15 June, Barcelona

Session 7. Differential gene expression based on RNASeq

RNA sequencing is an NGS method for sequencing RNA (converted to cDNA). Today, it has superseded most of other methods for studying transcriptomes.

RNAseq has many applications. Usually, researchers are comparing gene expression levels of one organism in different conditions to see how the condition alters the expression levels, which hints to the functions of the genes .

Here we will find differentially expressed genes between two conditions (planktonic and biofilm) of pathogenic fungus *Candida parapsilosis* using RNAseq. Differential gene expression analysis will show us which genes are important for this yeast in transformation from planktonic (non-pathogenic) to biofilm (pathogenic) morhotypes.

Holland et al. (https://www.ncbi.nlm.nih.gov/pubmed/25233198) have performed the experimental part – they have grown *C. parapsilosis* in normal conditions and in conditions that induce biofilm formation. For each of these conditions they have repeated the experiment 3 times (3 biological replicates), extracted RNA and did RNA sequencing.

Here, we will analyze the data coming from this RNAseq experiment. Important to note, this is the most basic experimental design for RNAseq study (comparing two conditions), much more sophisticated experiments can be performed (many conditions, covariates, time-series, etc). However, most of the steps in RNAseq data analysis even in more complicated experiments will very be similar to what we will do here.

Let’s start.

As an NGS-based technology, RNAseq generates massive datasets that have to be analyzed by a set of multiple software, which is collectively called “pipeline”.
The main steps of the RNAseq data analysis (differential gene expression analysis) pipeline include:

1. Quality control
2. Trimming
3. Read mapping and read summarization
4. Normalization and differential gene expression analysis
5. Functional analysis

1. Quality control

Raw RNAseq data is represented by reads. Reads are short stretches of cDNA that were actually sequenced. Reads are stored in files in ***fastq*** format. Fastq files are usually very big, that’s why we compress them to save space.
Specifications of fastq format can be found here. In our case we have strand-specific (TruSeq, reversely stranded) paired-end reads.
Before doing any analysis, we need to ensure that the quality of read is high, there are no any contaminations, adapters sequences, etc. For this purpose, we can use the popular software called **fastqc.**

For that we do:
cd QC

cat QC.sh

Let’s examine the command. Then we do

bash QC.sh

The output of **fastqc** are in ***html*** and ***zip*** formats. Let’s open ***html*** file with a browser.

You can see that data has some adapters and some reads have low quality score. We need to remove them.

2. Trimming

The process of removing adapters, low quality reads and bases, etc, is called trimming. We will use the software called **trimmomatic**.

Let’s go to

cd ../trimming

cat trimming.sh

Let’s examine the command. When done, we run the command by invoking the bash script with it:

bash trimming.sh

For paired end data, trimmomatic generates 4 files – 2 files for paired reads and 2 files for unpaired reads (in case if one read from a pair was discarded after trimming). To check if we have successfully removed low quality reads and adapters, we run **fastqc** on these 4 files and examine the results. As you can see, trimming was successful and we can continue with further analysis.

3. Mapping and read counting

When the raw data is clean, it has to be aligned against the corresponding reference sequence. The alignment of reads to the reference genome is called mapping. Today, for many model and even non-model organisms the reference genomes are already sequenced and publicly available. Otherwise one has to do a *de novo* transcriptome assembly, which we don’t cover here, or align the reads to closely related species, whose genome is sequenced.
For *Candida parapsilosis* the reference genome is available from Candida Genome Database.

Read mapping is computationally intensive task, and to reduce it we will map the reads to only one chromosome.

Software which we use for mapping RNAseq data to the reference genome is called STAR.
In general read mapping consists of two steps:

- **a.** indexing of reference genome
- **b.** and mapping itself

Indexing of reference genome is algorithmically complex. You can think of genome indexing as indexing of a book, when you write down the location of each of the word in the book. So when you have the index you can easily and quickly find the location of each word in that book. A similar principle works for genomes and reads – index allows to quickly find from which location of the genome each read is coming.

To perform the indexing, do

cd ../mapping/ref_gen/

cat indexing.sh

Explore the command and the run it:

bash indexing.sh

We have generated the genome (one chromosome) index. Once it is ready, we proceed with mapping. We go back one directory.

cd ..

In our example there are 3 replicates of paired end reads called rep1_subset_chr504_r1.fastq.gz, rep1_subset_chr504_r2.fastq.gz, rep2_subset_chr504_r1.fastq.gz, etc.

Let’s explore the mapping command.

cat mapping.sh

Now let’s run it:

bash mapping.sh

Read mapping generates so called ***sam*** files, and binary (compressed) version of ***sam*** file is called ***bam*** file. These files are usually very big. Specification of ***sam/bam*** files can be found here. Let’s have a closer look at it.

Once the mapping is done, we need to evaluate how successful was it. The information about mapping statistics is written to log files of STAR. Let’s open one file:

cat rep1_Log.final.out

Important parameters among others are % of uniquely mapped reads, % of reads mapped to multiple loci and number of splices. Normally for high quality reads and good reference genome unique mapping rate is around 90-95%.

Once the reads are mapped we need to assess gene expression levels. To do this, we need to count how many reads overlap with each gene, and this number will be proportional to the expression level of the gene. This process is called read summarization. Different software can do this task, but STAR has an option to do it, which we have already used. So the read counts are written to repN_ReadsPerGene.out.tab files.

We can inspect the file

head rep1_ReadsPerGene.out.tab

There are 4 columns in the file – gene id, non-stranded counts, forward-stranded counts, reverse-stranded counts. As you remember, our data is reversely strand-specific (TrueSeq protocol), that’s why we need column 4.

NOTES:

Note that read counts are not equal to real gene expression levels (i.e. the number of mRNA molecules in a cell at a given time) but are **proportional** to them.
A very rough example:
Imagine a cell with two genes A and B and they have the same length. At a given time there are 10 and 50 mRNA molecules for each gene, respectively (the proportion between them is 1:5). If we do RNAseq, depending on **sequencing depth** the read counts can vary: if we sequence a little amount of reads it can be 300 and 1505, if we sequence a lot it can be 5020 and 25200, etc. But their proportion will be **very close** to the real proportion of mRNA molecules which was initially 1:5 (if there are no any technical biases). Knowing this we can be confident that our analysis reflects the true expression levels.

Important note: in RNASeq data analysis the **normalization** plays a critical role. There are two main factors which have to be taken into account – **gene/transcript length** and **sequencing depth**.

Longer genes will have more reads than shorter genes, even if in reality they are equally expressed. To account for this, one can divide the number of reads of each gene by the length of the gene. This is called normalization by feature length.

Sequencing depth (or library size) also matters. It denotes the total number of reads that were sequenced in a given sample. As discussed above, sequencing depth varies in different samples, and in order to cancel out this differences one can divide the number of reads of each gene on the total number of reads. This is called library size normalization.

So, for each gene one can do library size and feature length normalization. Once both steps are done, the obtained value is called Read Per Kilobase per Million **(or RPKM)** in case of single-end reads and Fragments Per Kilobase per Million (or **FPKM**) for paired-end data.

Here is the formula:

$$\text{rpkm} = \text{raw counts} \times \frac{1{,}000{,}000\text{reads}}{\text{all reads}} \times \frac{1{,}000\text{base}}{\text{gene length}}$$

We multiply by 1000000 and by 1000 just to make RPKM values easily readable numbers, otherwise they will be very small.

There are many other normalization types in RNAseq apart of R(F)PKM. You can find other normalization types here

Nevertheless, many studies have shown that R(F)PKM values are not suitable for between sample comparisons, that is why many differential gene expression software use raw counts to perform more sophisticated normalization procedures. Anyway, the concept of RPKM and similar normalizations is still widely used of other types of analysis, especially in case of within-sample comparisons.

4. Differential gene expression analysis

Now we have read counts for all samples, so let’s find differentially expressed genes between two conditions of *C. parapsilosis*.

We do:

cd ../DE

The previous mapping example was a toy example where we used only 1 chromosome and subset of reads to speed up the things. In the current folder you can find the real count files that have genes from the whole genome. We will analyze the data with R (Rstudio).

Let’s open Rstudio, load the R script *Dif_expression.R* and inspect it.

The first part of the script loads all files iteratively and saves only the 4th column and gene ids.
We also need to remove the first three raws, since they are not relevant for the analysis.

When the data is ready, we already can perform differential gene expression analysis. We will use DESeq2 tool for that. The software utilizes raw read counts and performs quite complicated statistical approach to infer differentially expressed genes, which we will not cover here.

After running the program, it generates the tables of results, which we have written to *res* variable.
Display the *res* and inspect the table of results.

This file contains the following information:

- baseMean – mean of normalized counts for all samples
- log2FoldChange – log2 fold change between tested conditions
- lfcSE – standard error
- stat – Wald statistic
- pvalue – Wald test p-value
- padj – adjusted p-value

We will consider a gene to be differentially expressed if it expression has changed more than 4 fold (|log2FoldCahnge|>2), and padj<0.01 (these filters can be different depending on how strict you want to be).

Let’s choose up-regulated genes (genes that are expressed higher in biofilm compared to planktonic condition), and write them into a separate file.

5. Functional analysis

To understand what are these genes, we can perform Gene Ontology enrichment analysis to see if there are some groups of genes that are involved in particular function or pathway.

Let's copy the gene names, open the browser and go to http://www.candidagenome.org/, then click GO>GO Term finder, select *Candida parapsilosis*, paste the gene names into the window "Enter Gene/ORF name" and click "Search". After the search we can see that many genes are enriched in transmembrane transport, lipid oxidation, oxidation-reduction, etc.

Conclusions

During this class using RNAseq data we have performed differential gene expression analysis of human pathogen *C. parapsilosis* in pathogenic and non-pathogenic states. We have identified genes and groups of genes that change their gene expression levels during plankton-biofilm transformation, meaning that these genes can play an important role in biofilm formation.

From the technical point of view, we have covered the main steps of RNASeq data analysis, including QC, trimming, mapping, read counting, differential expression and some functional analysis.