Transcriptomics exercise – BT2100

Preparations

How to connect to the local server krabbe for the transcriptomics exercise

1. Connecting to krabbe

Windows-users:

We will use a program called mobaXterm to connect to krabbe. Go to:

https://mobaxterm.mobatek.net/download-home-edition.html and download the portable edition. (Download, unzip, save and click on the .exe file).

If a command-line window appears, write:

```
ssh mol8008_2@krabbe.medisin.ntnu.no
```

Type the password given. You should now be logged in to krabbe.

If an interactive window appears:

Write krabbe.medisin.ntnu.no in “Remote Host” window, tick “Specify Username” and write mol8008_2.

Type the password. You should now be logged in to your user account on krabbe.

Type “ls” when you have logged in. This will list a set of folders. Find the folder with your name on and type cd *[“folder_name”]* to go into the folder. This will be your working folder (directory) for the exercise.

Mac-users:

You could ssh directly from your terminal.

Activate mol8008 conda environment

To run the programs and tools in this course we have created conda environments where all the programs and tools you need have been installed. To access conda, type:

```
source /local/anaconda2/bin/activate
```

You have to type this every time you log in to korall with a new shell.

This environment for this exercise is called mol8008. To start this environment, type:

```
conda activate mol8008
```

To test that the environment has loaded properly, type `hisat2.` This should bring up a list of program parameters for HISAT2. You can also type `hisat2 --version,` which should display that the HISAT2 version installed is 2.2.1.

(Note: To exit any conda environment, type `conda deactivate.`)

Workflow

Differential expression of genes from RNA sequences using **HISAT2, featureCounts** and **Voom.**

1.1 Data set.

The RNA-Seq data used in this exercise is downloaded from Gene Expression Omnibus (GEO) with idemtifier **GSE75035.** It includes 12 sequence samples with paired-end RNA-Seq from one prostate cancer cell-line (LNCaP, 6 samples) and one normal prostate cell-line (RWPE, 6 samples) under various experimental conditions. In this exercise we just focus on the general differences between the LNCaP and RWPE cell-lines. A short summary of the meta-data for this experiment meta_data_GSE75035.xlsx, can be found in the RNA-Seq course folder, which also include all other files you will use in this exercise:

Course folder address for RNA-Seq exercise: **/local/work/biocore/mol8008/RNA/**

Before you start the exercise, I recommend that you make your own working directory called RNASeq (or something similar) where you run commands and store files related to this exercise.

From your starting directory: (for example: mol8008_25@mh-ikom02:~$), type

`mkdir RNASeq` (make the directory)
`cd RNASeq` (move into the directory)

Note that the sequence-files we use in this exercise are quite large! Due to limited space, we recommend that you do **not** copy these large files into your working directory, but run the programs from your working directory using an address to the large files in the course folder. (However, it is OK to copy small files, like scripts etc). For example, an address for the .fastq paired-end sequence file for the RWPE cell-line, replicate 6, first pair is:

```
/local/work/biocore/mol8008/RNA/RWPE_rep6_1.fastq
```

1.2 Reference genome and transcriptome

To align reads to the genome we need a reference genome. HISAT2 uses indexed reference genomes for fast alignment. The RNA-Seq reads for this exercise are from human prostate cancer cell line LNCaP and normal human prostate cell-line RWPE, which means we need a human reference genome. Pre-indexed genomes for the most common organisms are available at the HISAT2 home-page:

http://daehwankimlab.github.io/hisat2/

Have a look at the listed reference genomes (klick ***Download*** in the righthand menu) and see if you can explain what the different suffixes and notations means.

For this exercise we have already downloaded the spliced indexed reference genome *genome_tran* for *H. sapiens, GRCh38* using the *wget* command in linux. The suffix *_tran* indicated that a reference transcriptome (usually a .gtf file) has also been used to construct spliced indexes, which makes alignment of reads covering splice-sites more efficient. For further analysis we also need to download a reference transcriptome. For this exercise we have downloaded the reference transcriptome corresponding to the transcriptome used to construct *genome_tran*, named Homo_sapiens.GRCh38.84.gtf. For more information on the .gtf format you can go to:

ftp://ftp.ensembl.org/pub/release-84/gtf/homo_sapiens/README

We will need the transcriptome reference file later in the pipeline when using *featureCounts.*

Note that if you work on an organism which does not have a HISAT2 pre-indexed genome, you will need to build this yourself. A script for how to do this is included in the HISAT2 genome download files, and is called make_hg38.sh. We will not index new genomes in this exercise, since this requires too much computer resources. Explanations on how to set up the indexing is also in the manual page. See explanation under the option "--known-splicesite-infile <path>*" under "Spliced alignment options" to see how to generate your own spliced indexed reference genome (with suffix _tran).*

1.3 Alignment of RNA-Seq reads using HISAT2

The first alignment step of the RNA-Seq processing workflow aligns each sample individually (that is, one sample file does not need any information from any other sample files to generate the alignment results). For this exercise we will thus test alignment on only one of the samples. We have prepared alignments from all 12 samples in the course folder, which you can use in later stages for the RNA processing workflow.

To lower the computational burden from many course participants, we have reduced the number of sequences in the test sample substantially to contain only 5% of the sequences from the raw file. The two reduced paired-end .fastq files from LNCaP cell-line replicate 6 is:

```
forward_reads: /local/work/biocore/mol8008/RNA/LNCaP_red_rep6_1.fastq
```

```
reverse_reads: /local/work/biocore/mol8008/RNA/LNCaP_red_rep6_2.fastq
```

The reference genome is located at the address:

```
reference_genome: /local/work/biocore/mol8008/RNA/Human_hg20/genome_tran
```

The following basic command is used to align paired-end RNA-Seq reads in *HISAT2*:

```
hisat2 -p 1 --dta -x reference_genome -1 forward_reads -2 reverse_reads -S
LNCaP_rep6.sam 2>summary.txt
```

-p: Number of computing cores. P>1 speeds up analysis, but also requires more resources.
-x: Prefix and location of reference genome file.
-1: File-name of first (forward) pair of paired end reads with "_1" suffix
-2: File-name of second (reverse) pair of paired end reads with "_2" suffix
-S: Name of output-file (aligned reads) in SAM-format.
-dta: This parameter is strictly not necessary to perform an alignment. However, it provides and optimizes the output for subsequent isoform discovery in StringTie (which we will do in the next exercise), so we include it here for convenience.
-2>summary.txt: Adding this command writes alignment statistics to the file summary.txt. You can inspect the alignment summary by typing: `more summary.txt.` How would you rate the alignment results? Was it a success?

There are a million other options in HISAT2 as well, but unless you know what you are doing, I would recommend to use the default settings. An overview of parameters is found at the home-page.

1.4 Assign aligned RNA-Seq reads to gene features using *featureCounts*

After having aligned the reads to the genome, we want to assign reads to pre-defined genomic features (usually genes or transcripts), and obtain a count of the number of reads that belongs to each feature in each sample. For this we use the program *featureCounts.*

FeatureCounts is part of the *Subread*-package available at http://subread.sourceforge.net/

It is preferable to convert .sam files into sorted .bam files before counting (though featureCounts alse handles .sam files directly). Since .bam files are compressed, they take up much less space, and counting is also faster. Converting to .bam is generally convenient, since a lot of programs prefer sorted .bam files as input rather than .sam. You can use the *samtools* package to convert .sam files into sorted .bam files.

Converting: `samtools view -bS LNCaP_rep6.sam > LNCaP_rep6.bam`

You can type `ls –sh` to see and compare the size of the .sam and .bam file.

Sorting: `samtools sort LNCaP_rep6.bam -o LNCaP_rep6_sorted.bam`

You can also use *samtools* to view the alignments from .bam files. Use the command

```
samtools view LNCaP_rep6_sorted.bam | less -S
```

The following command create a count for each gene in a single sample using the reference transcriptome Homo_sapiens.GRCh38.84.gtf and the sorted .bam alignment file from HISAT2.

```
Reference_trans: /local/work/biocore/mol8008/RNA/Homo_sapiens.GRCh38.84.gtf
```

```
featureCounts -T 1 -t exon -g gene_id -O -a Reference_trans -o count-1smp.txt
LNCaP_rep6_sorted.bam
```

-T: The number of cores (or threads) used for counting
-t: Feature type to use in the counting. Here ***exon*** is used, which is also the default. This means that only features tagged as ***exon*** is used for counting.
-g: Specify how features should be grouped into meta-features. In this case ***exon*** features will be combined into ***gene*** meta-features. Thus only ***gene***-counts will be returned in the final count-table. (Note: It is possible to return ***exon***-counts as well by adjusting the parameters).
-O: Allow reads to be assigned to more than one feature.
-a: Transcriptome annotation file in .gtf format.
-o: Name of output-table with genes and counts.
*-**b**: Indicate that input files are in .bam format (Note: Not necessary in this version)*
At the end: the aligned and sorted sequence .bam file(s) to be counted

Look at the resulting table `less -S count-1smp.txt`. See if you can identify the table-column with read-counts.

Using the exact same processing you just performed on one paired-end .fastq sample, we have processed all 12 samples and made sorted .bam files ready to use in the course-folder. Use these files as inputs in the next step. *Note: These are sorted .bam files made from complete (not reduced) sequence files, and are thus considerably larger than the .bam file you just created.*

We will now make a count- table from the whole experiment using *featureCounts* with several sorted .bam files as input:

```
featureCounts -T 1 -t exon -g gene_id -O -a
/local/work/biocore/mol8008/RNA/Homo_sapiens.GRCh38.84.gtf -o count-table.txt
/local/work/biocore/mol8008/RNA/LNCaP_rep1.bam
/local/work/biocore/mol8008/RNA/LNCaP_rep2.bam
/local/work/biocore/mol8008/RNA/LNCaP_rep3.bam
/local/work/biocore/mol8008/RNA/LNCaP_rep4.bam
/local/work/biocore/mol8008/RNA/LNCaP_rep5.bam
/local/work/biocore/mol8008/RNA/LNCaP_rep6.bam
/local/work/biocore/mol8008/RNA/RWPE_rep1.bam
/local/work/biocore/mol8008/RNA/RWPE_rep2.bam
/local/work/biocore/mol8008/RNA/RWPE_rep3.bam
/local/work/biocore/mol8008/RNA/RWPE_rep4.bam
/local/work/biocore/mol8008/RNA/RWPE_rep5.bam
/local/work/biocore/mol8008/RNA/RWPE_rep6.bam
```

In this case *featureCounts* will make a table where each sample file-name and counts are listed in columns in the output table. (This should take about 10-15 mins with one thread. Grab a coffee!)

The raw output table from featureCounts can be a little awkward to handle. I have thus made a script *fcnts2dseq.py* to make it simpler. Copy the script to your working folder:

```
cp /local/work/biocore/mol8008/RNA/fcnts2dseq.py .
```

And run the script on count-table.txt.

```
python fcnts2dseq.py count-table.txt
```

which returns a simpler *count-table.tsv* file, and can be used as input to differential expression analysis in *Voom* and *R* in the next step.

1.5 Differential expression of genes using Voom and R

A script for running Voom, *Voom_script.r*. is provided in the course folder. Copy the script to your working directory. To learn the steps for a typical differential gene expression analysis in R, open a session of R (you start a session by typing "R" in a new linux-window. Make sure that you stand in your working directory where you have stored your *count-table.tsv* when you start R). Open the script in a text-editor, for example *emacs*. Run the script line-by-line by copy-pasting each line from text-editor into the R-session. Look at the output from each line. The final line writes a table named Voom_diffexp.txt of differentially expressed genes between LNCaP and RWPE cells, including fold-changes , p-values etc.

One issue with the transcriptome reference provided here, is that it prefers ENS- gene ids in the count table rather than conventional gene-symbols (in my experience, most people, including myself, prefer the gene symbols). I have thus made a script, replace_ENS_with_gene_symbol_py3.py which converts ENS- ids to gene symbols in the final differential expression table. Copy the script together with the reference file ENS_to_gene-names.txt. from the course folder to your working directory.

```
python replace_ENS_with_gene_symbol_py3.py Voom_diffexp.txt >
Voom_diffexp_gnms.txt
```

You should now have the conventional gene symbols in the table Voom_diffexp_gnms.txt. A convenient way to look at the first lines in this file is to use

```
less -S Voom_diffexp_gnms.txt
```

You can also run the complete R-script from command-line without starting an R-session.

```
Rscript --vanilla Voom_script.r
```

To create an MA and Volcano plot, you can use a python script plot_MA_Volcano.py interactively using ipython3 (it is a bit like Jupyter notebook where you can run scripts and code interactively). First deactivate the mol8008 environment.

```
mol8008 deactivate
```

Start ipython3 with interactive plots:

```
ipython3 --pylab
```

Run the script interactively. It generates two plots: MA_plot.pdf and Volcano_plot.pdf. Note that you can adjust in the script how many genes to display. Default is 100.