**RNA-Seq exercise – BI311F-1 20V Akvagenomikk og bioinformatikk**

**Login server**

**From Mac:** Open terminal and type ssh [username@158.39.32.203](mailto:username@158.39.32.203) where username is your username

**From windows:** Follow this procedure: <https://mobaxterm.mobatek.net/download-home-edition.html>

**Workflow 1**

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 from Tilapia. We will use samples from 3 domesticated and 3 wild tilapia**.** In this exercise we make it simple, and just focus on the general differences between domesticated and wild tilapia. A short summary of the meta-data for this experiment meta\_data.txt, can be found in the course folder, which also include all other files you will use in this exercise:

Course folder address for RNA-Seq exercise: /home2/BI311F

Before we start working on the linux server, type “bash” in the command line

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

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

Make a directory to store genome files

mkdir Genome

Note that sequence-files are usually 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).

The files are located in the folder /home2/BI311F/reads and looks like this:

robin@fba-bod-p1:/home2/BI311F/reads$ ls -lh

total 19G

-rwxrwxrwx 1 artem artem 1.7G Mar 5 08:37 H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.8G Mar 5 08:37 H3T72BBXY\_07-W63\_CGATGT\_L005\_R2\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1011M Mar 5 08:38 H3T72BBXY\_08-W66\_GCCAAT\_L005\_R1\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.1G Mar 5 08:38 H3T72BBXY\_08-W66\_GCCAAT\_L005\_R2\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.8G Mar 5 08:38 H3T72BBXY\_09-W68\_ACTTGA\_L005\_R1\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.8G Mar 5 08:38 H3T72BBXY\_09-W68\_ACTTGA\_L005\_R2\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.8G Mar 5 08:40 H3T72BBXY\_13-D63\_CACGAT\_L005\_R1\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.9G Mar 5 08:40 H3T72BBXY\_13-D63\_CACGAT\_L005\_R2\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.8G Mar 5 11:46 H3T72BBXY\_14-D66\_GACGAC\_L005\_R1\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.9G Mar 5 11:47 H3T72BBXY\_14-D66\_GACGAC\_L005\_R2\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.4G Mar 5 11:45 H3T72BBXY\_15-D68\_ATCACG\_L005\_R1\_001.fastq.gz

-rwxrwxrwx 1 artem artem 1.4G Mar 5 11:45 H3T72BBXY\_15-D68\_ATCACG\_L005\_R2\_001.fastq.gz

**Trimming and QC of sequencing data**

To determine the quality of the sequencing data we run a QC (Quality control) of the data using a program calles fastqc. This is done using the following command:

fastqc H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

robin@fba-bod-p1:/home2/BI311F/robin/RNASeq$ fastqc H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Started analysis of H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 5% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 10% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 15% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 20% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 25% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 30% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 35% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 40% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 45% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 50% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 55% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 60% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 65% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 70% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 75% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 80% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 85% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 90% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

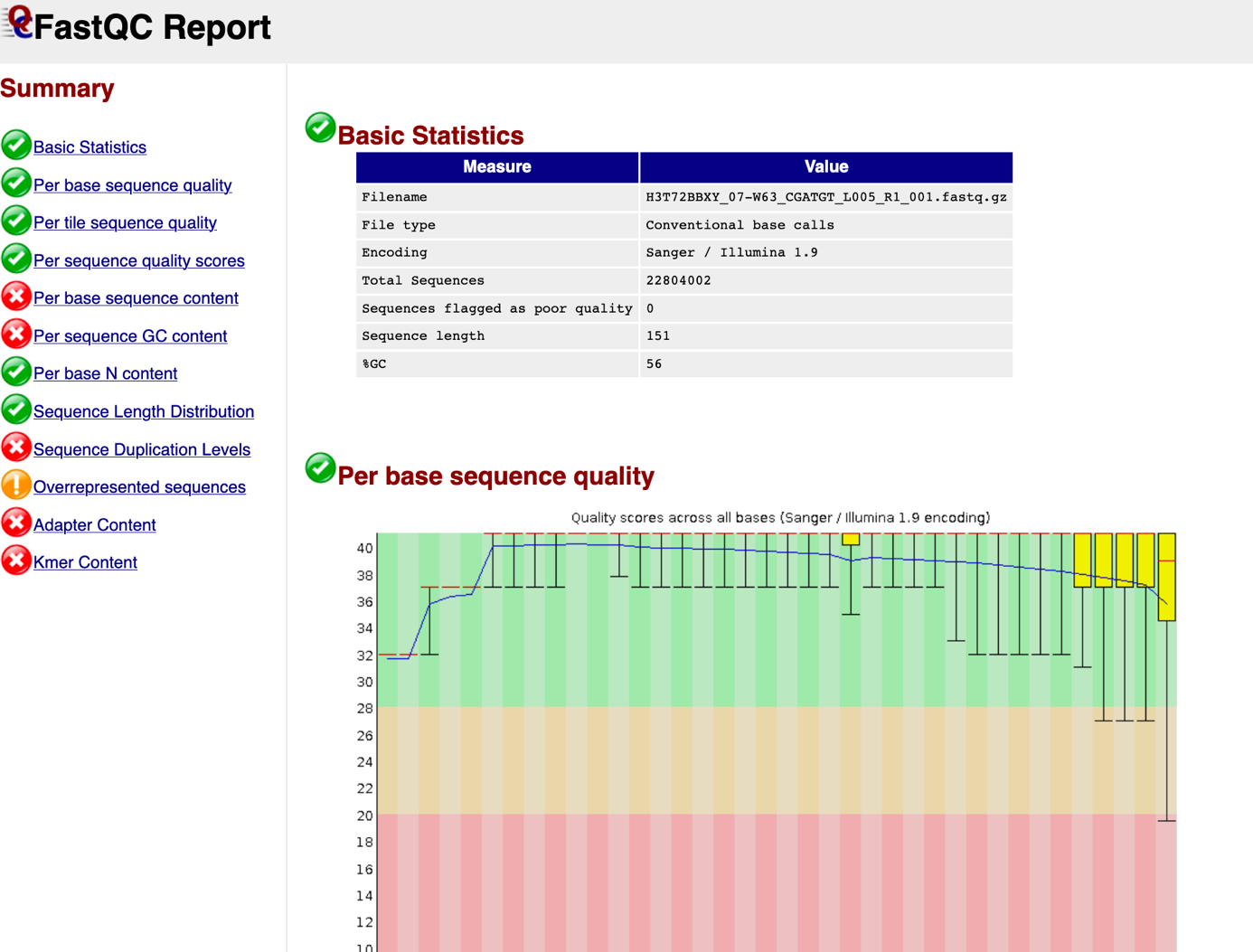
Approx 95% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Approx 100% complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

Analysis complete for H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

This will generate a zip file (H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001\_fastqc.zip) with files including html files with images. Download this to your desktop using scp. Open a new terminal without logging in to the server and use scp like this by typing the password when asked (change the username):

scp robin@158.39.32.203:/home2/BI311F/robin/H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001\_fastqc.zip ./

****

The html file loos like the above. You can see that the beginning and end of the reads have lower quality. We can now decide if we want to trim the beginning and end of the reads. If the quality is below 20, consider removing bases.

**Trimming of reads:**

Bases can be removed using a trimming tool. We will use cutadapt. Cutadapt accepts .gz input files, but produces unzipped output files.

For example, to remove the first five bases of each read:

cutadapt -u 5 -o Trim\_H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

For example, to remove the seven last bases of each read:

cutadapt -u -5 -o Trim\_H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

If you want to remove reads with quality below a certain quality threshold, for instance 20, use the following command:

cutadapt -q 20 -o Trim\_H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz

I recommend using the quality trimming like above using -q 20

Then run a new fastqc on one of the files to compare the quality and find number of reads.

You can also find the number of reads in a fastq file using this command:

wc -l file.fastq

Then divide this number by four, to get the number of sequences. Compare the number of sequences in the files before and after cutadapt.

Continue using the trimmed files if you think trimming is needed.

**1.2 Access HISAT2**

Differential expression of genes using ***HISAT2, featureCounts*** and ***Voom***

To run **HISAT2**, you simply type “hisat2” which should bring up a list of program parameters if your path specification was successful.

**1.3 Reference genome and transcriptome**

To align reads to the genome we need a reference genome. HISAT2 uses indexed reference genomes for fast alignment. Pre-indexed genomes for the most common organisms are available at the HISAT2 home-page:

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

<https://ccb.jhu.edu/software/hisat2/index.shtml>

Have a look at the listed reference genomes and see if you can explain what the different suffixes and notations means.

**NB!** Since Tilapia is not path of the genomes on the website, we need to make the genome file using the command:

hisat2-build

hisat2-build [options]\* <reference\_in> <ht2\_base>

The reference\_in has to be downloaded from

[https://www.ncbi.nlm.nih.gov/genome/?term=txid8128[orgn](https://www.ncbi.nlm.nih.gov/genome/?term=txid8128%5borgn)]

Download these files when you are in the Genome directory

wget <ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/858/045/GCF_001858045.2_O_niloticus_UMD_NMBU/GCF_001858045.2_O_niloticus_UMD_NMBU_genomic.fna.gz>

**At the same time, we can download the annotation file:**

wget ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/858/045/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff.gz

**Unzip by:**

*gunzip* GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff.gz

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

**Download it using:**

*wget* [*ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/858/045/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_rna.fna.gz*](ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/001/858/045/GCF_001858045.2_O_niloticus_UMD_NMBU/GCF_001858045.2_O_niloticus_UMD_NMBU_rna.fna.gz)

***NB!* hisat2-build accepts only unzipped files:**

*gunzip* gunzip GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.fna.gz

**To make the genome-index run:**

hisat2-build GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.fna Tilapia\_GCF\_001858045.2

**The you get something like this:**

-rw-rw-r-- 1 robin robin 1018624591 Mar 4 15:03 GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.fna

-rw-rw-r-- 1 robin robin 21131777 Mar 4 15:03 GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff.gz

-rw-rw-r-- 1 robin robin 339817677 Mar 4 17:23 Tilapia\_GCF\_001858045.2.1.ht2

-rw-rw-r-- 1 robin robin 251406644 Mar 4 17:23 Tilapia\_GCF\_001858045.2.2.ht2

-rw-rw-r-- 1 robin robin 27107 Mar 4 17:11 Tilapia\_GCF\_001858045.2.3.ht2

-rw-rw-r-- 1 robin robin 251406638 Mar 4 17:11 Tilapia\_GCF\_001858045.2.4.ht2

-rw-rw-r-- 1 robin robin 453411695 Mar 4 17:25 Tilapia\_GCF\_001858045.2.5.ht2

-rw-rw-r-- 1 robin robin 255767808 Mar 4 17:25 Tilapia\_GCF\_001858045.2.6.ht2

-rw-rw-r-- 1 robin robin 12 Mar 4 17:12 Tilapia\_GCF\_001858045.2.7.ht2

-rw-rw-r-- 1 robin robin 8 Mar 4 17:12 Tilapia\_GCF\_001858045.2.8.ht2

The ht2 is the index-files.

**1.4 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 6 samples in the course folder, which you can use in later stages for the RNA processing workflow.

We will analyze paired-end sequencing data:

<https://www.illumina.com/content/dam/illumina-marketing/documents/products/illumina_sequencing_introduction.pdf>

Each sample has one **forward read** and one **reverse read**:

The following basic command is used to align paired-end RNA-Seq reads in *HISAT2* for the samples W63:

hisat2 -p 1 --dta -x ./Genome/Tilapia\_GCF\_001858045.2 -1 ../../reads/H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz -2 ../../reads/H3T72BBXY\_07-W63\_CGATGT\_L005\_R2\_001.fastq.gz -S W63.sam 2>W63\_summary.csv

**-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.*

**1.5 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 athttp://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 W63.sam > W63.bam

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

Sorting: samtools sort W63.bam -o W63.sorted.bam  
 samtools sort W63.bam -T /home2/BI311F/username -o W63.sorted.bam

*Note: You may get an error running the first command (blue). If this happens, you can use the second command (red) with the –T option. The reason is that samtools create temporary files during the sorting (which is deleted afterwards), but sometimes they fail to write to the default folder (for some reason), so the program fails. So we specify another folder to write the temporary files using –T. Replace username with your own username. This is a problem specific for our course-setup, and under normal setups the command in blue is sufficient to perform this operation!*

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

samtools view W63.sorted.bam | less -S

The following command create a count for each gene in a single sample using the reference transcriptome annotation file GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff and the sorted W63.sorted.bam alignment file from HISAT2.   
  
**Reference\_trans**:

/home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff

featureCounts -T 1 -t mRNA -g gene -O -a **Reference\_trans** -b -o Counts\_W63.txt W63.sorted.bam

**-T**: The number of cores (or threads) used for counting  
**-t**: Feature type to use in the counting. Here ***mRNA*** is used, which is also the default. This means that only features tagged as ***mRNA*** 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  
At the end: the aligned and sorted sequence .bam file(s) to be counted  
  
Look at the resulting table less -S Counts\_W63.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, **try to process the remaining 5 samples.**

Use all six files as inputs in the next step.

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

Here we use the sam-files as input.

featureCounts -T 1 -b -t gene -g gene -O -a /home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff -o W63.txt /home2/BI311F/robin/RNASeq/W63.sam

featureCounts -T 1 -b -t gene -g gene -O -a /home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff -o W66.txt /home2/BI311F/robin/RNASeq/W66.sam

featureCounts -T 1 -b -t gene -g gene -O -a /home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff -o W68.txt /home2/BI311F/robin/RNASeq/W68.sam

featureCounts -T 1 -b -t gene -g gene -O -a /home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff -o D63.txt /home2/BI311F/robin/RNASeq/D63.sam

featureCounts -T 1 -b -t gene -g gene -O -a /home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff -o D66.txt /home2/BI311F/robin/RNASeq/D66.sam

featureCounts -T 1 -b -t gene -g gene -O -a /home2/BI311F/robin/RNASeq/Genome/GCF\_001858045.2\_O\_niloticus\_UMD\_NMBU\_genomic.gff -o D68.txt /home2/BI311F/robin/RNASeq/D68.sam

NB! Change “username” to your username

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 MakeDataFrame.R in the folder /home2/BI311F/scripts to make it simpler.

Copy the script to your working folder where the counts-files are:

cp /home2/BI311F/scripts/MakeDataFrame.R ./

And run the script like this

Rscript MakeDataFrame.R

which returns a Count\_DataFrame.csv file, and can be used as input to differential expression analysis in *Voom* and *R* in the next step.

less Count\_DataFrame.csv

ctsd 818 1105 503 1186 1776 1574

LOC102082062 100 95 24 364 304 463

LOC109202176 1 0 0 5 4 4

LOC100693279 1 7 1 0 9 3

sirt3 108 117 121 67 89 84

psmd13 524 797 870 661 846 622

LOC100694086 8 0 16 13 15 13

kiaa0895l 253 289 273 330 362 402

LOC112843579 4 8 0 10 2 6

znf319 86 73 62 95 55 152

**1.6 Differential expression of genes using Voom and R**

A script for running Voom, *limma.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\_DataFrame.csv when you start R).

The final line writes a table named topTable\_W\_vs\_1stGen.csv of differentially expressed genes between W and D samples, including fold-changes , p-values etc.

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

Rscript --vanilla *limma.R*

**Make a volcano plot by running the Vulcano.R script in the script-folder. First copy the script to your own directory.**

Copy the result pdf to your own computer by opening a new terminal (do not log into the server) then type: scp username@158.39.32.203:/home2/BI311F/username/RNASeq/\*.pdf ./

**Workflow 2**

Differential expression of annotated isoforms using **kallisto** and **sleuth**.

**3.1 Indexed *kallisto* transcriptome**

The *kallisto* program skips genome alignment and instead aligns sequences directly to a properly indexed reference transcriptome, and then estimates isoforms and abundancies. Differential expression analysis at the isoform level is then performed by *sleuth*, which is particularly tailored for *kallisto*-outputs.

The homepage for kallisto is

https://pachterlab.github.io/kallisto/

**Create Indexed reference transcriptomes for *kallisto***

In *kallisto* we only need the reference transcriptome (not genome), but since *kallisto* also relies on sequence alignment, we need a special version of the reference transcriptome where the sequence for each transcript is also included. Such transcriptomes for the most organisms are possible to retrieve from ensemble.

http://www.ensembl.org/info/data/ftp/index.html

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

In this exercise we have already downloaded a reference transcriptome to the course-folder. However, we here provide a brief explanation on how to do it. **You don’t have to do this in the exercise.**

Click on the FASTA file under Human and cDNA, then use the wget command to retrieve it

wget ftp://ftp.ensembl.org/pub/release-89/fasta/oreochromis\_niloticus/cdna/Oreochromis\_niloticus.Orenil1.0.cdna.all.fa.gz

Use the gunzip command to unpack it.

gunzip [Oreochromis\_niloticus.Orenil1.0.cdna.all.fa.gz](ftp://ftp.ensembl.org/pub/release-89/fasta/oreochromis_niloticus/cdna/Oreochromis_niloticus.Orenil1.0.cdna.all.fa.gz)

There is also an explanation for how to create transcriptomes from reference genome and .gtf files at

<https://github.com/griffithlab/rnaseq_tutorial/wiki/Kallisto>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For now we use the ensembl downloaded *kallisto* transcriptome with sequences:

Oreochromis\_niloticus.Orenil1.0.cdna.all.fa

You can look at the first entries of the file to see what it looks like.

less Oreochromis\_niloticus.Orenil1.0.cdna.all.fa

Index the reference transcriptome by typing:

kallisto index -i transcripts.idx Oreochromis\_niloticus.Orenil1.0.cdna.all.fa

Note that we can use the .gz packed transcriptome directly in the command-line. The transcripts.idx is the created indexed transcriptome. If indexing takes too long, you can also use the transcripts.idx file prepared in the course folder.

**3.2 Isoform abundance estimation in *kallisto* using the indexed transcriptome**

Use the following command to perform isoform abundance estimation with *kallisto*

NB! This will take some time for all samples.

/home2/BI311F/robin/Software/kallisto/kallisto quant -i ./Genome/transcripts.idx -t 1 -b 5 -o W63 ../../reads/H3T72BBXY\_07-W63\_CGATGT\_L005\_R1\_001.fastq.gz ../../reads/H3T72BBXY\_07-W63\_CGATGT\_L005\_R2\_001.fastq.gz

/home2/BI311F/robin/Software/kallisto/kallisto quant -i ./Genome/transcripts.idx -t 1 -b 5 -o W66 ../../reads/H3T72BBXY\_08-W66\_GCCAAT\_L005\_R1\_001.fastq.gz ../../reads/H3T72BBXY\_08-W66\_GCCAAT\_L005\_R2\_001.fastq.gz

/home2/BI311F/robin/Software/kallisto/kallisto quant -i ./Genome/transcripts.idx -t 1 -b 5 -o W68 ../../reads/H3T72BBXY\_09-W68\_ACTTGA\_L005\_R1\_001.fastq.gz ../../reads/H3T72BBXY\_09-W68\_ACTTGA\_L005\_R2\_001.fastq.gz

/home2/BI311F/robin/Software/kallisto/kallisto quant -i ./Genome/transcripts.idx -t 1 -b 5 -o D63 ../../reads/H3T72BBXY\_13-D63\_CACGAT\_L005\_R1\_001.fastq.gz ../../reads/H3T72BBXY\_13-D63\_CACGAT\_L005\_R2\_001.fastq.gz

/home2/BI311F/robin/Software/kallisto/kallisto quant -i ./Genome/transcripts.idx -t 1 -b 5 -o D66 ../../reads/H3T72BBXY\_14-D66\_GACGAC\_L005\_R1\_001.fastq.gz ../../reads/H3T72BBXY\_14-D66\_GACGAC\_L005\_R2\_001.fastq.gz

/home2/BI311F/robin/Software/kallisto/kallisto quant -i ./Genome/transcripts.idx -t 1 -b 5 -o D68 ../../reads/H3T72BBXY\_15-D68\_ATCACG\_L005\_R1\_001.fastq.gz ../../reads/H3T72BBXY\_15-D68\_ATCACG\_L005\_R2\_001.fastq.gz

Output files can be found in specified folder for each sample, for instance ./W63, where the file abundance.tsv includes transcripts and abundancies. Note that you will need to specify a new catalogue name for each sample, otherwise results from a new run will overwrite results from the last run. Note also that you need to turn on the –b bootstrap option is if you want to perform differential expression analysis in *sleuth* afterwards. This will take somewhat longer run-time, however, this can be compensated by increasing the number of threads (option –t. Do not do this in this exercise due to limited computational resources ). Note that there is no gain in using more threads than bootstraps. Normally we have used 5 bootstraps. Output-folders for all samples have been prepared in the course-folder, which you can use as input to *sleuth* for differential expression in the next step.

**3.3 Differential expression of *kallisto* isoforms with *sleuth***

The home-page for *sleuth* can be found at

<https://pachterlab.github.io/sleuth/>

# Differential expression analysis in *sleuth* is performed in “R"

**Install sleuth:**

Open R by typing R in the terminal.

library(sleuth, lib.loc = "/home2/BI311F//R/x86\_64-pc-linux-gnu-library/3.6/")

**Sleuth is now ready to use.**

https://pachterlab.github.io/sleuth\_walkthroughs/pval\_agg/analysis.html

Make a metadata file using nano:

sample group pair path

W63 WT 63 ./W63/abundance.h5

W66 WT 66 ./W66/abundance.h5

W68 WT 68 ./W68/abundance.h5

D63 D 63 ./D63/abundance.h5

D66 D 66 ./D63/abundance.h5

D68 D 68 ./D68/abundance.h5

and read file in R:

> metadata <- read.table(file="metadata\_test.txt",sep="\t", header=T)

> metadata

sample group pair path

W63 WT 63 ./W63/abundance.h5

W66 WT 66 ./W66/abundance.h5

W68 WT 68 ./W68/abundance.h5

D63 D 63 ./D63/abundance.h5

D66 D 66 ./D63/abundance.h5

D68 D 68 ./D68/abundance.h5

Convert path column to character:

metadata$path <- as.character(metadata$path)

so <- sleuth\_prep(metadata)

so <- sleuth\_fit(so, ~group+pair, 'reduced')

so <- sleuth\_fit(so, ~group+pair, 'full')

so <- sleuth\_lrt(so, 'reduced', 'full')

sleuth\_table\_gene <- sleuth\_results(so, 'reduced:full', 'lrt', show\_all = FALSE)

head(sleuth\_table\_gene)

head(sleuth\_table\_gene[order(sleuth\_table\_gene$qval,decreasing=F),])