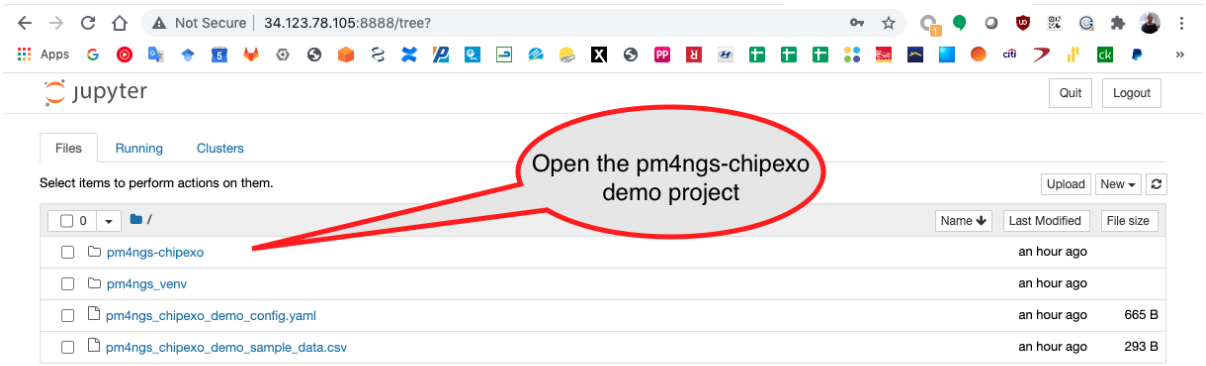
Go to the SSH terminal and copy the Jupyter token.



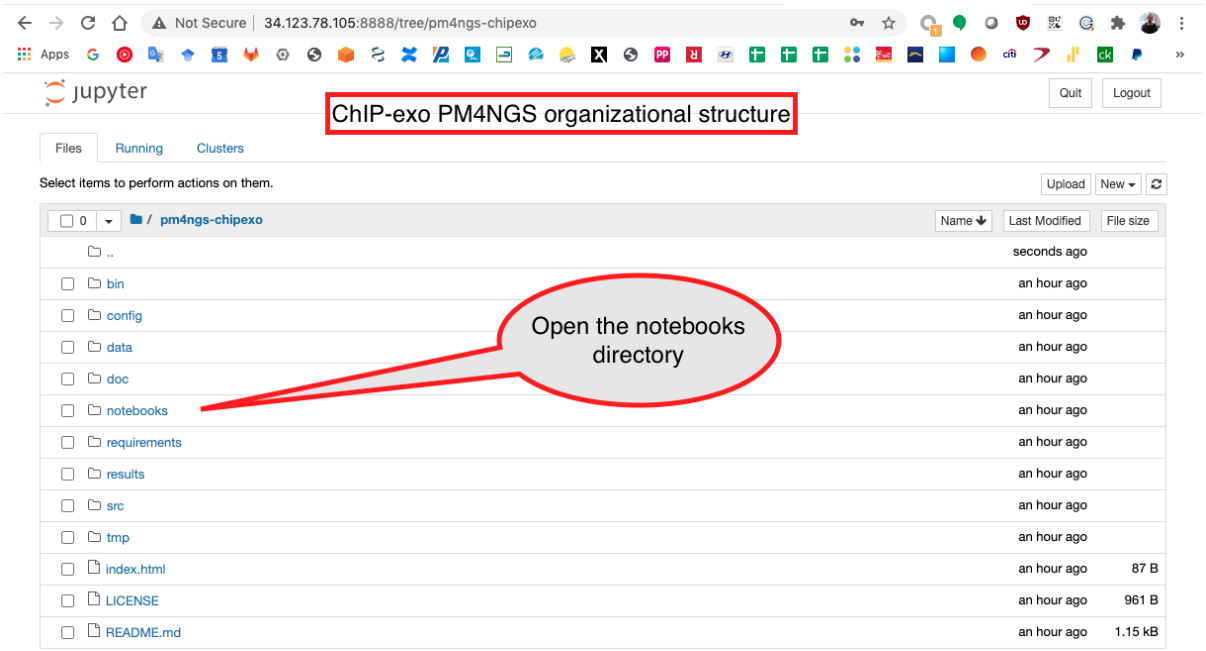
Paste the token in the input bar and click Log in



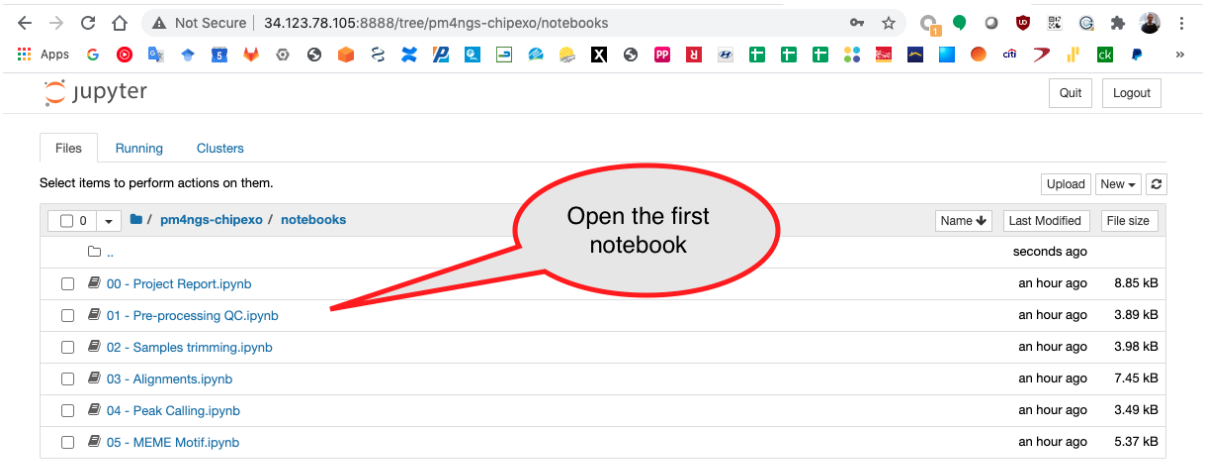
Open the **pm4ngs-chipexo** directory



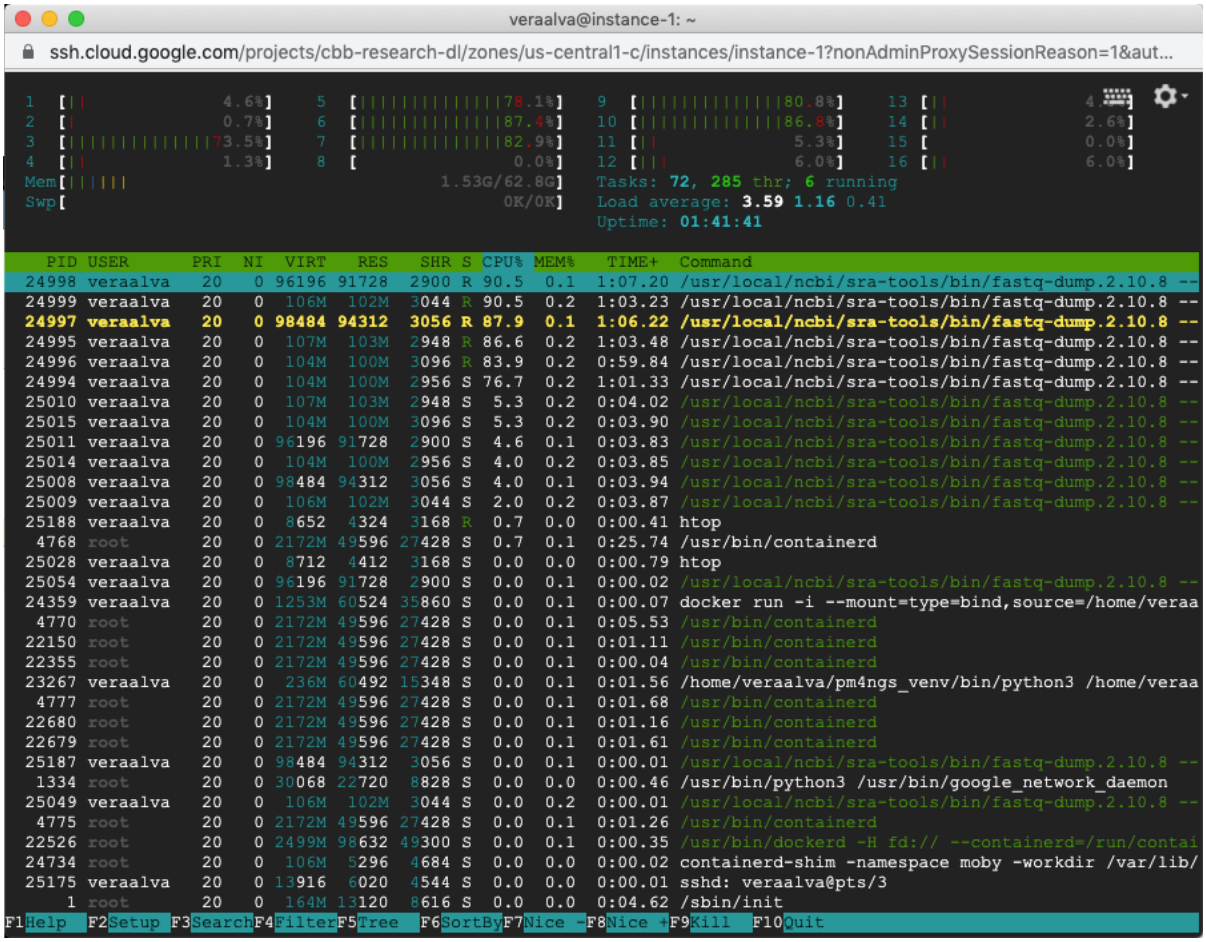
Then, open the **notebooks** directory



Start running the notebook **01 - Pre-processing QC.ipynb**



Open a different VM terminal to run the command **htop** to see the process running. In this casewe are seeing multiple **fastq-dump** command being executed.

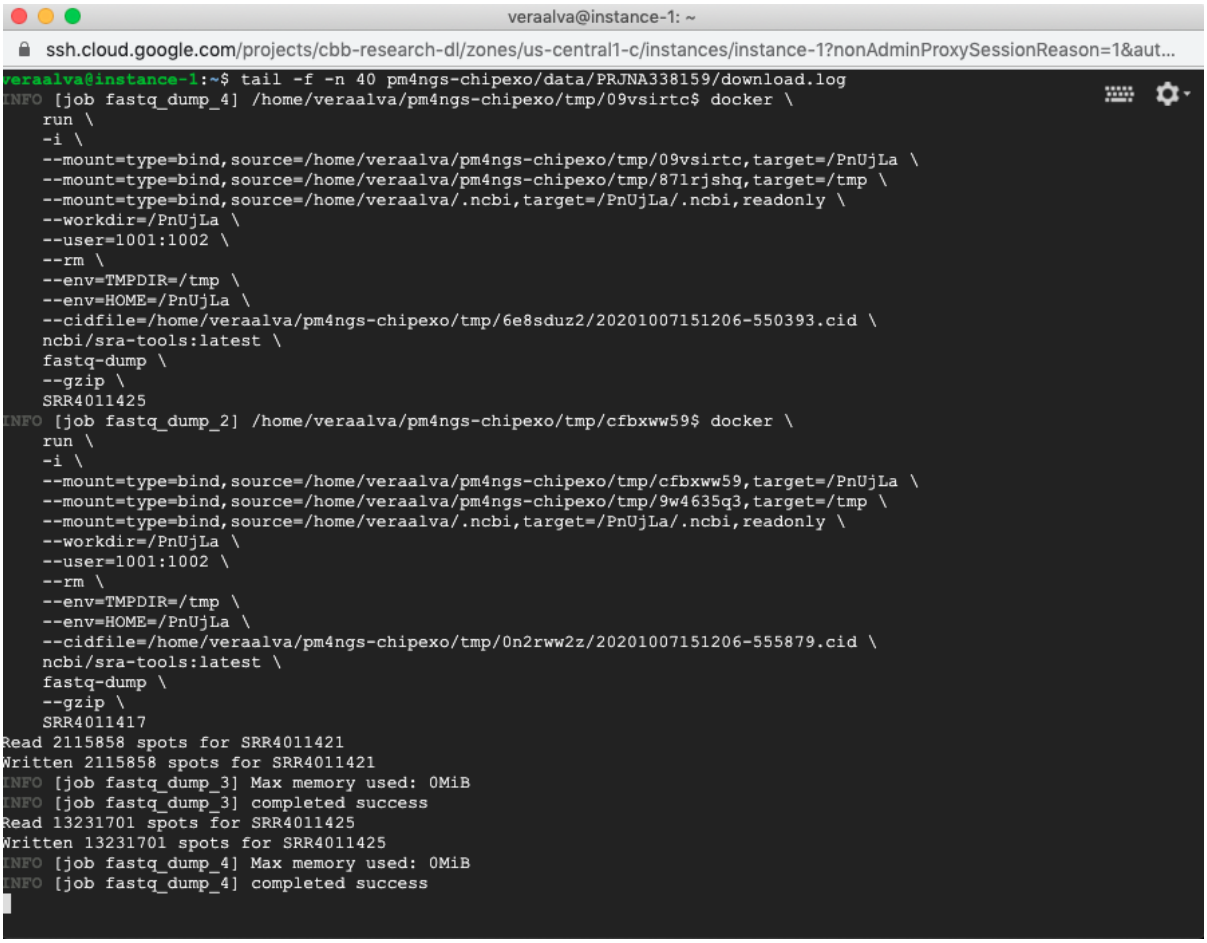


Wait for that process to finish. The log can be checked running the **Checking command output** cell

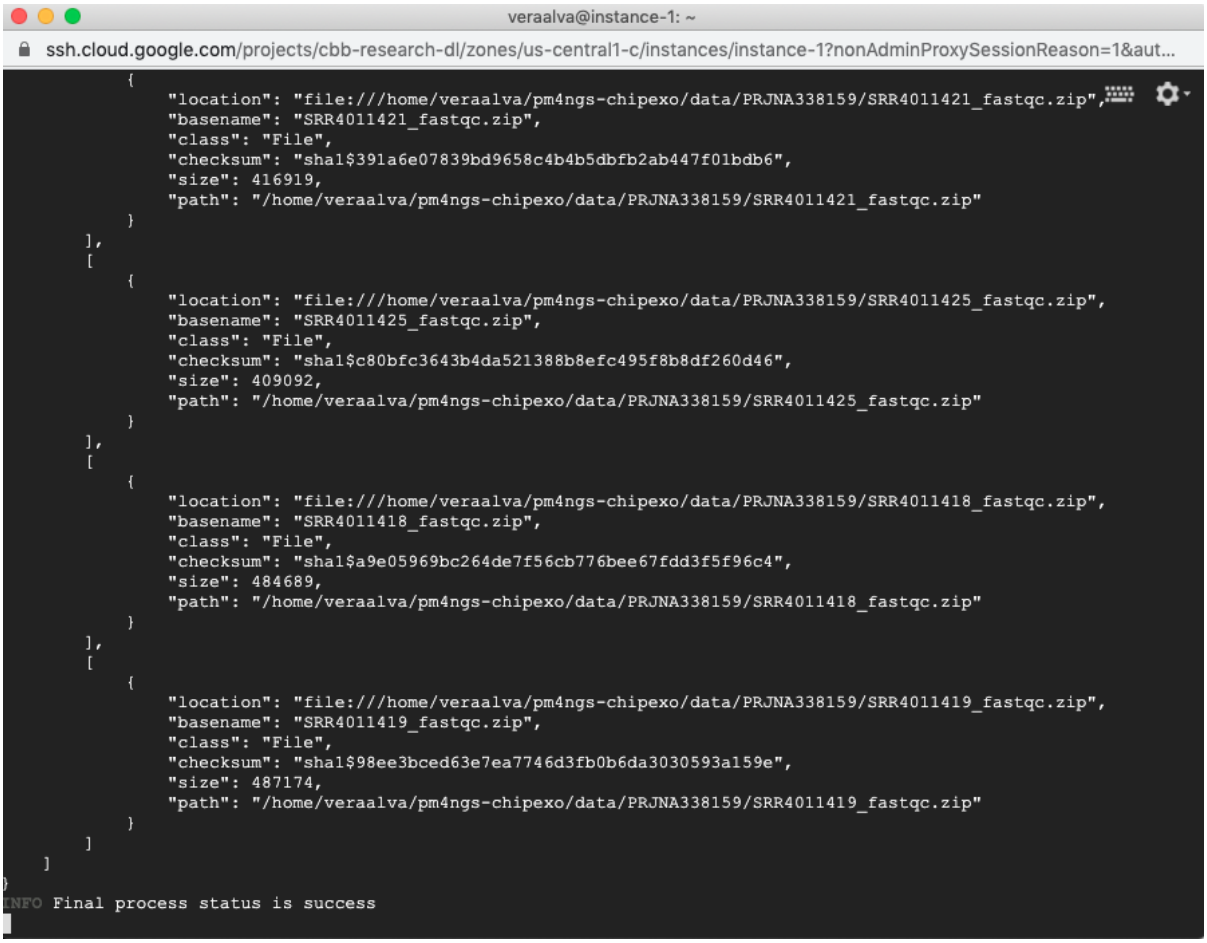


In the VM terminal you can use the command **tail** to see the process log

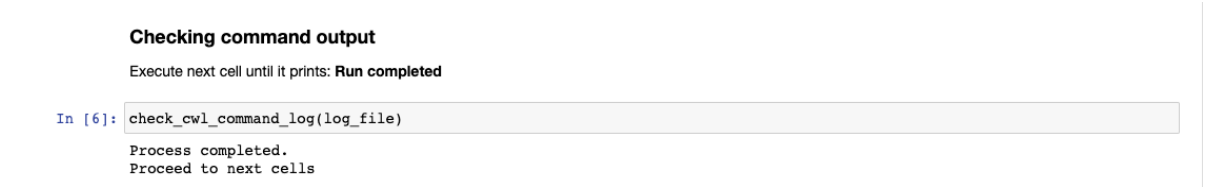
```
(pm4ngs_venv) veraalva@instance-1:~$ tail -f -n 40 pm4ngs-chipexo/data/PRJNA338159/
↪download.log
```



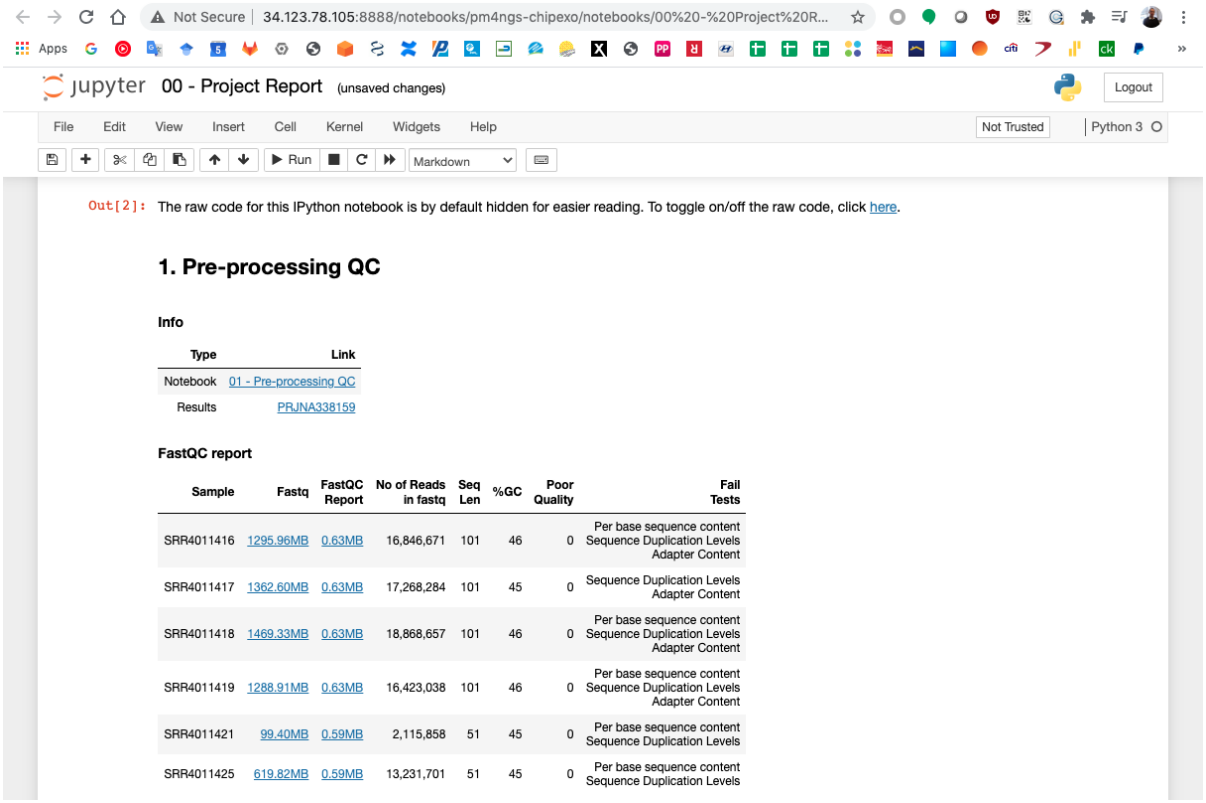
The process will finish with a message: **Final process status is success**



Running the **Checking command output** cell again



Finish the **01 - Pre-processing QC.ipynb** notebook and go to the project report **00 - Project Report.ipynb**. Execute the first and second cell to visualize the Pre-processing report table.



Follow this procedure for each notebook in the project.

CHAPTER 5

Sample sheet

A CSV file describing samples, conditions and replicate number is required during the PM4NGS project creation.

PM4NGS will copy the sample sheet file to the folder **data/{{dataset_name}}** with the standard name **sample_table.csv**.

5.1 Single-end example

This is an sample sheet example for **single-end** sequencing technology data:

sample_name	file	condition	replicate
SRR4011416	/net/rawdata/SRR4011416.fastq.gz	Exp_O2_growth_no_rifampicin	1
SRR4011417	/net/rawdata/SRR4011417.fastq.gz	Exp_O2_growth_no_rifampicin	2
SRR4011421	/net/rawdata/SRR4011421.fastq.gz	Exp_O2_growth_rifampicin	1
SRR4011425	/net/rawdata/SRR4011425.fastq.gz	Exp_O2_growth_rifampicin	2
SRR4011418	/net/rawdata/SRR4011418.fastq.gz	Stat_02_growth_no_rifampicin	1
SRR4011419	/net/rawdata/SRR4011419.fastq.gz	Stat_02_growth_no_rifampicin	2

Table source: sample_sheet_single_end.csv

5.2 Paired-end example

For **paired-end** sequencing technology data, the | should be used to separate forward and reverse fastq files:

sample_name	file	condition	replicate
SRR2126784	http://myserver.net/SRR2126784_1.fastq.gz|http://myserver.net/SRR2126784_2.fastq.gz	PRE_NACT	1
SRR2126785	http://myserver.net/SRR2126785_1.fastq.gz|http://myserver.net/SRR2126785_2.fastq.gz	PRE_NACT	1
SRR2126786	http://myserver.net/SRR2126786_1.fastq.gz|http://myserver.net/SRR2126786_2.fastq.gz	PRE_NACT	1
SRR2126787	http://myserver.net/SRR2126787_1.fastq.gz|http://myserver.net/SRR2126787_2.fastq.gz	PRE_NACT	1
SRR3383790	http://myserver.net/SRR3383790_1.fastq.gz|http://myserver.net/SRR3383790_2.fastq.gz	PRE_NACT	1

Source: sample_sheet_paired_end.csv

PM4NGS will copy or download the raw fastq files to the **data/{{dataset_name}}/** directory during the project creation if the [–copy-rawdata] is used.

5.3 Processing data from the NCBI SRA

For data in the NCBI SRA database, the *file* column should be empty. PM4NGS will download the files during the pre-processing quality control step.

sample_name	file	condition	replicate
SRR7549105		ES	1
SRR7549106		ES	2
SRR7549109		MEF	1
SRR7549110		MEF	2
SRR7549114		rB	1
SRR7549113		rB	2

Source: sample_sheet_sra.csv

5.4 Sample sheet column names and description

Note: Columns names are required and are case sensitive.

Columns

- **sample_name**: Sample names. It can be different of sample file name.
- **file**: This is the absolute path or URL to the raw fastq file.

  For paired-end data the files should be separated using the unix pipe | as **SRR4053795_1.fastq.gz|SRR4053795_2.fastq.gz** must exist.

  The data files will be copied to the folder **data/{{dataset_name}}/**.
- **condition**: Conditions to group the samples. Use only alphanumeric characters.

  For RNASeq projects the differential gene expression will be generated comparing these conditions. If there are multiple conditions all comparisons will be generated. It must be at least two conditions.

  For ChIPSeq projects differential binding events will be detected comparing these conditions. If there are multiple conditions all comparisons will be generated. It must be at least two conditions.

  For ChIPexo projects the samples of the same condition will be grouped for the peak calling with MACE.
- **replicate**: Replicate number for samples.

CHAPTER 6

Differential Gene expression and GO enrichment from RNA-Seq data

Differential expression (DE) analysis allows the comparison of RNA expression levels between multiple conditions.

The differential gene expression and GO enrichment pipeline is comprised of five steps, shown in next Table. The first step involves downloading samples from the NCBI SRA database, if necessary, or executing the pre-processing quality control tools on local samples. Subsequently, sample trimming, alignment, and quantification processes are executed. Once all samples are processed, groups of differentially expressed genes are identified per condition, using DESeq and EdgeR. Over- and under-expressed genes are reported by each program, and the interception of their results is computed.

Finally, once differentially expressed genes are identified, a GO enrichment analysis is executed to provide key biological processes, molecular functions, and cellular components for identified genes.

Table 1: DGA and Go enrichment, RNASeq pipeline

Step	Jupyter Notebook	Workflow CWL	Tool	Input	Output	CWL Tool
Sample Download and Quality Control	01 - Pre-processing QC	download_quality_control.cwl	SRA Tools	SRA accession	Fastq	fastq-dump.cwl
			FastQC	Fastq	FastQC HTML and Zip	fastqc.cwl
Trimming	02 - Samples trimming		Trimmomatic	Fastq	Fastq	trimmomatic-PE.cwl trimmomatic-SE.cwl
Alignments and Quantification	03 - Alignments and Quantification	rnaseq-alignment-quantification.cwl	STAR	Fastq	BAM	star.cwl
			Samtools	SAM	BAM Sorted BAM BAM index BAM stats BAM flagstats	samtools-view.cwl samtools-sort.cwl samtools-index.cwl samtools-stats.cwl samtools-flagstat.cwl
			IGVtools	Sorted BAM	TDF	igvtools-count.cw
			RSeQC	Sorted BAM	Alignment quality control (TXT and PDF)	rseqc-bam_stat.cwl rseqc-infer_experiment.cwl rseqc-junction_annotation.cwl rseqc-junction_saturation.cwl rseqc-read_distribution.cwl rseqc-read_quality.cwl
			TPMCalculator	Sorted BAM	Read counts (TSV) TPM values (TSV)	tpmcalculator.cwl
Differential Gene Analysis	04 - DGA		DeSEq2	Read Matrix	TSV	deseq2-2conditions.cwl
			EdgeR	Read Matrix	TSV	edgeR-2conditions.cwl
Go enrichment	05 - GO enrichment		goenrichment	gene IDs	TSV	Python code in the notebook

6.1 Input requirements

The input requirement for the DGA pipeline is the *Sample sheet* file.

6.2 Pipeline command line

The RNASeq based project can be created using the following command line:

```
localhost:~> pm4ngs-rnaseq
usage: Generate a PM4NGS project for RNA-Seq data analysis [-h] [-v]
                                                           --sample-sheet
                                                           SAMPLE_SHEET
                                                           [--config-file CONFIG_FILE]
                                                           [--copy-rawdata]
```

Options:

- **sample-sheet**: Sample sheet with the samples metadata
- **config-file**: YAML file with configuration for project creation
- **copy-rawdata**: Copy the raw data defined in the sample sheet to the project structure. (The data can be hosted locally or in an http server)

6.3 Creating the DGA and GO enrichment from RNA-Seq data project

The **pm4ngs-rnaseq** command line executed with the **–sample-sheet** option will let you type the different variables required for creating and configuring the project. The default value for each variable is shown in the brackets. After all questions are answered, the CWL workflow files will be cloned from the github repo ncbi/cwl-ngs-workflows-cbb to the folder **bin/cwl**.

```
localhost:~> pm4ngs-rnaseq --sample-sheet my-sample-sheet.tsv
Generating RNA-Seq data analysis project
author_name [Roberto Vera Alvarez]:
email [veraalva@ncbi.nlm.nih.gov]:
project_name [my_ngs_project]:
dataset_name [my_dataset_name]:
is_data_in_SRA [y]:
Select sequencing_technology:
1 - single-end
2 - paired-end
Choose from 1, 2 [1]: 2
create_demo [y]: y
number_spots [1000000]:
organism [human]:
genome_name [hg38]:
genome_dir [hg38]:
aligner_index_dir [hg38/STAR]:
genome_fasta [hg38/genome.fa]:
genome_gtf [hg38/genes.gtf]:
genome_bed [hg38/genes.bed]:
fold_change [2.0]:
fdr [0.05]:
use_docker [y]:
max_number_threads [16]:
Cloning Git repo: https://github.com/ncbi/cwl-ngs-workflows-cbb to /home/veraalva/tmp/
↪cookiecutter/my_ngs_project/bin/cwl
```

(continues on next page)

(continued from previous page)

```
Updating CWLs dockerPull and SoftwareRequirement from: /home/veraalva/my_ngs_project/
↪requirements/conda-env-dependencies.yaml
bioconductor-diffbind with version 2.16.0 update image to: quay.io/biocontainers/
↪bioconductor-diffbind:2.16.0--r40h5f743cb_0
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-pca.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/macs-cutoff.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/dga_heatmaps.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/diffbind.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/edgeR-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/volcano_plot.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/readQC.cwl with old image replaced:␣
↪quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /home/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
Copying file /home/veraalva/Work/Developer/Python/pm4ngs/pm4ngs-rnaseq/example/pm4ngs_
↪rnaseq_demo_sample_data.csv  to /Users/veraalva/my_ngs_project/data/my_dataset_name/
↪sample_table.csv
20 files loaded
Using table:
   sample_name file       condition  replicate
0    SRR2126784             PRE_NACT          1
1    SRR2126785             PRE_NACT          1
2    SRR2126786             PRE_NACT          1
3    SRR2126787             PRE_NACT          1
4    SRR3383790             PRE_NACT          1
5    SRR3383791             PRE_NACT          1
6    SRR3383792             PRE_NACT          1
7    SRR3383794             PRE_NACT          1
8    SRR3383796             PRE_NACT          1
9    SRR3383797             PRE_NACT          1
10   SRR3383798             PRE_NACT          1
11   SRR2126788       POST_NACT_CRS3          1
12   SRR2126789       POST_NACT_CRS3          1
13   SRR2126790       POST_NACT_CRS3          1
14   SRR2126791       POST_NACT_CRS3          1
15   SRR2126792       POST_NACT_CRS3          1
16   SRR2126793       POST_NACT_CRS3          1
17   SRR2126794       POST_NACT_CRS3          1
18   SRR2126795       POST_NACT_CRS3          1
19   SRR2126796       POST_NACT_CRS3          1
 Done
```

The **pm4ngs-rnaseq** command line will create a project structure as:

```
.
├── LICENSE
├── README.md
├── bin
│   └── cwl
├── config
│   └── init.py
```

(continues on next page)

(continued from previous page)

```
├── data
│   └── my_dataset_name
│       └── sample_table.csv
├── doc
├── index.html
├── notebooks
│   ├── 00 - Project Report.ipynb
│   ├── 01 - Pre-processing QC.ipynb
│   ├── 02 - Samples trimming.ipynb
│   ├── 03 - Alignments and Quantification.ipynb
│   ├── 04 - DGA.ipynb
│   └── 05 - GO enrichment.ipynb
├── requirements
│   └── conda-env-dependencies.yaml
├── results
│   └── my_dataset_name
├── src
└── tmp

61 directories, 239 files
```

Note: RNASeq based project variables

- **author_name:** Default: [Roberto Vera Alvarez]
- **email:** Default: [veraalva@ncbi.nlm.nih.gov]
- **project_name:** Name of the project with no space nor especial characters. This will be used as project folder's name.

  Default: [my_ngs_project]
- **dataset_name:** Dataset to process name with no space nor especial characters. This will be used as folder name to group the data. This folder will be created under the **data/{{dataset_name}}** and **results/{{dataset_name}}**.

  Default: [my_dataset_name]
- **is_data_in_SRA:** If the data is in the SRA set this to y. A CWL workflow to download the data from the SRA database to the folder **data/{{dataset_name}}** and execute FastQC on it will be included in the **01 - Pre-processing QC.ipynb** notebook.

  Set this option to **n**, if the fastq files names and location are included in the sample sheet.

  Default: [y]
- **Select sequencing_technology:** Select one of the available sequencing technologies in your data.

  Values: 1 - single-end, 2 - paired-end
- **create_demo:** If the data is downloaded from the SRA and this option is set to y, only the number of spots specified in the next variable will be downloaded. Useful to test the workflow.

  Default: [y]: y
- **number_spots:** Number of sport to download from the SRA database. It is ignored is the **create_demo** is set to **n**.

  Default: [1000000]
- **organism:** Organism to process, e.g. human. This is used to link the selected genes to the NCBI gene database.

Default: [human]

- **genome_name:** Genome name , e.g. hg38 or mm10.

  Default: [hg38]

- **genome_dir:** Absolute path to the directory with the genome annotation (genome.fa, genes.gtf) to be used by the workflow or the name of the genome.

  If the name of the genome is used, PM4NGS will include a cell in the **03 - Alignments and Quantification.ipynb** notebook to download the genome files. The genome data will be at **data/{{dataset_name}}/{{genome_name}}/**

  Default: [hg38]

- **aligner_index_dir:** Absolute path to the directory with the genome indexes for STAR.

  If **{{genome_name}}/STAR** is used, PM4NGS will include a cell in the **03 - Alignments and Quantification.ipynb** notebook to create the genome indexes for STAR.

  Default: [hg38/STAR]

- **genome_fasta:** Absolute path to the genome fasta file

  If **{{genome_name}}/genome.fa** is used, PM4NGS will use the downloaded fasta file.

  Default: [hg38/genome.fa]

- **genome_gtf:** Absolute path to the genome GTF file

  If **{{genome_name}}/genes.gtf** is used, PM4NGS will use the downloaded GTF file.

  Default: [hg38/gene.gtf]

- **genome_bed:** Absolute path to the genome BED file

  If **{{genome_name}}/genes.bed** is used, PM4NGS will use the downloaded BED file.

  Default: [hg38/genes.bed]

- **fold_change:** A real number used as fold change cutoff value for the DG analysis, e.g. 2.0.

  Default: [2.0]

- **fdr:** Adjusted P-Value to be used as cutoff in the DG analysis, e.g. 0.05.

  Default: [0.05]

- **use_docker:** Set this to y if you will be using Docker. Otherwise Conda needs to be installed in the computer.

  Default: [y]

- **max_number_threads:** Number of threads available in the computer.

  Default: [16]

6.4 Jupyter server

PM4NGS uses Jupyter as interface for users. After project creation the jupyter server should be started as shown below. The server will open a browser windows showing the project's structure just created.

```
localhost:~> jupyter notebook
```

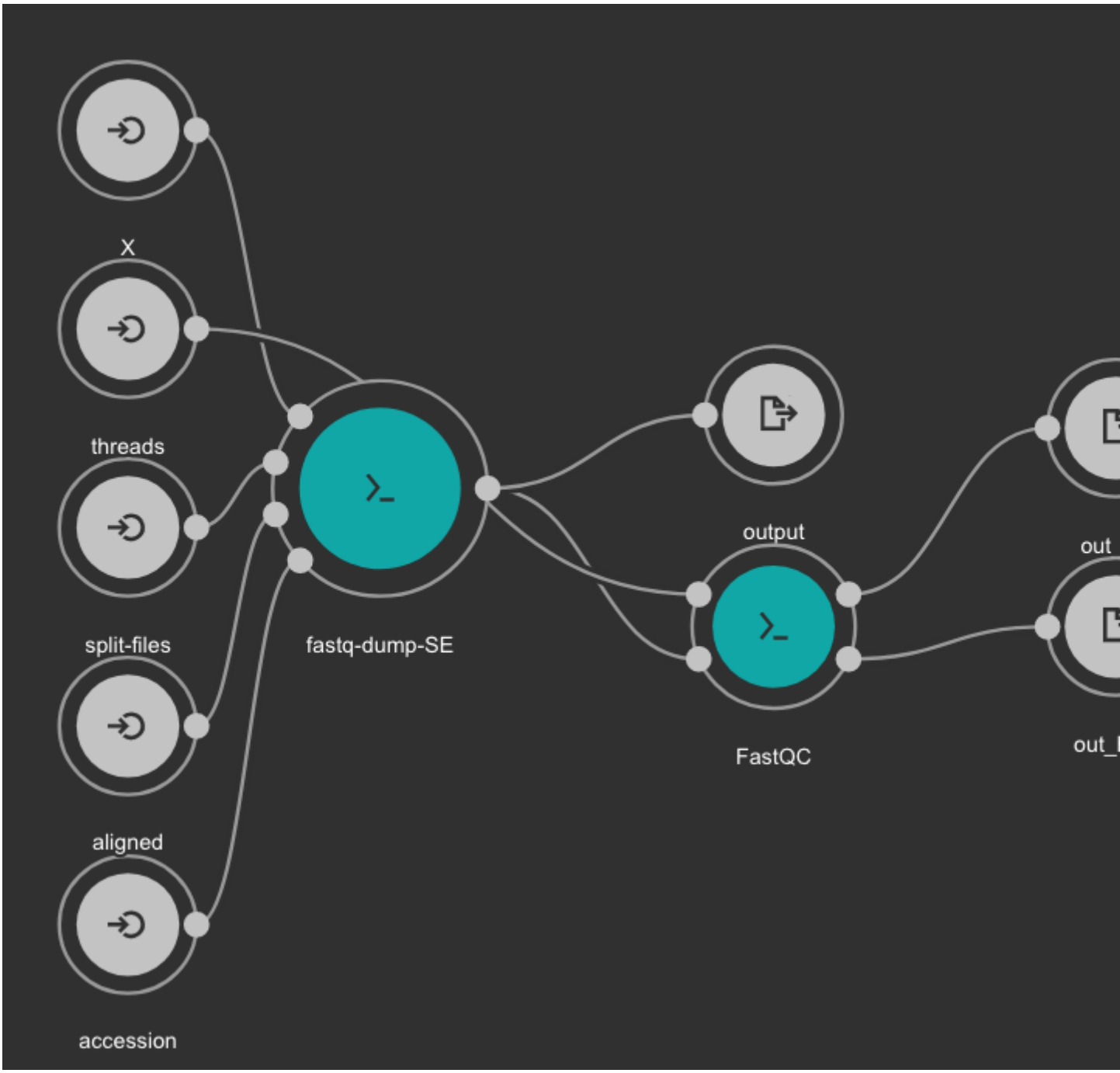
6.5 Data processing

Start executing the notebooks from 01 to 05 waiting for each step completion. The **00 - Project Report.ipynb** notebook can be executed after each notebooks to see the progress in the analysis.

6.6 CWL workflows

6.6.1 SRA download and QC workflow

This CWL workflow is designed to download FASTQ files from the NCBI SRA database using fastq-dump and then, execute fastqc generating a quality control report of the sample.



Inputs

- **accession**: SRA accession ID. Type: string. Required.
- **aligned**: Used it to download only aligned reads. Type: boolean. Optional.
- **split-files**: Dump each read into separate file. Files will receive suffix corresponding to read number. Type: boolean. Optional.
- **threads**: Number of threads. Type: int. Default: 1. Optional.

- **X**: Maximum spot id. Optional.

Outputs

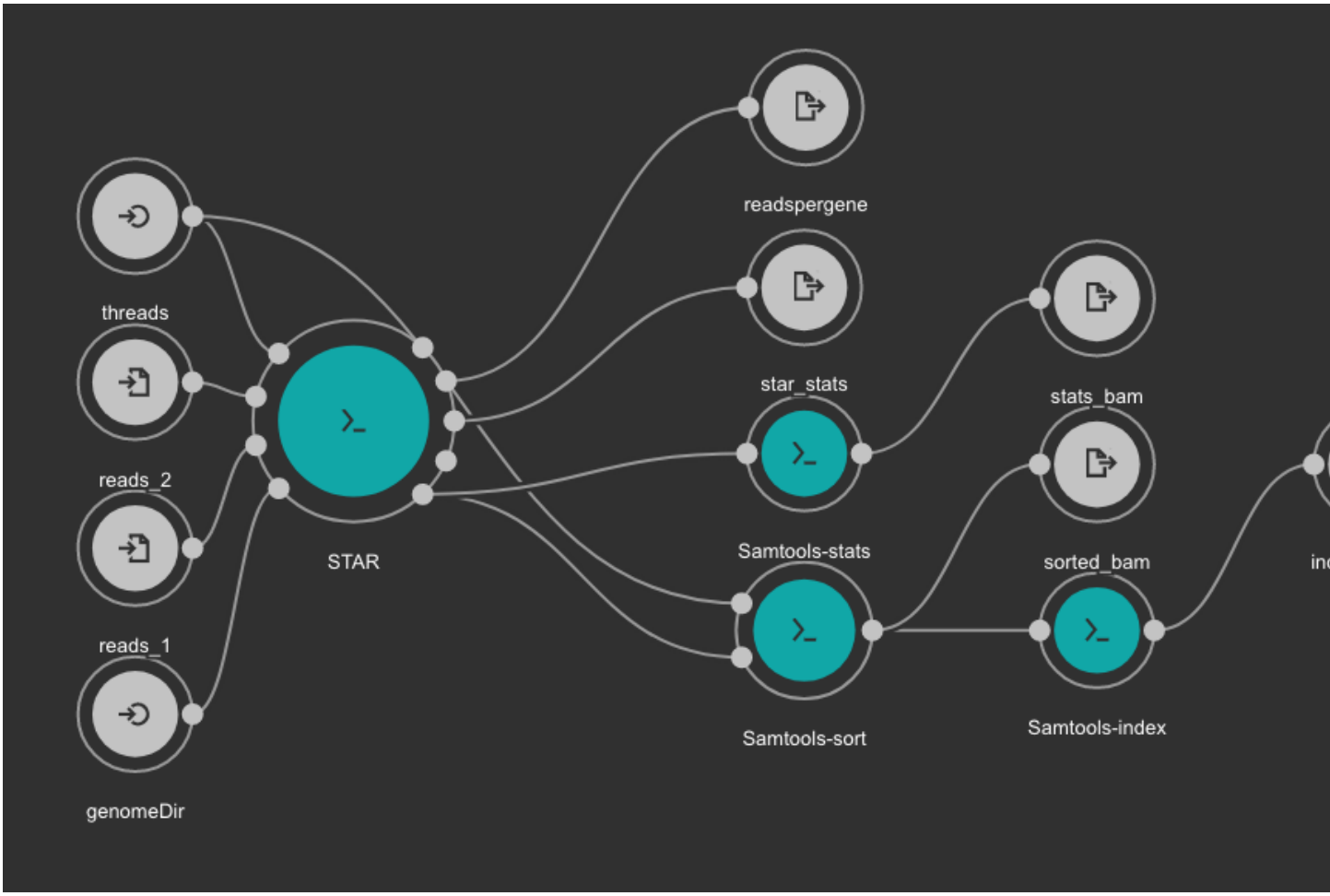
- **output**: Fastq files downloaded. Type: File[]
- **out_zip**: FastQC report ZIP file. Type: File[]
- **out_html**: FastQC report HTML. Type: File[]

6.6.2 Samples Trimming

Our workflows uses Trimmomatic for read trimming. The Jupyter notebooks uses some basic Trimmomatic options that need to be modified depending on the FastQC quality control report generated for the sample.

6.6.3 STAR based alignment and sorting

This workflows use STAR for alignning RNA-Seq reads to a genome. The obtained BAM file is sorted using SAMtools. Statistics outputs from STAR and SAMtools are returned as well.



Inputs

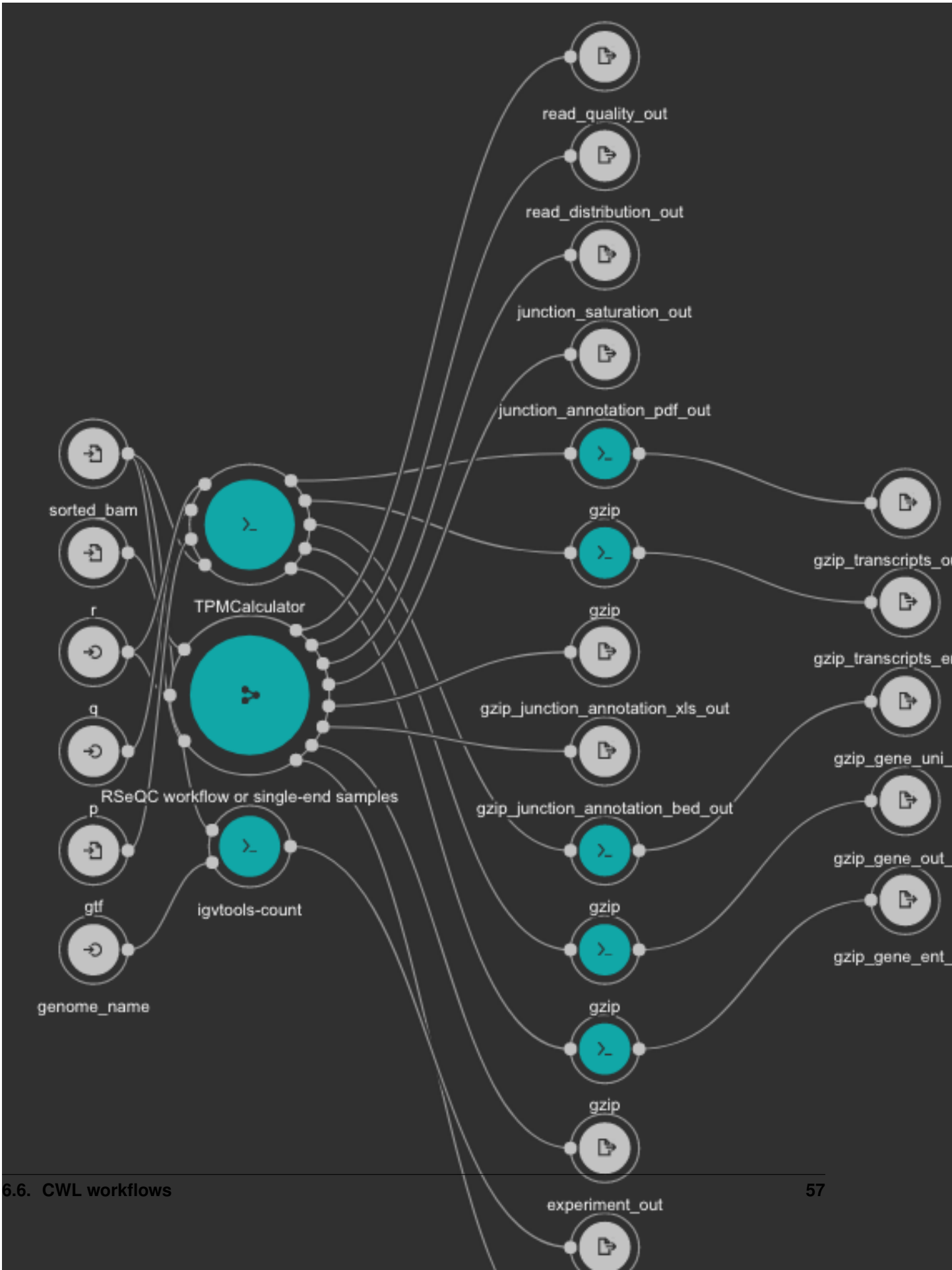
- **genomeDir**: Aligner indexes directory. Type: Directory. Required. Variable ALIGNER_INDEX in the Jupyter Notebooks.
- **threads**: Number of threads. Type: int. Default: 1. Optional.
- **reads_1**: FastQ file to be processed for paired-end reads _1. Type: File. Required.
- **reads_2**: FastQ file to be processed for paired-end reads _2. Type: File. Required.

Outputs

- **sorted_bam**: Final BAM file filtered and sorted. Type: File.
- **indexed_bam**: BAM index file. Type: File.
- **star_stats**: STAR alignment statistics. Type: File.
- **readspergene**: STAR reads per gene output. Type: File.
- **stats_bam**: SAMtools stats output: Type: File.

6.6.4 RNA-Seq quantification and QC workflow using TPMCalculator

This workflow uses TPMCalculator to quantify the abundance of genes and transcripts from the sorted BAM file. Additionally, RSeQC is executed to generate multiple quality control outputs from the sorted BAM file. At the end, a TDF file is generated using igvtools from the BAM file for a quick visualization.


read_quality_out
read_distribution_out
junction_saturation_out
junction_annotation_pdf_out
sorted_bam
gzip
gzip_transcripts_o
TPMCalculator
r
gzip
gzip_transcripts_e
q
gzip_junction_annotation_xls_out
gzip_gene_uni
RSeQC workflow or single-end samples
p
gzip_junction_annotation_bed_out
gzip_gene_out_
gtf
igvtools-count
gzip
gzip_gene_ent_
genome_name
gzip
gzip
experiment_out

Inputs

- **gtf**: Genome GTF file. Variable GENOME_GTF in the Jupyter Notebooks. Type: File. Required.
- **genome_name**: Genome name as defined in IGV for TDF conversion. Type: string. Required.
- **q**: Minimum MAPQ value to use reads. We recommend 255. Type: int. Required.
- **r**: Reference Genome in BED format used by RSeQC. Variable GENOME_BED in the Jupyter Notebooks. Type: File. Required.
- **sorted_bam**: Sorted BAM file to quantify. Type: File. Required.

Outputs

- **bam_to_tdf_out**: TDF file created with igvtools from the BAM file for quick visualization. Type: File.
- **gzip_gene_ent_out**: TPMCalculator gene ENT output gzipped. Type: File.
- **gzip_gene_out_out**: TPMCalculator gene OUT output gzipped. Type: File.
- **gzip_gene_uni_out**: TPMCalculator gene UNI output gzipped. Type: File.
- **gzip_transcripts_ent_out**: TPMCalculator transcript ENT output gzipped. Type: File.
- **gzip_transcripts_out_out**: TPMCalculator transcript OUT output gzipped. Type: File.
- **bam_stat_out**: RSeQC BAM stats output. Type: File.
- **experiment_out**: RSeQC experiment output. Type: File.
- **gzip_junction_annotation_bed_out**: RSeQC junction annotation bed. Type: File.
- **gzip_junction_annotation_xls_out**: RSeQC junction annotation xls. Type: File.
- **junction_annotation_pdf_out**: RSeQC junction annotation PDF figure. Type: File.
- **junction_saturation_out**: RSeQC junction saturation output. Type: File.
- **read_distribution_out**: RSeQC read distribution output. Type: File.
- **read_quality_out**: RSeQC read quality output. Type: File.

6.6.5 Differential Gene Expression analysis from RNA-Seq data

Our notebooks are designed to execute a Differential Gene Expression analysis using two available tools: DESeq2 and EdgeR. Also, the results for the interception of both tools output is reported with volcano plots, heatmaps and PCA plots.

The workflow use the **sample_sheet.tsv** file to generate an array with all combinations of **conditions**. The code to generate this array is very simple and can be found in the cell number 3 in the **05 - DGA.ipynb** notebook.

```
comparisons = []
for s in itertools.combinations(factors['condition'].unique(), 2):
    comparisons.append(list(s))
```

Let's suppose we have a **factors.txt** file with three conditions: **cond1**, **cond2** and **cond3**. The **comparisons** array will look like:

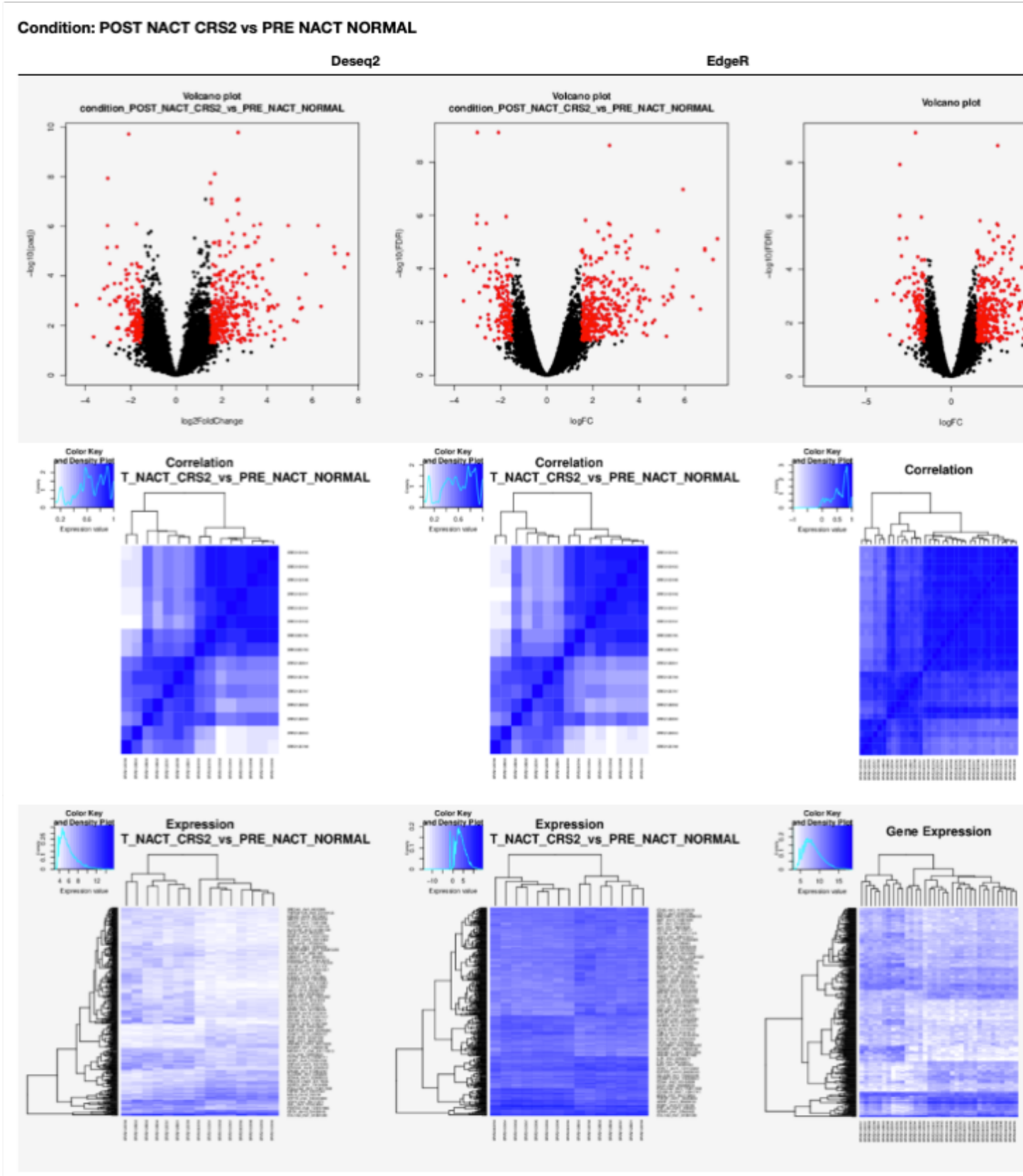
```
comparisons = [
    ['cond1', 'cond2'],
    ['cond1', 'cond3'],
    ['cond2', 'cond3']
]
```

To avoid this behavior and execute the comparison just in a set of conditions, you should remove the code in the cell number 3 in the **05 - DGA.ipynb** notebook and manually create the array of combinations to be compared as:

```
comparisons = [
    ['cond1', 'cond3'],
]
```

The R code used for running DESeq2 is embedded in deseq2-2conditions.cwl from line 25 to line 169. The R code used for running EdgeR is embedded in edgeR-2conditions.cwl from line 25 to line 159.

A table with DGA plots is generated for each condition in the **00 - Project Report.ipynb** as shown next.


Condition: POST NACT CRS2 vs PRE NACT NORMAL
Deseq2
EdgeR
Volcano plot
condition_POST_NACT_CRS2_vs_PRE_NACT_NORMAL
log2FoldChange
logFC
Correlation
T_NACT_CRS2_vs_PRE_NACT_NORMAL
Expression
Gene Expression
PCA

6.6.6 GO enrichment from RNA-Seq data

The GO enrichment analysis is executed with an *in-house* developed python package named goenrichment. This tools uses the hypergeometric distribution test to estimate the probability of successes in selecting GO terms from a list of differentially expressed genes. The GO terms are represented as a network using the python library NetworkX.

The tool uses a pre-computed database, currently available for human and mouse, at https://ftp.ncbi.nlm.nih.gov/pub/goenrichment/. However, the project web page describe how to create your own database from a set of reference databases.

The workflow uses the **factors.txt** file to generate an array with all combinations of **conditions**. The code to generate this array is very simple and can be found in the cell number 3 in the **06 - GO enrichment.ipynb** notebook.

```
comparisons = []
for s in itertools.combinations(factors['condition'].unique(), 2):
    comparisons.append(list(s))
```

Let's suppose we have a **factors.txt** file with three conditions: **cond1**, **cond2** and **cond3**. The **comparisons** array will look like:

```
comparisons = [
    ['cond1', 'cond2'],
    ['cond1', 'cond3'],
    ['cond2', 'cond3']
]
```

To avoid this behavior and execute the comparison just in a set of conditions, you should remove the code in the cell number 3 in the **06 - GO enrichment.ipynb** notebook and manually create the array of combinations to be compared as:

```
comparisons = [
    ['cond1', 'cond3'],
]
```

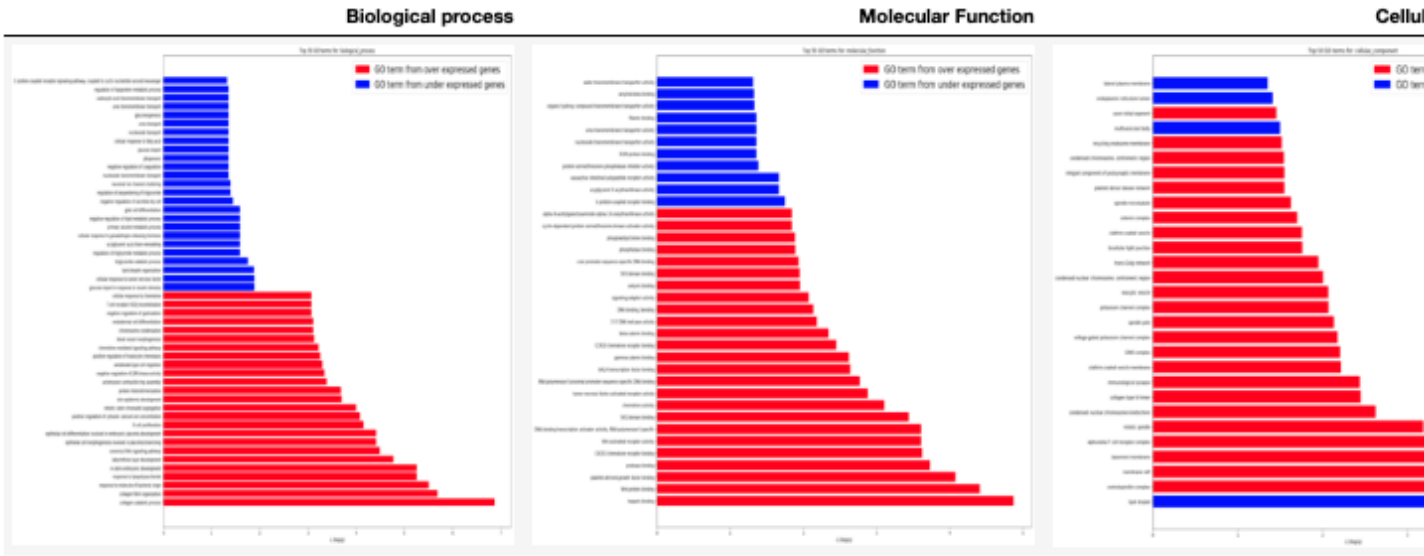
Additionally, the workflow requires three cutoff that are defined in the cell number 5 of the same notebook.

```
min_category_depth=4
min_category_size=3
max_category_size=500
```

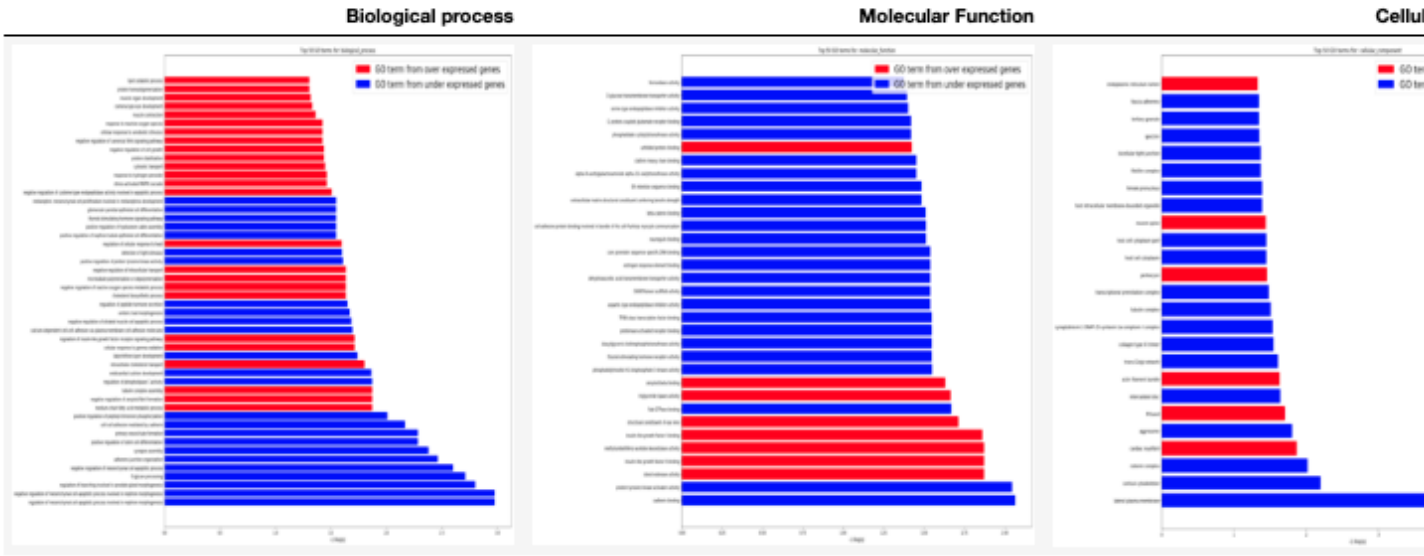
Cutoffs definition

- min_category_depth: Min GO term graph depth to include in the report. Default: 4
- min_category_size: Min number of gene in a GO term to include in the report. Default: 3
- max_category_size: Max number of gene in a GO term to include in the report. Default: 500

A table with GO terms plots is generated for each condition in the **00 - Project Report.ipynb** as shown next. In these plots the red bars are for GO terms selected from the over expressed genes and the blue bars are for GO terms selected from the under expressed genes. It is important to clarify that the two sets of GO terms don't overlap each other.



Condition: POST_NACT_CRS3 vs POST_NACT_CRS2



6.7 Demo

PM4NGS includes a demo project that users can use to test the framework. It is pre-configured to use Docker as execution environment.

The RNASeq based demo process samples from the BioProject PRJNA290924.

Use this command to create the project structure in your local computer

```
localhost:~> pm4ngs-rnaseq-demo
```

Once it finish, start the jupyter server and execute the notebooks as it is indicated on them

```
localhost:~> jupyter notebook
[I 14:12:52.956 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 14:12:52.956 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 14:12:52.956 NotebookApp] http://localhost:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp]  or http://127.0.0.1:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmatio
n).
[C 14:12:52.959 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-23251-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
     or http://127.0.0.1:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
```

The results of this analysis is https://ftp.ncbi.nlm.nih.gov/pub/pm4ngs/examples/rnaseq-sra-paired/

CHAPTER 7

Differential Binding detection from ChIP-Seq data

Differential binding analysis pipeline from ChIP-Seq data.

The workflow is comprised of five steps, as shown in next table. The first step, sample download and quality control, is for downloading samples from the NCBI SRA database, if necessary, or for executing the pre-processing quality control tools on all samples. Sample trimming and alignment are executed next. Post-processing quality control on ChIP-Seq samples is executed using Phantompeakqualtools, as recommended by the ENCODE consortia]. Then, a peak calling step is performed using MACS2, and peak annotation is executed with Homer. Peak reproducibility is explored with IDR. Finally, a differential binding analysis is completed with DiffBind.

Table 1: Differential binding analysis, ChIP-Seq pipeline

Step	Jupyter Notebook	Workflow CWL	Tool	Input	Output	CWL Tool
Sample Download and Quality Control	01 - Pre-processing QC	download_quality_workflow.cwl	SRA Tools	SRA accession	Fastq	fastq-dump.cwl
			FastQC	Fastq	FastQC HTML and Zip	fastqc.cwl
Trimming	02 - Samples trimming		Trimmomatic	Fastq	Fastq	trimmomatic-PE.cwl trimmomatic-SE.cwl
Alignments and Quantification	03 - Alignments	chip-seq-alignment.cwl	BWA	Fastq	BAM	bwa-mem.cwl
			Samtools	SAM	BAM Sorted BAM BAM index BAM stats BAM flagstats	samtools-view.cwl samtools-sort.cwl samtools-index.cwl samtools-stats.cwl samtools-flagstat.cwl
Peak Calling and IDR	04 - Peak Calling and IDR	peak-calling-MACS2.cwl	IGVtools	Sorted BAM	TDF	igvtools-count.cw
			MACS2	tagAlign	Peaks (TSV), plots	macs2-callpeak.cwl
			Homer	BAM	Annotation (TSV) FPKM values (TSV)	homer-annotatePeaks.cwl homer-makeTagDirectory.cwl
			IDR	Peaks	Peaks (TSV), plots	idr.cwl
Differential binding Analysis	05 - Differential binding Detection		DiffBind	Read Matrix	Peaks (TSV), plots	diffBind.cwl

7.1 Input requirements

The input requirement for the ChIPSeq pipeline is the *Sample sheet* file.

7.2 Pipeline command line

The ChIPSeq based project can be created using the following command line:

```
localhost:~> pm4ngs-chipseq
usage: Generate a PM4NGS project for ChIPSeq data analysis [-h] [-v]
```

(continues on next page)

(continued from previous page)

```
                                                    --sample-sheet
                                                    SAMPLE_SHEET
                                                    [--config-file CONFIG_FILE]
                                                    [--copy-rawdata]
```

Options:

- **sample-sheet**: Sample sheet with the samples metadata
- **config-file**: YAML file with configuration for project creation
- **copy-rawdata**: Copy the raw data defined in the sample sheet to the project structure. (The data can be hosted locally or in an http server)

7.3 Creating the Differential Binding detection from ChIP-Seq data project

The **pm4ngs-chipseq** command line executed with the **–sample-sheet** option will let you type the different variables required for creating and configuring the project. The default value for each variable is shown in the brackets. After all questions are answered, the CWL workflow files will be cloned from the github repo ncbi/cwl-ngs-workflows-cbb to the folder **bin/cwl**.

```
localhost:~> pm4ngs-chipseq --sample-sheet my-sample-sheet.tsv
Generating ChIP-Seq data analysis project
author_name [Roberto Vera Alvarez]:
email [veraalva@ncbi.nlm.nih.gov]:
project_name [my_ngs_project]:
dataset_name [my_dataset_name]:
is_data_in_SRA [None]:
Select sequencing_technology:
1 - single-end
2 - paired-end
Choose from 1, 2 [1]: 2
create_demo [y]:
number_spots [1000000]:
organism [human]:
genome_name [hg38]:
genome_dir [hg38]:
aligner_index_dir [hg38/BWA]:
genome_fasta [hg38/genome.fa]:
genome_gtf [hg38/genome.gtf]:
genome_mappable_size [hg38]:
fdr [0.05]:
use_docker [y]:
max_number_threads [16]:
Cloning Git repo: https://github.com/ncbi/cwl-ngs-workflows-cbb to /Users/veraalva/my_
↪ngs_project/bin/cwl
Updating CWLs dockerPull and SoftwareRequirement from: /Users/veraalva/my_ngs_project/
↪requirements/conda-env-dependencies.yaml
bedtools with version 2.29.2 update image to: quay.io/biocontainers/bedtools:2.29.2--
↪hc088bd4_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bedtools/bedtools-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bedtools:2.28.0--hdf88d34_0
```

(continues on next page)

(continued from previous page)

```
bioconductor-diffbind with version 2.16.0 update image to: quay.io/biocontainers/
↪bioconductor-diffbind:2.16.0--r40h5f743cb_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-pca.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/macs-cutoff.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/dga_heatmaps.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/diffbind.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/edgeR-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/volcano_plot.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/readQC.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
bwa with version 0.7.17 update image to: quay.io/biocontainers/bwa:0.7.17--hed695b0_7
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bwa/bwa-docker.yml with old image␣
↪replaced: quay.io/biocontainers/bwa:0.7.17--h84994c4_5
homer with version 4.11 update image to: quay.io/biocontainers/homer:4.11--
↪pl526h9a982cc_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/homer/homer-docker.yml with old␣
↪image replaced: quay.io/biocontainers/homer:4.11--pl526h2bce143_2
Copying file /Users/veraalva/Work/Developer/Python/pm4ngs/pm4ngs-chipseq/example/
↪pm4ngs_chipseq_demo_sample_data.csv  to /Users/veraalva/my_ngs_project/data/my_
↪dataset_name/sample_table.csv
6 files loaded
Using table:
  sample_name file condition  replicate
0  SRR7549105             ES          1
1  SRR7549106             ES          2
2  SRR7549109            MEF          1
3  SRR7549110            MEF          2
4  SRR7549114             rB          1
5  SRR7549113             rB          2
 Done
```

The **pm4ngs-chipseq** command line will create a project structure as:

```
.
├── LICENSE
├── README.md
├── bin
│   └── cwl
├── config
│   └── init.py
├── data
│   └── my_dataset_name
├── doc
├── index.html
├── notebooks
│   ├── 00 - Project Report.ipynb
│   ├── 01 - Pre-processing QC.ipynb
│   ├── 02 - Samples trimming.ipynb
│   ├── 03 - Alignments.ipynb
```

(continues on next page)

(continued from previous page)

```
│   ├── 04 - Peak Calling and IDR.ipynb
│   └── 05 - Differential binding Detection.ipynb
├── requirements
│   └── conda-env-dependencies.yaml
├── results
│   └── my_dataset_name
├── src
└── tmp

12 directories, 11 files
```

Note: ChIP-Seq based project variables

- **author_name:** Default: [Roberto Vera Alvarez]
- **email:** Default: [veraalva@ncbi.nlm.nih.gov]
- **project_name:** Name of the project with no space nor especial characters. This will be used as project folder's name.

  Default: [my_ngs_project]
- **dataset_name:** Dataset to process name with no space nor especial characters. This will be used as folder name to group the data. This folder will be created under the **data/{{dataset_name}}** and **results/{{dataset_name}}**.

  Default: [my_dataset_name]
- **is_data_in_SRA:** If the data is in the SRA set this to y. A CWL workflow to download the data from the SRA database to the folder **data/{{dataset_name}}** and execute FastQC on it will be included in the **01 - Pre-processing QC.ipynb** notebook.

  Set this option to **n**, if the fastq files names and location are included in the sample sheet.

  Default: [y]
- **Select sequencing_technology:** Select one of the available sequencing technologies in your data.

  Values: 1 - single-end, 2 - paired-end
- **create_demo:** If the data is downloaded from the SRA and this option is set to y, only the number of spots specified in the next variable will be downloaded. Useful to test the workflow.

  Default: [y]: y
- **number_spots:** Number of sport to download from the SRA database. It is ignored is the **create_demo** is set to **n**.

  Default: [1000000]
- **organism:** Organism to process, e.g. human. This is used to link the selected genes to the NCBI gene database.

  Default: [human]
- **genome_name:** Genome name , e.g. hg38 or mm10.

  Default: [hg38]
- **genome_dir:** Absolute path to the directory with the genome annotation (genome.fa, genes.gtf) to be used by the workflow or the name of the genome.

  If the name of the genome is used, PM4NGS will include a cell in the **03 - Alignments.ipynb** notebook to download the genome files. The genome data will be at **data/{{dataset_name}}/{{genome_name}}/**

Default: [hg38]

- **aligner_index_dir:** Absolute path to the directory with the genome indexes for BWA.

  If **{{genome_name}}/BWA** is used, PM4NGS will include a cell in the **03 - Alignments.ipynb** notebook to create the genome indexes for BWA.

  Default: [hg38/BWA]

- **genome_fasta:** Absolute path to the genome fasta file

  If **{{genome_name}}/genome.fa** is used, PM4NGS will use the downloaded fasta file.

  Default: [hg38/genome.fa]

- **genome_gtf:** Absolute path to the genome GTF file

  If **{{genome_name}}/genes.gtf** is used, PM4NGS will use the downloaded GTF file.

  Default: [hg38/gene.gtf]

- **genome_bed:** Absolute path to the genome BED file

  If **{{genome_name}}/genes.bed** is used, PM4NGS will use the downloaded BED file.

  Default: [hg38/genes.bed]

- **genome_mappable_size:** Genome mappable size used by MACS. For human can be hg38 or in case of other genomes it is a number.

  Default: [hg38]

- **fdr:** Adjusted P-Value to be used as cutoff in the DG analysis, e.g. 0.05.

  Default: [0.05]

- **use_docker:** Set this to y if you will be using Docker. Otherwise Conda needs to be installed in the computer.

  Default: [y]

- **max_number_threads:** Number of threads available in the computer.

  Default: [16]

7.4 Jupyter server

PM4NGS uses Jupyter as interface for users. After project creation the jupyter server should be started as shown below. The server will open a browser windows showing the project's structure just created.

```
localhost:~> jupyter notebook
```

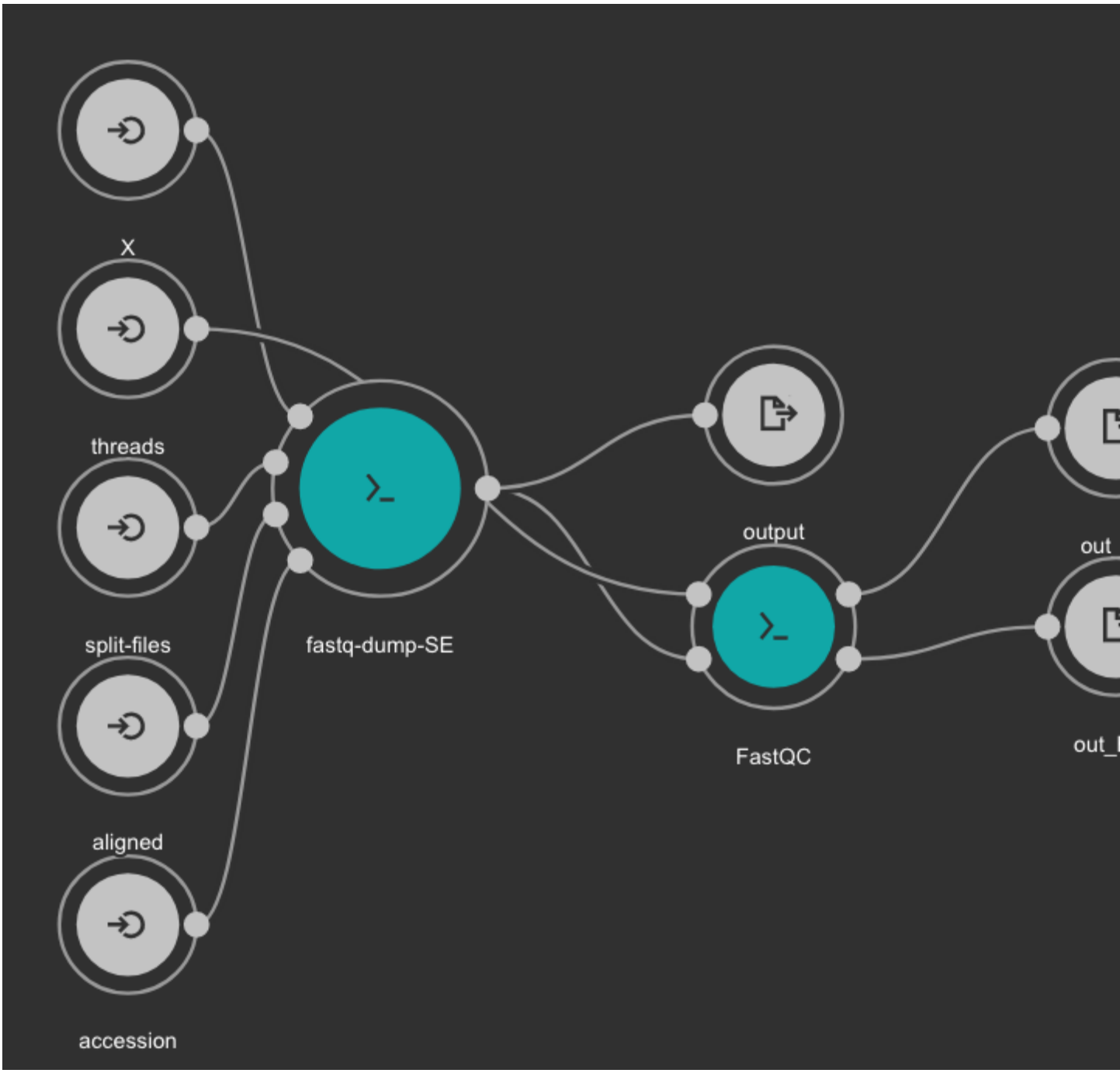
7.5 Data processing

Start executing the notebooks from 01 to 05 waiting for each step completion. The **00 - Project Report.ipynb** notebook can be executed after each notebooks to see the progress in the analysis.

7.6 CWL workflows

7.6.1 SRA download and QC workflow

This CWL workflow is designed to download FASTQ files from the NCBI SRA database using fastq-dump and then, execute fastqc generating a quality control report of the sample.



Inputs

- **accession**: SRA accession ID. Type: string. Required.
- **aligned**: Used it to download only aligned reads. Type: boolean. Optional.
- **split-files**: Dump each read into separate file. Files will receive suffix corresponding to read number. Type: boolean. Optional.
- **threads**: Number of threads. Type: int. Default: 1. Optional.
- **X**: Maximum spot id. Optional.

Outputs

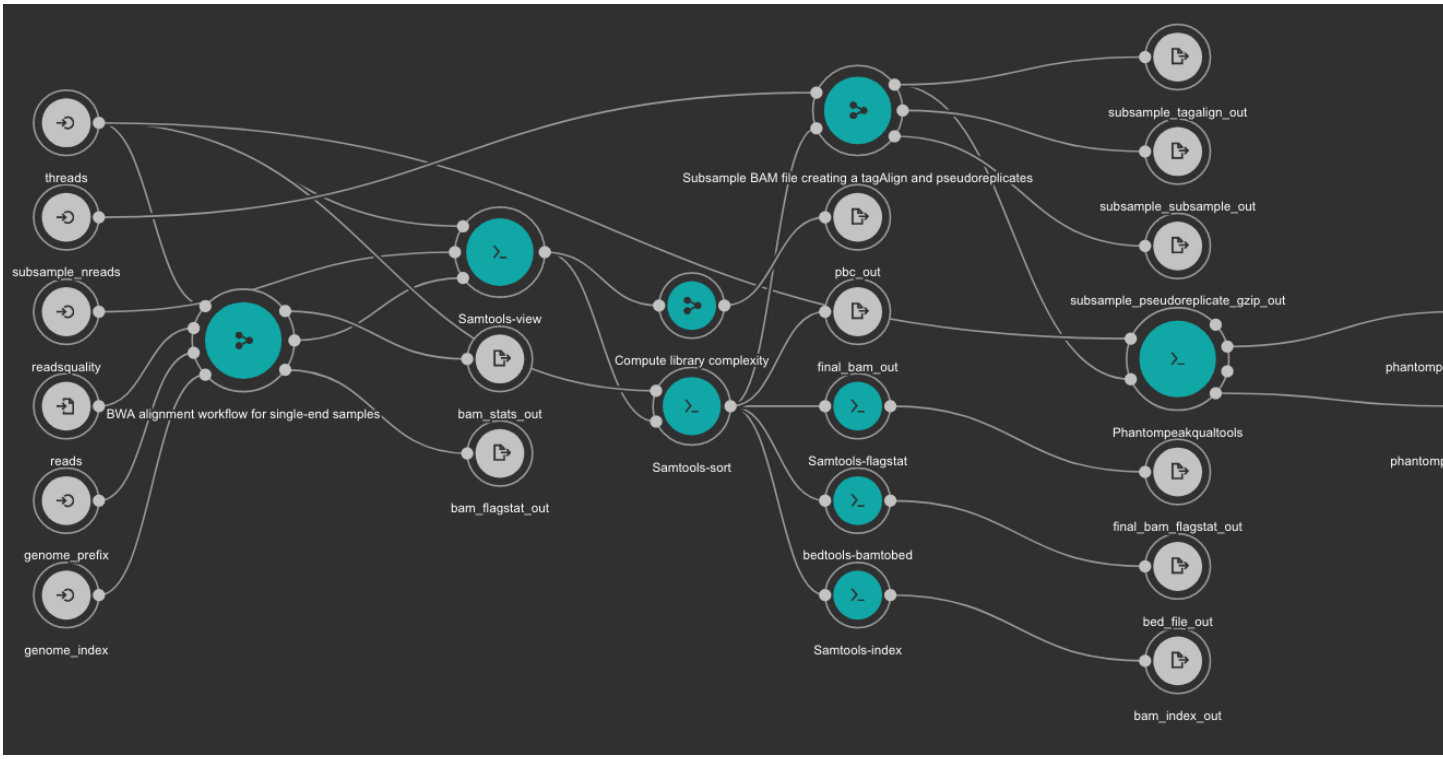
- **output**: Fastq files downloaded. Type: File[]
- **out_zip**: FastQC report ZIP file. Type: File[]
- **out_html**: FastQC report HTML. Type: File[]

7.6.2 Samples Trimming

Our workflows uses Trimmomatic for read trimming. The Jupyter notebooks uses some basic Trimmomatic options that need to be modified depending on the FastQC quality control report generated for the sample.

7.6.3 BWA based alignment and quality control workflow

This workflow use BWA as base aligner. It also use SAMtools and bedtools for file conversion and statistics report. Finally, Phantompeakqualtools is used to generate quality control report for the processed samples.



Inputs

- **genome_index**: Aligner indexes directory. Type: Directory. Required. Variable ALIGNER_INDEX in the Jupyter Notebooks.
- **genome_prefix**: Prefix of the aligner indexes. Generally, it is the name of the genome FASTA file. It can be used as os.path.basename(GENOME_FASTA) in the Jupyter Notebooks. Type: string. Required.
- **readsquality**: Minimum MAPQ value to use reads. We recommend for ChIP_exo dataa value of: 30. Type: int. Required.
- **threads**: Number of threads. Type: int. Default: 1. Optional.
- **subsample_nreads**: Number of reads to be used for the subsample. We recomend for ChIP_exo dataa value of: 500000. Type: int. Required.
- **reads**: FastQ file to be processed. Type: File. Required.

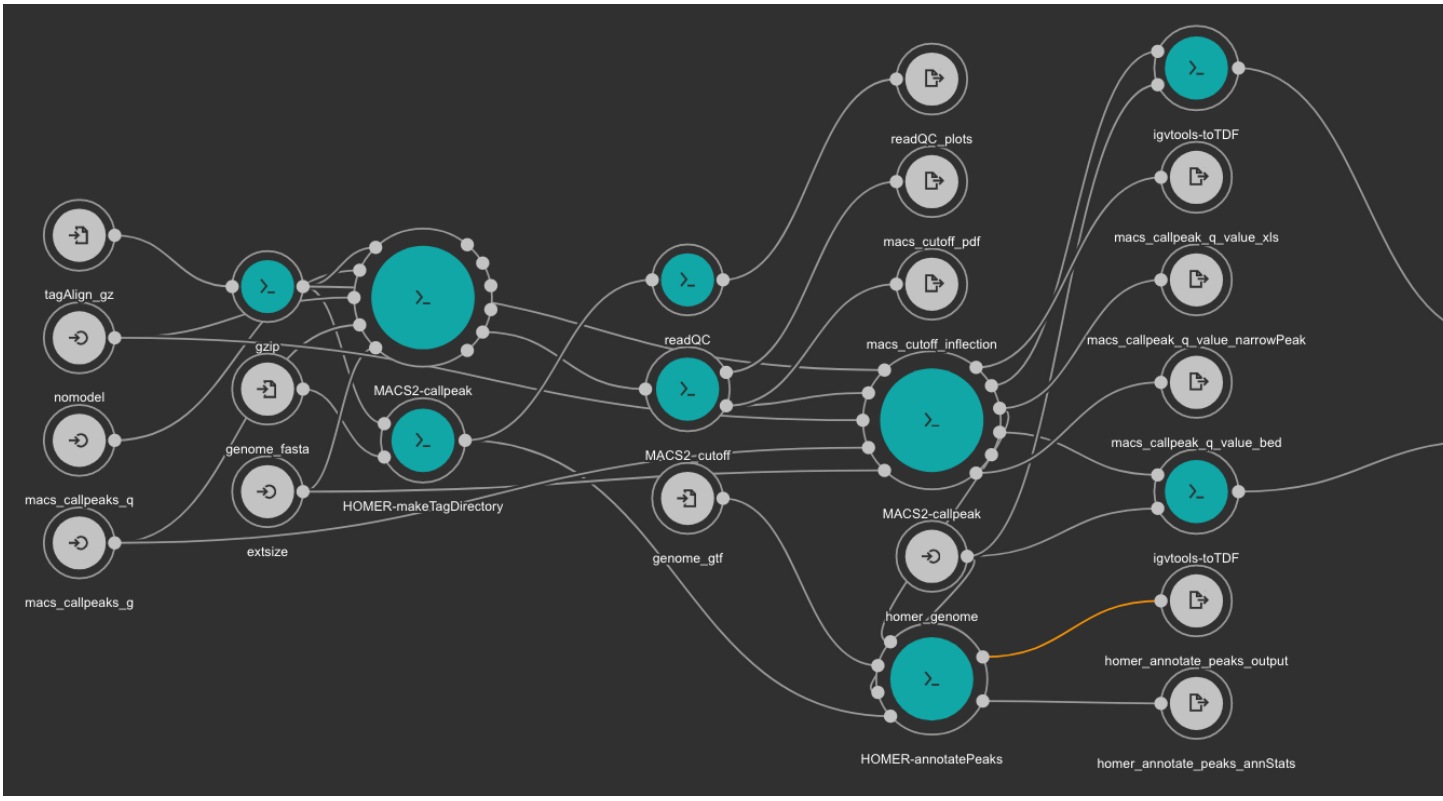
Outputs

- **bam_flagstat_out**: SAMtools flagstats for unfiltered BAM file. Type: File.
- **bam_stats_out**: SAMtools stats for unfiltered BAM file. Type: File.
- **final_bam_flagstat_out**: SAMtools flagstats for filtered BAM file. Type: File.
- **bed_file_out**:: Aligned reads in BED format. Type: File.
- **final_bam_out**: Final BAM file filtered and sorted. Type: File.
- **bam_index_out**: BAM index file. Type: File.

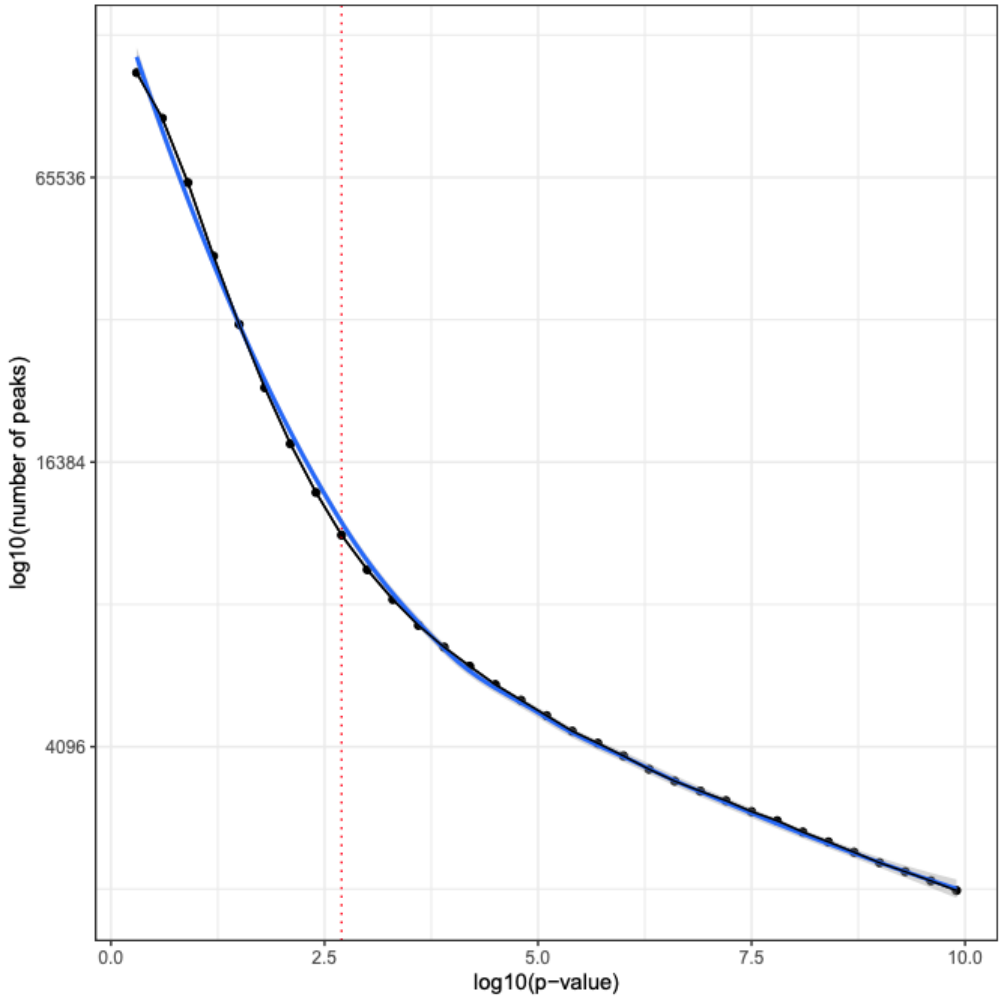
- **pbc_out**: Library complexity report. Type: File.
- **phantompeakqualtools_output_out**: Phantompeakqualtools main output. Type: File.
- **phantompeakqualtools_output_savp**: Phantompeakqualtools SAVP output. Type: File.
- **subsample_pseudoreplicate_gzip_out**: Subsample pseudoreplicates tagAlign gzipped. Type: File[].
- **subsample_tagalign_out**: Subsample tagAlign gzipped. Type: File[].
- **subsample_subsample_out**: Subsample shuffled tagAlign gzipped. Type: File[].

7.6.4 Peak caller workflow using MACS2

This workflow uses MACS2 as peak caller tool. The annotation is created using Homer and TDF files are created with igvtools.



MACS2 is executed two times. First, the **cutoff-analysis** option is used to execute a cutoff value analysis which is used to estimate a proper value for the p-value used by MACS2 (for more detailed explanation read this thread).



RSeQC is also executed for quality control.

Inputs

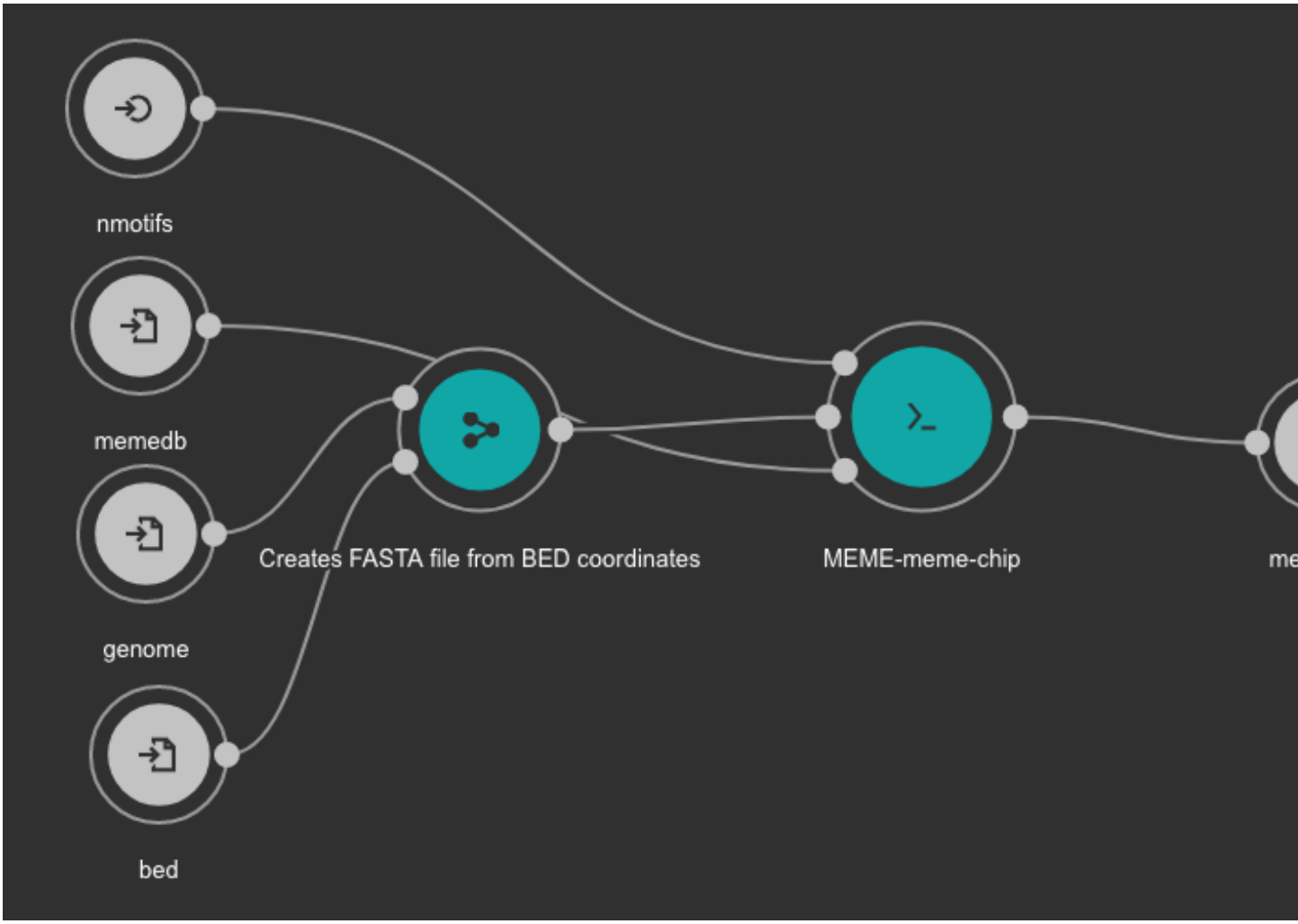
- **homer_genome**: Homer genome name. Type: string. Required.
- **genome_fasta** Genome FASTA file. Type: File. Required. Variable GENOME_FASTA in the Jupyter Notebooks.
- **genome_gtf**: Genome GTF file. Type: File. Required. Variable GENOME_GTF in the Jupyter Notebooks.
- **tagAlign_gz**: Tag aligned file created with the BWA based alignment and quality control workflow. Type: File. Required.
- **macs_callpeaks_g**: Genome mapeable size as defined in MACS2. Type: string. Required. Variable GENOME_MAPPABLE_SIZE in the Jupyter Notebooks.
- **macs_callpeaks_q**: MACS2 **q** option. Starting qvalue (minimum FDR) cutoff to call significant regions. Type: float. Required. Variable fdr in the Jupyter Notebooks.
- **nomodel**: MACS2 nomodel option. MACS will bypass building the shifting model. Type: boolean. Optional.
- **extsize**: MACS2 extsize option. MACS uses this parameter to extend reads in 5'->3' direction to fix-sized fragments. Type: int. Optional.

Outputs

- **readQC_plots**: RSeQC plots. Type: File[]
- **macs_cutoff_pdf** MACS2 cutoff analysis plot in PDF format. Type: File
- **macs_cutoff_inflection**: MACS2 inflection q value used for the second round. Type: File
- **macs_callpeak_q_value_narrowPeak**: Final MACS2 narrowpeak file. Type: File
- **macs_callpeak_q_value_xls**: Final MACS2 XLS file. Type: File
- **macs_callpeak_q_value_bed**: Final MACS2 BED file. Type: File
- **homer_annotate_peaks_output**: Homer annotated BED file. Type: File
- **homer_annotate_peaks_annStats**: Homer annotation statistics. Type: File
- **lambda_tdf_out**: MACS2 lambda file in TDF format. Type: File
- **pileup_tdf_out**: MACS2 pileup file in TDF format. Type: File

7.6.5 MEME Motif detection workflow

Motif detection is executed using the MEME suite.



Inputs

- **bed**: BED file with detected peaks. Type: File. Required.
- **memedb**: MEME database for use by Tomtom and CentriMo. Type: File. Required.
- **genome**: Genome FASTA file. Variable GENOME_FASTA in the Jupyter Notebooks. Type: File. Required.
- **nmotifs**: Maximum number of motifs to find. We recommend use 10. Type: int. Required.

Outputs

- **meme_out**: MEME output directory. Type: Directory

7.6.6 MEME databases

MEME workflow depends on the MEME databases. Go to the MEME Suite Download page: http://meme-suite.org/doc/download.html

Download the latest version for the Motif Databases and GOMo Databases.

The downloaded files should be uncompressed in a directory **data/meme**. The final directory should be:

```
localhost:~> cd data
localhost:~> mkdir meme
localhost:~> cd meme
localhost:~> wget http://meme-suite.org/meme-software/Databases/motifs/motif_
↪databases.12.18.tgz
localhost:~> wget http://meme-suite.org/meme-software/Databases/gomo/gomo_databases.3.
↪2.tgz
localhost:~> tar xzf motif_databases.12.18.tgz
localhost:~> tar xzf gomo_databases.3.2.tgz
localhost:~> rm gomo_databases.3.2.tgz motif_databases.12.18.tgz
localhost:~> tree -d
.
├── gomo_databases
└── motif_databases
    ├── ARABD
    ├── CIS-BP
    ├── CISBP-RNA
    ├── ECOLI
    ├── EUKARYOTE
    ├── FLY
    ├── HUMAN
    ├── JASPAR
    ├── MALARIA
    ├── MIRBASE
    ├── MOUSE
    ├── PROKARYOTE
    ├── PROTEIN
    ├── RNA
    ├── TFBSshape
    ├── WORM
    └── YEAST

19 directories
```

See also an example in our test project: https://ftp.ncbi.nlm.nih.gov/pub/pm4ngs/examples/chipexo-single/data/

7.6.7 Differential binding analysis with Diffbind

Differential binding event is detected with Diffbind. Thsi tool will be executed for all comparisons added to the **comparisons** array. See cell number 3 in the notebook **05 - Differential binding analysis.ipynb** (ChIP-Seq workflow).

Inputs

- **bamDir**: Directory with the BAM files. Type: Directory. Required.
- **bedDir**: Directory with BED files created from MACS2 peak calling workflow Type: Directory. Required.

Outputs

- **outpng**: PNG files created from Diffbind. Type File[]

- **outxls**: XLS files created from Diffbind. Type: File[]
- **outbed** BED files created from Diffbind. Type: File[]

7.7 Demo

PM4NGS includes a demo project that users can use to test the framework. It is pre-configured to use Docker as execution environment.

The ChIPSeq based demo process samples from the BioProject PRJNA481982.

Use this command to create the project structure in your local computer

```
localhost:~> pm4ngs-chipseq-demo
```

Once it finish, start the jupyter server and execute the notebooks as it is indicated on them

```
localhost:~> jupyter notebook
[I 14:12:52.956 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 14:12:52.956 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 14:12:52.956 NotebookApp] http://localhost:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp]  or http://127.0.0.1:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmatio
n).
[C 14:12:52.959 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-23251-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
     or http://127.0.0.1:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
```

The results of this analysis is https://ftp.ncbi.nlm.nih.gov/pub/pm4ngs/examples/chipseq-hmgn1/

CHAPTER 8

Detection of binding events from ChIP-exo data

The workflow is comprised of five steps, as shown in next table. The first step, sample download and quality control, is for downloading samples from the NCBI SRA database, if necessary, or for executing the pre-processing quality control tools on all samples. After that, sample trimming and alignment are executed. Post-processing quality control on the ChIP-exo samples is performed, using Phantompeakqualtools. After that, the peak calling is performed, using MACE. Finally, DNA motif annotation is performed with the MEME Suite.

Table 1: DNA motif identification, ChIP-exo data pipeline

Step	Jupyter Notebook	Workflow CWL	Tool	Input	Output	CWL Tool
Sample Download and Quality Control	01 - Pre-processing QC	download_quality_workflow.cwl	SRA Tools	SRA accession	Fastq	fastq-dump.cwl
			FastQC	Fastq	FastQC HTML and Zip	fastqc.cwl
Trimming	02 - Samples trimming		Trimmomatic	Fastq	Fastq	trimmomatic-PE.cwl trimmomatic-SE.cwl
Alignments and Quantification	03 - Alignments	chip-seq-alignment.cwl	BWA	Fastq	BAM	bwa-mem.cwl
			Samtools	SAM	BAM Sorted BAM BAM index BAM stats BAM flagstats	samtools-view.cwl samtools-sort.cwl samtools-index.cwl samtools-stats.cwl samtools-flagstat.cwl
Peak Calling and IDR	04 - Peak Calling	peak-caller-MACE-SE.cwl	IGVtools	Sorted BAM	TDF	igvtools-count.cw
			MACE	tagAlign	Peaks (TSV), plots	mace.cwl
			BAMScale	BAM	TPM values	bamscale-cov.cwl bamscale-scale.cwl
DNA motif identification	05 - MEME Motif		MEME Suite	Peaks	DNA Motif , plots	meme-chip.cwl

8.1 Input requirements

The input requirement for the ChIPexo pipeline is the *Sample sheet* file.

8.2 Pipeline command line

The ChIPexo based project can be created using the following command line:

```
localhost:~> pm4ngs-chipexo
usage: Generate a PM4NGS project for ChIPexo data analysis [-h] [-v]
                                                           --sample-sheet
                                                           SAMPLE_SHEET
                                                           [--config-file CONFIG_FILE]
                                                           [--copy-rawdata]
```

Options:

- **sample-sheet**: Sample sheet with the samples metadata
- **config-file**: YAML file with configuration for project creation
- **copy-rawdata**: Copy the raw data defined in the sample sheet to the project structure. (The data can be hosted locally or in an http server)

8.3 Creating the Detection of binding events from ChIP-exo data project

The **pm4ngs-chipexo** command line executed with the **–sample-sheet** option will let you type the different variables required for creating and configuring the project. The default value for each variable is shown in the brackets. After all questions are answered, the CWL workflow files will be cloned from the github repo ncbi/cwl-ngs-workflows-cbb to the folder **bin/cwl**.

```
localhost:~> pm4ngs-chipexo  --sample-sheet my-sample-sheet.tsv
Generating ChIP-exo data analysis project
author_name [Roberto Vera Alvarez]:
email [veraalva@ncbi.nlm.nih.gov]:
project_name [my_ngs_project]:
dataset_name [my_dataset_name]:
is_data_in_SRA [None]:
Select sequencing_technology:
1 - single-end
2 - paired-end
Choose from 1, 2 [1]:
create_demo [y]:
number_spots [5000000]:
organism [human]:
genome_name [hg38]:
genome_dir [hg38]:
aligner_index_dir [hg38/BWA]:
genome_fasta [hg38/genome.fa]:
genome_gtf [hg38/genes.gtf]:
genome_chromsizes [hg38/genome.sizes]:
use_docker [y]:
max_number_threads [16]:
Cloning Git repo: https://github.com/ncbi/cwl-ngs-workflows-cbb to /Users/veraalva/my_
↪ngs_project/bin/cwl
Updating CWLs dockerPull and SoftwareRequirement from: /Users/veraalva/my_ngs_project/
↪requirements/conda-env-dependencies.yaml
bamscale with version 0.0.3 update image to: quay.io/biocontainers/bamscale:0.0.3--
↪ha85820d_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bamscale/bamscale-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bamscale:0.0.5--h18f8b1d_1
bedtools with version 2.29.2 update image to: quay.io/biocontainers/bedtools:2.29.2--
↪hc088bd4_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bedtools/bedtools-docker.yml with␣
↪old image replaced: quay.io/biocontainers/bedtools:2.28.0--hdf88d34_0
bioconductor-diffbind with version 2.16.0 update image to: quay.io/biocontainers/
↪bioconductor-diffbind:2.16.0--r40h5f743cb_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-pca.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
```

(continues on next page)

(continued from previous page)

```
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/macs-cutoff.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/dga_heatmaps.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/diffbind.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/edgeR-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/volcano_plot.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/readQC.cwl with old image␣
↪replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/R/deseq2-2conditions.cwl with old␣
↪image replaced: quay.io/biocontainers/bioconductor-diffbind:2.16.0--r40h5f743cb_2
bwa with version 0.7.17 update image to: quay.io/biocontainers/bwa:0.7.17--hed695b0_7
    /Users/veraalva/my_ngs_project/bin/cwl/tools/bwa/bwa-docker.yml with old image␣
↪replaced: quay.io/biocontainers/bwa:0.7.17--h84994c4_5
There is not biocontainer image for gffread version 0.12.1
homer with version 4.11 update image to: quay.io/biocontainers/homer:4.11--
↪pl526h9a982cc_2
    /Users/veraalva/my_ngs_project/bin/cwl/tools/homer/homer-docker.yml with old␣
↪image replaced: quay.io/biocontainers/homer:4.11--pl526h2bce143_2
mace with version 1.2 update image to: quay.io/biocontainers/mace:1.2--py27h99da42f_0
    /Users/veraalva/my_ngs_project/bin/cwl/tools/mace/mace-docker.yml with old image␣
↪replaced: quay.io/biocontainers/mace:1.2--py27h99da42f_1
meme with version 5.1.1 update image to: quay.io/biocontainers/meme:5.1.1--
↪py37pl526h072abfd_3
    /Users/veraalva/my_ngs_project/bin/cwl/tools/meme/meme-docker.yml with old image␣
↪replaced: quay.io/biocontainers/meme:5.1.1--py27pl526h53063a7_3
Copying file /Users/veraalva/Work/Developer/Python/pm4ngs/pm4ngs-chipexo/example/
↪pm4ngs_chipexo_demo_sample_data.csv  to /Users/veraalva/my_ngs_project/data/my_
↪dataset_name/sample_table.csv
6 files loaded
Using table:
  sample_name file                      condition  replicate
0  SRR4011416        Exp_O2_growth_no_rifampicin          1
1  SRR4011417        Exp_O2_growth_no_rifampicin          2
2  SRR4011421           Exp_O2_growth_rifampicin          1
3  SRR4011425           Exp_O2_growth_rifampicin          2
4  SRR4011418       Stat_O2_growth_no_rifampicin          1
5  SRR4011419       Stat_O2_growth_no_rifampicin          2
 Done
```

The **pm4ngs-chipexo** command line will create a project structure as:

```
.
├── LICENSE
├── README.md
├── bin
│   └── cwl
├── config
│   └── init.py
├── data
│   └── my_dataset_name
├── doc
├── index.html
├── notebooks
```

(continues on next page)

(continued from previous page)

```
    ├── 00 - Project Report.ipynb
    ├── 01 - Pre-processing QC.ipynb
    ├── 02 - Samples trimming.ipynb
    ├── 03 - Alignments.ipynb
    ├── 04 - Peak Calling.ipynb
    └── 05 - MEME Motif.ipynb
├── requirements
    └── conda-env-dependencies.yaml
├── results
    └── my_dataset_name
├── src
└── tmp

12 directories, 11 files
```

Note: ChIPexo based project variables

- **author_name:** Default: [Roberto Vera Alvarez]
- **email:** Default: [veraalva@ncbi.nlm.nih.gov]
- **project_name:** Name of the project with no space nor especial characters. This will be used as project folder's name.

  Default: [my_ngs_project]
- **dataset_name:** Dataset to process name with no space nor especial characters. This will be used as folder name to group the data. This folder will be created under the **data/{{dataset_name}}** and **results/{{dataset_name}}**.

  Default: [my_dataset_name]
- **is_data_in_SRA:** If the data is in the SRA set this to y. A CWL workflow to download the data from the SRA database to the folder **data/{{dataset_name}}** and execute FastQC on it will be included in the **01 - Pre-processing QC.ipynb** notebook.

  Set this option to **n**, if the fastq files names and location are included in the sample sheet.

  Default: [y]
- **Select sequencing_technology:** Select one of the available sequencing technologies in your data.

  Values: 1 - single-end, 2 - paired-end
- **create_demo:** If the data is downloaded from the SRA and this option is set to y, only the number of spots specified in the next variable will be downloaded. Useful to test the workflow.

  Default: [y]: y
- **number_spots:** Number of sport to download from the SRA database. It is ignored is the **create_demo** is set to **n**.

  Default: [1000000]
- **organism:** Organism to process, e.g. human. This is used to link the selected genes to the NCBI gene database.

  Default: [human]
- **genome_name:** Genome name , e.g. hg38 or mm10.

  Default: [hg38]

- **genome_dir:** Absolute path to the directory with the genome annotation (genome.fa, genes.gtf) to be used by the workflow or the name of the genome.

  If the name of the genome is used, PM4NGS will include a cell in the **03 - Alignments.ipynb** notebook to download the genome files. The genome data will be at **data/{{dataset_name}}/{{genome_name}}/**

  Default: [hg38]

- **aligner_index_dir:** Absolute path to the directory with the genome indexes for BWA.

  If **{{genome_name}}/BWA** is used, PM4NGS will include a cell in the **03 - Alignments.ipynb** notebook to create the genome indexes for BWA.

  Default: [hg38/BWA]

- **genome_fasta:** Absolute path to the genome fasta file

  If **{{genome_name}}/genome.fa** is used, PM4NGS will use the downloaded fasta file.

  Default: [hg38/genome.fa]

- **genome_gtf:** Absolute path to the genome GTF file

  If **{{genome_name}}/genes.gtf** is used, PM4NGS will use the downloaded GTF file.

  Default: [hg38/gene.gtf]

- **genome_chromsizes:** TSV file with the genome chromosomes name and size.

  If **{{genome_name}}/genome.sizes** is used, PM4NGS will use the downloaded file.

  Default: [hg38/genome.sizes]

- **use_docker:** Set this to y if you will be using Docker. Otherwise Conda needs to be installed in the computer.

  Default: [y]

- **max_number_threads:** Number of threads available in the computer.

  Default: [16]

8.4 Jupyter server

PM4NGS uses Jupyter as interface for users. After project creation the jupyter server should be started as shown below. The server will open a browser windows showing the project's structure just created.

```
localhost:~> jupyter notebook
```

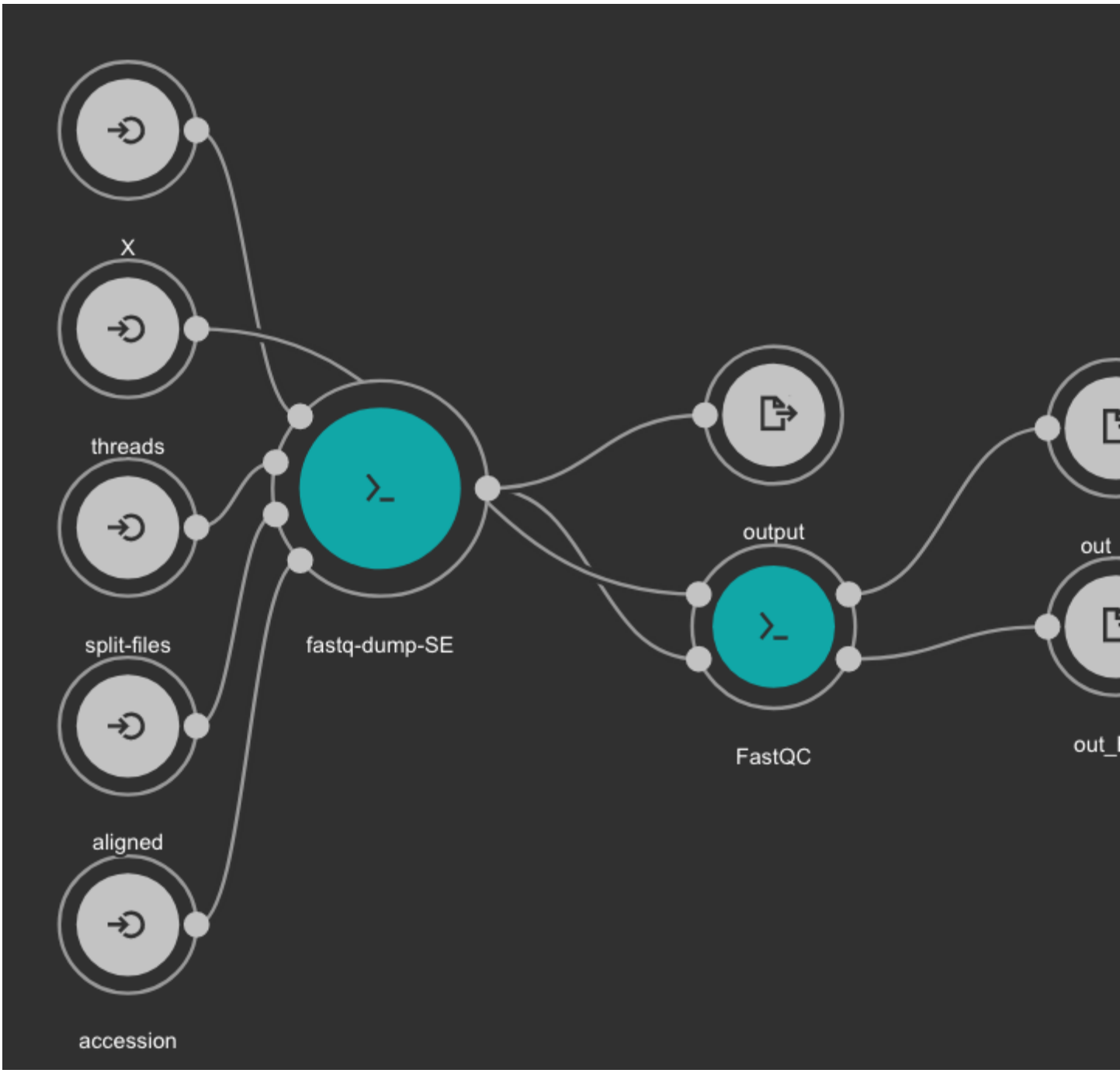
8.5 Data processing

Start executing the notebooks from 01 to 05 waiting for each step completion. The **00 - Project Report.ipynb** notebook can be executed after each notebooks to see the progress in the analysis.

8.6 CWL workflows

8.6.1 SRA download and QC workflow

This CWL workflow is designed to download FASTQ files from the NCBI SRA database using fastq-dump and then, execute fastqc generating a quality control report of the sample.



Inputs

- **accession**: SRA accession ID. Type: string. Required.
- **aligned**: Used it to download only aligned reads. Type: boolean. Optional.
- **split-files**: Dump each read into separate file. Files will receive suffix corresponding to read number. Type: boolean. Optional.
- **threads**: Number of threads. Type: int. Default: 1. Optional.
- **X**: Maximum spot id. Optional.

Outputs

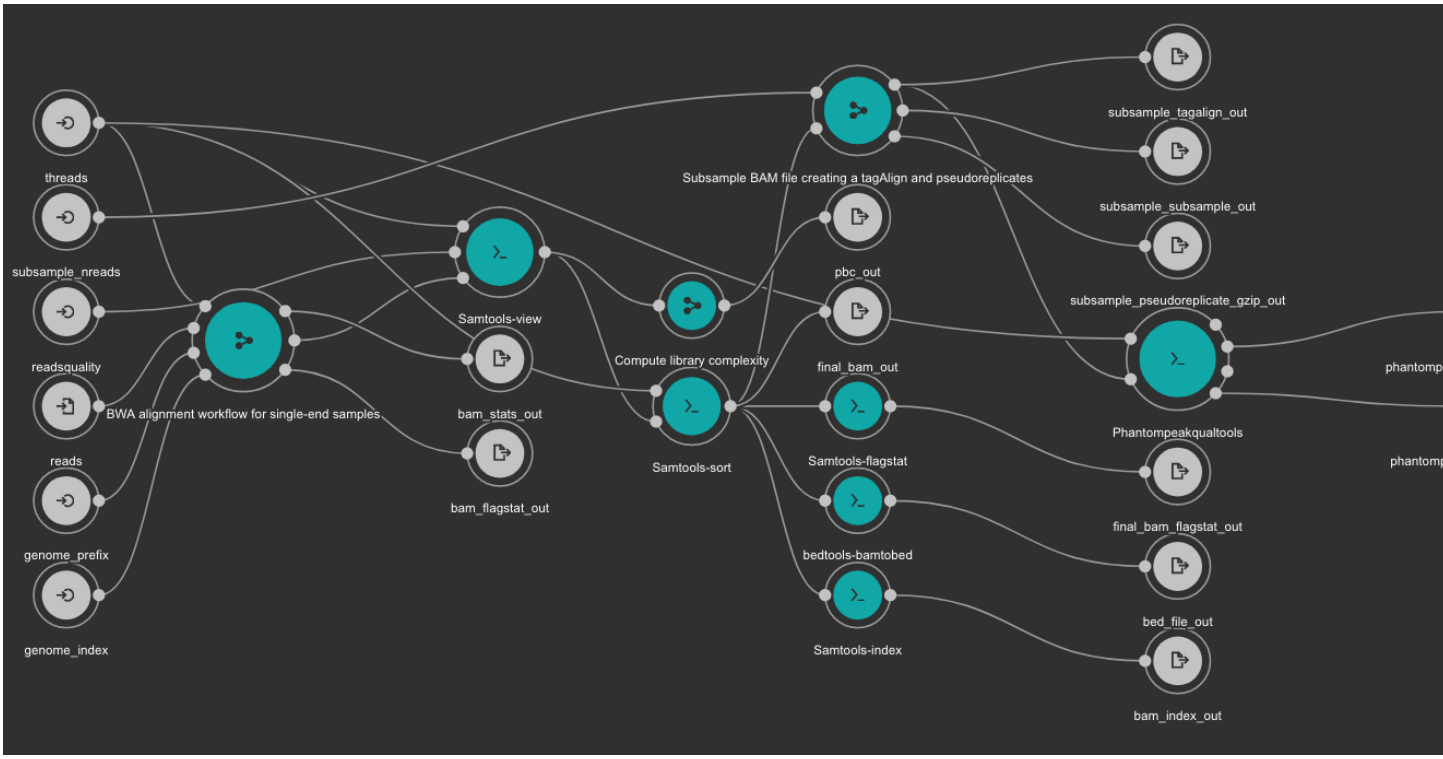
- **output**: Fastq files downloaded. Type: File[]
- **out_zip**: FastQC report ZIP file. Type: File[]
- **out_html**: FastQC report HTML. Type: File[]

8.6.2 Samples Trimming

Our workflows uses Trimmomatic for read trimming. The Jupyter notebooks uses some basic Trimmomatic options that need to be modified depending on the FastQC quality control report generated for the sample.

8.6.3 BWA based alignment and quality control workflow

This workflow use BWA as base aligner. It also use SAMtools and bedtools for file conversion and statistics report. Finally, Phantompeakqualtools is used to generate quality control report for the processed samples.



Inputs

- **genome_index**: Aligner indexes directory. Type: Directory. Required. Variable ALIGNER_INDEX in the Jupyter Notebooks.
- **genome_prefix**: Prefix of the aligner indexes. Generally, it is the name of the genome FASTA file. It can be used as os.path.basename(GENOME_FASTA) in the Jupyter Notebooks. Type: string. Required.
- **readsquality**: Minimum MAPQ value to use reads. We recommend for ChIP_exo dataa value of: 30. Type: int. Required.
- **threads**: Number of threads. Type: int. Default: 1. Optional.
- **subsample_nreads**: Number of reads to be used for the subsample. We recomend for ChIP_exo dataa value of: 500000. Type: int. Required.
- **reads**: FastQ file to be processed. Type: File. Required.

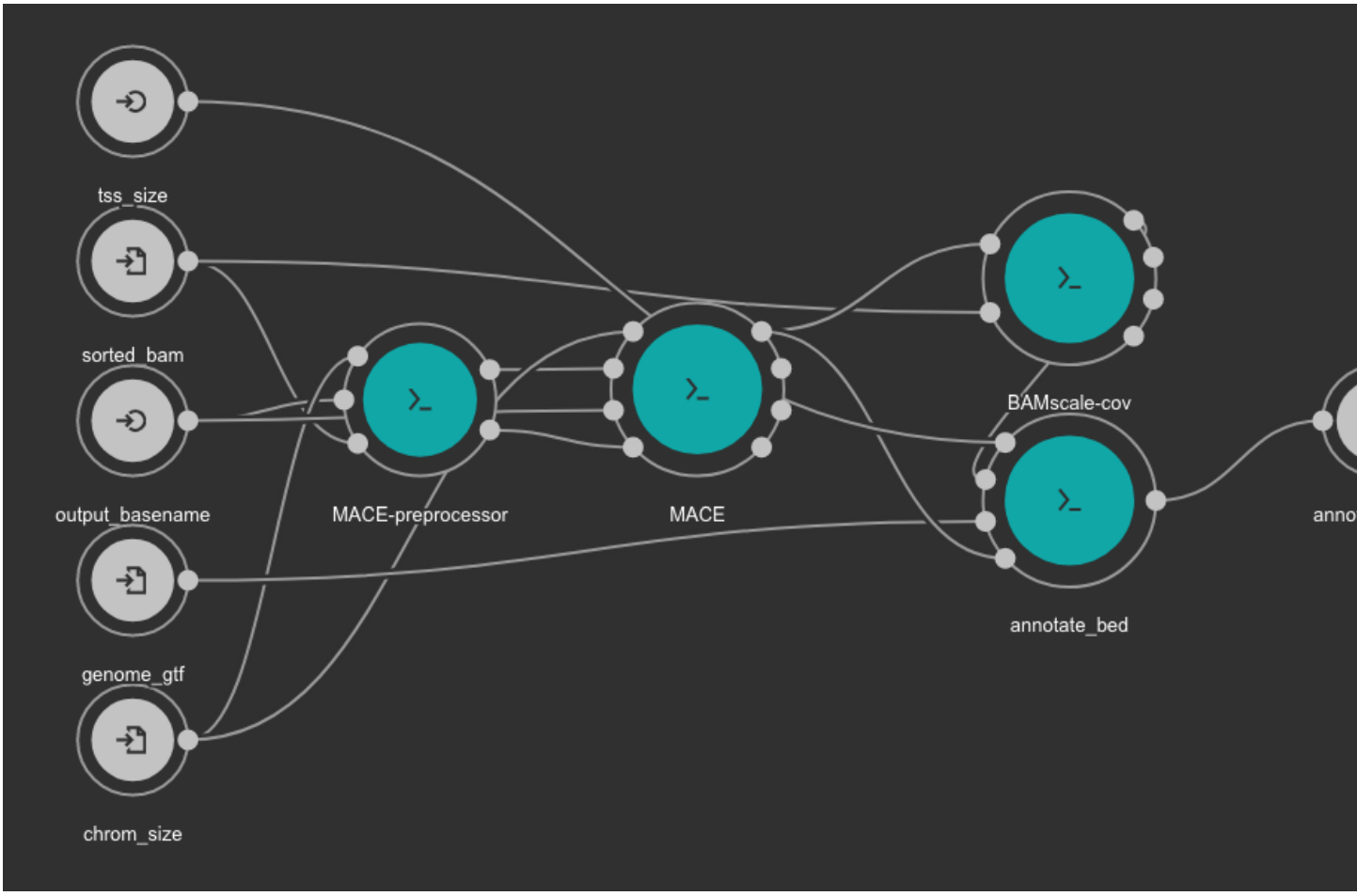
Outputs

- **bam_flagstat_out**: SAMtools flagstats for unfiltered BAM file. Type: File.
- **bam_stats_out**: SAMtools stats for unfiltered BAM file. Type: File.
- **final_bam_flagstat_out**: SAMtools flagstats for filtered BAM file. Type: File.
- **bed_file_out**:: Aligned reads in BED format. Type: File.
- **final_bam_out**: Final BAM file filtered and sorted. Type: File.
- **bam_index_out**: BAM index file. Type: File.

- **pbc_out**: Library complexity report. Type: File.
- **phantompeakqualtools_output_out**: Phantompeakqualtools main output. Type: File.
- **phantompeakqualtools_output_savp**: Phantompeakqualtools SAVP output. Type: File.
- **subsample_pseudoreplicate_gzip_out**: Subsample pseudoreplicates tagAlign gzipped. Type: File[].
- **subsample_tagalign_out**: Subsample tagAlign gzipped. Type: File[].
- **subsample_subsample_out**: Subsample shuffled tagAlign gzipped. Type: File[].

8.6.4 Peak caller workflow using MACE

This workflow uses MACE as peak caller tool. The annotation is created from the GTF file using a *in-house* python script available here. BAMscale is used for the quantification of the resulting peaks.



Inputs

- **chrom_size**: Chromosome size file. Tab or space separated text file with 2 columns: first column is chromosome name, second column is size of the chromosome. Type: File. Required. Variable GENOME_CHROMSIZES in the Jupyter Notebooks.

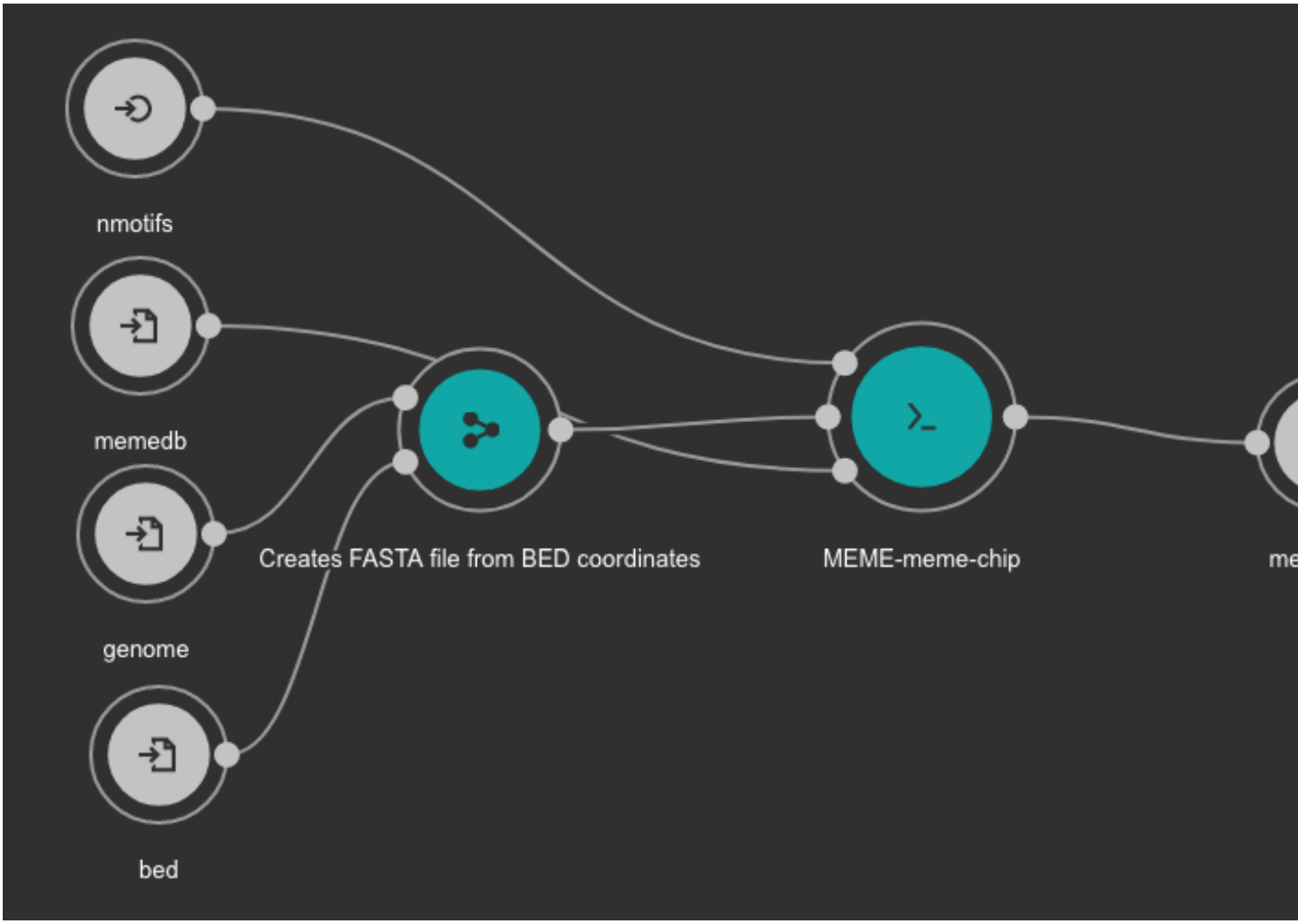
- **output_basename**: Prefix for the output file. Type: string. Required.
- **genome_gtf**: Genome GTF file. Variable GENOME_GTF in the Jupyter Notebooks. Type: File. Required.
- **tss_size**: Number of bp to use for TSS definition. We recommend use 1000. Type: int. Required.

Outputs

- **annotated_bed**: Annotated detected peaks in BED format. Type: File

8.6.5 MEME Motif detection workflow

Motif detection is executed using the MEME suite.



Inputs

- **bed**: BED file with detected peaks. Type: File. Required.

- **memedb**: MEME database for use by Tomtom and CentriMo. Type: File. Required.
- **genome**: Genome FASTA file. Variable GENOME_FASTA in the Jupyter Notebooks. Type: File. Required.
- **nmotifs**: Maximum number of motifs to find. We recommend use 10. Type: int. Required.

Outputs

- **meme_out**: MEME output directory. Type: Directory

8.6.6 MEME databases

MEME workflow depends on the MEME databases. Go to the MEME Suite Download page: http://meme-suite.org/doc/download.html

Download the latest version for the Motif Databases and GOMo Databases.

The downloaded files should be uncompressed in a directory **data/meme**. The final directory should be:

```
localhost:~> cd data
localhost:~> mkdir meme
localhost:~> cd meme
localhost:~> wget http://meme-suite.org/meme-software/Databases/motifs/motif_
↪databases.12.18.tgz
localhost:~> wget http://meme-suite.org/meme-software/Databases/gomo/gomo_databases.3.
↪2.tgz
localhost:~> tar xzf motif_databases.12.18.tgz
localhost:~> tar xzf gomo_databases.3.2.tgz
localhost:~> rm gomo_databases.3.2.tgz motif_databases.12.18.tgz
localhost:~> tree -d
.
├── gomo_databases
└── motif_databases
    ├── ARABD
    ├── CIS-BP
    ├── CISBP-RNA
    ├── ECOLI
    ├── EUKARYOTE
    ├── FLY
    ├── HUMAN
    ├── JASPAR
    ├── MALARIA
    ├── MIRBASE
    ├── MOUSE
    ├── PROKARYOTE
    ├── PROTEIN
    ├── RNA
    ├── TFBSshape
    ├── WORM
    └── YEAST

19 directories
```

See also an example in our test project: https://ftp.ncbi.nlm.nih.gov/pub/pm4ngs/examples/chipexo-single/data/

8.7 Demo

PM4NGS includes a demo project that users can use to test the framework. It is pre-configured to use Docker as execution environment.

The ChIPexo based demo process samples from the BioProject PRJNA338159.

Use this command to create the project structure in your local computer

```
localhost:~> pm4ngs-chipexo-demo
```

Once it finish, start the jupyter server and execute the notebooks as it is indicated on them

```
localhost:~> jupyter notebook
[I 14:12:52.956 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 14:12:52.956 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 14:12:52.956 NotebookApp] http://localhost:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp]  or http://127.0.0.1:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmatio
n).
[C 14:12:52.959 NotebookApp]

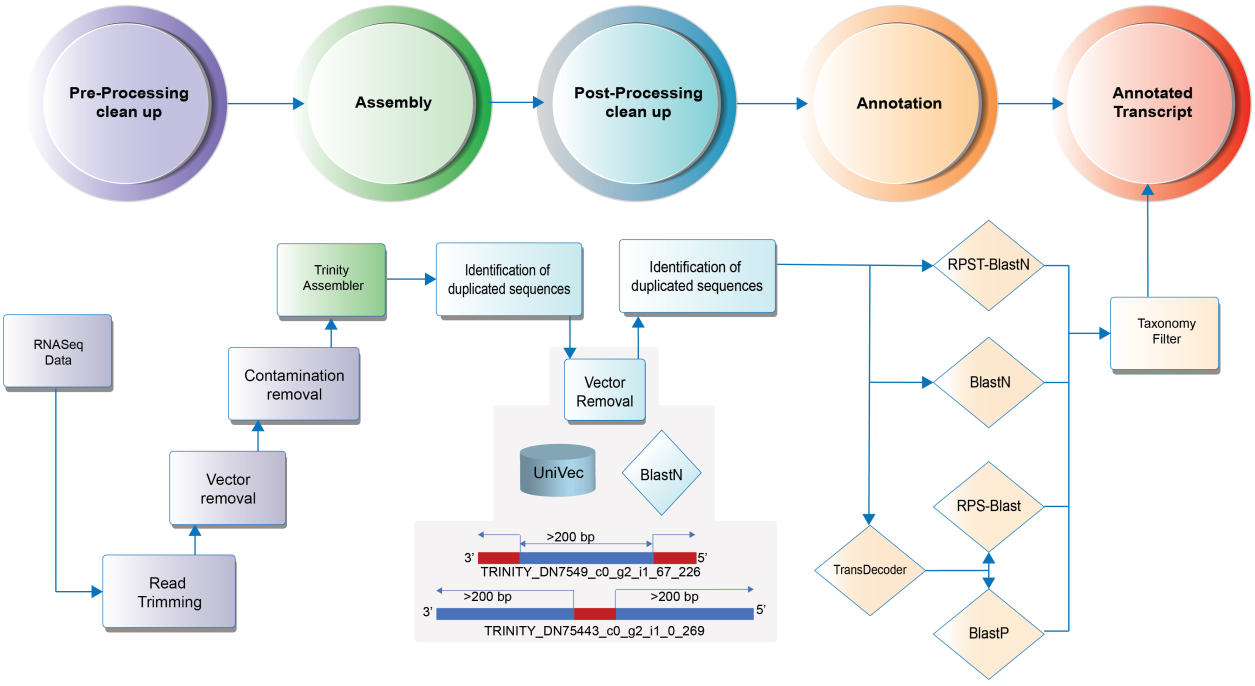
    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-23251-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
     or http://127.0.0.1:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
```

The results of this analysis is https://ftp.ncbi.nlm.nih.gov/pub/pm4ngs/examples/chipexo-single/

CHAPTER 9

Transcriptome Annotation pipeline for non-model organisms

9.1 Annotation Workflow



Attention: This annotation pipeline uses Google Cloud platform for all computing tasks. Users should have installed and configured Cloud SDK.

9.2 GCP configuration

Cloud SDK should be installed and configured. The pipelines are based on the Cloud Life Sciences (beta) API.

To install the beta commands use this command line:

```
localhost:~>  gcloud components install beta
```

kubectl command is also required. This is used by Elastic-blast to configure the Kubernetes cluster on GCP.

It can be installed:

```
localhost:~>  gcloud components install kubectl
```

9.3 Input requirements

The input requirement for this pipeline is the *Sample sheet* file.

9.4 Pipeline command line

The annotation project can be created using the following command line:

```
localhost:~> pm4ngs-transcriptome-annotation
usage: Generate a PM4NGS project for Transcriptome-Annotation data analysis [-h] [-v]
                                 --sample-sheet SAMPLE_SHEET [--config-file CONFIG_
↪FILE] [--copy-rawdata]

Generate a PM4NGS project for Transcriptome-Annotation data analysis:
error: the following arguments are required: --sample-sheet
```

Options:

- **sample-sheet**: Sample sheet with the samples SRA run IDs in the first column
- **config-file**: YAML file with configuration for project creation
- **copy-rawdata**: Copy the raw data defined in the sample sheet to the project structure. (The data can be hosted locally or in an http server)

9.5 Creating the annotation project

The **pm4ngs-transcriptome-annotation** command line executed with the **–sample-sheet** option will let you type the different variables required for creating and configuring the project. The default value for each variable is shown in the brackets.

```
localhost:~> pm4ngs-transcriptome-annotation --sample-sheet my-sample-sheet.tsv
Generating Transcriptome-Annotation data analysis project
author_name [Roberto Vera Alvarez]:
email [veraalva@ncbi.nlm.nih.gov]:
```

(continues on next page)

(continued from previous page)

```
project_name [my_ngs_project]: nopal-annotation
dataset_name [my_dataset_name]: PRJNA320545
Select sequencing_technology:
1 - single-end
2 - paired-end
Choose from 1, 2 [1]: 1
Copying file /home/veraalva/my-sample-sheet.tsv  to /home/veraalva/nopal-annotation/
↪data/PRJNA320545/sample_table.csv
 Done
```

The **pm4ngs-rnaseq** command line will create a project structure as:

```
.
├── LICENSE
├── README.md
├── bin
│   └── gcp
│       ├── pipeline-blastn.json
│       ├── pipeline-contamination-cleanup.json
│       ├── pipeline-download-sra.json
│       ├── pipeline-read-assignment.json
│       ├── pipeline-split-fasta.json
│       ├── pipeline-transcriptome-annotation-rpsblast.json
│       ├── pipeline-transcriptome-annotation-rpstblastn.json
│       ├── pipeline-transcriptome-annotation.json
│       ├── pipeline-transcriptome-cleanup.json
│       ├── pipeline-transcriptome-fastq-cleanup.json
│       ├── pipeline-trimming-fastq-pe.json
│       ├── pipeline-trimming-fastq-se.json
│       └── pipeline-trinity.json
├── config
│   └── init.py
├── data
│   └── PRJNA320545
│       └── sample_table.csv
├── doc
├── notebooks
│   ├── 01 - Download and pre-processing quality control.ipynb
│   ├── 02 - Sample Trimming.ipynb
│   ├── 03 - Vector removal.ipynb
│   ├── 04 - Detecting Contamination.ipynb
│   ├── 05 - Trinity assembly.ipynb
│   ├── 06 - Vector Detection and data Partitioning.ipynb
│   ├── 07 - Transcriptome annotation.ipynb
│   ├── 08 - Transcript Annotation - Blast.ipynb
│   ├── 09 - Transcript Annotation - CDD.ipynb
│   ├── 10 - Transcript Submission to TSA.ipynb
│   ├── 11 - Alignment of raw read to the transcriptome.ipynb
│   └── 12 - Quantifying transcripts.ipynb
├── requirements
│   └── conda-env-dependencies.yaml
├── results
│   └── PRJNA320545
├── src
└── tmp

12 directories, 29 files
```

Note: RNASeq based project variables

- **author_name:** Default: [Roberto Vera Alvarez]
- **email:** Default: [veraalva@ncbi.nlm.nih.gov]
- **project_name:** Name of the project with no space nor especial characters. This will be used as project folder's name.

  Default: [my_ngs_project]
- **dataset_name:** Dataset to process name with no space nor especial characters. This will be used as folder name to group the data. This folder will be created under the **data/{{dataset_name}}** and **results/{{dataset_name}}**.

  Default: [my_dataset_name]
- **Select sequencing_technology:** Select one of the available sequencing technologies in your data.

  Values: 1 - single-end, 2 - paired-end

9.6 Jupyter server

PM4NGS uses Jupyter as interface for users. After project creation the jupyter server should be started as shown below. The server will open a browser windows showing the project's structure just created.

```
localhost:~> jupyter notebook
```

9.7 Data processing

Start executing the notebooks from 01 to 12 waiting for each step completion.

9.8 Demo

PM4NGS includes a demo project that users can use to test the framework. It is pre-configured to use Docker as execution environment.

The annotated based demo process samples from the BioProject PRJNA320545.

Use this command to create the project structure in your local computer

```
localhost:~> pm4ngs-transcriptome-annotation-demo
```

Once it finish, start the jupyter server and execute the notebooks as it is indicated on them

```
localhost:~> jupyter notebook
[I 14:12:52.956 NotebookApp] Serving notebooks from local directory: /home/veraalva
[I 14:12:52.956 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 14:12:52.956 NotebookApp] http://localhost:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
[I 14:12:52.956 NotebookApp]  or http://127.0.0.1:8888/?
↪token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
```

(continues on next page)

(continued from previous page)

```
[I 14:12:52.956 NotebookApp] Use Control-C to stop this server and shut down all␣
↪kernels (twice to skip confirmatio
n).
[C 14:12:52.959 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/veraalva/.local/share/jupyter/runtime/nbserver-23251-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
     or http://127.0.0.1:8888/?token=eae6a8d42ad12d6ace23f5d0923bcec14d0f798127750122
```

CHAPTER 10

Create a new PM4NGS pipeline

The recommended way to create a new pipeline is modifying one of the current available pipelines. Before reading this instructions, the user should understand the basics concepts of the Cookiecutter.

10.1 PM4NGS based pipeline folder structure

The PM4NGS based pipeline folder structure is:

```
.
|── LICENSE
├── README.md
├── cookiecutter.json
├── example
│   ├── pm4ngs_chipseq_demo_config.yaml
│   └── pm4ngs_chipseq_demo_sample_data.csv
├── hooks
│   ├── post_gen_project.py
│   └── pre_gen_project.py
└── {{cookiecutter.project_name}}
    ├── LICENSE
    ├── README.md
    ├── bin
    ├── config
    │   └── init.py
    ├── data
    │   └── {{cookiecutter.dataset_name}}
    ├── doc
    ├── index.html
    ├── notebooks
    │   ├── 00 - Project Report.ipynb
    │   ├── 01 - Pre-processing QC.ipynb
    │   ├── 02 - Samples trimming.ipynb
    │   ├── 03 - Alignments.ipynb
```

(continues on next page)

(continued from previous page)

```
    │   ├── 04 - Peak Calling and IDR.ipynb
    │   └── 05 - Differential binding Detection.ipynb
    ├── requirements
    │   └── conda-env-dependencies.yaml
    ├── results
    │   └── {{cookiecutter.dataset_name}}
    ├── src
    └── tmp

14 directories, 18 files
```

Folders:

- **cookiecutter.json**: the default cookiecutter JSON file where all input variables are defined
- **example**: this folder should include the sample sheet and yaml config file for the pipeline demo
- **hooks**: Cookiecutter pre and post execution scripts. PM4NGS uses these pre and post scripts to execute tasks required to configure the pipeline like cloning the CWL repo, copying the raw data and the sample sheet.

  In the **post_gen_project.py** is where the CWL repository is defined. Users should modify this variable with the localization of their own CWL workflows.

  > CWL_WORKFLOW_REPO = 'https://github.com/ncbi/cwl-ngs-workflows-cbb'

  This repo will be clone to a folder named **cwl** in the project **bin** directory.
- **{{cookiecutter.project_name}}**: this is a folder with the cookiecutter variable for project name.

  This folder store all pipeline's files.
- **config/init.py**: This is the configuration file for the workflow that should be loaded at the beginning of all notebooks. Any function for creating tables, figures and plots not included in PM4NGS should be included in the **init.py** file. Users can also do PR on the PM4NGS repo to the package: pm4ngs.jupyterngsplugin to include their functions in the PM4NGS package but this is not required.
- **notebooks**: In this folder all jupyter notebook are stored.

  The specific requirement for the notebooks are:

  1. Run the **init.py** at the first cell of the notebook

     ```
     %run ../config/init.py
     ```

  2. Use relative paths all the time.
  3. Global variables should be defined in the **init.py** file.
  4. HTML code can be used to display information on the notebook:

     To show a link to the sample sheet this can be used on the notebook cell:

     ```
     filename = os.path.relpath(os.path.join(DATA, DATASET, 'sample_sheet.tsv'))
     html_link = '<a href="{}" target="_blank">sample_sheet.tsv</a>'.
     ↪format(filename.replace(' ', '%20'))
     display(Markdown(html))
     ```

  5. PM4NGS include functions to create links from an image or PDF file: Note that this functions require Poppler (https://poppler.freedesktop.org/) installed.

```
from pm4ngs.jupyterngsplugin.markdown.utils import get_link_image

width = 450
height = 450
filename = os.path.relpath(os.path.join(RESULTS, DATASET, 'dga', 'condition_
↪POST_NACT_CRS2_vs_PRE_NACT_NORMAL_deseq2_pca.pdf'))
html_link = get_link_image(filename, width, height, ' --- ')
display(Markdown(html))
```

- **requirements/conda-env-dependencies.yaml**: Define the conda packages and versions tu run the pipeline.

10.2 CWL tools and workflows specifications

The workflow repository should include two main directories: **tools** and **workflows**. The first directory, **tools**, includes all computational tools used by the workflows in CWL format. The second folder, **workflows**, includes all workflows.

Each CWL tool should include two YAML files with suffixes **bioconda.yml** and **docker.yml** that are imported in the **hints** block.

```
hints:
   - $import: bwa-docker.yml
   - $import: bwa-bioconda.yml
```

See example bwa-mem.cwl

As it is indicated by its names, the **bioconda.yml** files stores the software requirements for executing the CWL tool using Conda. The files specify the package name and version. The CWL runner will create a Conda environment and install the package, if it doesnt exists, at runtime.

```
class: SoftwareRequirement
packages:
   - package: 'bwa'
     version:
       - '0.7.17'
     specs:
       - https://anaconda.org/bioconda/bwa
```

See example bwa-bioconda.yml

The docker.yml file defines the Biocontainers docker image to be used. This image will be pulled by the CWL runner at runtime.

```
class: DockerRequirement
dockerPull: quay.io/biocontainers/bwa:0.7.17--h84994c4_5
```

See example bwa-docker.yml

PM4NGS uses the Biocontainers Registry through its python interface named bioconda2biocontainer to keep CWL docker images defined in the **docker.yml** file updated to its latest tag. The Bioconda package name and version defined in the **bioconda.yml** file is passed as an argument to the update_cwl_docker_from_tool_name method in bioconda2biocontainer returning the latest docker image available for the tool. PM4NGS after cloning the CWL repository, reads the Bioconda package names and version from the **bioconda.yml** files and updates all defined docker images to its latest tags modifying all **docker.yml** files.

CHAPTER 11

Utils

11.1 STAR Genomic Indexes

The **genome.fa** and **genes.gtf** files should be copied to the genome directory.

```
localhost:~> mkdir genome
localhost:~> cd genome
localhost:~> mkdir STAR
localhost:~> cd STAR
localhost:~> cwl-runner https://github.com/ncbi/cwl-ngs-workflows-cbb/blob/master/
↪tools/star/star-index.cwl --runThreadN 16 --genomeDir . --genomeFastaFiles ../
↪genome.fa  --sjdbGTFfile ../genes.gtf
localhost:~> cd ..
localhost:~> tree
.
├── genes.gtf
├── genome.fa
└── STAR
    ├── chrLength.txt
    ├── chrNameLength.txt
    ├── chrName.txt
    ├── chrStart.txt
    ├── exonGeTrInfo.tab
    ├── exonInfo.tab
    ├── geneInfo.tab
    ├── Genome
    ├── genomeParameters.txt
    ├── Log.out
    ├── SA
    ├── SAindex
    ├── sjdbInfo.txt
    ├── sjdbList.fromGTF.out.tab
    ├── sjdbList.out.tab
    └── transcriptInfo.tab
```

(continues on next page)

(continued from previous page)

```
1 directory, 18 files
```

Here all files inside the directory **STAR** are created by the workflow.

11.2 BWA Genomic Indexes

The **genome.fa** file should be copied to the genome directory.

```
localhost:~> mkdir genome
localhost:~> cd genome
localhost:~> mkdir BWA
localhost:~> cd BWA
localhost:~> cwl-runner --no-container https://github.com/ncbi/cwl-ngs-workflows-cbb/
↪blob/master/tools/tools/BWA/bwa-index.cwl --sequences ../genome.fa
localhost:~> cd ..
localhost:~> tree
.
├── BWA
│   ├── genome.fa
│   ├── genome.fa.amb
│   ├── genome.fa.ann
│   ├── genome.fa.bwt
│   ├── genome.fa.pac
│   └── genome.fa.sa
└── genome.fa

1 directory, 7 files
```

11.3 Create BED from GTF

For generating a BED file from a GTF.

The **genes.gtf** file should be copied to the genome directory.

```
localhost:~> mkdir genome
localhost:~> cd genome
localhost:~> cwl-runner --no-container https://github.com/ncbi/cwl-ngs-workflows-cbb/
↪blob/master/workflows/UCSC/gtftobed.cwl --gtf genes.gtf
localhost:~> tree
.
├── genes.bed
├── genes.genePred
└── genome.gtf

0 directory, 3 files
```

Here the files **genes.bed** and **genes.genePred** are created from the workflow.

CHAPTER 12

PM4NGS available genomes

A set of genomes are available for download with the required files already configured. If the genome name is used, PM4NGS will include in the alignment notebook a cell to download, uncompress the genome file and create the aligner indexes.

Note: We recommend to move our of the first project the generated genome folder **data/{{genome_name}}/** to a different folder so it can be reutilized for other projects. This will avoid the extra time for the index creation and will save space.

Table 1: Available genomes

Name	Organism	Source
hg38	Homo sapiens	UCSC
hg19	Homo sapiens	UCSC
mm10	Mus musculus (Mouse)	UCSC
mm9	Mus musculus (Mouse)	UCSC
NC_000913.3	Escherichia coli strain K12, MG1655	NCBI
canFam3	Canis familiaris (Dog)	UCSC
dm6	Drosophila melanogaster	UCSC
TAIR10	Arabidopsis thaliana	NCBI

CHAPTER 13

Introduction

PM4NGS was designed to generate a standard organizational structure for Next Generation Sequencing (NGS) data analysis. It includes a directory structure for the project, several Jupyter notebooks for data management and CWL workflows for pipeline execution.

Our work was inspired by a manuscript by Prof. William Noble in 2009: A Quick Guide to Organizing Computational Biology Projects. We recommend reading this paper for a better understanding of the guiding principles of our project.

The project is composed of three main parts.

1. A project organizational structure which define standard files and directories for the project
2. Jupyter Notebooks as user interfaces for data management and visualization
3. CWL workflows that execute the data analysis

PM4NGS source code includes the templates used by **cookiecutter** to generate the project organizational structure and the Jupyter notebooks. The CWL workflows are defined in a separate GitHub project named: cwl-ngs-workflows-cbb.

CHAPTER 14

Features

- NGS data integration, management and analysis uses Jupyter notebooks, CWL workflows and cookiecutter project templates
- Easy installation and use with a minimum command line interaction
- Data analysis CWL workflows executed from the Jupyter notebook with automatic failing detection and can be validated with published data
- CWL workflows and Jupyter Notebooks are ready for cloud computing
- Project reports and dynamic content creation after data processing using CWL workflows are included
- Optional use of Docker/Biocontainers or Conda/Bioconda for Bioinformatics tool installations and managements are also included

CHAPTER 15

Citation

1. Vera Alvarez R, Pongor LS, Mariño-Ramírez L and Landsman D. PM4NGS, a project management framework for next-generation sequencing data analysis, GigaScience, Volume 10, Issue 1, January 2021, giaa141, https://doi.org/10.1093/gigascience/giaa141

2. Vera Alvarez R, Mariño-Ramírez L and Landsman D. Transcriptome annotation in the cloud: complexity, best practices, and cost, GigaScience, Volume 10, Issue 2, February 2021, giaa163, https://doi.org/10.1093/gigascience/giaa163

3. Vera Alvarez R, Pongor LS, Mariño-Ramírez L and Landsman D. Containerized open-source framework for NGS data analysis and management [version 1; not peer reviewed]. F1000Research 2019, 8(ISCB Comm J):1229 (poster) (doi: 10.7490/f1000research.1117155.1)

CHAPTER 16

Help and Support

For query/questions regarding snakePipes, please write on biostars with the tag **#PM4NGS**

For feature requests or bug reports, please open an issue on our GitHub Repository.

CHAPTER 17

Public Domain notice

National Center for Biotechnology Information.

This software is a "United States Government Work" under the terms of the United States Copyright Act. It was written as part of the authors' official duties as United States Government employees and thus cannot be copyrighted. This software is freely available to the public for use. The National Library of Medicine and the U.S. Government have not placed any restriction on its use or reproduction.

Although all reasonable efforts have been taken to ensure the accuracy and reliability of the software and data, the NLM and the U.S. Government do not and cannot warrant the performance or results that may be obtained by using this software or data. The NLM and the U.S. Government disclaim all warranties, express or implied, including warranties of performance, merchantability or fitness for any particular purpose.

Please cite NCBI in any work or product based on this material.