RNA-seq processing

DU Bii - N. Servant / M. Deloger

11th March 2020

Contents

Before sarting

RNA-seq experiments is one of the most complexe and diversified NGS application. From a pool of RNAs, one could be interested in measuring the gene expression level, others could be interested in the detection of new isoforms, in the detection of variants (ie. editing), in allele-specific analysis, in the detection of non-coding RNA, in the expression of nascent RNAs, etc.

Here, we will focus on the most classical and standard question ; how to quantify gene expression ? To do so, the idea is to start from raw fastq files and to generate a final table with for each gene and each sample a number of reads (counts level).

How to measure gene expression level ?

As in any sequencing experiment, the experimental design is usually driven by the biological question(s). So far, most of the RNA-seq project relies on paired-end (PE) sequencing using standard Illumina protocols. PE sequencing offers the opportunity to better characterize exon junctions, and are therefore of interest for RNA-seq experiments.

However, it is important to keep in mind that PE sequencing is not require to measure gene expression. For instance, recent protocols based on 3' sequencing give very good results, and are usually enough to quantify gene expression levels. Single-end (SE) sequencing are also frequently used for this purpose. In addition to the sequencing strategy, the number of reads (or sequencing depth) is also important. From our experience, between 10 to 30 Millions of sequenced fragments (in SE or PE) is enough for gene quantification. Other RNA-seq applications such as isoform detection can require up to 100 Millions PE reads.

Finally, last but not least, as gene expression experiments are usually set up to compare different conditions, keep in mind that working with biological replicates is mandatory for statistical analysis.

Overview of appropriate QC

Quality controls are an important step of the analysis which have to be done before any downstream analysis. Here is a list of points which could be interested to check in the context of RNA sequencing :

- rRNA mapping
  Which fraction of reads comes from ribosomal RNAs ? Was the ribo-depletion efficient ?
- Alignment
  How many reads aligned the reference genome ?
- Read distribution
  Where do my reads come from ? CDS ? intronic regions ? 5'/3' UTRs ? others ?
- Gene body coverage
  Do I cover all gene body ? or only the 3' end ?
- Strandness
  What is the sequencing strandness ?
- Sequencing complexity
  Did I sequence enough ? too much ?
- Gene-based saturation
  How many genes are detected as expressed ?
- Duplicates
  What about duplicated and potential PCR artefacts ?
- Exploratory analysis
  How my samples cluster together ?

Look at the `RNAseq_qc_report.html` for example. This report has been generated with the Institut Curie RNA-seq pipeline, derived from the nf-core RNA-seq pipeline.

How to generate a count table at the gene level ?

A count table represents the number of reads mapped per gene/transcripts in an RNA-seq experiment. This is the entry point of many downstream supervised or unsupervised analysis.

There are many different ways to generate a count table, and many tools can be used.
Usually, generating such table requires two mains steps :

1. Aligning the reads on a reference genome
2. Counting how many reads can be assigned to a given gene

The mapping step aims at positioning the sequencing reads on your reference genome. Different tools such as TopHat2[1], HiSat2[2], STAR[3], etc. are still commonly used. In theory, if well configured, these tools should give close results, although their mapping strategy and computational requirements might be different. Of note, recent methods/tools based on pseudo-mapping approaches such as Salmon[4], Kallisto[5], Rapmap[6], etc.

[1]Kim D., Pertea G., Trapnell C. (2013) TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. Genome Biology, 14(4).

[2]Kim D, Langmead B and Salzberg SL. (2015) HISAT: a fast spliced aligner with low memory requirements. Nature Methods

[3]Dobin A., Davis C.A., Schlesinger F. et al. (2013) STAR: ultrafast universal RNA-seq aligner, Bioinformatics, 29(1):15–21,

[4]Patro, R., Duggal, G., Love, M. I. et al. (2017). Salmon provides fast and bias-aware quantification of transcript expression. Nature Methods.

[5]Nicolas L Bray N.L., Pimentel H., Melsted P. et al. (2016) Near-optimal probabilistic RNA-seq quantification, Nature Biotechnology 34, 525–527

[6]Srivastava A., Sarkar H., Gupta N. et al. (2016) RapMap: a rapid, sensitive and accurate tool for mapping RNA-seq reads to transcriptomes, Bioinformatics. 32(12)

can also be used to quantify the gene expression from raw RNA-seq data (see *Bray et al. 2016* [7]).

Once the data are mapped on the genome, several tools can be used to count and assign reads to a given gene (exons).
Among the most popular tools, HTSeqCount[8] or FeatureCounts[9] are frequently used. Note that for this step, it is crucial to have details on the protocol used to generate the samples, and especially if the protocol was **stranded** or not.
This step also requires some gene annotations. Databases such as Ensembl, Refseq, or Gencode can be used. They all contain the most common coding genes but they also all have their own specifities.

Hands-on

In order to run the hands-on, first connect to the computational cluster using ssh. Do not forget than you need to submit the job to the cluster (and to write scripts if necessary).

Loading modules

The tools that we will use are all available through the `module` system.

```
module load star
module load subread
module load kallisto
```

Data

To speed up the computation, we will work on a small Mouse RNA-seq data (SRR1106775) which has been down-sampled to 1 million reads.

Here the annotation files that we will use:

```
## Mus musculus Gencode GTF
gtf="/shared/projects/dubii2020/data/rnaseq/gtf/gencode.vM22.annotation.gtf"
trs="/shared/projects/dubii2020/data/rnaseq/kallisto/gencode.vM22.transcripts.fa"

## Mus musculus mm10 STAR index
index="/shared/bank/mus_musculus/mm10/star-2.7.2b/"

## Toy dataset - SRR1106775
r1="/shared/projects/dubii2020/data/rnaseq/rawdata/SRR1106775-1M_1.fastq.gz"
r2="/shared/projects/dubii2020/data/rnaseq/rawdata/SRR1106775-1M_2.fastq.gz"
```

Mapping with STAR

Here, we will use the STAR mapper which is one of the most use software for RNA-seq reads mapping. Before starting, you can first have a look at the STAR manual

[7] Bray N.L. et al. (2016) Near-optimal probabilistic RNA-seq quantification. Nature Biotech., 34(5):525–527.

[8] Anders S., Pyl T.P., Huber W. (2015) HTSeq - A Python framework to work with high-throughput sequencing data. Bioinformatics 31(2):166-9

[9] Liao Y, Smyth GK and Shi W. (2014) featureCounts: an efficient general-purpose program for assigning sequence reads to genomic features. Bioinformatics, 30(7):923-30

Genome index

Indexing the genome is mandatory to run the alignment. However, it is also time and memory consuming.

```
## /!\ DO NOT RUN /!\
STAR \\
  --runMode genomeGenerate \\
  --runThreadN ${cpus} \\
  --genomeDir star/ \\
  --genomeFastaFiles ${fasta}
```

Note that the indexing step can be run with or without the `--sjdbGTFfile ${gtf}` option. STAR will extract splice junctions from this file and use them to greatly improve accuracy of the mapping. It is therefore highly recommanded to run STAR with this annotation. However, since version 2.4.1a, this option can be specified during the mappin step.

Genome mapping

STAR has a lot of options which can be tuned according to the downstream analysis (isoform detection, gene fusion, etc.). Here, we will use the parameters proposed by the gencode consortium.

```
starOpts="--outSAMmultNmax 1 --outFilterMismatchNmax 999 --outFilterMismatchNoverLmax 0.04 --outSAMprima

odir="./mapping"
prefix=$(basename $r1 | sed -e 's/.fastq.gz//')
cpus=4

mkdir -p ${odir}
STAR \\
  --genomeDir ${index} \\
  --sjdbGTFfile ${gtf} \\
  --readFilesIn ${r1} ${r2}  \\
  --runThreadN ${cpus} \\
  --runMode alignReads \\
  --outSAMtype BAM Unsorted  \\
  --readFilesCommand zcat \\
  --outFileNamePrefix ${odir}/${prefix}  \\
  --quantMode GeneCounts \\
  --outSAMattrRGline ID:${prefix} SM:${prefix} LB:Illumina PL:Illumina  \\
  ${starOpts}
```

Counts

Once the reads are mapped on the reference genome, the next step will be to count how many reads overlap gene annotations. To do so, most of the tools require the strandness parameter (forward/reverse/unstranded sequencing), and will therefore count the reads according to the genes' orientation.

`FeatureCounts` is part of the `subread` and is widely used to generate count matrix. Before starting, have a look to the help message.

```
bam="./mapping/SRR1106775-1M_1Aligned.out.bam"

featureCounts \
  -T ${cpus} \
  -a ${gtf} \
```

```
  -o counts.csv \
  -p \
  -s 0 \
  ${bam}
```

Pseudo-mapping with Kallisto

Finally, recent methods have been proposed a couple of years ago based on pseudo-mapping algorithm. Contrary to standard mapping strategy, these approach are based on kmer search and do not rely on a strick alignment on a reference genome.

Compared to standard mppaing/counting approach, these algorithm are usually much faster, and diectly generate a count table starting from the raw fastq files.

Here, we propose to use the `Kallisto`[10] tool. Before starting, Kallisto requires a dedicated index file which can be generate for each transcriptom as follow.

```
module load kallisto

kallistoIndex="/shared/projects/dubii2020/data/rnaseq/kallisto/gencode.vM22.transcripts_index"

## /!\ DO NOT RUN /!\
kallisto index -i ${kallistoIndex} ${trs}
```

Then, the gene counts will be directly calculated from the index and the fastq files.

```
kallisto quant \
    -i ${kallistoIndex} \
    -t ${cpus} \
    -b 100 \
    --genomebam \
    -g ${gtf} \
    -o ${odir} \
    ${r1} ${r2}
```

Going further

Gene counts

In the previous examples, we used 3 different methods to generate the gene count tables. Open a R session, load the count tables and compare them. Is there any difference ? Are the counts well correlated ?

Visualisation

Looking at the data by eyes is the best way to understand what you are doing. The IGV browser is widely used to visualize NGS data. An online version of IGV is available at https://igv.org/app/.

References

[10] Nicolas L Bray N.L., Pimentel H., Melsted P. et al. (2016) Near-optimal probabilistic RNA-seq quantification, Nature Biotechnology 34, 525–527