Workshop UCIBIO

Module 1 – High-throughput Sequencing Data

Ana Rita Grosso (argrosso@fct.unl.pt)

September 12th 2022

Introduction

This practical exercise will obtain high-throughput sequencing (HTS) data from public resources, check their quality and perform alignments using High Performance Computing (HPC).

Exercise 1. Prepare your working environment

First, we will create a directory to save the files from our analyses:
Run in terminal:
$ **mkdir Module1**
$ **cd Module1**

Second, we will load the INCD module containing all the bioinformatic tools:
Run in terminal:
$ **module load module1**

You can verify if all the bioinformatic tools are available in your environment:
Run in terminal:
$ **module list**

Finally, we will create a basic bash file to be used as template for each analysis step (e.g. "*basicScript.sh*"):
Run in terminal:
$ **nano basicScript.sh**

Complete the bash file with the basic bash lines:
#!/bin/bash
#SBATCH --partition=short
#SBATCH --tasks-per-node=1
#SBATCH --nodes=1

Do *Ctrl+X* to save and close the file (Y and "Enter").

Exercise 2. Download the sample file.
We will download and process the sample with the SRA accession number SRR10027460. The SRA-Tool downloads the file in SRA format (SRA standard format) and converts to fastq format.

First, we will have to create the bash file (e.g. "*getData.sh*") using our template file:

Run in terminal:
$ **cp basicScript.sh getData.sh**

$ **nano getData.sh**

Complete the bash file, providing the command line and the sample ID:

```
#!/bin/bash
#SBATCH --partition=short
#SBATCH --tasks-per-node=1
#SBATCH --nodes=1

fasterq-dump SRR10027460
```

Do *Ctrl+X* to save and close the file.
Now, we can submit our first job!
Run in terminal:
$ **sbatch getData.sh**

You can see the status of the job using *squeue* command.
Run in terminal:
$ **squeue -u <youUsername>**

The download will take some minutes and you can get more information about your sample browsing SRA data archive:

a) How many samples can you find related to cancer? Are all the datasets from human biological source?
b) Which biological material was sequenced in the sample SRR10027460? Which platform was used?

After, you can check if the job run successfully by examining the file "*slurm-<jobID>"*.

If the job finished properly, you should have a "*.fastq"* file in your folder (run "*ls"*. The file is very large, but you can see the first lines.

Run in terminal:
$ **head SRR10027460.fastq**

Exercise 3. Assess data quality.

The FastQC program will assess the quality of each raw data file and the output is a report with graphics in html format (you can open in a Web Browser).

First, we will have to create the bash file (e.g. "*dataQual.sh*") using our template file.

Run in terminal:
$ **cp basicScript.sh dataQual.sh**

$ **nano dataQual.sh**

Complete the bash file, providing the command line and the sample ID:

```
#!/bin/bash
#SBATCH --partition=short
#SBATCH --tasks-per-node=1
#SBATCH --nodes=1

fastqc SRR10027460.fastq
```

Do *Ctrl+X* to save and close the file.

Before submitting the job you should configurate the SRA tools to allow remote access:
Run in terminal:
$ **vdb-config -i**

Choose the Remote Access option, save and exit.

Now, you can submit your first job!

Run in terminal:
$ **sbatch dataQual.sh**

You can see the status of the job using *squeue* command.

After the quality assessment is finished you can inspect the report. Go to the "FCB" folder (in the Desktop) and open the html file in the Web browser:

a) What are the major problems identified for each file?

Exercise 4. Align the RNAseq data to the human transcriptome

The RNAseq data will be aligned to the human transcriptome using Kallisto. The human transcriptome (GENCODE v39) is already in the INCD folder: /data/tutorial/modulo1/data

First, we will have to create the bash file (e.g. "*alignTranscriptome.sh*") using our template file.

Run in terminal:
$ **cp basicScript.sh alignTranscriptome.sh**

$ **nano alignTranscriptome.sh**

Complete the bash file, providing the command line and the sample ID and changing the number of tasks per node (matching the multithread option -t):

```
#!/bin/bash
#SBATCH --partition=short
#SBATCH --tasks-per-node=4
#SBATCH --nodes=1

kallisto quant -i /data/tutorial/modulo1/data/gencode.v39.transcriptome.idx -o output_alignTranscriptome --single -t 4 -l 250 -s 50 SRR10027460.fastq
```

Do *Ctrl+X* to save and close the file.
Now, we can submit our first job!

Run in terminal:
$ **sbatch alignTranscriptome.sh**

You can see the status of the job using *squeue* command.

Finally, you can explore the output of Kallisto. How many files were produced?
You can see the first lines of the "abundance.tsv" file that will be used for the downstream analyses (next tutorial).

Run in terminal:
$ **head abundance.tsv**

Exercise 5. Align the RNAseq data to the human genome

The RNAseq data will be aligned to the human genome using STAR. The human genome (GRCh38) is already in the INCD folder: /data/tutorial/modulo1/data.

First, we will have to create the bash file (e.g. "*alignGenome.sh*") using our template file.

Run in terminal:
$ **cp basicScript.sh alignGenome.sh**

$ **nano alignGenome.sh**

Complete the bash file, providing the command line and the sample ID and changing the number of tasks per node (matching the multithread option - runThreadN):

```
#!/bin/bash
#SBATCH --partition=short
#SBATCH --tasks-per-node=4
#SBATCH --nodes=1

module purge
module load star/2.5.2b

STAR --runThreadN 4 --genomeDir /data/tutorial/modulo1/data/genomeIndex/ --outFileNamePrefix output_alignGenome --outSAMtype SAM --readFilesIn SRR10027460.fastq --quantMode GeneCounts --sjdbGTFfile /data/tutorial/modulo1/data/ref_annot.gtf
```

Do *Ctrl+X* to save and close the file.
Now, we can submit our first job!
Run in terminal:
$ **sbatch alignGenome.sh**

You can see the status of the job using *squeue* command.

After the quality assessment is finished you can inspect the summary report:
Run in terminal:
$ **cat output_alignGenomeLog.final.out**

How many reads were aligned?

You can get the number of reads mapped to each gene:
Run in terminal:
$ **head -n 50 output_alignGenomeReadsPerGene.out.tab**

Finally, you can see the alignment through the Sequence Alignment/Map (SAM) format, where each line (after header section) contains information for one alignment.
Run in terminal:
$ **sed -n '1000,1005p' output_alignGenomeAligned.out.sam**

(optional): if you want to further try to interpret the content of the file, read the SAM format specification: https://samtools.github.io/hts-specs/SAMv1.pdf