RNA-seq Data Analysis

Luigi Grassi < *lg490@medschl.cam.ac.uk* >
Guillermo Parada < *gp7@sanger.ac.uk* >

Introduction to RNA-seq data analysis
June, 2018

General information

The following standard icons are used in the hands-on exercises to help you locating:

ℹ Important Information

▤ General information / notes

👣 Follow the following steps

Q Questions to be answered

! Warning – Please take care and read carefully

✸ Optional Bonus exercise

✸ Optional Bonus exercise for a champion

Resources used

FastQC

HISAT2

stringtie

gffcompare

samtools

IGV genome browser

STAR RNA-seq aligner

Salmon

Read mapping and counting with STAR

STAR is a powerful RNA-seq aligner, designed to analyse the ENCODE transcriptome RNA-seq datasets. It is fast program with a high alignment sensitivity and precision. The software is already installed on your computer.

The documentation is available at this web address: https://github.com/alexdobin/STAR/raw/master/doc/STARmanual.pdf

Preparing the genome index

In order to align our reads we have to index the reference genome, as we made for *Hisat2*. Read the sections 1.2 and 2.1 of the manual to see how you should do it.

You do not need to install *STAR* or provide any advanced options.

If you are not in the `RNA-seq` directory, please change your working directory using `cd`.

Due to time constraints we will build the index only for one chromosome of the zebrafish genome. For this we need the chromosome sequence in fasta format.

Create a new directory `STAR_chr2` using the mkdir command.

Starting from the command we used to download the zebrafish chromosome 4 from the ensembl database, modify and run it, to download the chromosome 2 fasta file and unzip it in the directory `STAR_chr2`.

Then, generate the genome index using the following parameters:

- Number of threads: 8
- Genome dir: the genome directory you just created (STAR_chr2).
- Genome fasta file: The chr FASTA file of chromosome 2 you downloaded.
- Sjdb GTF file: The transcriptome annotation file from ENSEMBL (ends in `.gtf`, contained in the folder annotation) you used also in the previous session.
- Sjdb overhang: Your read length - 1 (Hint: use *FastQC* to check the read length of one of the fastq files in the `data` folder!).

Step 2: Run the alignment

Now you can align the fastq files onto the genome using the index made for the whole genome contained in the directory genome/STAR_genome. The commands for this are explained in section 3.1 of the manual.

Align the fastq files of the SRR1048063 sample to the genome, specifying the following parameters:

- Number of threads: 8
- Genome dir: `genome/STAR_genome`
- Fastq files: The two fastq files from the SRR1048063 sample, contained in the folder `data` (Remember: this is paired-end data, so you need to provide the file names of both files at the same time!)
- Specify that the fastq file are zipped (look in the manual the option `-readFilesCommand`)
- Add the `--quantMode GeneCounts` option to generate a text file reporting the read counts per gene (see section 7 of the STAR manual)
- Add the `outSAMtype` parameter to generate a BAM file sorted by coordinate (see section 4.3 of the STAR manual)

Questions

1. The read file `ReadsPerGene.out.tab` is made of 4 columns: the first reporting the gene ID and the other three with read counts corresponding to different strandedness option. Our RNA-seq library was "not strand specific": do you have confirmation of this by the read counts in the three strandedness options?

   ---

2. Look at the `gene_count_matrix.csv` file you produced in the previous session and compare the counts estimated for the genes "ENSDARG00000100294"; "ENSDARG00000074362", "ENSDARG00000078585" and "ENSDARG00000012789" .

   Hint: Use the unix `sort` command to sort the files according to the column of interest. With the `grep` command you can look for one specific gene

3. Do you find discrepancies between the results in `gene_count_matrix.csv` and the ones in `ReadsPerGene.out.tab` ?

4. Have a look at the log file generated by *STAR* called `Log.final.out`. How many reads could *STAR* map to the genome? How does that compare to *HISAT2*? Hint: You can find mapping statistics from Hisat2 in the corresponding summary file file generated in the same directory of the alignment

---

TRANSCRIPT QUANTIFICATION WITH SALMON

Salmon is a tool for quantifying the expression of transcripts using RNA-seq data. It is based on a new algorithm that couples the concept of quasi-mapping with a two-phase inference procedure, providing accurate expression estimates very quickly and using little memory. It quantifies the expression of the transcripts from a given annotation, so it is not able to identify non annotated genes and transcripts.

The documentation of the tool is available at https://salmon.readthedocs.io/en/latest/salmon.html

Salmon is able to quantify transcript expression by using a quasi-mappings algorithm. Quasi-mappings are mappings of reads to transcript positions that are computed without performing a base-to-base alignment of the read to the transcript. This approach is typically much faster to compute than traditional (or full) alignments, and can sometimes provide superior accuracy by being more robust to errors in the read or genomic variation from the reference sequence.

Preparing the transcriptome index

In order to run Salmon in a quasi-mapping-based mode, you first have to build an Salmon index for your transcriptome.

Visualise the fasta file `Danio_rerio.GRCz10.cdna_noversion.all.fa` present in the directory `annotation`. It contains all the set of transcripts you are going to quantify.

As first step create a directory inside the genome directory named Drer_SalmonQuasi.

Then run the Salmon indexer:

```
salmon index -t annotation/Danio_rerio.GRCz10.cdna_noversion.all.fa -i genome/Drer_SalmonQuasi
--type quasi -k 31 -p 8
```

Questions

1. At the end of the index creation have a look at the log file created in the `Drer_SalmonQuasi` and look at the warnings reported, did we set the right parameters for indexing the transcriptome?
2. We used the parameter `-k 31` to set the k-mer length to 31, do you think it was a good choice?

Quantifying transcript expression

Salmon can align the reads using the quasi-mappings algorithm or the SMEM-based mapping algorithm (the original lightweight-alignment method used by Salmon). The same quant command will work with either index (quasi-mapping or SMEM-based), and Salmon will automatically determine the type of index being read and perform the appropriate lightweight mapping accordingly.

Create a directory named `salmon_quant` with the unix command mkdir

Use the fastq files of the SRR1048063 sample to quantify transcript expression, specifying the following parameters:

- Number of threads: 8
- transcripts_index: the directory where you saved the quasi-mapping index
- Fastq files: The two fastq files from the SRR1048063 sample, contained in the folder `data` (Notice that you have gzipped files, look on the Salmon documentation how to use them on-the-fly)
- library type: inward unstranded

Explore the results saved in the `quant.sf` file.

Bonus exercise

Using the transcript quantification results from Salmon and stringtie look at the transcripts present in both analyses and test their correlation. To do it we could use the R language.

```r
# Set some options first
options(stringsAsFactors = F)
setwd("/home/participant/Course_Materials/RNA-seq")
# Read the `quant.sf` file into R
salmonres <- read.delim("salmon_quant/quant.sf", header = TRUE, sep="\t")
head(salmonres)
salmonres <-subset(salmonres,select=c(Name, NumReads))
colnames(salmonres)<-c("Transcript","Salmon_counts")
#Read the stringtie `transcript_count_matrix.csv` file into R
strgtieres <- read.delim("transcript_count_matrix.csv", header = TRUE, sep=",")
head(strgtieres)
strgtieres<-subset(strgtieres,select=c(transcript_id, SRR1048063))
colnames(strgtieres)<-c("Transcript","Stringtie_counts")
merged_quant<-merge(salmonres, strgtieres)
plot(merged_quant$Salmon_counts,merged_quant$Stringtie_counts)
cor.test(merged_quant$Salmon_counts,merged_quant$Stringtie_counts,
    method=c("spearman"))
```