
PM4NGS Documentation

Release 1.0.0

Roberto Vera Alvarez

Jun 28, 2021

CONTENTS

1	RNA-seq of the lobe-less (lol) mutants in <i>Drosophila melanogaster</i>	1
1.1	BioProject	2
1.2	SRA	3
2	GCP Instance Setup	7
2.1	Create a GCP instance	7
2.2	Accessing the instance with SSH	9
2.3	Installing dependencies on the GCP instance with Ubuntu	10
2.4	Testing the Docker daemon	10
3	Installing PM4NGS	11
3.1	What is a Python virtual environment?	11
3.2	Installing PM4NGS	11
3.3	Testing PM4NGS	12
4	The Jupyter Server	13
4.1	Accessing the instance with SSH	13
4.2	Activating pm4ngs_venv virtual environment	14
4.3	Running the Jupyter Server	14
4.4	Open the jupyter web interface in your local computer	15
5	PM4NGS for RNA-Seq data analysis	17
5.1	Sample sheet	19
5.2	Creating the PM4NGS RNA-Seq project	20
5.3	Pre-processing QC	24
5.4	Samples trimming	31
5.5	Alignment and Quantification	36
5.6	Differential Gene Expression Analysis	44
5.6.1	Volcano Plots	46
5.6.2	Sample correlation	47
5.6.3	Expression correlation	47
5.6.4	PCA plots	47
5.6.5	List of differentially expressed genes	48
5.7	Gene Ontology Enrichment	48
6	Introduction	53
7	Background reading	55
8	Help and Support	57

RNA-SEQ OF THE LOBE-LESS (LOL) MUTANTS IN DROSOPHILA MELANOGASTER

Lobe-less (*lol*) encodes a long noncoding RNA (lncRNA) required for the development of *Drosophila* mushroom body, a center of insect memories and learning. RNA-seq was carried out in the control (*w*[1118]) and *lol* mutant flies to identify genes regulated by *lol* lncRNA.

From the manuscript: Genetic interactions between Protein Kinase D and Lobe mutants during eye development of *Drosophila melanogaster*

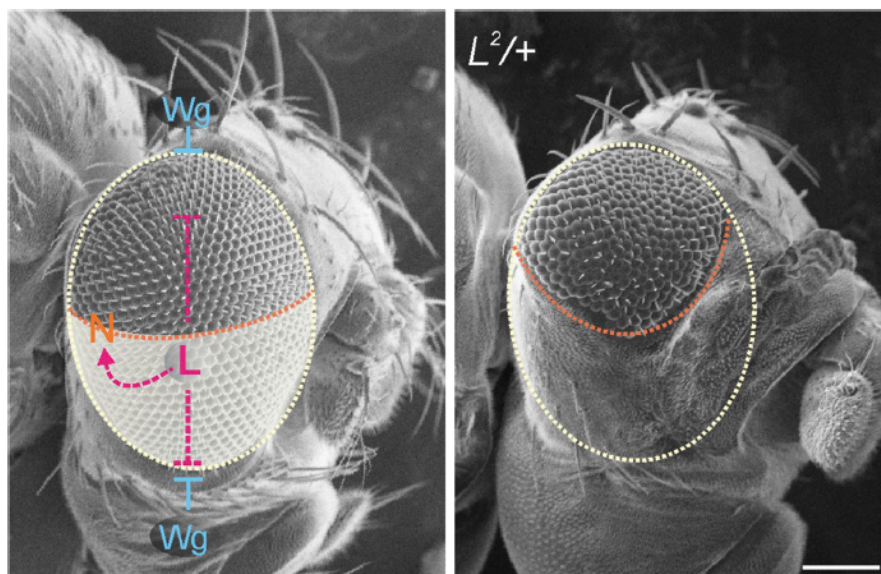


Fig. 1 Development of the ventral eye is affected by the *L* mutation. During development, the *Drosophila* eye field is confined by the activity of Wg. Establishment of the dorso-ventral boundary requires the activation of N. Concluded from genetic interactions, *Lobe* acts as a positive regulator of N and a negative regulator of Wg activities. Accordingly, *L*²/+ mutant flies display small eyes, where the ventral eye compartment is reduced at the expense of head epidermis. Arrows indicate positive, and bars negative interactions. Size bar, 100 μ m

1.1 BioProject

A BioProject is a collection of biological data related to a single initiative, originating from a single organization or from a consortium. A BioProject record provides users a single place to find links to the diverse data types generated for that project.

BioProject: [PRJDB5673](https://ncbi.nlm.nih.gov/bioproject/PRJDB5673)

The screenshot shows the NCBI BioProject page for PRJDB5673. The page includes a search bar, a navigation menu, and a main content area with various sections. A red arrow points to the 'SRA Experiments' row in the 'Project Data' table, which is labeled 'SRA samples'.

Project Data:

Resource Name	Number of Links
SEQUENCE DATA	
SRA Experiments	26
OTHER DATASETS	
BioSample	26

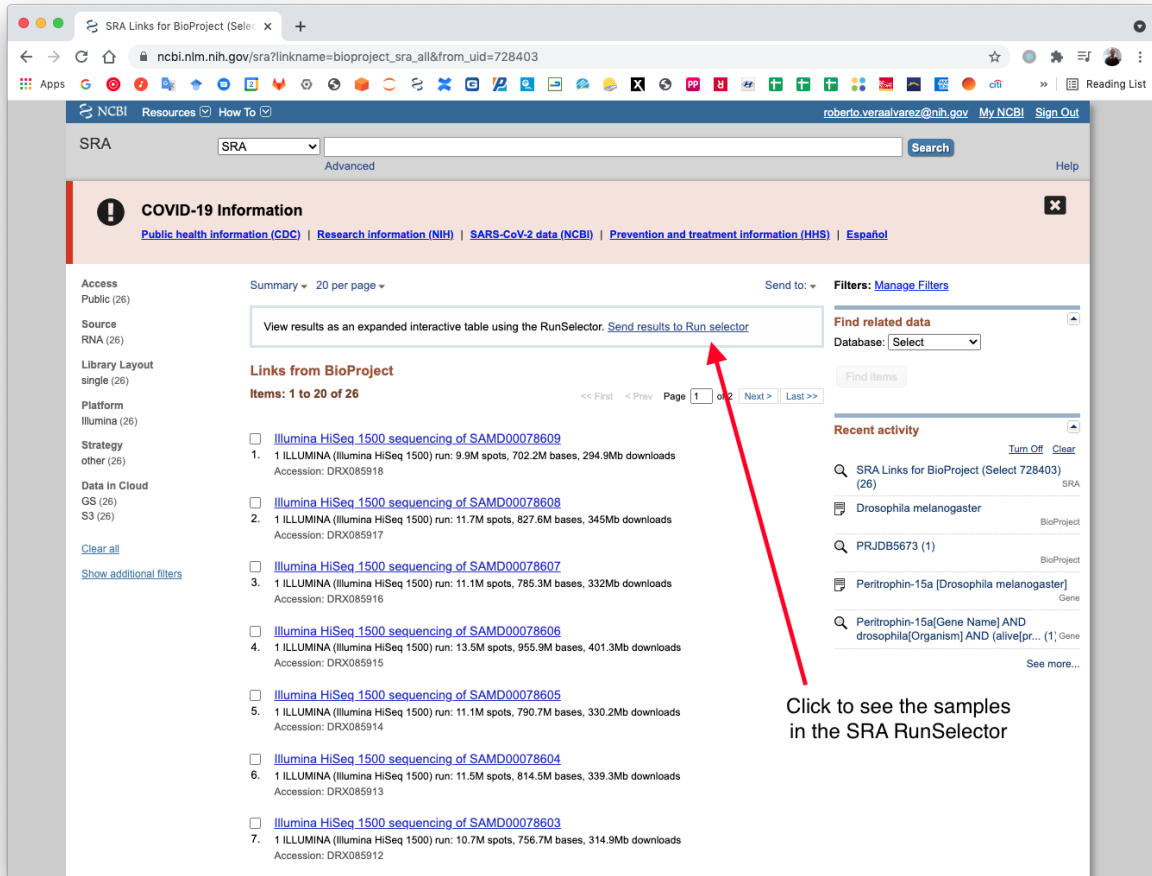
SRA Data Details:

Parameter	Value
Data volume, Gbases	20
Data volume, Mbytes	8985

1.2 SRA

Sequence Read Archive (SRA) data, available through multiple cloud providers and NCBI servers, is the largest publicly available repository of high throughput sequencing data.

List of SRA samples in the BioProject



The screenshot shows the NCBI SRA Links for BioProject (Select 728403) page. The page displays a list of 7 SRA runs, all from the Illumina HiSeq 1500 platform. A red arrow points to the 'Send results to Run selector' link in the top right corner of the results table.

COVID-19 Information
[Public health information \(CDC\)](#) | [Research information \(NIH\)](#) | [SARS-CoV-2 data \(NCBI\)](#) | [Prevention and treatment information \(HHS\)](#) | [Español](#)

Access: Public (26)
Source: RNA (26)
Library Layout: single (26)
Platform: Illumina (26)
Strategy: other (26)
Data in Cloud: GS (26), S3 (26)
[Clear all](#)
[Show additional filters](#)

Summary 20 per page
 Send to: [Manage Filters](#)

View results as an expanded interactive table using the RunSelector. [Send results to Run selector](#)

Links from BioProject
 Items: 1 to 20 of 26
 << First < Prev Page 1 of 2 Next > Last >>

Run ID	Platform	Run Length	Spots	Bases	Downloads
1. ILLUMINA (Illumina HiSeq 1500) run: 9.9M spots, 702.2M bases, 294.9Mb downloads Accession: DRX085918	Illumina HiSeq 1500	9.9M spots	702.2M bases	294.9Mb downloads	
2. ILLUMINA (Illumina HiSeq 1500) run: 11.7M spots, 827.6M bases, 345Mb downloads Accession: DRX085917	Illumina HiSeq 1500	11.7M spots	827.6M bases	345Mb downloads	
3. ILLUMINA (Illumina HiSeq 1500) run: 11.1M spots, 785.3M bases, 332Mb downloads Accession: DRX085916	Illumina HiSeq 1500	11.1M spots	785.3M bases	332Mb downloads	
4. ILLUMINA (Illumina HiSeq 1500) run: 13.5M spots, 955.9M bases, 401.3Mb downloads Accession: DRX085915	Illumina HiSeq 1500	13.5M spots	955.9M bases	401.3Mb downloads	
5. ILLUMINA (Illumina HiSeq 1500) run: 11.1M spots, 790.7M bases, 330.2Mb downloads Accession: DRX085914	Illumina HiSeq 1500	11.1M spots	790.7M bases	330.2Mb downloads	
6. ILLUMINA (Illumina HiSeq 1500) run: 11.5M spots, 814.5M bases, 339.3Mb downloads Accession: DRX085913	Illumina HiSeq 1500	11.5M spots	814.5M bases	339.3Mb downloads	
7. ILLUMINA (Illumina HiSeq 1500) run: 10.7M spots, 756.7M bases, 314.9Mb downloads Accession: DRX085912	Illumina HiSeq 1500	10.7M spots	756.7M bases	314.9Mb downloads	

Recent activity
 Turn Off Clear
 SRA Links for BioProject (Select 728403) (26) SRA
 Drosophila melanogaster BioProject
 PRJDB5673 (1) BioProject
 Pentrophin-15a [Drosophila melanogaster] Gene
 Pentrophin-15a[Gene Name] AND drosophila[Organism] AND (alive[pr... (1) Gene
 See more...

Click on the **Send results to Run selector** to see the SRA data in the Run selector app.

Filters List

- 1 Bases
- 2 Bytes
- 3 dev_stage
- 4 Replicate
- 5 sex
- 6 tissue_type

Common Fields

BioProject: PRJDB5673
 Consent: PUBLIC
 Assay Type: RNA-Seq
 AvgSpotLen: 71
 Center Name: RIKEN_CLST_DBFDI
 DATASTORE filetype: SRA
 DATASTORE provider: GS, NCBI, S3
 DATASTORE region: gs.US, ncbl.public, s3.us-east-1
 Instrument: Illumina HiSeq 1500

Select

	Runs	Bytes	Bases	Download	Cloud Data Delivery	Computing
Total	26	8.37 Gb	20.41 G	Metadata or Accession List		
Selected	0	0	0	Metadata or Accession List or JWT Cart	Deliver Data	Galaxy

Found 25 Items

	Run	BioSample	Bases	Bytes	Experiment	Library Name	Replicate	Sample Name	Sample_name	tissue_type	dev_stage	sex
☐	1	DRR092329	SAMD00078590	657.84 M	271.58 Mb	DRX085899	Wild type, stage 15-17 embryo, replicate 1	biological replicate 1	SAMD00078590	W_Embryo_Stage_15-17.1	embryo	
☐	2	DRR092330	SAMD00078591	742.99 M	305.27 Mb	DRX085900	Wild type, stage 15-17 embryo, replicate 2	biological replicate 2	SAMD00078591	W_Embryo_Stage_15-17.2	embryo	
☐	3	DRR092331	SAMD00078592	765.31 M	316.39 Mb	DRX085901	Wild type, stage 15-17 embryo, replicate 3	biological replicate 3	SAMD00078592	W_Embryo_Stage_15-17.3	embryo	
☐	4	DRR092332	SAMD00078593	899.55 M	371.30 Mb	DRX085902	Lol mutant, stage 15-17 embryo, replicate 1	biological replicate 1	SAMD00078593	lol_Embryo_Stage_15-17.1	embryo	
☐	5	DRR092333	SAMD00078594	728.55 M	301.00 Mb	DRX085903	Lol mutant, stage 15-17 embryo, replicate 2	biological replicate 2	SAMD00078594	lol_Embryo_Stage_15-17.2	embryo	
☐	6	DRR092334	SAMD00078595	712.66 M	296.39 Mb	DRX085904	Lol mutant, stage 15-17 embryo, replicate 3	biological replicate 3	SAMD00078595	lol_Embryo_Stage_15-17.3	embryo	
☐	7	DRR092335	SAMD00078596	767.17 M	322.06 Mb	DRX085905	Wild type, 1st instar larva, replicate 1	biological replicate 1	SAMD00078596	W_1st_Instar_Larva_1	larva	
☐	8	DRR092336	SAMD00078597	706.17 M	294.40 Mb	DRX085906	Wild type, 1st instar larva, replicate 2	biological replicate 2	SAMD00078597	W_1st_Instar_Larva_2	larva	

Select samples to be analyzed: DRR092341, DRR092342, DRR092343, DRR092344. Wild type, adult male head compared with Lol mutant, adult male head.

Filters List

- 1 Bases
- 2 Bytes
- 3 dev_stage
- 4 Replicate
- 5 sex
- 6 tissue_type

Common Fields

Assay Type: RNA-Seq
 AvgSpotLen: 71
 Center Name: RIKEN_CLST_DBFDI
 DATASTORE filetype: SRA
 DATASTORE provider: GS, NCBI, S3
 DATASTORE region: gs.US, ncbl.public, s3.us-east-1
 Instrument: Illumina HiSeq 1500
 LibraryLayout: SINGLE
 LibrarySelection: POLYA

Select

	Runs	Bytes	Bases	Download	Cloud Data Delivery	Computing
Total	26	8.37 Gb	20.41 G	Metadata or Accession List		
Selected	4	1.26 Gb	3.09 G	Metadata or Accession List or JWT Cart	Deliver Data	Galaxy

Show only selected samples

Click to request data delivery to a cloud provider

You selected 4 Items

	Run	BioSample	Bases	Bytes	Experiment	Library Name	Replicate	Sample Name	Sample_name	tissue_type	dev_stage	sex
☒	1	DRR092341	SAMD00078602	731.37 M	307.54 Mb	DRX085911	Wild type, adult male head, replicate 1	biological replicate 1	SAMD00078602	W_Adult_Male_Head_1	head	adult male
☒	2	DRR092342	SAMD00078603	756.69 M	314.91 Mb	DRX085912	Wild type, adult male head, replicate 2	biological replicate 2	SAMD00078603	W_Adult_Male_Head_2	head	adult male
☒	3	DRR092343	SAMD00078604	814.54 M	339.34 Mb	DRX085913	Lol mutant, adult male head, replicate 1	biological replicate 1	SAMD00078604	lol_Adult_Male_Head_1	head	adult male
☒	4	DRR092344	SAMD00078605	790.68 M	330.15 Mb	DRX085914	Lol mutant, adult male head, replicate 2	biological replicate 2	SAMD00078605	lol_Adult_Male_Head_2	head	adult male

Click on **Deliver Data** to request NCBI to transfer the data to a cloud bucket.

Deliver Data

Follow the steps below to request Sequence Read Archive (SRA) data be delivered to an Amazon Web Services (AWS) or Google Cloud Platform (GCP) bucket of your choice.

- 1 Select runs to deliver**
DRR092341, DRR092342, DRR092343, DRR092344
[Change selected runs](#)
- 2 Choose destination bucket**
There is no previous bucket on your record, please add one to deliver data.
Cloud provider
☐ AWS ☒ GCP
Bucket name

To grant permissions, assign the **Storage Admin** role for your bucket to the NCBI service accounts:
csvm-service@nih-sra-datastore.iam.gserviceaccount.com
csvm-service@nih-sra-datastore-protected.iam.gserviceaccount.com
For instructions, see [Adding a member to a bucket-level policy](#) in the *Google Cloud Documentation*.
[Add bucket](#)
- 3 Select one or more source file types**
Add a bucket to proceed.

Helpful information
Additional details to help you make your request are available below.

- [Creating a bucket](#)
- [Basic cost information](#)
- [Selecting region & storage class](#)
- [NCBI restoration & delivery limits](#)

[Feedback](#)

GCP INSTANCE SETUP

2.1 Create a GCP instance

Create a GCP instance with 16 CPUs and 60GB of RAM (machine type: **n1-standard-16**). Click on **Change** button in the **Boot disk** box to select Operating system: Ubuntu, Version: Ubuntu 20.04 LTS. Click on **Management** to select **Preemptibility: ON**

Name ?
Name is permanent

Labels ? (Optional)

Region ?
Region is permanent

Zone ?
Zone is permanent

Machine configuration**Machine family**

☒ General-purpose ☐ Compute-optimized ☐ Memory-optimized

Machine types for common workloads, optimized for cost and flexibility

Series

Powered by Intel Skylake CPU platform or one of its predecessors

Machine type

vCPU

16

Memory

60 GB

GPUs

-

☒ CPU platform and GPU

Confidential VM service ?

☐ Enable the Confidential Computing service on this VM instance.

Container ?

☐ Deploy a container image to this VM instance. [Learn more](#)

Boot disk ?

New 100 GB balanced persistent disk

Image

☒ Ubuntu 20.04 LTS

[Change](#)

Identity and API access ?**Service account** ?
Access scopes ?

Select a service account to enable API access

Firewall ?

Add tags and firewall rules to allow specific network traffic from the Internet

- ☐ Allow HTTP traffic
☐ Allow HTTPS traffic

☒ [Management, security, disks, networking, sole tenancy](#)

The following options have been customized:

Preemptibility
Automatic restart
On host maintenance

You will be billed for this instance. [Compute Engine pricing](#)

[Create](#) [Cancel](#)

Equivalent [REST](#) or [command line](#)

\$126.80 monthly estimate

That's about \$0.174 hourly

Pay for what you use: No upfront costs and per second billing

Item	Estimated costs
16 vCPUs + 60 GB memory	\$116.80/month
100 GB balanced persistent disk	\$10.00/month
Sustained use discount ?	- \$0.00/month
Total	\$126.80/month

[Compute Engine pricing](#)

[^ Less](#)

2.2 Accessing the instance with SSH

The instance will be available with two IP address, one internal for GCP and another external accessible from Internet. Click on the **SSH** button to login.

INSTANCES INSTANCE SCHEDULE

Filter Enter property name or value

Click in SSH to login

☐	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect
☐	✓	instance-veraalva	us-east1-b			10.142.0.3 (nic0)	35.231.191.114	SSH

After clicking the **SSH** button, a new windows will be open. This is a Linux terminal running directly in the instance.

```
r78v10a07@instance-veraalva: ~
ssh.cloud.google.com/projects/pm4ngs-rnaseq-tutorial/zones/us-east1-b/instances/instance-veraalva?authuser=0&hl=en_US&p...
* Documentation: https://help.ubuntu.com
* Management:   https://landscape.canonical.com
* Support:      https://ubuntu.com/advantage

System information as of Tue Jun  8 19:13:27 UTC 2021

System load:  0.0           Processes:    249
Usage of /:   1.6% of 96.75GB Users logged in: 0
Memory usage: 0%           IPv4 address for ens4: 10.142.0.3
Swap usage:  0%

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

r78v10a07@instance-veraalva:~$
```

2.3 Installing dependencies on the GCP instance with Ubuntu

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

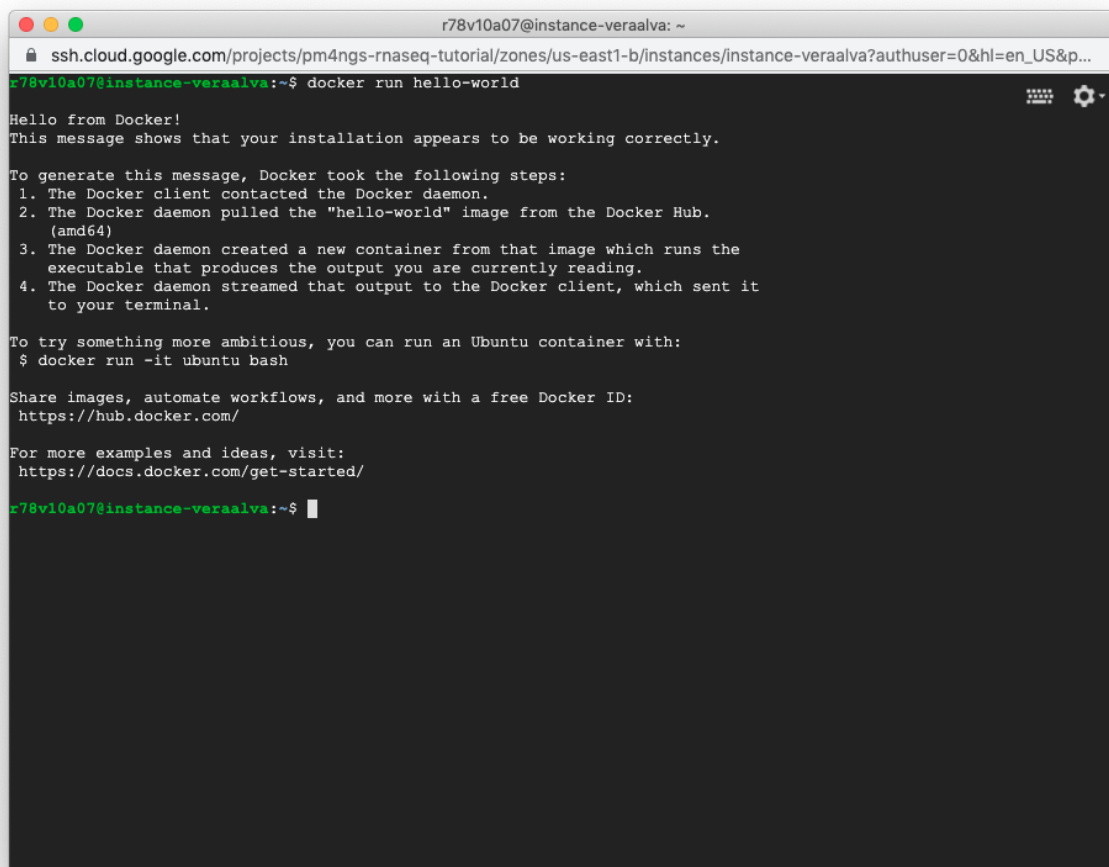
```
veraalva@instance-veraalva:~$ sudo apt-get update
veraalva@instance-veraalva:~$ sudo apt-get install docker.io python3 python3-pip python3-
venv python3-dev poppler-utils gcc nodejs tree
veraalva@instance-veraalva:~$ sudo usermod -aG docker $USER
veraalva@instance-veraalva:~$ logout
```

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.

2.4 Testing the Docker daemon

```
veraalva@instance-veraalva:~$ docker run hello-world
```

Docker will pull the **hello-world** image and run it in a container. A **Hello from Docker!** message is displayed in the terminal.



```
r78v10a07@instance-veraalva: ~
ssh.cloud.google.com/projects/pm4ngs-rnaseq-tutorial/zones/us-east1-b/instances/instance-veraalva?authuser=0&hl=en_US&p...
r78v10a07@instance-veraalva:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

r78v10a07@instance-veraalva:~$
```

INSTALLING PM4NGS

Read PM4NGS published manuscript in [GIGAScience](#)

3.1 What is a Python virtual environment?

A python virtual environment is a "new Python installation" with their own site directories, optionally isolated from system site directories. Each virtual environment has its own Python binary (which matches the version of the binary that was used to create this environment) and can have its own independent set of installed Python packages in its site directories.

Creation of virtual environments is done by executing the command `venv`:

Note: `python3 -m venv /path/to/new/virtual/environment`

Running this command creates the target directory (creating any parent directories that don't exist already), a common name for the target directory uses as suffix `_venv`. It also creates a `bin` subdirectory containing a copy/symlink of the Python binary/binaries. It also creates an (initially empty) `lib/pythonX.Y/site-packages` subdirectory for the new packages. If an existing directory is specified, it will be re-used.

Once a virtual environment has been created, it can be “**activated**” using a script in the virtual environment's binary directory. The invocation of the script is platform-specific (`<_venv>` must be replaced by the path of the directory containing the virtual environment):

Note: `$ source /path/to/new/virtual/environment/bin/activate`

3.2 Installing PM4NGS

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

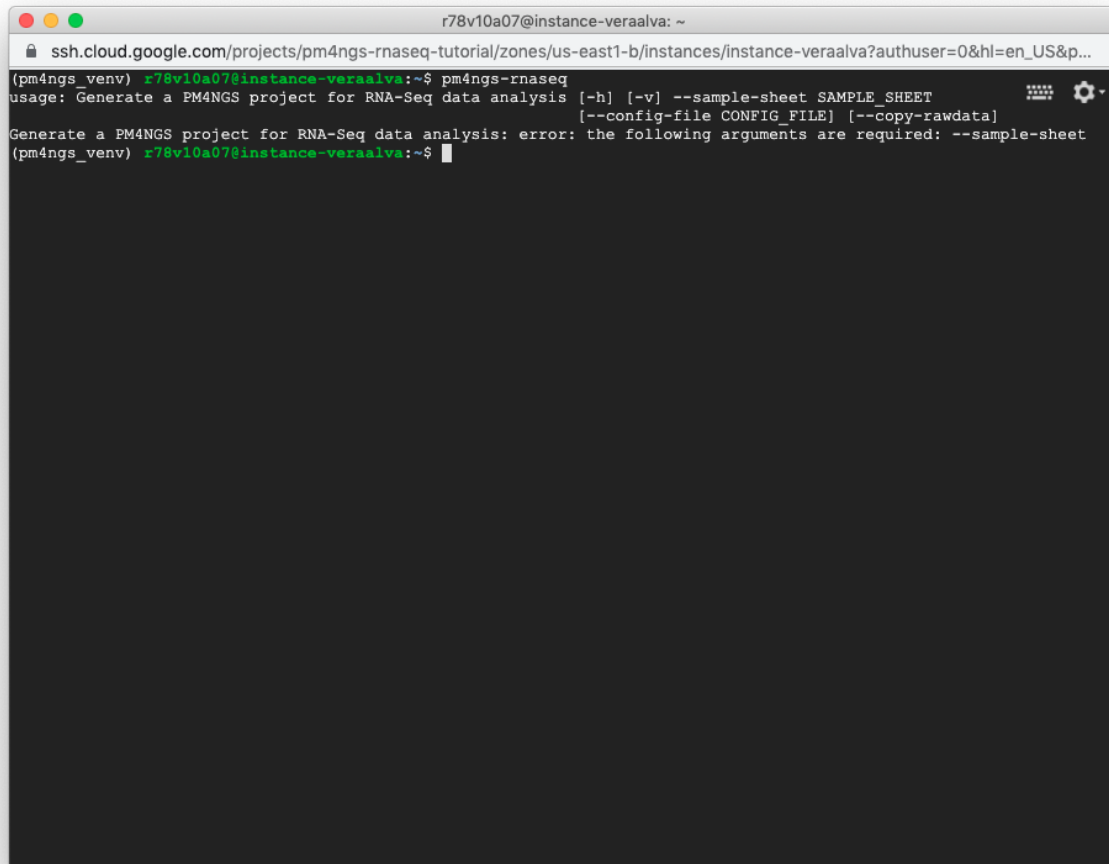
```
veraalva@instance-veraalva:~$ which python3
/usr/bin/python3
veraalva@instance-veraalva:~$ python3 -m venv pm4ngs_venv
veraalva@instance-veraalva:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) veraalva@instance-veraalva:~$ which python3
/home/r78v10a07/pm4ngs_venv/bin/python3
(pm4ngs_venv) veraalva@instance-1:~$ pip install wheel
(pm4ngs_venv) veraalva@instance-1:~$ pip install pm4ngs
```

These commands need to be executed only one time during the tutorial.

3.3 Testing PM4NGS

Test PM4NGS RNA-Seq module:

```
(pm4ngs_venv) veraalva@instance-veraalva:~$ pm4ngs-rnaseq
```



The screenshot shows a terminal window with a title bar indicating the user is connected via SSH to an instance named 'veraalva'. The terminal shows the command `pm4ngs-rnaseq` being executed, which results in an error message: `usage: Generate a PM4NGS project for RNA-Seq data analysis [-h] [-v] --sample-sheet SAMPLE_SHEET [--config-file CONFIG_FILE] [--copy-rawdata]`. The error message states: `Generate a PM4NGS project for RNA-Seq data analysis: error: the following arguments are required: --sample-sheet`. The terminal prompt is `(pm4ngs_venv) r78v10a07@instance-veraalva:~$`.

THE JUPYTER SERVER

Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document. Computational notebooks are essentially laboratory notebooks for scientific computing that keep track of all steps in a research project. Jupyter notebooks can be shared with collaborators and are used to guarantee reproducibility.

The Jupyter notebook has two components. Users input programming code or text in rectangular cells in a front-end web page. The browser then passes that code to a back-end ‘kernel’, which runs the code and returns the results. The kernel reside in the server, executing all code in that computer and the front-end in the user local browser.

Read this interesting article about Jupyter Notebooks for data scientists: <https://www.nature.com/articles/d41586-018-07196-1>

4.1 Accessing the instance with SSH

Click on the **SSH** button to login.

INSTANCES

INSTANCE SCHEDULE

Filter

Enter property name or value

Click in SSH to login

?

☰

☐	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect	
☐	✓	instance-veaalva	us-east1-b			10.142.0.3 (nic0)	35.231.191.114	SSH	⋮

After clicking the **SSH** button, a new windows will be open. This is a Linux terminal running directly in the instance.

```
r78v10a07@instance-veraalva: ~  
ssh.cloud.google.com/projects/pm4ngs-rnaseq-tutorial/zones/us-east1-b/instances/instance-veraalva?authuser=0&hl=en_US&p...  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
System information as of Tue Jun  8 19:13:27 UTC 2021  
  
System load:  0.0           Processes:            249  
Usage of /:   1.6% of 96.75GB Users logged in:          0  
Memory usage: 0%           IPv4 address for ens4: 10.142.0.3  
Swap usage:   0%  
  
1 update can be applied immediately.  
To see these additional updates run: apt list --upgradable  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
r78v10a07@instance-veraalva:~$
```

4.2 Activating pm4ngs_venv virtual environment

Everytime we login into the instance, we need to activate the **pm4ngs_venv** virtual environment. This mean that we will be using the python packages installed in this virtual environment instead of the packages in the instance.

```
veraalva@instance-veraalva:~$ source pm4ngs_venv/bin/activate
```

4.3 Running the Jupyter Server

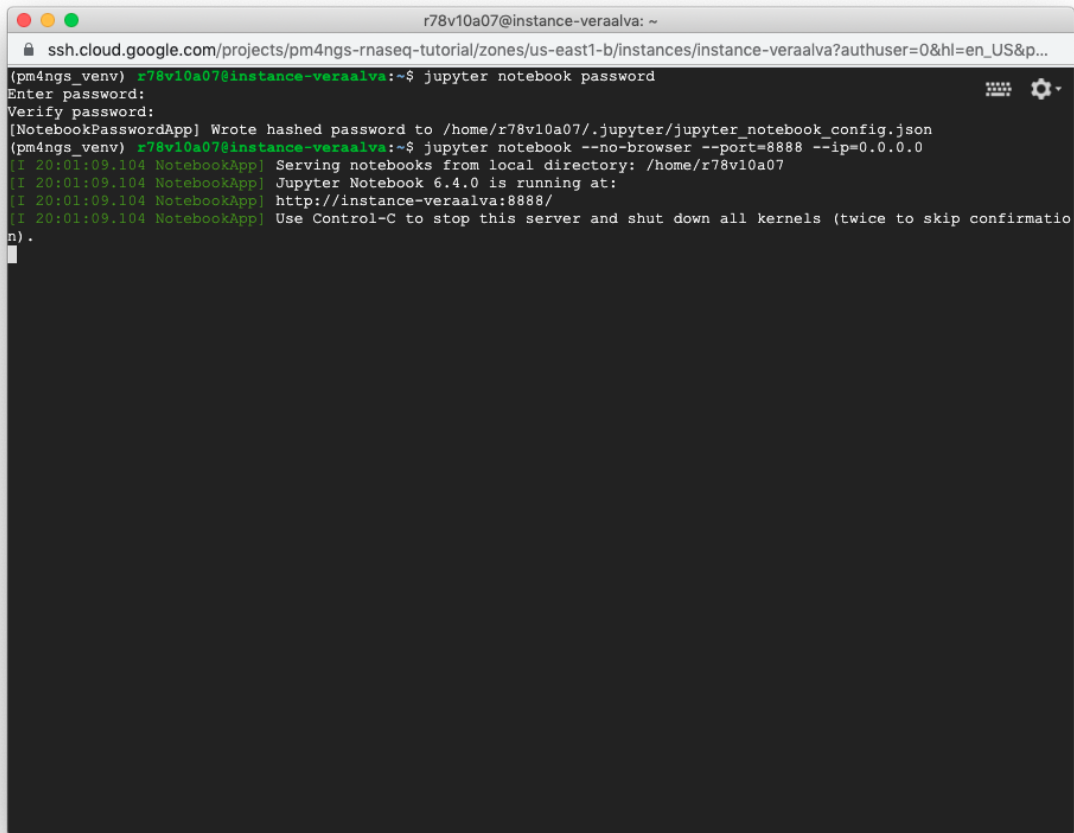
Open a terminal and activate the **pm4ngs_venv** virtual environment and run the jupyter server. As the GCP instance is a remote computer, we need to run the jupyter server with the **--port** and **--ip** options.

Creates a password for the Jupyter server:

```
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ jupyter notebook password
```

Start the Jupyter server

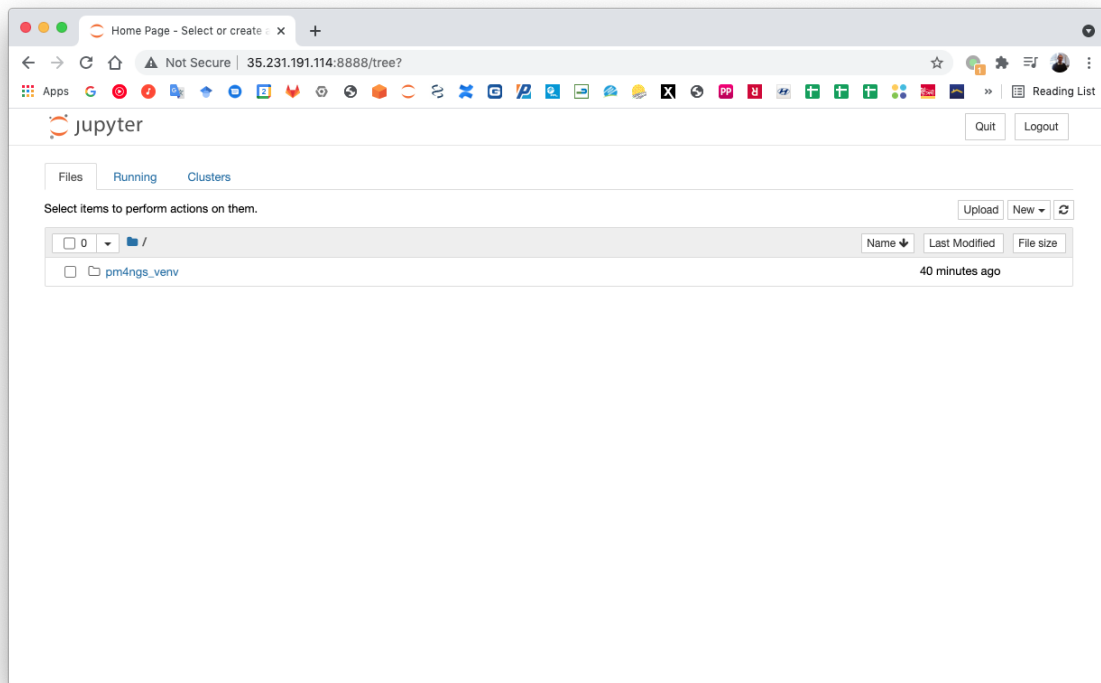
```
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ jupyter notebook --no-browser --port=8888 --  
↪ ip=0.0.0.0
```



```
r78v10a07@instance-veraalva: ~  
ssh.cloud.google.com/projects/pm4ngs-rnaseq-tutorial/zones/us-east1-b/instances/instance-veraalva?authuser=0&hl=en_US&p...  
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ jupyter notebook password  
Enter password:  
Verify password:  
[NotebookPasswordApp] Wrote hashed password to /home/r78v10a07/.jupyter/jupyter_notebook_config.json  
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ jupyter notebook --no-browser --port=8888 --ip=0.0.0.0  
[I 20:01:09.104 NotebookApp] Serving notebooks from local directory: /home/r78v10a07  
[I 20:01:09.104 NotebookApp] Jupyter Notebook 6.4.0 is running at:  
[I 20:01:09.104 NotebookApp] http://instance-veraalva:8888/  
[I 20:01:09.104 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmatio  
n).
```

4.4 Open the jupyter web interface in your local computer

Get the **External IP** for your instance in the GCp Cloud console: **VM instances**. Then, type the address in your local browser plus the jupyter server port: **:8888**



PM4NGS FOR RNA-SEQ DATA ANALYSIS

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	<i>Pre-processing QC</i>	down-load_quality_control.cwl	SRA-Tools	SRA accession	Fastq	fastq-dump.cwl
			FastQC	Fastq	FastQC HTML and Zip	fastqc.cwl
Trimming	<i>Samples trimming</i>		Trimmomatic	Fastq	Fastq	trimmomatic-PE.cwl trimmomatic-SE.cwl
Alignments and Quantification	<i>Alignment and Quantification</i>	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	<i>Differential Gene Expression Analysis</i>		DeSeq2	Read Matrix	TSV	deseq2-2conditions.cwl
			EdgeR	Read Matrix	TSV	edgeR-2conditions.cwl
Go enrichment	<i>Gene Ontology Enrichment</i>		goenrichment	gene IDs	TSV	Python code in the notebook

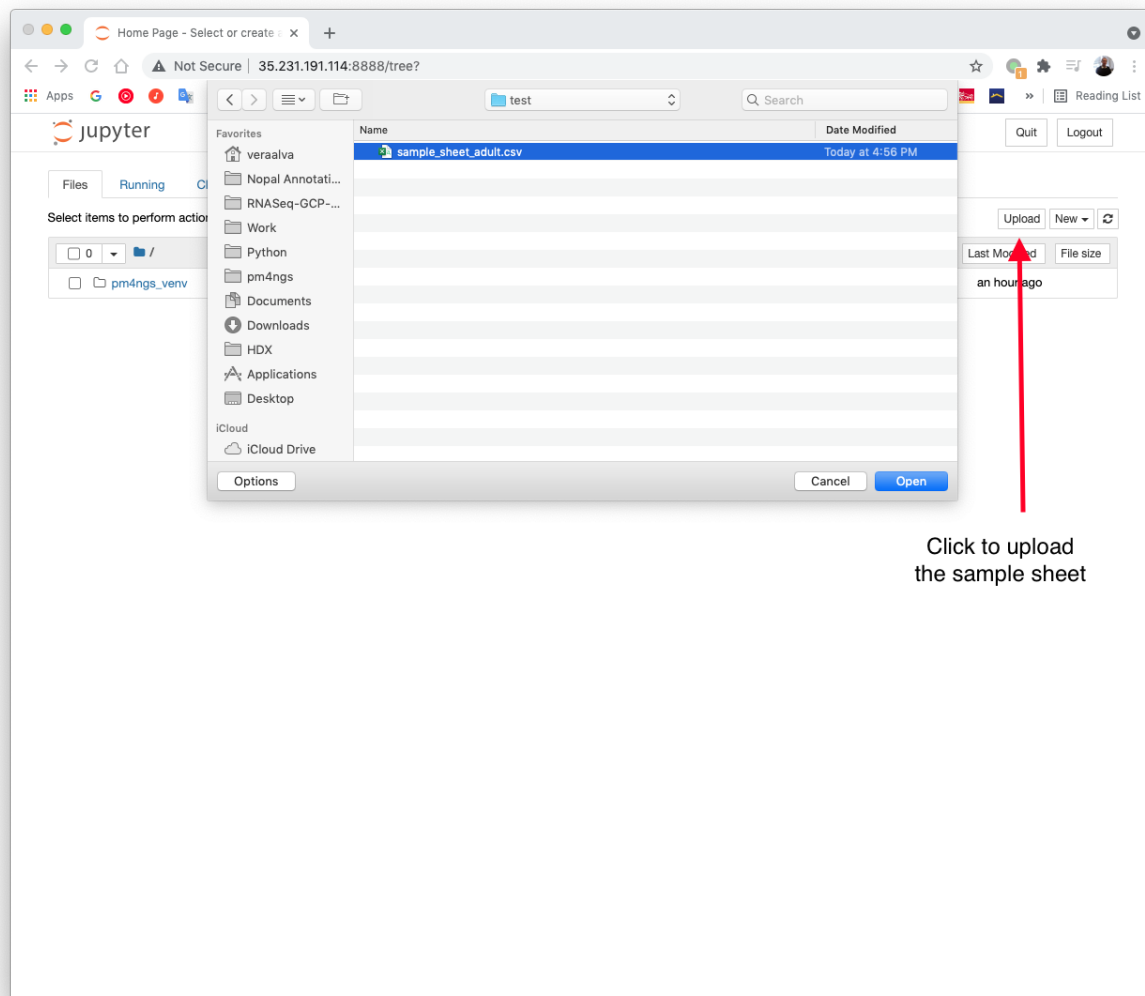
5.1 Sample sheet

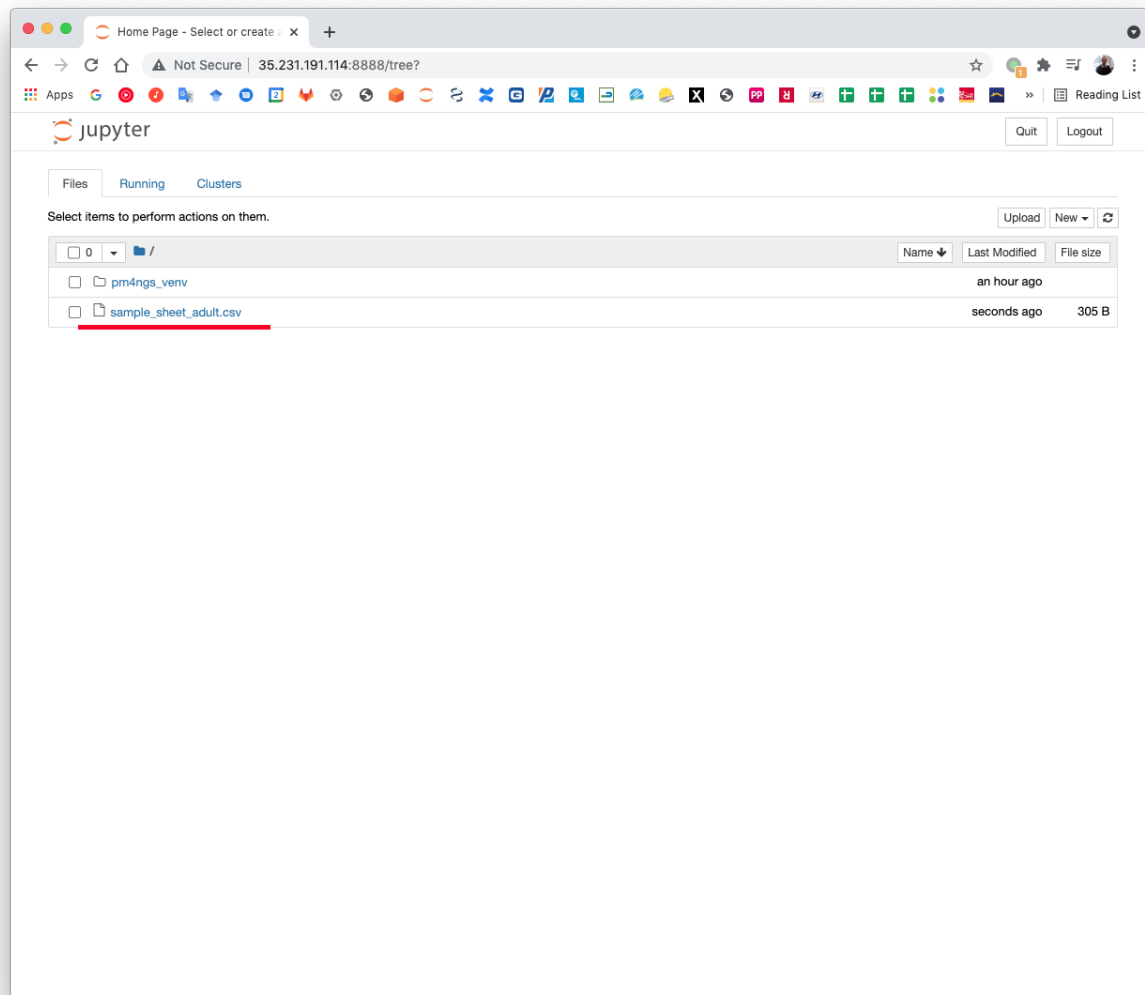
Sample Sheet for Drosophila BioProject [PRJDB5673](#)

sample_name	file	description	replicate	condition
DRR092341		Wild type, adult male head, replicate 1	1	WTAdult
DRR092342		Wild type, adult male head, replicate 2	2	WTAdult
DRR092343		Lol mutant, adult male head, replicate 1	1	Lo-Adult
DRR092344		Lol mutant, adult male head, replicate 2	2	Lo-Adult

Download the sample sheet file to your local computer: [sample_sheet_adult.csv](#)

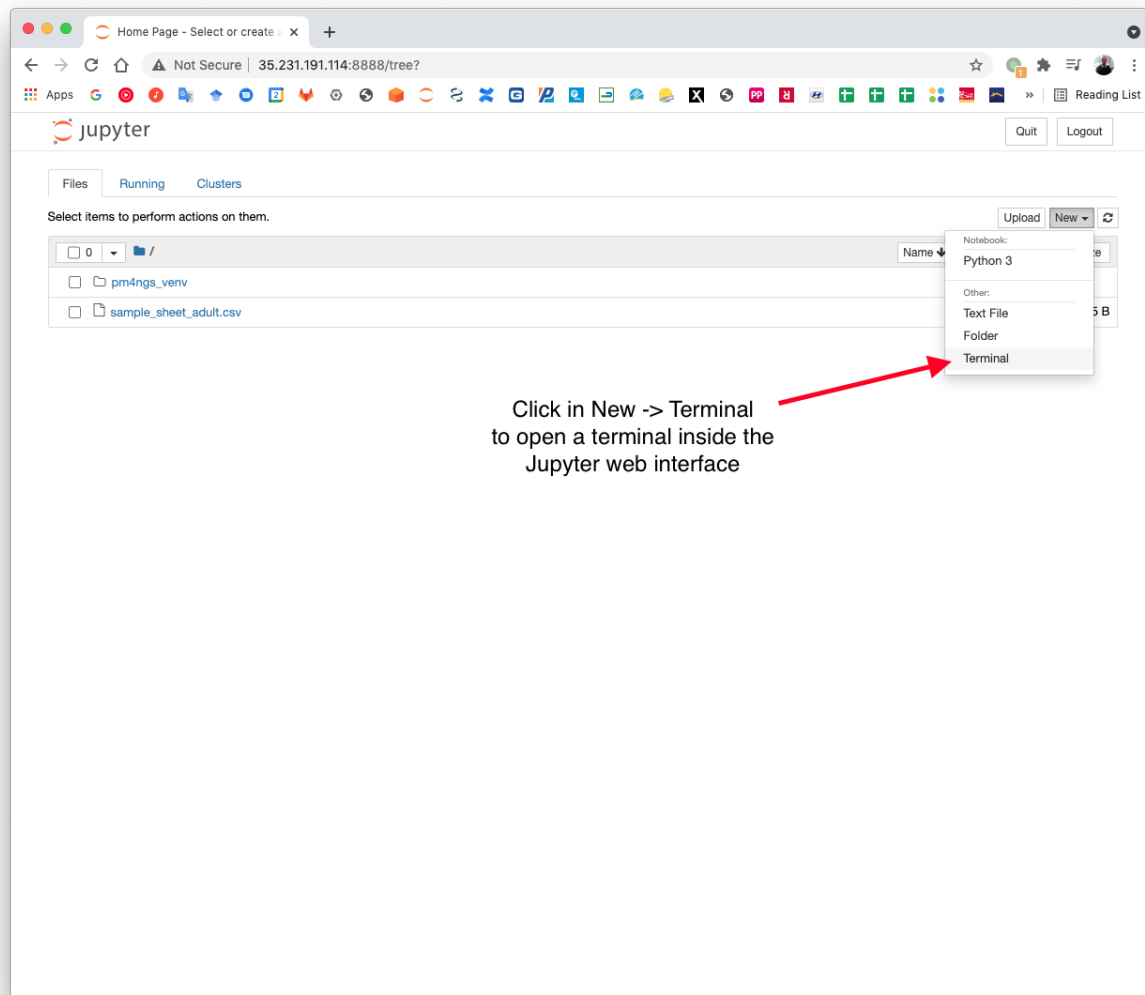
Upload the sample sheet file to the GCP instance through the Jupyter server





5.2 Creating the PM4NGS RNA-Seq project

Open a terminal to execute *pm4ngs-rnaseq*. This command will create an organizational data structure for the analysis.



In the terminal, activate the *pm4ngs_venv* virtual environment and execute the *pm4ngs-rnaseq* command.

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**.

```
r78v10a07@instance-veraalva:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ ls
pm4ngs_venv  sample_sheet_adult.csv
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ pm4ngs-rnaseq --sample-sheet sample_sheet_
↩adult.csv
```

Note:

- **author_name:** Use your full name
- **email:** Use your email
- **project_name:** Name of the project with no space nor especial characters. This will be used as project folder's

name.

Use: **Dros_lol_mut**

- **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}}**.

Use: **PRJDB5673**

- **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.

Use: **y**

- **Select sequencing technology:** Select one of the available sequencing technologies in your data.

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

Use: **1**

- **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.

Use: **n**

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

Press Enter

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

Use: **drosophila**

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

Use: **dm6**

- **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}}/**

Press Enter

- **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.

Press Enter

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

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

Press Enter

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

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

Press Enter

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

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

Press Enter

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

Press Enter

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

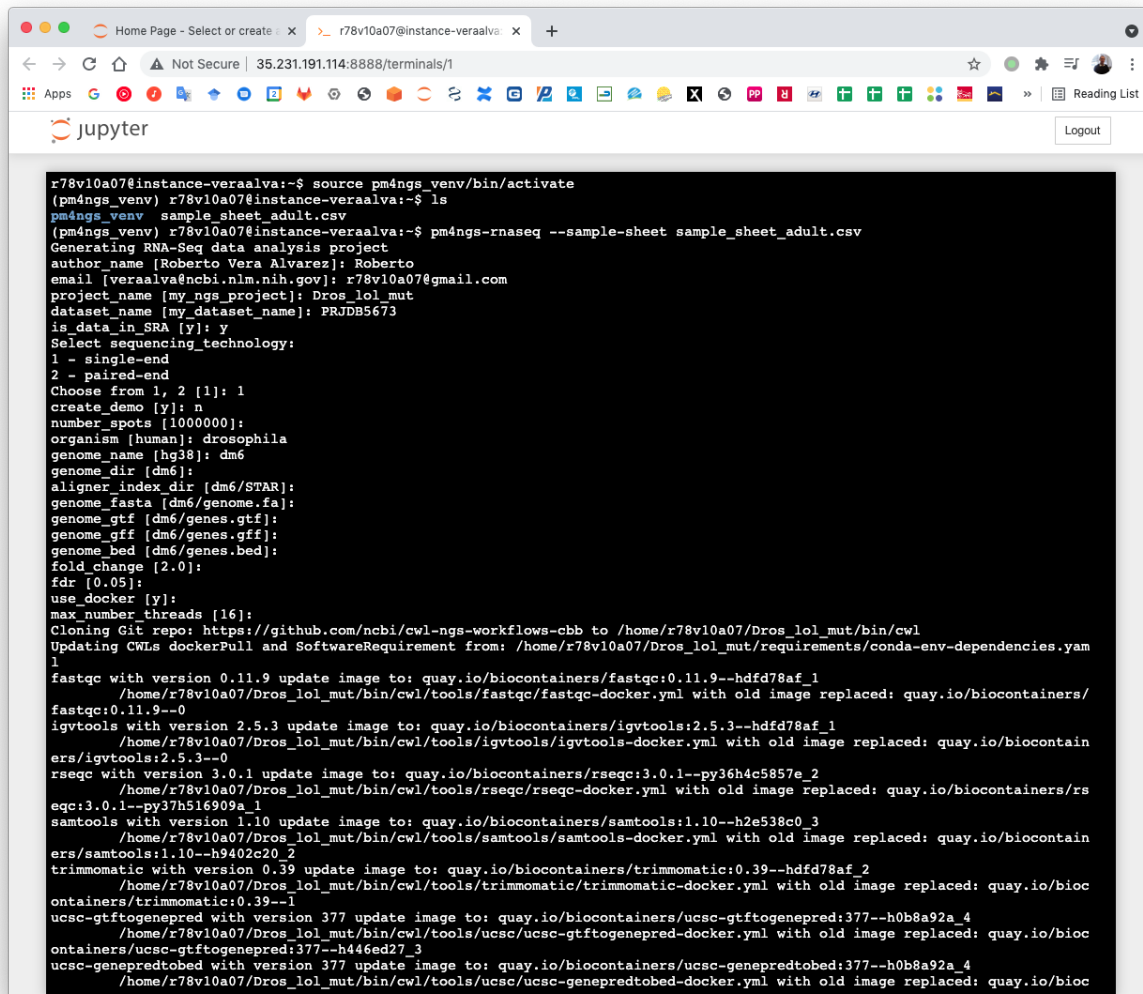
Press Enter

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

Press Enter

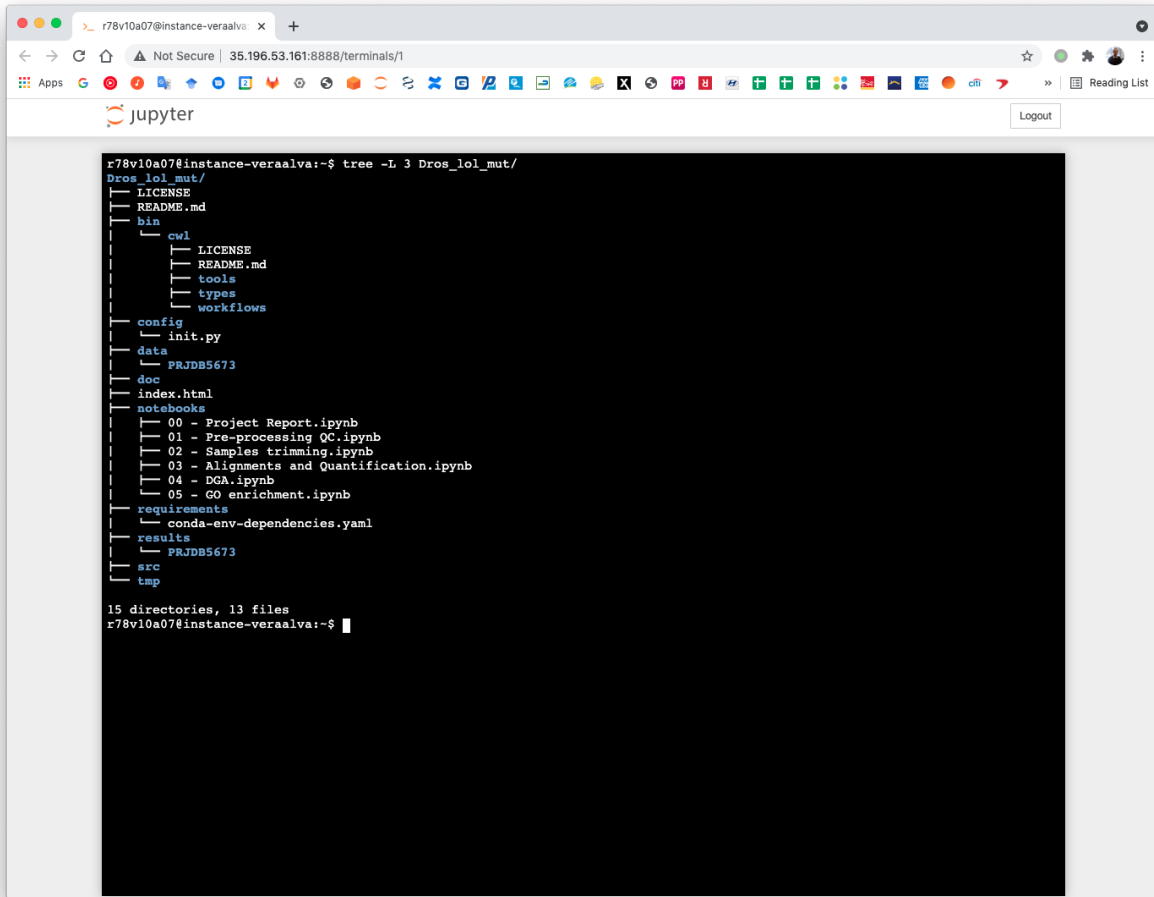
- **max_number_threads:** Number of threads available in the computer.

Press Enter



```
r78v10a07@instance-veeraalva:~$ source pm4ngs_venv/bin/activate
(pm4ngs_venv) r78v10a07@instance-veeraalva:~$ ls
pm4ngs_venv  sample_sheet_adult.csv
(pm4ngs_venv) r78v10a07@instance-veeraalva:~$ pm4ngs-rnaseq --sample-sheet sample_sheet_adult.csv
Generating RNA-Seq data analysis project
author_name [Roberto Vera Alvarez]: Roberto
email [veraalva@ncbi.nlm.nih.gov]: r78v10a07@gmail.com
project_name [my_ngo_project]: Dros_lol_mut
dataset_name [my_dataset_name]: PRJDB5673
is_data_in_SRA [y]: y
Select sequencing technology:
1 - single-end
2 - paired-end
Choose from 1, 2 [1]: 1
create_demo [y]: n
number_spots [1000000]:
organism [human]: drosophila
genome_name [hg38]: dm6
genome_dir [dm6]:
aligner_index_dir [dm6/STAR]:
genome_fasta [dm6/genome.fa]:
genome_gtf [dm6/genes.gtf]:
genome_gff [dm6/genes.gff]:
genome_bed [dm6/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/r78v10a07/Dros_lol_mut/bin/cwl
Updating CWLs dockerPull and SoftwareRequirement from: /home/r78v10a07/Dros_lol_mut/requirements/conda-env-dependencies.yaml
fastqc with version 0.11.9 update image to: quay.io/biocontainers/fastqc:0.11.9--hdfd78af_1
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/fastqc/fastqc-docker.yml with old image replaced: quay.io/biocontainers/
fastqc:0.11.9--0
igvtools with version 2.5.3 update image to: quay.io/biocontainers/igvtools:2.5.3--hdfd78af_1
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/igvtools/igvtools-docker.yml with old image replaced: quay.io/biocontain
ers/igvtools:2.5.3--0
rseqc with version 3.0.1 update image to: quay.io/biocontainers/rseqc:3.0.1--py36h4c5857e_2
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/rseqc/rseqc-docker.yml with old image replaced: quay.io/biocontainers/rs
eqc:3.0.1--py37h516909a_1
samtools with version 1.10 update image to: quay.io/biocontainers/samtools:1.10--h2e538c0_3
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/samtools/samtools-docker.yml with old image replaced: quay.io/bioc
ainers/samtools:1.10--h9402c20_2
trimmomatic with version 0.39 update image to: quay.io/biocontainers/trimmomatic:0.39--hdfd78af_2
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/trimmomatic/trimmomatic-docker.yml with old image replaced: quay.io/bioc
ontainers/trimmomatic:0.39--1
ucsc-gtftogepred with version 377 update image to: quay.io/biocontainers/ucsc-gtftogepred:377--h0b8a92a_4
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/ucsc/ucsc-gtftogepred-docker.yml with old image replaced: quay.io/bioc
ainers/ucsc-gtftogepred:377--h446ed27_3
ucsc-gtftogepred with version 377 update image to: quay.io/biocontainers/ucsc-gtftogepred:377--h0b8a92a_4
/home/r78v10a07/Dros_lol_mut/bin/cwl/tools/ucsc/ucsc-gtftogepred-docker.yml with old image replaced: quay.io/bioc
```

The project organizational data structure is:



The screenshot shows a JupyterLab interface with a terminal window open. The terminal displays the output of the command `tree -L 3 Dros_lol_mut/`. The directory structure is as follows:

```
Dros_lol_mut/
├── LICENSE
├── README.md
├── bin
│   └── cwl
│       ├── LICENSE
│       ├── README.md
│       ├── tools
│       ├── types
│       └── workflows
├── config
│   └── init.py
├── data
│   └── PRJDB5673
├── 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
│   └── PRJDB5673
├── src
└── tmp

15 directories, 13 files
r78v10a07@instance-veeraalva:~$
```

5.3 Pre-processing QC

The first notebook download the SRA data using the accessions defined in the sample sheet. Execute all cells until the **Retrieving data using fastq-dump**. This cell will submit the CWL workflow. Open a terminal to check that the **fastq-dump** command is working.


```

In [1]: !run ../config/init.py

Setting workdir to data/PRJDB5673

In [2]: data_dir = working_dir(os.path.join(DATA, DATASET))

Loading sample table file

The sample table file named: sample_table.csv file should be in the folder data/PRJDB5673.

The "sample_table.csv" file should have at least the following columns:
sample_name,file,condition,replicate
Columns:
Example:

    sample_name,file,condition,replicate
    SRR2126784,,PRE_NACT,1
    SRR2126785,,PRE_NACT,1
    SRR2126786,,PRE_NACT,1

In [3]: sample_table_file = os.path.join(DATA, DATASET, 'sample_table.csv')
sample_table = pandas.read_csv(sample_table_file, keep_default_na=False)
sample_table.head()

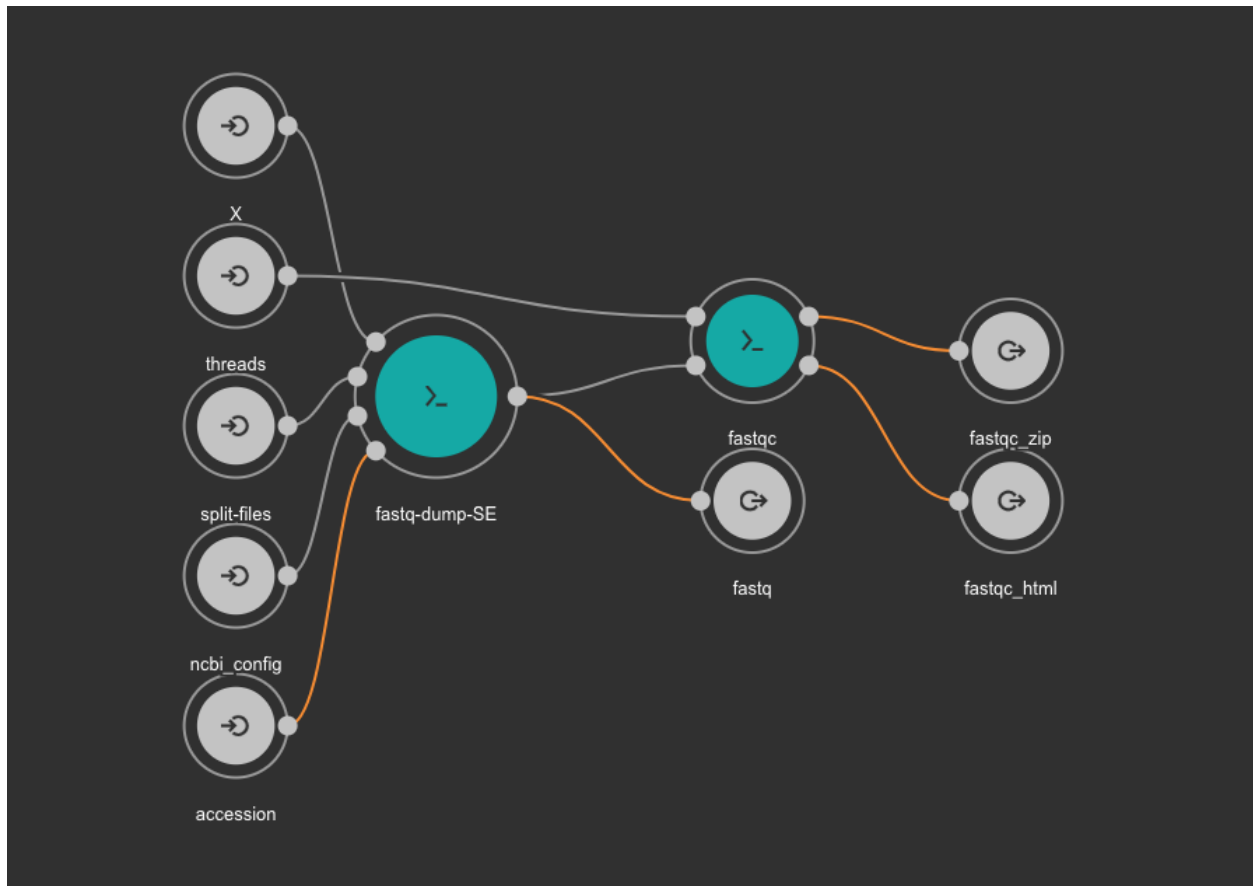
Out[3]:
  sample_name  file  description  replicate  condition
0  DRR092341      Wild type, adult male head, replicate 1      1  WTAdult
1  DRR092342      Wild type, adult male head, replicate 2      2  WTAdult
2  DRR092343  Loi mutant, adult male head, replicate 1      1  LoiAdult
3  DRR092344  Loi mutant, adult male head, replicate 2      2  LoiAdult

Retrieving data using fastq-dump

In [4]: log_file = 'download.log'

```

The CWL workflow for this step is: [download_quality_control.cwl](#)



Once all cells are execute completely the *fastq* samples will be available at the **data/PRJDB5673** directory. Run the **tree** command to visualize the data structure.

```
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ tree -L 3 Dros_lol_mut/
```

```

r78v10a07@instance-veraalva:~$ tree -L 3 Dros_lol_mut/
Dros_lol_mut/
├── LICENSE
├── README.md
├── bin
│   ├── cwl
│   │   ├── LICENSE
│   │   ├── README.md
│   │   ├── tools
│   │   ├── types
│   │   └── workflows
├── config
│   └── init.py
├── data
│   └── PRJDB5673
│       ├── DRR092341.fastq.gz
│       ├── DRR092341_fastqc.html
│       ├── DRR092341_fastqc.zip
│       ├── DRR092342.fastq.gz
│       ├── DRR092342_fastqc.html
│       ├── DRR092342_fastqc.zip
│       ├── DRR092343.fastq.gz
│       ├── DRR092343_fastqc.html
│       ├── DRR092343_fastqc.zip
│       ├── DRR092344.fastq.gz
│       ├── DRR092344_fastqc.html
│       ├── DRR092344_fastqc.zip
│       ├── download.log
│       └── 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
│   └── PRJDB5673
├── src
└── tmp

15 directories, 27 files
r78v10a07@instance-veraalva:~$

```

The pre-processing table is available in the *00 - Project Report* notebook. The links in the table gives access to the [FastQC](#) reports for each sample. The reports are used to select the trimming parameters.

Out[1]: The raw code for this IPython notebook is by default hidden for easier reading. To toggle on/off the raw code, click [here](#).

1. Pre-processing QC

Info

Type	Link
Notebook	01 - Pre-processing QC
Results	PRJDB5673

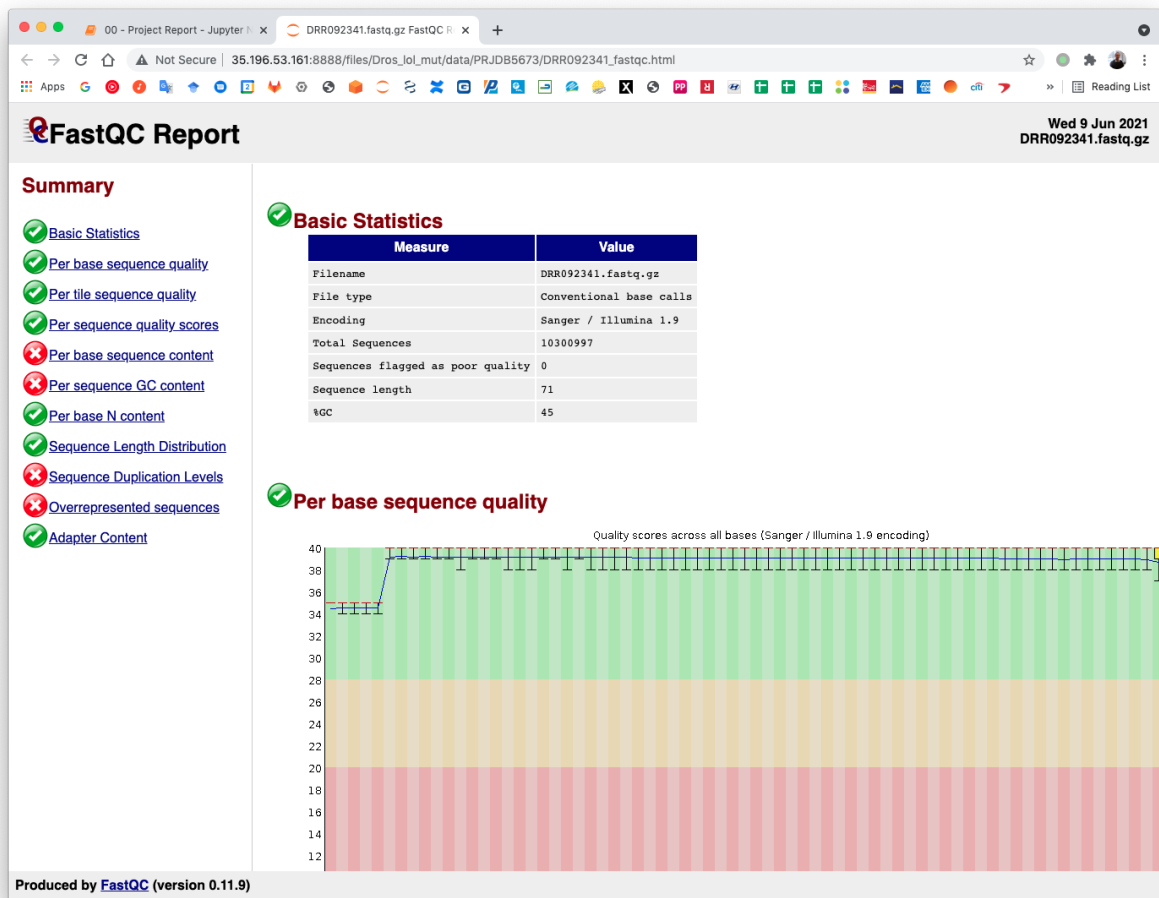
FastQC report

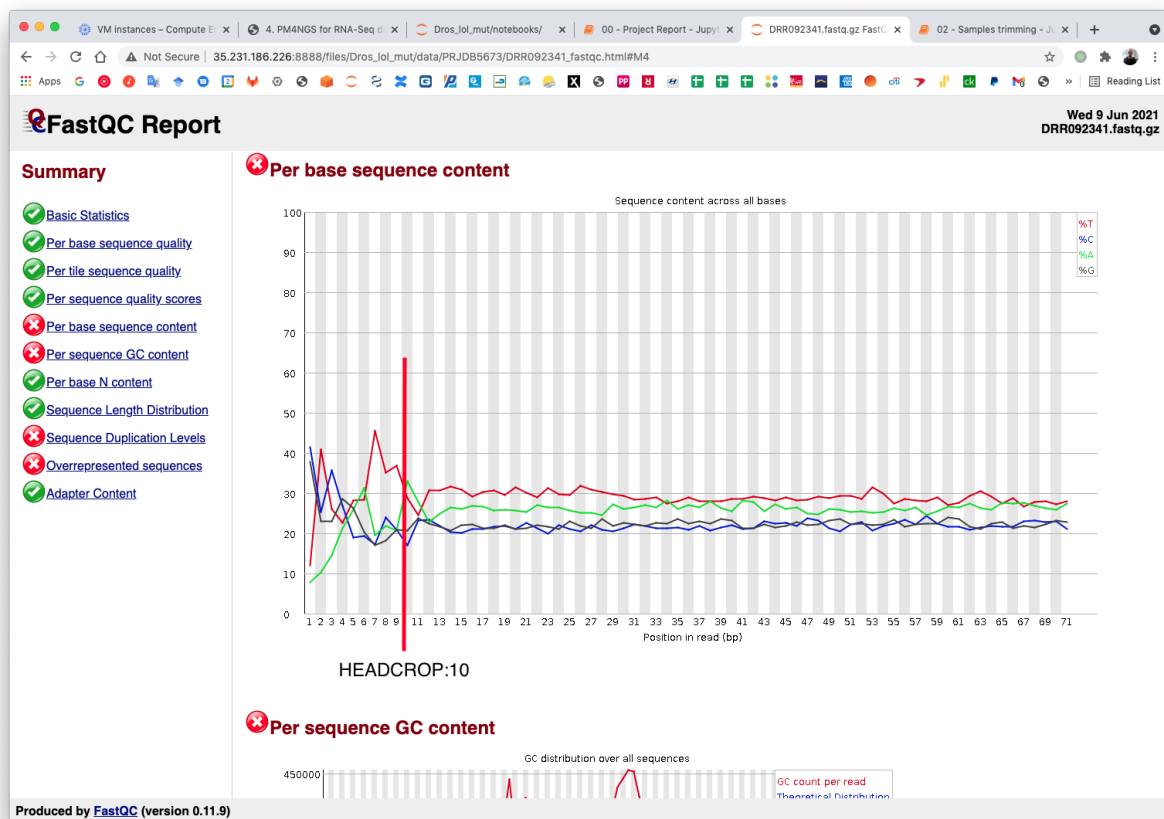
Sample	Fastq	FastQC Report	No of Reads in fastq	Seq Len	%GC	Poor Quality	Fail Tests
DRR092341	503.76MB	0.69MB	10,300,997	71	45	0	Per base sequence content Per sequence GC content Sequence Duplication Levels Overrepresented sequences
DRR092342	519.27MB	0.69MB	10,657,672	71	45	0	Per base sequence content Per sequence GC content Sequence Duplication Levels Overrepresented sequences
DRR092343	555.24MB	0.70MB	11,472,370	71	45	0	Per base sequence content Per sequence GC content Sequence Duplication Levels Overrepresented sequences
DRR092344	538.83MB	0.70MB	11,136,350	71	44	0	Per base sequence content Per sequence GC content Sequence Duplication Levels Overrepresented sequences

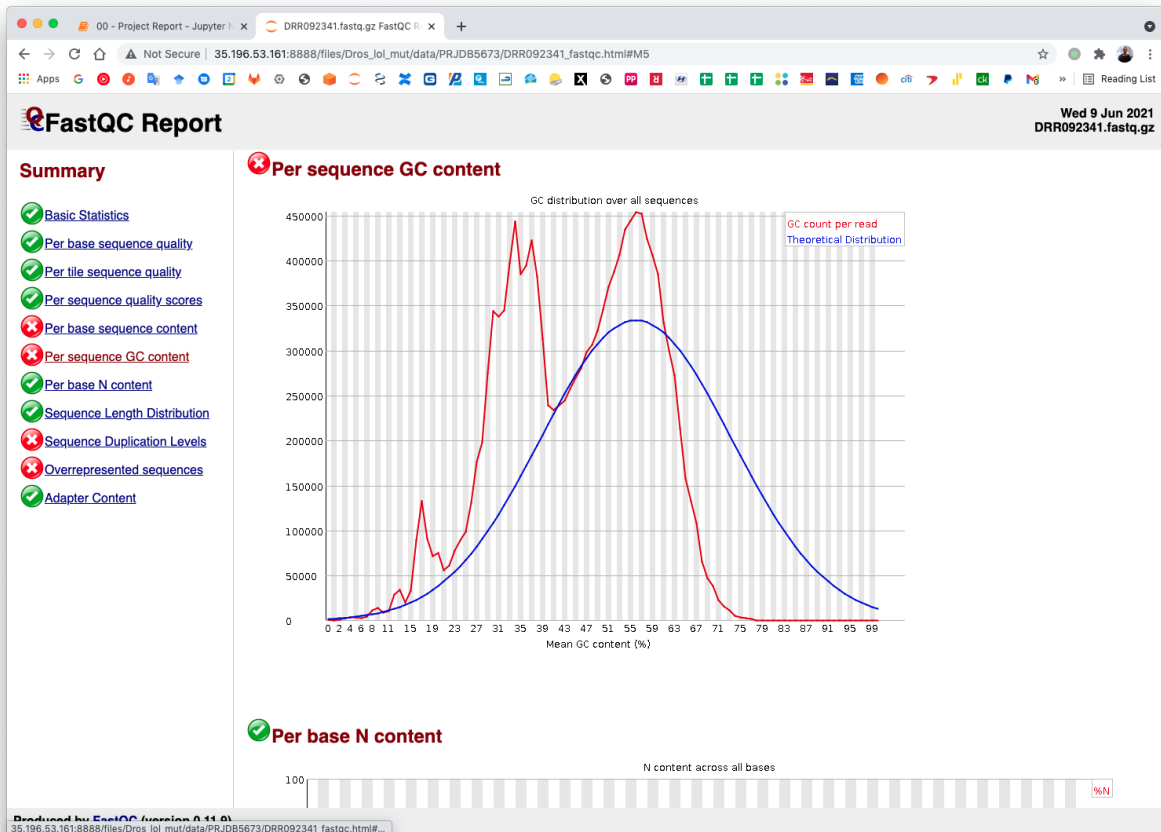
2. Trimming

3. Alignments and Quantification

3.1. Alignment-QC







5.4 Samples trimming

Trimmomatic performs a variety of useful trimming tasks for illumina paired-end and single ended data. The selection of trimming steps and their associated parameters are supplied on the command line.

The current trimming steps are:

- **ILLUMINACLIP:** Cut adapter and other illumina-specific sequences from the read.
- **SLIDINGWINDOW:** Perform a sliding window trimming, cutting once the average quality within the window falls below a threshold.
- **LEADING:** Cut bases off the start of a read, if below a threshold quality
- **TRAILING:** Cut bases off the end of a read, if below a threshold quality
- **CROP:** Cut the read to a specified length
- **HEADCROP:** Cut the specified number of bases from the start of the read
- **MINLEN:** Drop the read if it is below a specified length
- **TOPHRED33:** Convert quality scores to Phred-33
- **TOPHRED64:** Convert quality scores to Phred-64

It works with FASTQ (using phred + 33 or phred + 64 quality scores, depending on the Illumina pipeline used), either uncompressed or gzipped FASTQ. Use of gzip format is determined based on the .gz extension.

PM4NGS uses standard options for Trimmomatic but in this example we need to add **HEADCROP:10** as it is shown in next figure:

```

In [3]: log_file = 'trimming.log'
# Edit these values accordingly with the FastQC report and Trimmomatic path for adapters
HEADCROP = 10
MINLEN = 25
AVGQUAL = 30
LEADING = 30
TRAILING = 30

TRIMMOMATIC_ADAPTER = 'TruSeq3-SE.fa:2:30:10'

trimming_yaml = {
    'threads': 2,
    'illuminaClip': os.path.join(TRIMMOMATIC_ADAPTERS, TRIMMOMATIC_ADAPTER),
    'headcrop': 10,
    'minlen': MINLEN,
    'avgqual': AVGQUAL,
    'leading': LEADING,
    'trailing': TRAILING,
    'input_files': []
}

for i, r in sample_table.iterrows():
    f = os.path.join(DATA, DATASET, r['file'])
    if not os.path.exists(r['file']):
        trimming_yaml['input_files'].append({'class': 'File', 'path': f})

if trimming_yaml['input_files']:
    write_to_yaml(trimming_yaml, 'trimming.yaml')

    cmd_header = '{} {} /pre-processing/trimming-qc-se.cwl trimming.yaml > {} 2>&1 &'.format(
        CWLRUNNER, CWLWORKFLOWS, log_file)

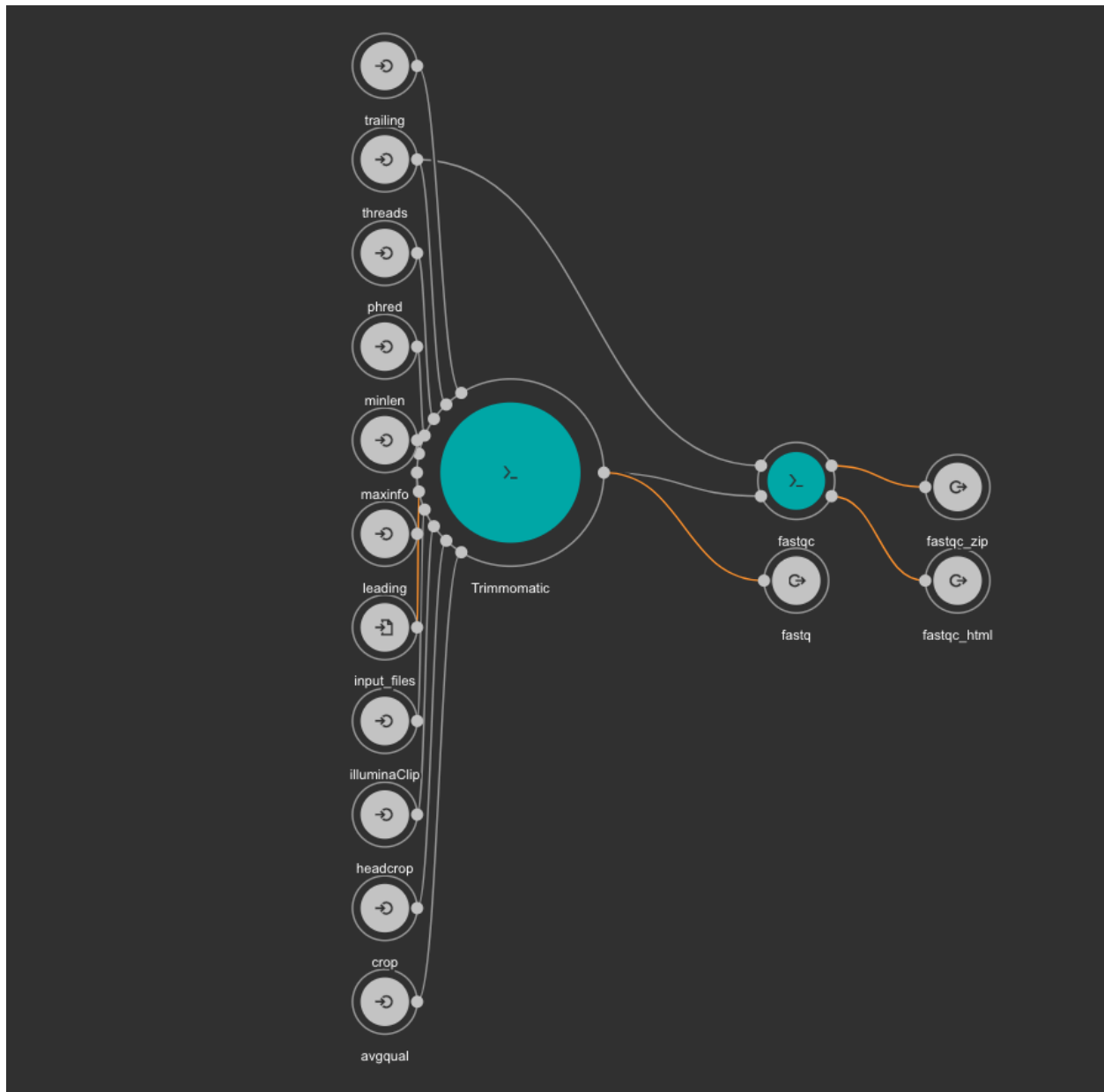
    run_command(cmd_header)

Executing: /home/r78v10a07/pm4ngs_env/bin/cwltool --no-read-only --beta-use-biocontainers --parallel --on-error cont
inue --rm-tmpdir --tmp-outdir-prefix=/home/r78v10a07/Dros_lol_mut/tmp/ --tmpdir-prefix=/home/r78v10a07/Dros_lol_mut/t
mp/ /home/r78v10a07/Dros_lol_mut/bin/cwl/workflows/pre-processing/trimming-qc-se.cwl trimming.yaml > trimming.log 2>&
1 &
0

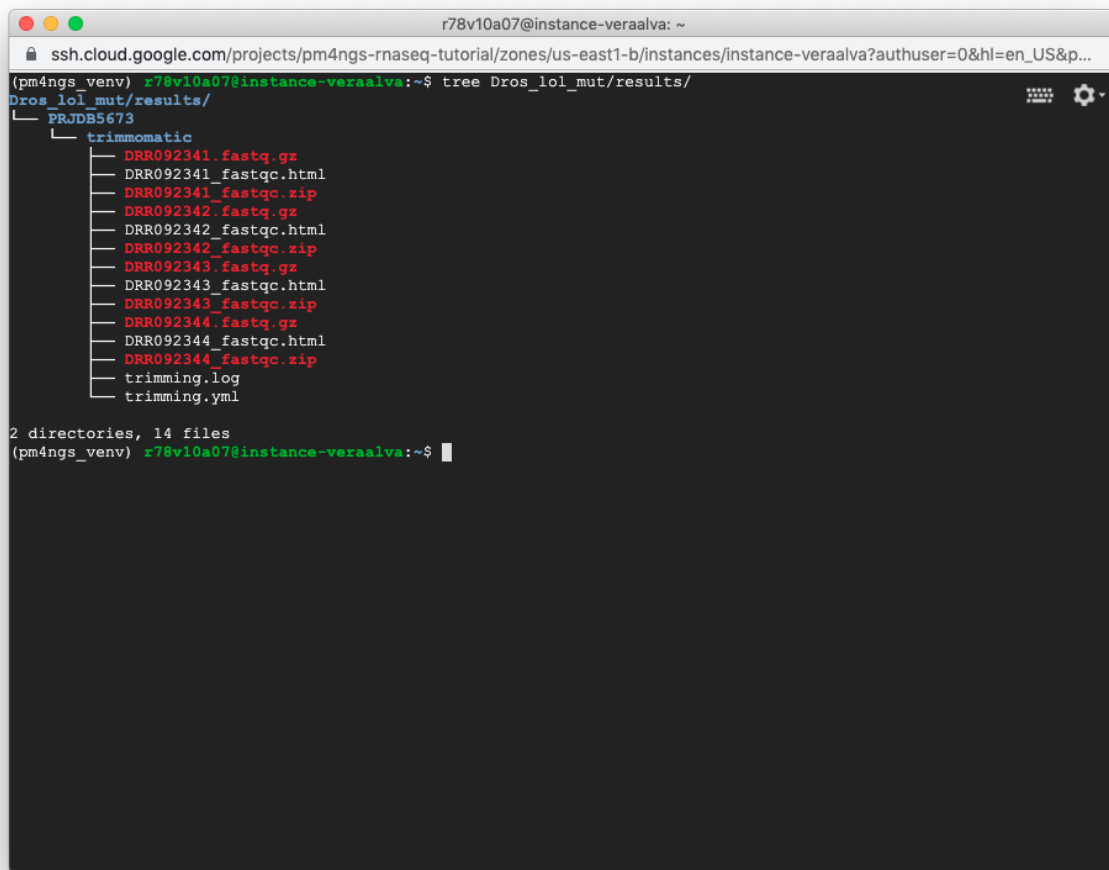
```

Checking command output
Execute next cell until it prints: Run completed

The CWL workflow for this step is: [trimming-qc-se.cwl](#)



The result files for the trimming workflow are available at: [results/PRJDB5673/trimmomatic](https://results.prjdb5673/trimmomatic)



```
r78v10a07@instance-veraalva: ~  
ssh.cloud.google.com/projects/pm4ngs-rnaseq-tutorial/zones/us-east1-b/instances/instance-veraalva?authuser=0&hl=en_US&p...  
(pm4ngs_venv) r78v10a07@instance-veraalva:~$ tree Dros_lol_mut/results/  
Dros_lol_mut/results/  
├── PRJDB5673  
│   └── trimmomatic  
│       ├── DRR092341.fastq.gz  
│       ├── DRR092341_fastqc.html  
│       ├── DRR092341_fastqc.zip  
│       ├── DRR092342.fastq.gz  
│       ├── DRR092342_fastqc.html  
│       ├── DRR092342_fastqc.zip  
│       ├── DRR092343.fastq.gz  
│       ├── DRR092343_fastqc.html  
│       ├── DRR092343_fastqc.zip  
│       ├── DRR092344.fastq.gz  
│       ├── DRR092344_fastqc.html  
│       ├── DRR092344_fastqc.zip  
│       ├── trimming.log  
│       └── trimming.yml  
2 directories, 14 files  
(pm4ngs_venv) r78v10a07@instance-veraalva:~$
```

Updating the 00 - *Project Report* notebook.

2. Trimming

Info

Type	Link
Notebook	02 - Samples trimming
Results	trimmomatic

Sample	Trimmed Fastq	FastQC Report	No of Reads in fastq	Removed Reads	Seq Len	%GC	Poor Quality	Fail Tests
DRR092341	447.88MB	659.27KB	10,145,750	155,247	25-61	44	0	Per sequence GC content Sequence Duplication Levels Overrepresented sequences
DRR092342	462.24MB	658.15KB	10,502,697	154,975	25-61	44	0	Per sequence GC content Sequence Duplication Levels Overrepresented sequences
DRR092343	483.89MB	658.93KB	11,301,311	171,059	25-61	45	0	Per sequence GC content Sequence Duplication Levels Overrepresented sequences
DRR092344	479.72MB	660.19KB	10,973,790	162,560	25-61	44	0	Per sequence GC content Sequence Duplication Levels Overrepresented sequences

3. Alignments and Quantification

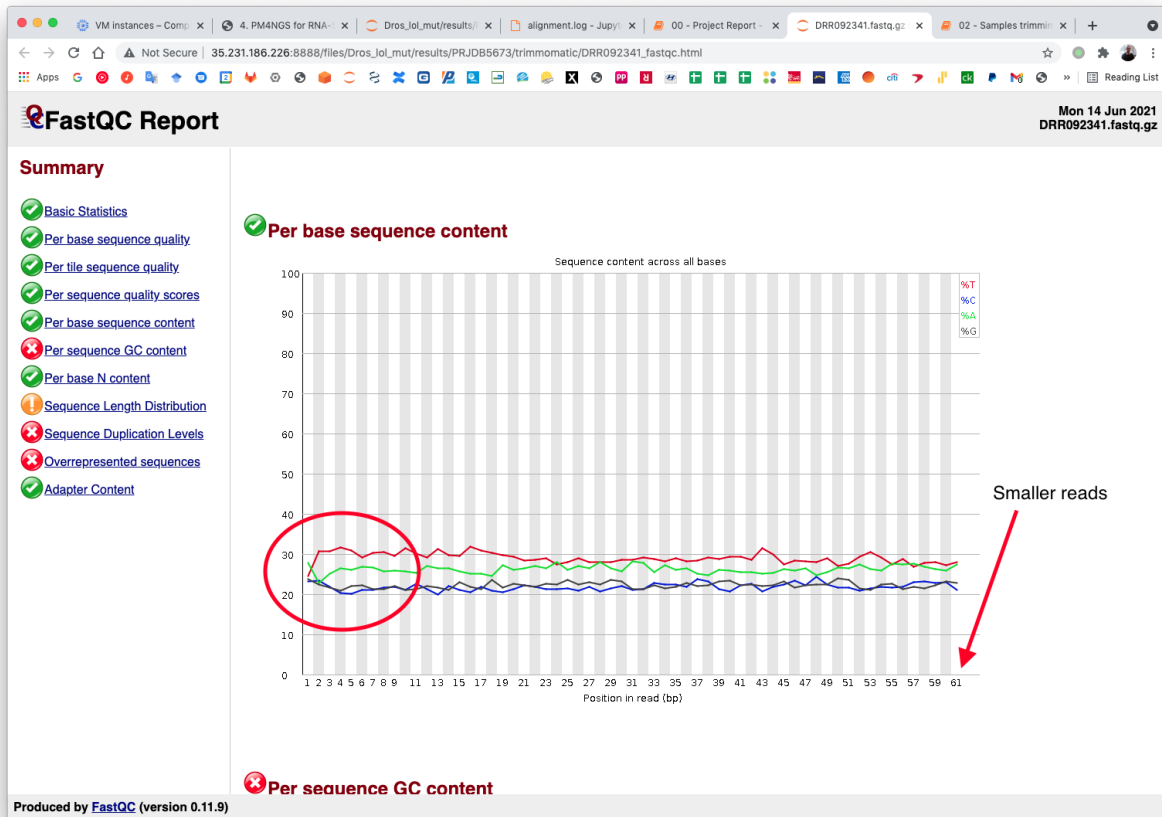
3.1. Alignment-QC

3.2. Quantification

4. Differential Gene Expression Analysis

5. GO enrichment

Check the FastQC reports to check if the trimming reduced the distortion in the first 10 bases.



5.5 Alignment and Quantification

The alignment and quantification workflow uses [STAR](#) as aligner, [Samtools](#) for filtering and sorting BAM files, [TPM-Calculator](#) for RNA-Seq quantification, [RSeQC](#) for post processing quality control and [IGVtools](#) for creating visualization files.

In this tutorial we are analysing *Drosophila* samples. Therefore, we need the *Drosophila* genome sequence and annotations. PM4NGS provides the **dm6** genome pre-formatted for the alignment and quantification workflow. For a complete list of PM4NGS pre-formatted genomes see: <https://pm4ngs.readthedocs.io/en/latest/pipelines/genomes.html>

Creating Genome indexes if they don't exists

```
In [3]: if not os.path.exists(GENOME):
        working_dir(GENOME)
        !curl -o {GENOME_NAME}.tar.gz https://ftp.ncbi.nlm.nih.gov/pub/pm4ngs/resources/{GENOME_NAME}.tar.gz
        !tar xzfv {GENOME_NAME}.tar.gz --strip 1
        !rm {GENOME_NAME}.tar.gz
        working_dir(ALIGNER_INDEX)
        log_file = 'star_index.log'
        index_cmd = '{0} {1}/star/star-index.cwl --runThreadN {2} --genomeDir .'.format(CWLRUNNER, CWLTOOLS, 16)
        index_cmd += '--genomeFastaFiles ../genome.fa --sjdbGTFfile ../genes.gtf > index.log 2>&1 &'
        run_command(index_cmd)

        % Total      % Received % Xferd  Average Speed   Time    Time     Time  Current
                               Dload  Upload    Total   Spent    Left   Speed

100 47.8M  100 47.8M    0     0  44.9M      0  0:00:01  0:00:01 --:--:--  44.9M
dm6/dm6-blacklist.bed
dm6/dm6.chrom.sizes
dm6/genes.bed
dm6/genes.genePred
dm6/genes.gtf
dm6/genome.fa
dm6/refGene.txt
Executing: /home/r78v10a07/pm4ngs_venv/bin/cwltool --no-read-only --beta-use-biocontainers --parallel --on-error cont
inue --rm-tmpdir --tmp-outdir-prefix=/home/r78v10a07/Dros_lol_mut/tmp/ --tmpdir-prefix=/home/r78v10a07/Dros_lol_mut/t
mp/ /home/r78v10a07/Dros_lol_mut/bin/cwl/tools/star/star-index.cwl --runThreadN 16 --genomeDir . --genomeFastaFiles
../genome.fa --sjdbGTFfile ../genes.gtf > index.log 2>&1 &
0
```

```
In [5]: if os.path.exists('index.log'):
        check_cwl_command_log('index.log')

Process completed.
Proceed to next cells
```

This workflow is the most time consuming part and require setting proper computer resources like number of cores and RAM. For this tutorial we need at least 64 GB of RAM and 16 cores. GCP machine type *n1-standard-16* provides those resources.

Processing samples

```
In [6]: working_dir(result_dir)
        log_file = 'alignment.log'

        alignment_yaml = {
            'ramMaxRSeQC': 16000,
            'ramMaxSTAR': 16000,

            'threads': 16,
            'genomeDir': {'class': 'Directory', 'path': ALIGNER_INDEX },
            'genome_bed': {'class': 'File', 'path': GENOME_BED },
            'genome_gtf': {'class': 'File', 'path': GENOME_GTF },
            'q': 255,
            'reads': []
        }

        for i, r in sample_table.iterrows():
            f = os.path.join(data_dir, r['file'])
            if not os.path.exists(r['file'].replace('.fastq.gz', '_genes.out.gz')):
                alignment_yaml['reads'].append({
                    'class': 'File', 'path': f})

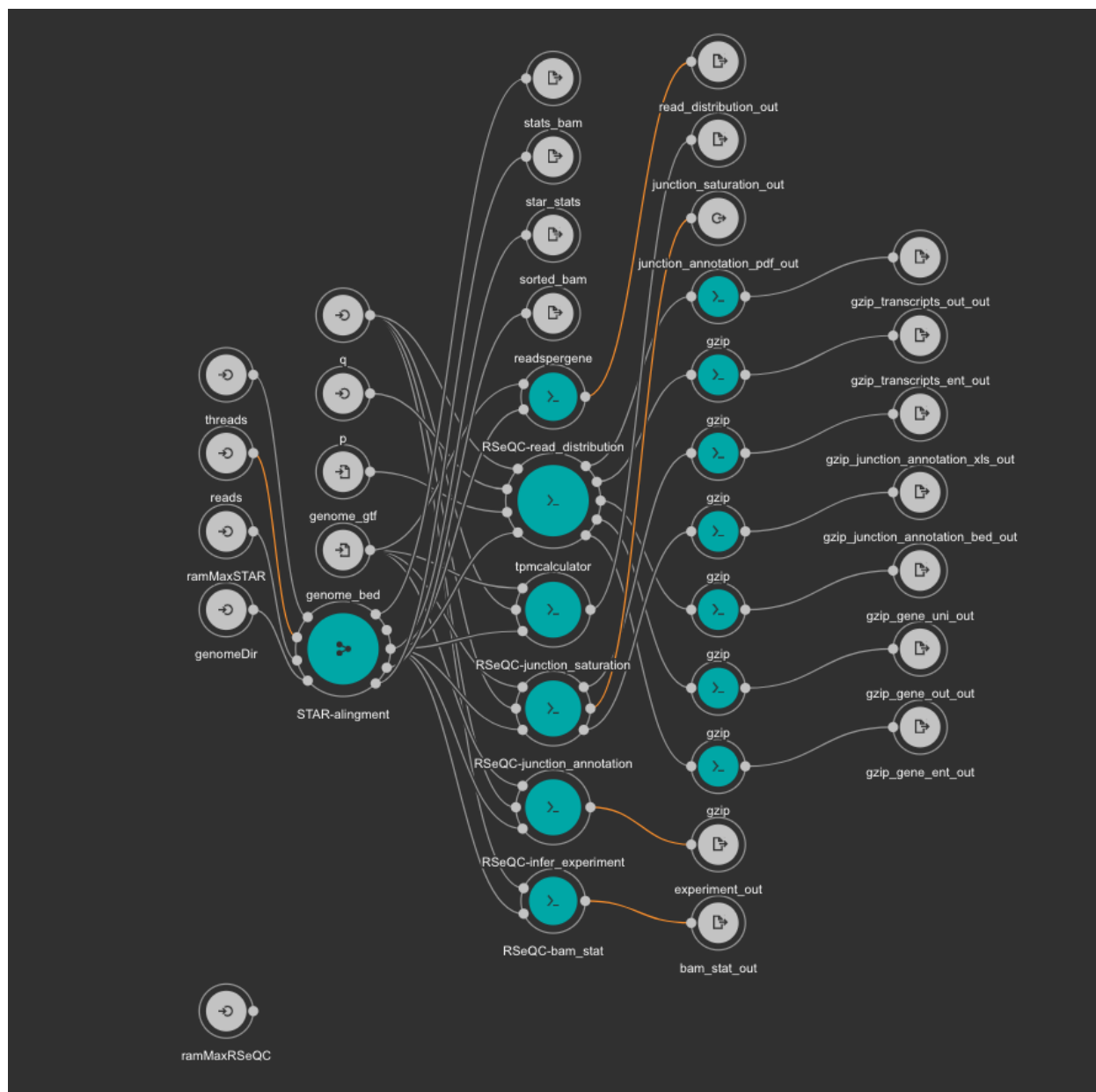
        if alignment_yaml['reads']:
            write_to_yaml(alignment_yaml, 'alignment.yaml')
            cmd_header = '{0} {1}/RNA-Seq/rnaseq-alignment-quantification.cwl alignment.yaml > {2} 2>&1 &'.format(
                CWLRUNNER, CWLWORKFLOWS, log_file)
            run_command(cmd_header)
```

Adjust to machine RAM
depending on the genome.
Use 32 000 for human

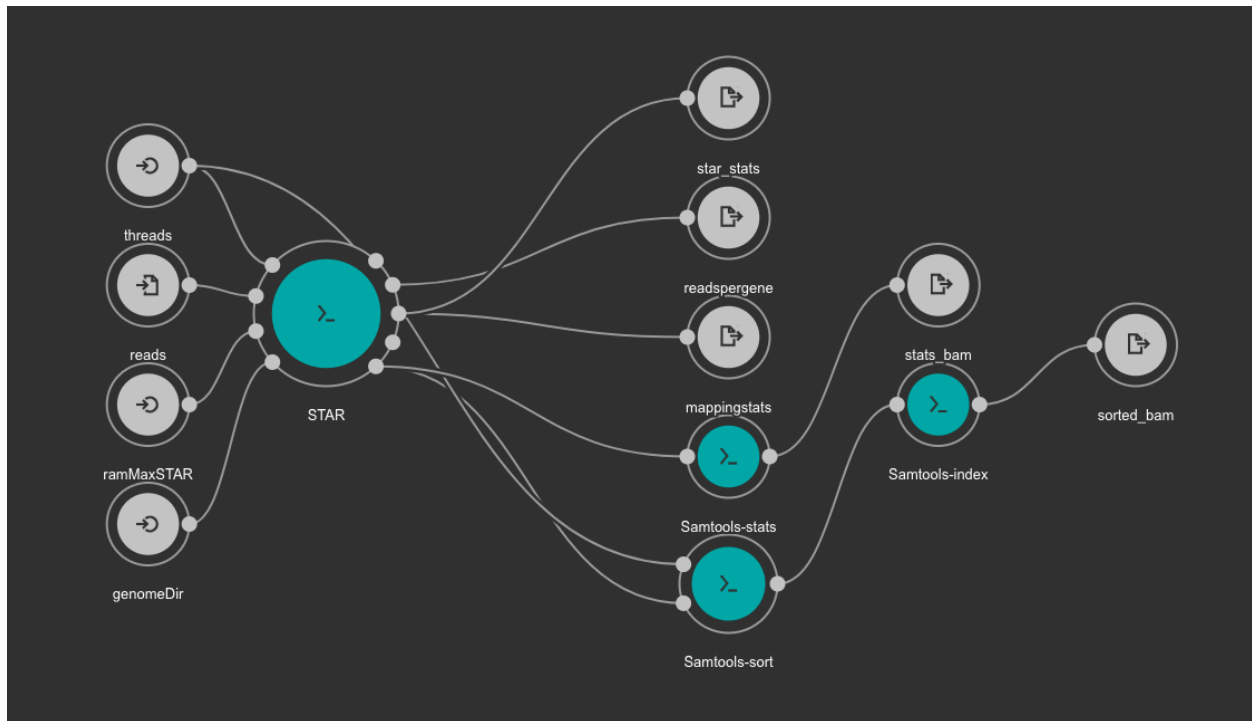
Adjust to number of cores

```
Executing: /home/r78v10a07/pm4ngs_venv/bin/cwltool --no-read-only --beta-use-biocontainers --parallel --on-error cont
inue --rm-tmpdir --tmp-outdir-prefix=/home/r78v10a07/Dros_lol_mut/tmp/ --tmpdir-prefix=/home/r78v10a07/Dros_lol_mut/t
mp/ /home/r78v10a07/Dros_lol_mut/bin/cwl/workflows/RNA-Seq/rnaseq-alignment-quantification.cwl alignment.yaml > align
ment.log 2>&1 &
0
```

The CWL workflow for this step is: [rnaseq-alignment-quantification.cwl](#)



It is based in another workflow for the alignment: [star-alignment.cwl](#)



Once finished, the alignment and quantification workflow will create for each sample these files:

Note: STAR alignment stats

- DRR092341Aligned.out.stats
- DRR092341Log.final.out
- DRR092341ReadsPerGene.out.tab

Alignments files sorted from SAMtools

- DRR092341_sorted.bam
- DRR092341_sorted.bam.bai

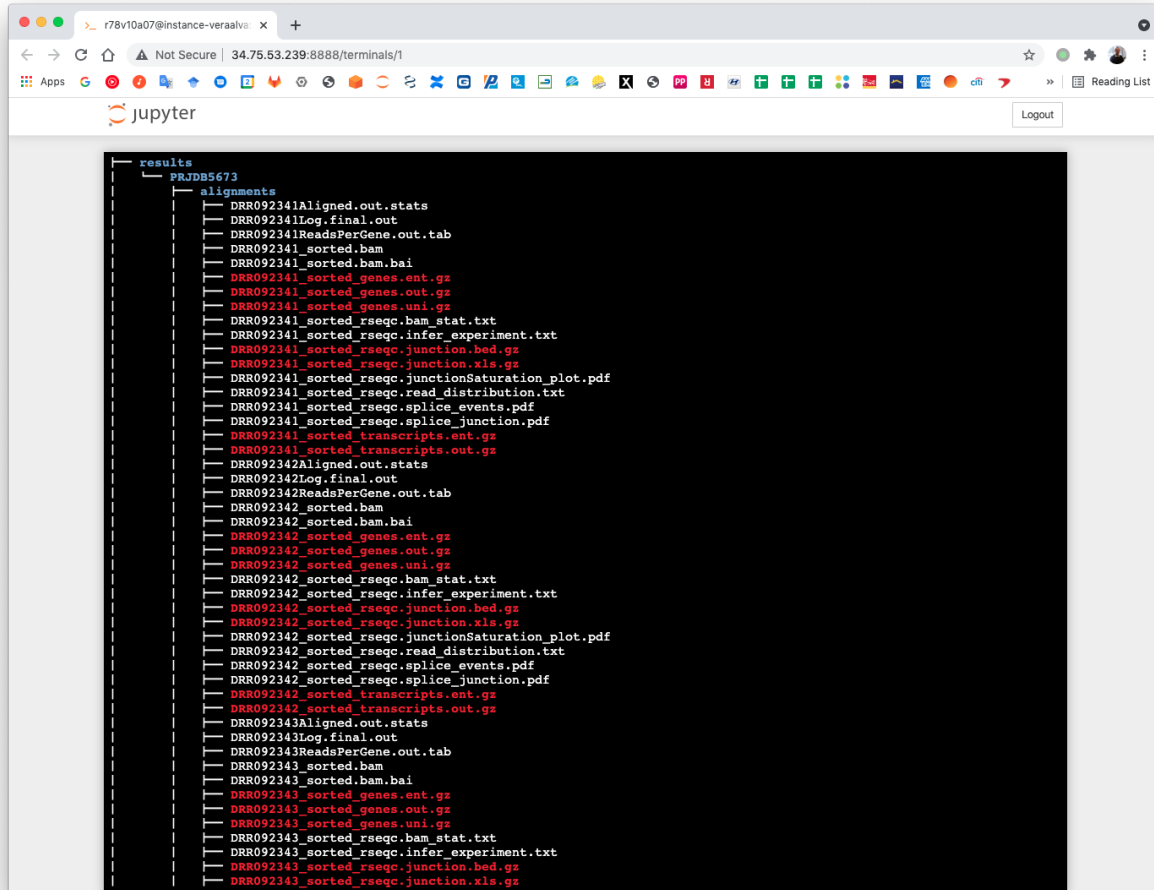
Read counts and TPMs from TPMCalculator

- DRR092341_sorted_genes.ent.gz
- DRR092341_sorted_genes.out.gz
- DRR092341_sorted_genes.uni.gz
- DRR092341_sorted_transcripts.ent.gz
- DRR092341_sorted_transcripts.out.gz

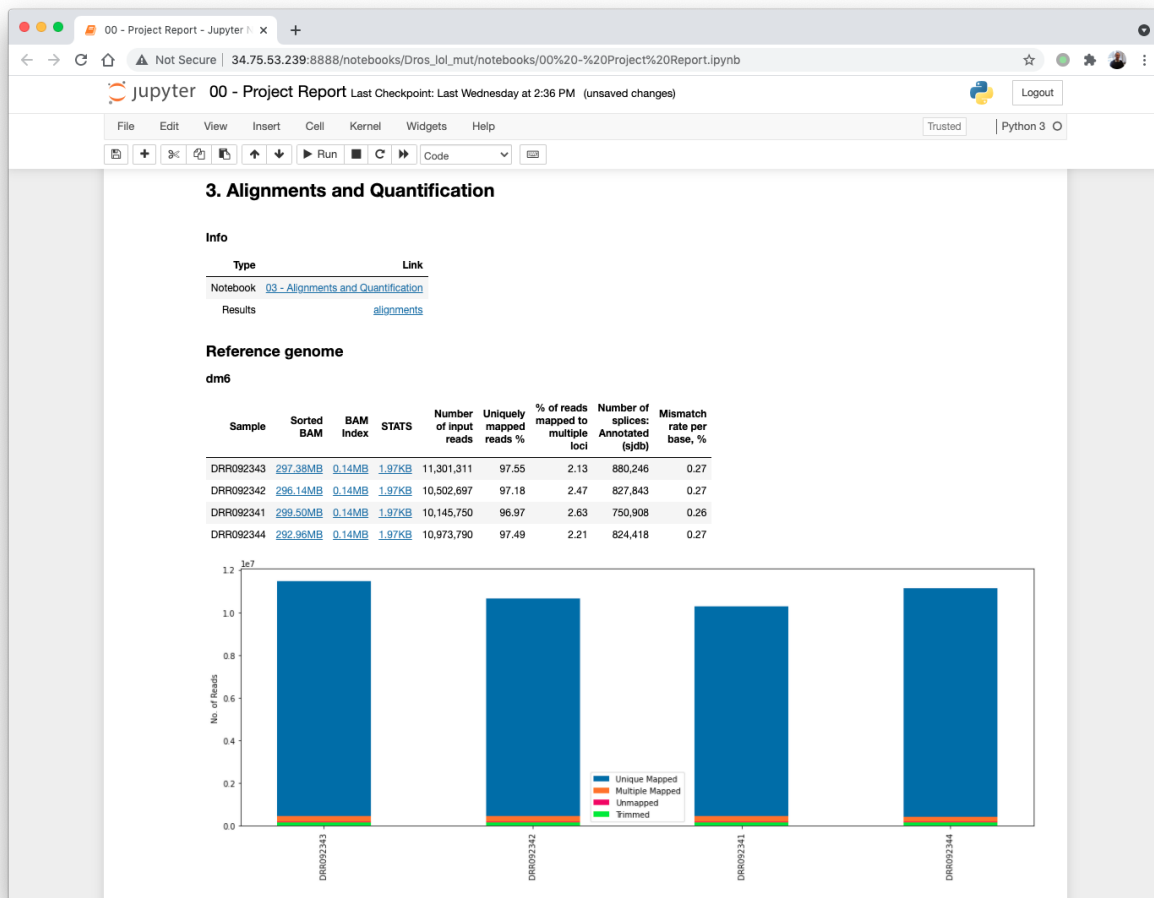
** Post-processing QC from RSeQC**

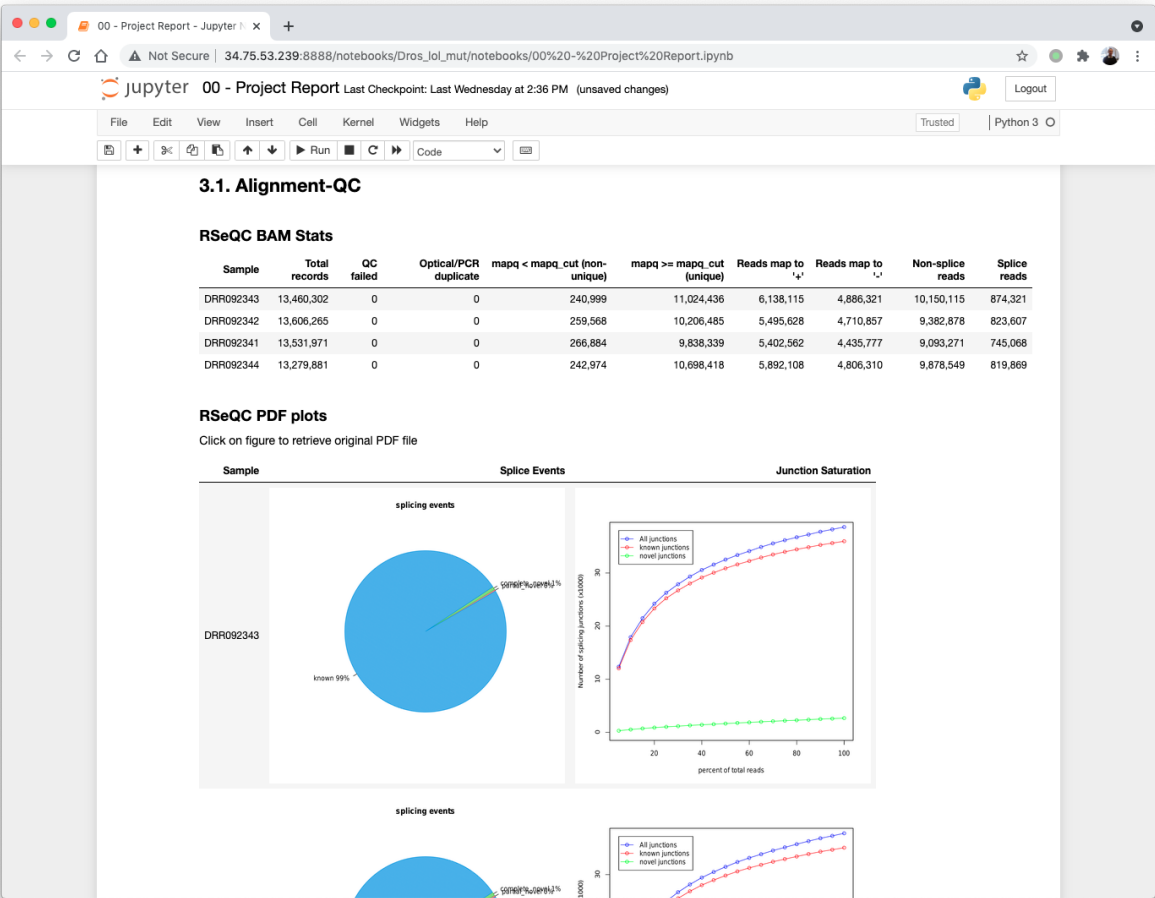
- DRR092341_sorted_rseqc.bam_stat.txt
- DRR092341_sorted_rseqc.infer_experiment.txt
- DRR092341_sorted_rseqc.junction.bed.gz
- DRR092341_sorted_rseqc.junction.xls.gz
- DRR092341_sorted_rseqc.junctionSaturation_plot.pdf

- DRR092341_sorted_rseqc.read_distribution.txt
- DRR092341_sorted_rseqc.splice_events.pdf
- DRR092341_sorted_rseqc.splice_junction.pdf

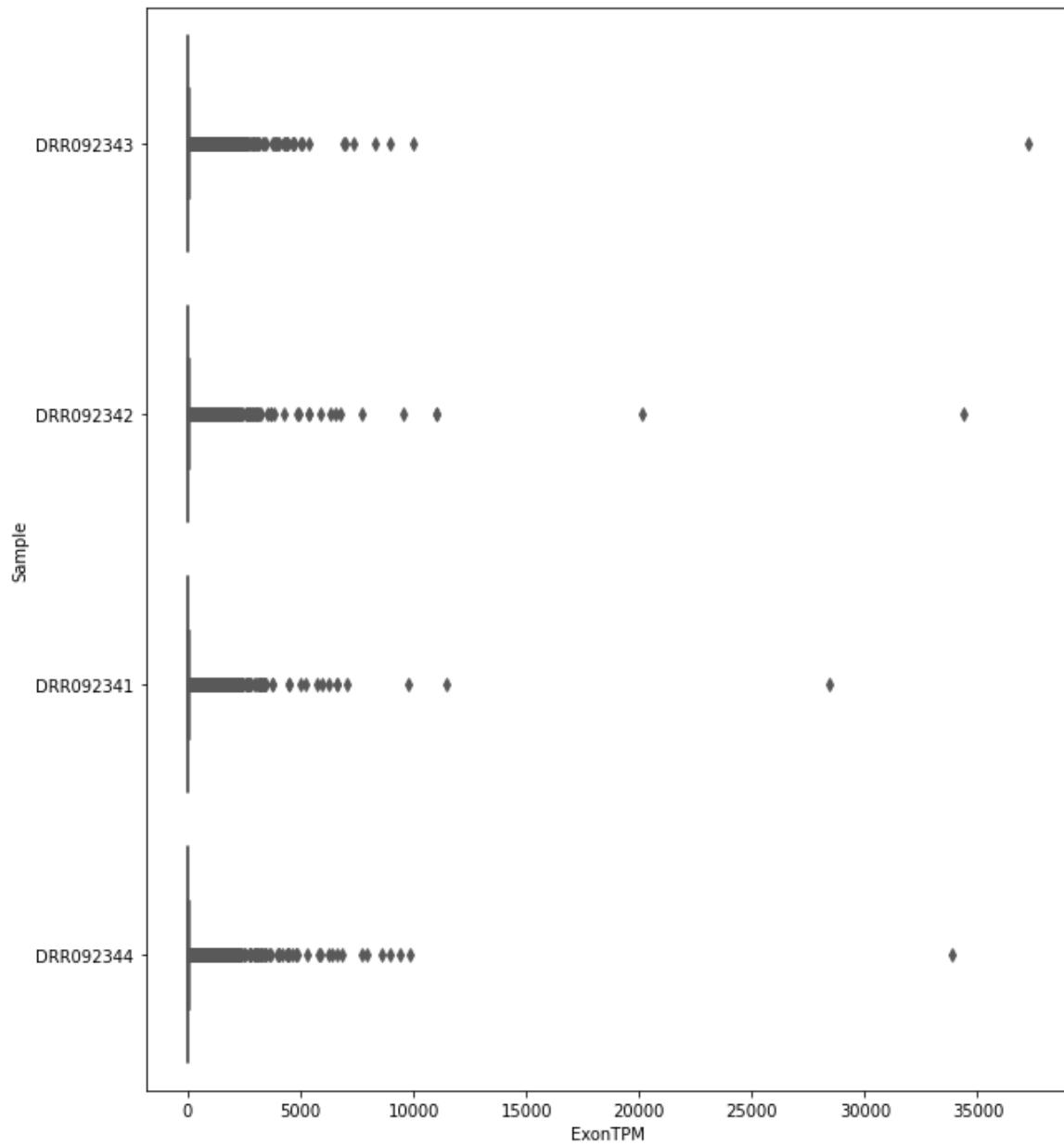


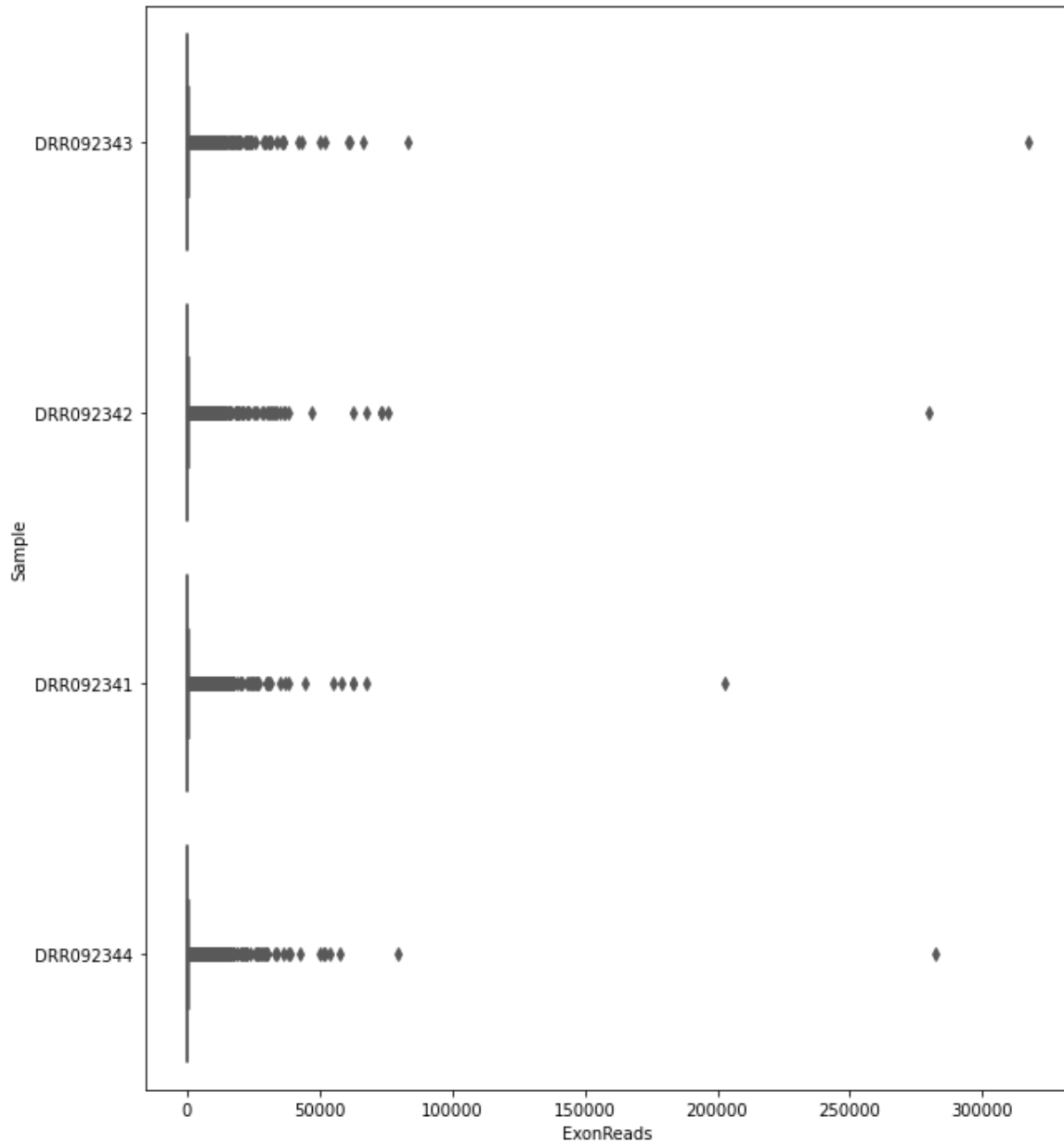
Updating the 00 - Project Report notebook.





The quantification values read counts and TPM are shown in a boxplot for easy comparison.

Exon TPM and reads distribution per sample



5.6 Differential Gene Expression Analysis

The DGE analysis is executed in the **04 - DGA** notebook. The workflow uses [DESeq2](#) and [EdgeR](#) for the identification of differentially expressed genes. R scripts are available at [deseq2-2conditions.cwl](#) and [edgeR-2conditions.cwl](#).

In our approach, genes are designated as differentially expressed if they are identified by DESeq2 and EdgeR. Those genes are shown under the **Intersection** results.

```
results/PRJDB5673/dga/
├── WTAdult_vs_LolAdult_deseq2_dga.log
├── WTAdult_vs_LolAdult_edge_dga.log
```

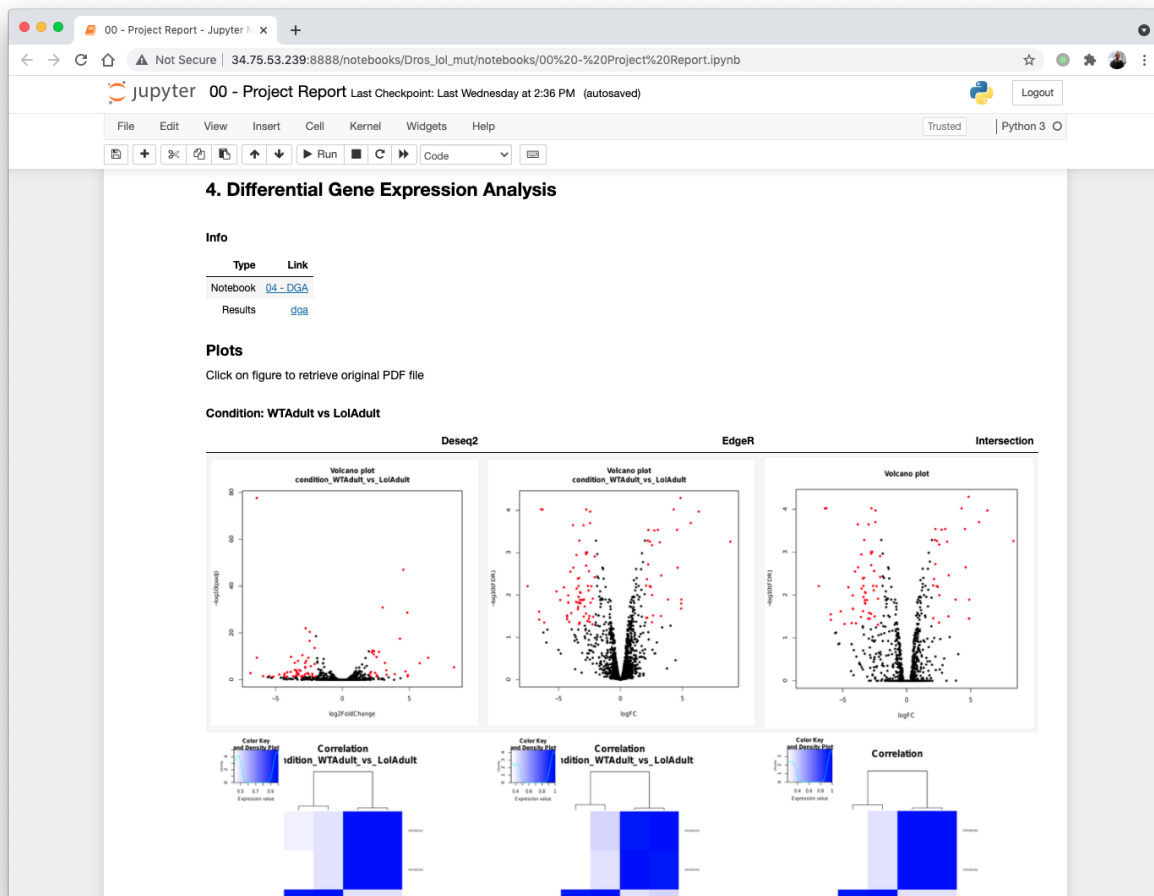
(continues on next page)

(continued from previous page)

```
— WTAdult_vs_LolAdult_heatmap_union.log
— WTAdult_vs_LolAdult_volcano_union.log
— condition_WTAdult_vs_LolAdult_deseq2.csv
— condition_WTAdult_vs_LolAdult_deseq2_correlation_heatmap.pdf
— condition_WTAdult_vs_LolAdult_deseq2_expression_heatmap.pdf
— condition_WTAdult_vs_LolAdult_deseq2_pca.csv
— condition_WTAdult_vs_LolAdult_deseq2_pca.pdf
— condition_WTAdult_vs_LolAdult_deseq2_volcano.pdf
— condition_WTAdult_vs_LolAdult_edgeR.csv
— condition_WTAdult_vs_LolAdult_edgeR_correlation_heatmap.pdf
— condition_WTAdult_vs_LolAdult_edgeR_expression_heatmap.pdf
— condition_WTAdult_vs_LolAdult_edgeR_pca.csv
— condition_WTAdult_vs_LolAdult_edgeR_pca.pdf
— condition_WTAdult_vs_LolAdult_edgeR_volcano.pdf
— condition_WTAdult_vs_LolAdult_intersection.csv
— condition_WTAdult_vs_LolAdult_intersection_correlation_heatmap.pdf
— condition_WTAdult_vs_LolAdult_intersection_expression_heatmap.pdf
— condition_WTAdult_vs_LolAdult_intersection_over-expressed.csv
— condition_WTAdult_vs_LolAdult_intersection_pca.pdf
— condition_WTAdult_vs_LolAdult_intersection_under-expressed.csv
— condition_WTAdult_vs_LolAdult_intersection_volcano.pdf
```

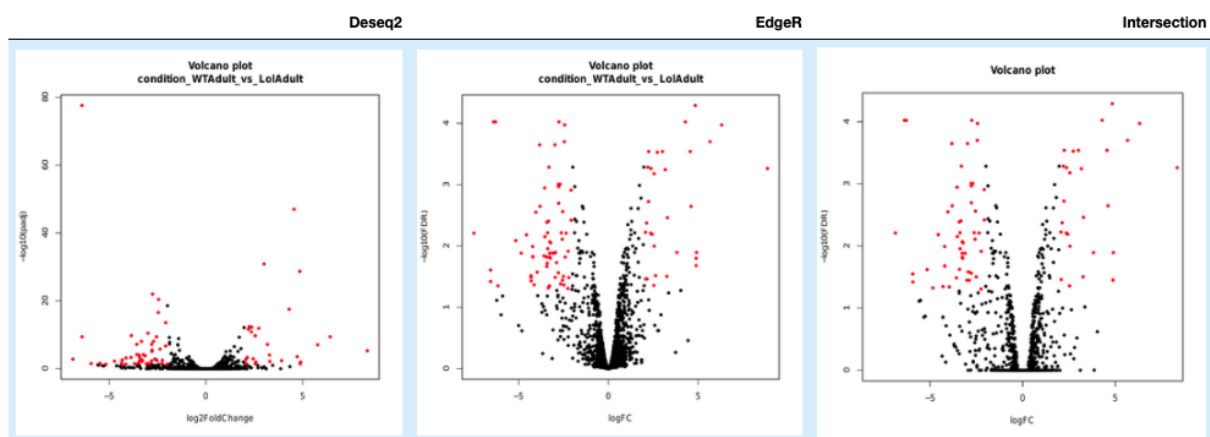
0 directories, 23 files

Updating the 00 - Project Report notebook.

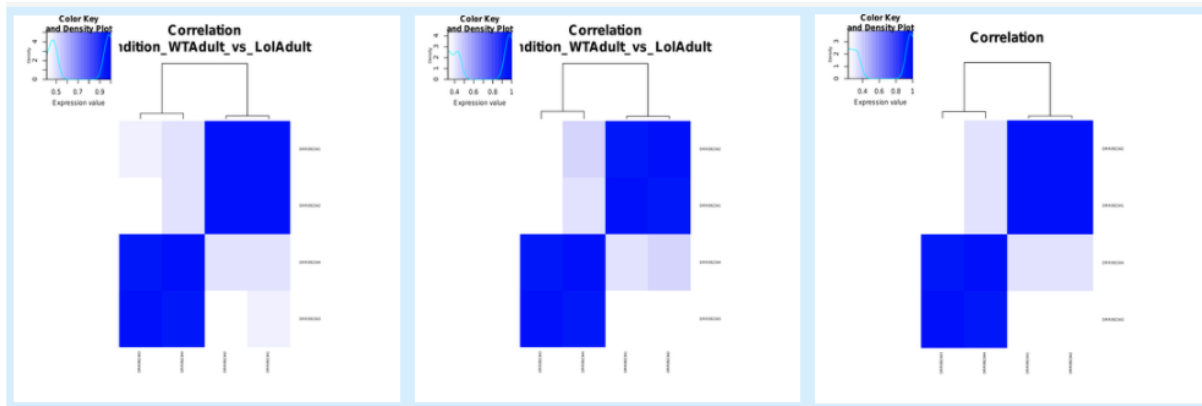


DGE plots are automatically generated.

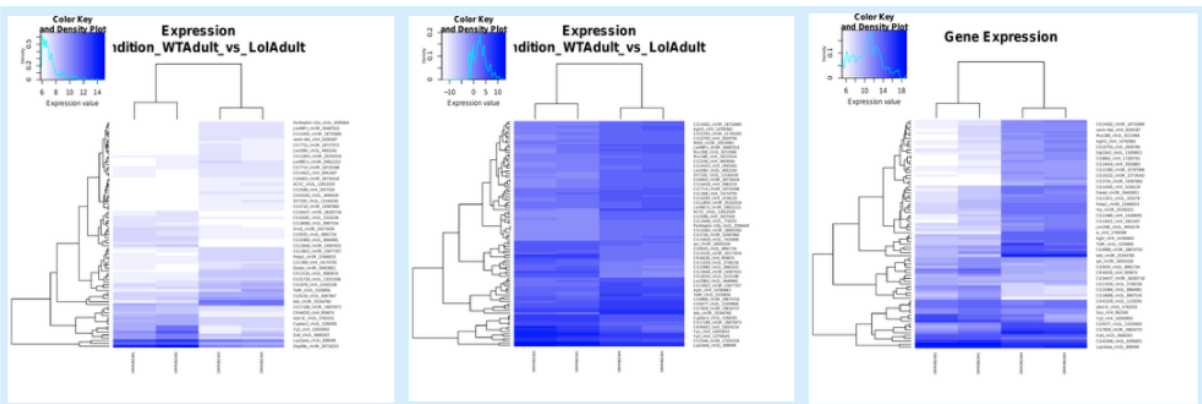
5.6.1 Volcano Plots



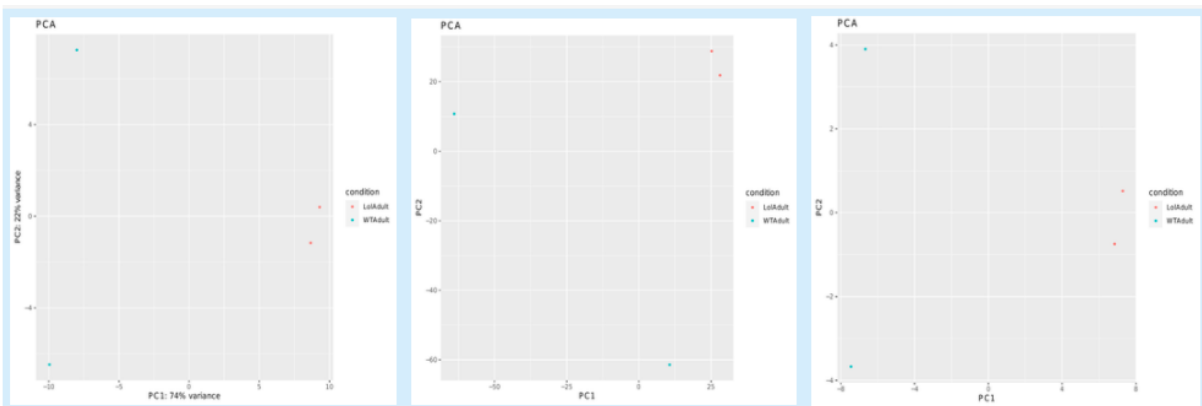
5.6.2 Sample correlation



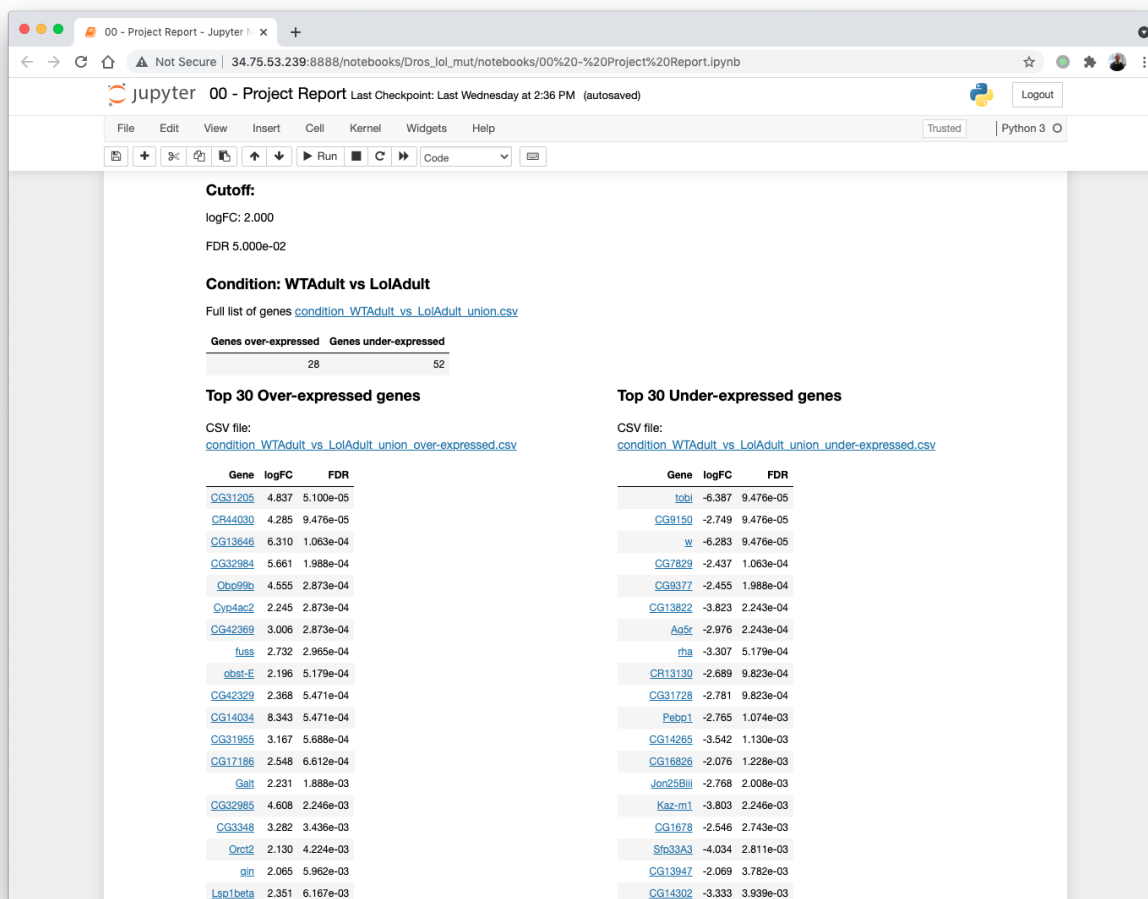
5.6.3 Expression correlation



5.6.4 PCA plots



5.6.5 List of differentially expressed genes



5.7 Gene Ontology Enrichment

Gene Ontology enrichment analysis is executed with an *in-house* developed python package named [goenrichment](#). This tool creates a database by integrating information from [Gene Ontology](#), [NCBI Gene](#) using the files [gene_info](#) and [gene2go](#). Read more in the package web page to create a database for another organism.

Available databases can be downloaded from <https://ftp.ncbi.nlm.nih.gov/pub/goenrichment/>

Next image shows the pre-defined code in PM4NGS for downloading the GO enrichment database for *Drosophila*. Current version of the database contains **43 987** GO terms cross-referenced to **13 741** genes.

Loading Gene2GO database from NCBI FTP

This database is created with the GO-enrichment python package: <https://gitlab.com/r78v10a07/goenrichment>

There are pre-computed database at: <ftp://ftp.ncbi.nlm.nih.gov/pub/goenrichment>

Loading Gene2GO database for human ¶

```
In [4]: goenrichDB = "https://ftp.ncbi.nlm.nih.gov/pub/goenrichment/goenrichDB_drosophila.pickle"
godb = load_goenrichdb(goenrichDB)
print('There are %d alternative ids in database' % (len(godb['alt_id'])))
print('There are %d GO terms' % (len(godb['graph'].nodes())))
print('There are %d genes in database' % (godb['M']))

Loading go enrichment DB from: https://ftp.ncbi.nlm.nih.gov/pub/goenrichment/goenrichDB_drosophila.pickle
There are 3297 alternative ids in database
There are 43987 GO terms
There are 13741 genes in database
```

The GO enrichment tool uses three parameters to identify differential GO terms between the two conditions.

Note: 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

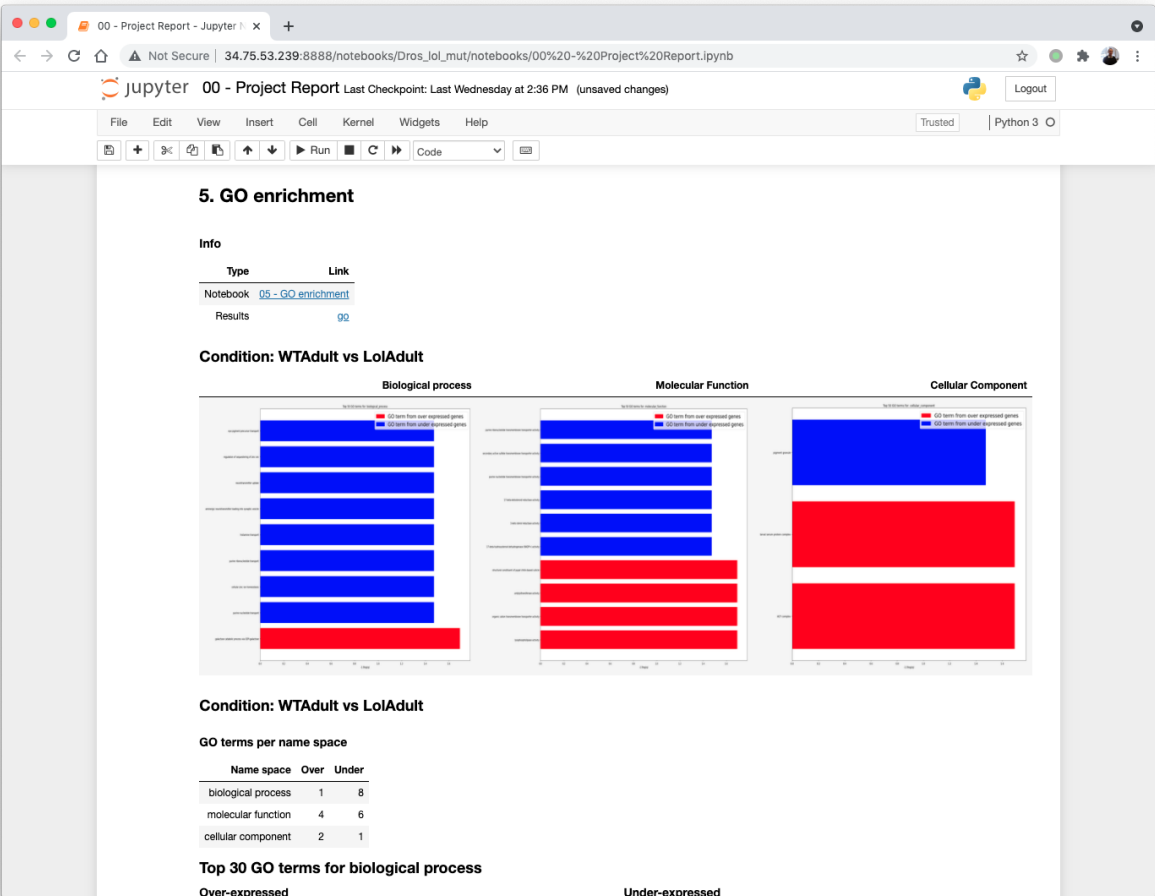
Max number of gene in a GO term to include in the report. Default: 500 This 500 is good for very well annotated organism as human or mouse.

In the case of *Drosophila* use 10 to get relevant results.

In the Jupyter Notebook change the values to**GO enrichment parameters**

```
In [33]: min_category_depth=4
min_category_size=3
max_category_size=10
```

Updating the 00 - Project Report notebook.



The list of differential GO terms are available in the tables per GO name space.

Top 30 GO terms for biological process

Over-expressed

CSV file:

[raw file](#)

GO term	Name	FDR
GO:0033499	galactose catabolic process via UDP-gala	2.020e-02

Under-expressed

CSV file:

[raw file](#)

GO term	Name	FDR
GO:0015865	purine nucleotide transport	3.356e-02
GO:0006882	cellular zinc ion homeostasis	3.356e-02
GO:0015868	purine ribonucleotide transport	3.356e-02
GO:0051608	histamine transport	3.356e-02
GO:0015842	aminergic neurotransmitter loading into	3.356e-02
GO:0001504	neurotransmitter uptake	3.356e-02
GO:0061088	regulation of sequestering of zinc ion	3.356e-02
GO:0006856	eye pigment precursor transport	3.356e-02

Top 30 GO terms for molecular function**Over-expressed**

CSV file:

[raw file](#)

GO term	Name	FDR
GO:0004622	lysophospholipase activity	2.020e-02
GO:0015101	organic cation transmembrane transporter	2.020e-02
GO:0070569	uridylyltransferase activity	2.020e-02
GO:0008011	structural constituent of pupal chitin-b	2.020e-02

Under-expressed

CSV file:

[raw file](#)

GO term	Name	FDR
GO:0072582	17-beta-hydroxysteroid dehydrogenase (NA	3.356e-02
GO:0000253	3-keto sterol reductase activity	3.356e-02
GO:0072555	17-beta-ketosteroid reductase activity	3.356e-02
GO:0015216	purine nucleotide transmembrane transpor	3.356e-02
GO:0008271	secondary active sulfate transmembrane t	3.356e-02
GO:0005346	purine ribonucleotide transmembrane tran	3.356e-02

Top 30 GO terms for cellular component**Over-expressed**

CSV file:

[raw file](#)

GO term	Name	FDR
GO:0016590	ACF complex	2.020e-02
GO:0005616	larval serum protein complex	2.020e-02

Under-expressed

CSV file:

[raw file](#)

GO term	Name	FDR
GO:0048770	pigment granule	3.356e-02

INTRODUCTION

This tutorial will show how to create and configure a Google Cloud Platform instance for Next Generation Sequencing (NGS) data analysis. PM4NGS is a program manager for NGS data analysis that is designed to generate a standard organizational structure including directory structures for the project, Jupyter notebooks for data management and CWL workflows for the pipeline execution.

BACKGROUND READING

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>

HELP AND SUPPORT

For feature requests or bug reports, please open an issue on our [GitHub Repository](#).