Introduction to RNA-seq Data Analysis

Myrto Kostadima
Romina Petersen < *rp520@medschl.cam.ac.uk* >

High throughput sequencing workshop
University of Cambridge
February, 2016

General information

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

 Important Information

 General information / notes

 Follow the following steps

 Questions to be answered

 Warning – Please take care and read carefully

 Optional Bonus exercise

 Optional Bonus exercise for a champion

Resources used

Tophat: http://ccb.jhu.edu/software/tophat/index.shtml

Cufflinks: http://cole-trapnell-lab.github.io/cufflinks/

Samtools: http://samtools.sourceforge.net/

IGV genome browser: http://www.broadinstitute.org/igv/

HTSeq-count: http://www-huber.embl.de/users/anders/HTSeq/doc/count.html

DESeq2: http://bioconductor.org/packages/release/bioc/html/DESeq2.html

DEXSeq: http://bioconductor.org/packages/release/bioc/html/DEXSeq.html

STAR: https://github.com/alexdobin/STAR

Original data can be found here: http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE18508

Introduction

The goal of this hands-on session is to perform some basic tasks in the downstream analysis of RNA-seq data. We will start from RNA-seq data aligned to the zebrafish 

genome using *Tophat*. We will perform transcriptome reconstruction using *Cufflinks* and we will compare the gene expression between two different conditions in order to identify differentially expressed genes using *Cuffdiff*.

Prepare environment

We will use a dataset derived from sequencing of mRNA from *Danio rerio* embryos in two different developmental stages. Sequencing was performed on the Illumina platform and generated 76bp paired-end sequence data using poly-(A)+ selected RNA. Due to the time constraints of the practical we will only use a subset of the reads.

The data files are contained in the subdirectory called data and are the following:

- `2cells_1.fastq` and `2cells_2.fastq`: these files are based on RNA-seq data of a 2-cell zebrafish embryo, and
- `6h_1.fastq` and `6h_2.fastq`: these files are based on RNA-seq data of zebrafish embryos 6h post fertilisation.

Open the Terminal.

First, go to the folder, where the data are stored.

```
cd ~/Desktop/RNA-seq
```

Check that the `data` folder contains the above-mentioned files by typing:

```
ls -l data
```

Note that all commands that are given in this tutorial should be run within the main folder `RNA-seq`.

Alignment

There are numerous tools performing short read alignment and the choice of aligner should be carefully made according to the analysis goals/requirements. Here we will use *Tophat*, a widely used ultrafast aligner that performs spliced alignments.

Tophat is based on *Bowtie* to perform alignments and uses an indexed genome for the alignment to keep its memory footprint small. We have already seen how to index the genome (see Alignment hands-on session), therefore for time purposes we have already generated the index for the zebrafish genome and placed it under the `genome` subdirectory.

Tophat has a number of parameters in order to perform the alignment. To view them all type

```
tophat --help
```

The general format of the tophat command is:

```
tophat [options]* <index_base> <reads_1> <reads_2>
```

Where the last two arguments are the `.fastq` files of the paired end reads, and the argument before is the basename of the indexed genome.

Like with *Bowtie* before you run Tophat, you have to know which quality encoding the fastq formatted reads are in.

Questions

1. Can you tell which quality encoding our fastq formatted reads are in? _____

   Hint: Look at the first few reads of the file `data/2cells_1.fastq` by typing:

```
head -n 20 data/2cells_1.fastq
```

2. Compare the quality strings with the table found at http://en.wikipedia.org/wiki/FASTQ_format#Encoding

Some other parameters that we are going to use to run *Tophat* are listed below:

```
-g maximum number of multihits allowed. Short reads are likely to map to
   more than one locations in the genome even though these reads can
   have originated from only one of these regions. In RNA-seq we allow
   for a restricted number of multihits, and in this case we ask Tophat
   to report only reads that map at most onto 2 different loci.
-p use these many threads to align reads
--library-type before performing any type of RNA-seq analysis you need
   to know a few things about the library preparation. Was it done using
   a strand-specific protocol or not? If yes, which strand? In our data
   the protocol was NOT strand specific.
-J improve spliced alignment by providing *Tophat* with annotated splice
   junctions. Pre-existing genome annotation is an advantage when
   analysing RNA-seq data. This file contains the coordinates of
   annotated splice junctions from Ensembl. These are stored under the
   sub-directory annotation in a file called `ZV9.spliceSites`.
-o this specifies in which subdirectory *Tophat* should save the output
   files. Given that for every run the name of the output files is the
   same, we specify different folders for each run.
```

Now we will proceed with the alignment of the paired-end data for the two different conditions. Due to the fact that the spliced alignment takes long even for a subset of the reads, we will only align one of the two datasets. The other one has been already aligned for you.

Questions

1. Given that we used the following command to align the `2cells` dataset:

```
tophat --solexa-quals -g 2 -p 8 --library-type fr-unstranded -j
    annotation/Danio_rerio.Zv9.66.spliceSites -o tophat/ZV9_2cells
    genome/ZV9 data/2cells_1.fastq data/2cells_2.fastq
```

   What is the command to align the '6h' dataset? Run this command on the terminal. _____

_____

Note: You will have to change the input fastq files and the output folder. If you don't change the output folder, then these results will overwrite the ones for the `2cells` dataset.

The alignment will take approximately 5 minutes. In the meantime please move on with the practical and we will get back to the terminal once the alignment is done.

We will firstly look at some of the files produced by *Tophat*. For this please open the RNA-seq folder which can be found on your `/home/htstraining##/workspace/`. Click on the tophat subfolder and then on the folder called `ZV9_2cells`.

Tophat reports the alignments in a BAM file called `accepted_hits.bam`. Among others it also creates a `junctions.bed` files that stores the coordinates of the splice junctions present in your dataset, as these have been extracted from the spliced alignments.

Now we will load the BAM file and the splice junctions onto *IGV* to visualise the alignments reported by *Tophat*.

In order to launch *IGV* type on the terminal

```
igv &
```

Ignore any warnings and when it opens you have to load the genome of interest.

On the top left of your screen choose from the drop down menu *Zebrafish (Zv9)*. Then in order to load the desire files go to:

File –> Load from File

On the pop up window navigate to **home –> participant –> Desktop –> RNA-seq –> tophat –> ZV9_2cells** folder and select the file `accepted_hits.sorted.bam`.

Once the file is loaded right-click on the name of the track on the left and choose Rename Track. Give the track a meaningful name.

Follow the same steps in order to load the `junctions.bed` file from the same folder.

Finally following the same process load the Ensembl annotation `Danio_rerio.Zv9.66.gtf` stored under folder annotation under the `RNA-seq` folder.

On the top middle box you can specify the region you want your browser to zoom. Type `chr12:20,270,921- 20,300,943`.

Right-click on the name of the Ensembl track and choose **Expanded**.

Questions

1. Can you identify the splice junctions from the BAM file? _____
   _____
2. Are the junctions annotated for CBY1 consistent with the annotation? _____
   _____
3. Are all annotated genes (both from RefSeq and Ensembl) expressed? _____
   _____
   _____

Once you are done with the questions above, please close *IGV*.

We already know that in order to load a BAM file onto IGV we need to have this file sorted by genomic location and indexed. Here's a reminder of the commands to perform these: Sort the BAM file using samtools:

```
samtools sort [bam file to be sorted] [prefix of sorted bam output file]
```

Index the sorted file.

```
samtools index [sorted bam file]
```

Once *Tophat* finishes running, sort the output `.bam` file and then index the sorted `.bam` file using the information above to guide you.

Launch *IGV* again. This time we will change the configuration of *IGV* as described below. Go to:

View –> Preferences

Click on the tab **Alignments** and further down on the window tick the option **Show junction track**. Then type 5 in the box of the **Min junction coverage**. Click **OK**.

Finally, load the alignments for the two datasets onto IGV following the steps described above. Please, load the Ensembl annotation as well.

Isoform expression and transcriptome assembly

There are a number of tools that perform reconstruction of the transcriptome and for this workshop we are going to use *Cufflinks*. Cufflinks can do transcriptome assembly either ab initio or using a reference annotation. It also quantifies the isoform expression in FPKMs.

Cufflinks has a number of parameters in order to perform transcriptome assembly and quantification. To view them all type

```
cufflinks --help
```

We aim to reconstruct the transcriptome for both samples by using the Ensembl annotation both strictly and as a guide. In the first case *Cufflinks* will only report isoforms that are included in the annotation, while in the latter case it will report novel isoforms as well.

A reminder from the presentation this morning that FPKM stands for Fragments Per Kilobase of exon per Million fragments mapped.

The annotation from Ensembl of *Danio rerio* is stored under the folder annotation in a file called `Danio_rerio.Zv9.66.gtf`.

The general format of the cufflinks command is:

```
cufflinks [options]* <aligned_reads.(sam/bam)>
```

Where the input is the aligned reads (either in SAM or BAM format).

Some of available parameters of Cufflinks that we are going to use to run *Cufflinks* are listed below:

```
-o output directory
-G tells *Cufflinks* to use the supplied annotation strictly in order to
   estimate isoform annotation.
-b instructs *Cufflinks* to run a bias detection and correction
   algorithm which can significantly improve accuracy of transcript
   abundance estimates. To do this *Cufflinks* requires a multi-fasta
   file with the genomic sequences against which we have aligned the
   reads.
-u tells *Cufflinks* to do an initial estimation procedure to more
   accurately weight reads mapping to multiple locations in the genome
   (multi-hits).
--library-type see *Tophat* parameters.
-p see *Tophat parameters.
```

In the terminal type:

```
cufflinks  -o cufflinks/ZV9_2cells_gff -G
   annotation/Danio_rerio.Zv9.66.gtf -p 8 -b
   genome/Danio_rerio.Zv9.66.dna.fa -u --library-type fr-unstranded
   tophat/ZV9_2cells/accepted_hits.bam
```

Questions

1. Given the previous command for `2cells` dataset, how would you run *Cufflinks* for the other dataset `6h`? Run this command on the terminal. Don't forget to change the output folder. Otherwise the second command will overwrite the results of the previous run. _____

Take a look at the output folders that have been created. The results from *Cufflinks* are stored in 4 different files named:

- `genes.fpkm_tracking`
- `isoforms.fpkm_tracking`
- `skipped.gtf`
- `transcripts.gtf`

Here's a short description of these files:

- `genes.fpkm_tracking`: contains the estimated gene-level expression values.
- `isoforms.fpkm_tracking`: contains the estimated isoform-level expression values.
- `transcripts.gtf`: This GTF file contains *Cufflinks* assembled isoforms

The complete documentation can be found at: http://cole-trapnell-lab.github.io/cufflinks/cufflinks/#cufflinks-output-files

Now in order to perform guided transcriptome assembly (transcriptome assembly that reports novel transcripts as well) we will have to change the `-G` option of the previous command. In its place we will use the `-g` option that tells *Cufflinks* to assemble the transcriptome using the supplied annotation as a guide and allowing for novel transcripts.

Questions

Due to time constraints, please do not run the command for guided transcriptome analysis. Instead, write the **cufflinks** command you would use to perform a guided transcriptome assembly for the 2cells dataset in the space below. _____

Performing the guided transcriptome analysis for the `2cells` and `6h` data sets would take 15-20min each. Therefore, we have pre-computed these for you and have the results under subdirectories: **cufflinks**`/ZV9_2cells` and **cufflinks**`/ZV9_6h`.

Go back to the *IGV* browser and load the file `transcripts.gtf` which is located in the subdirectory **cufflinks**`/ZV9_2cells/`. Rename the track into something meaningful.

This file contains the transcripts that *Cufflinks* assembled based on the alignment of our reads onto the genome.

Questions

In the search box type ENSDART00000082297 in order for the browser to zoom in to the gene of interest. Compare between the already annotated transcripts and the ones assembled by *Cufflinks*. Do you observe any difference?

Differential Expression

One of the stand-alone tools that perform differential expression analysis is *Cuffdiff*. We use this tool to compare between two conditions; for example different conditions could be control and disease, or wild-type and mutant, or various developmental stages. In our case we want to identify genes that are differentially expressed between two developmental stages; a 2 cell embryo and 6h post fertilization.

The general format of the **cuffdiff** command is:

```
cuffdiff [options]* <transcripts.gtf>
   <sample1_replicate1.sam[,...,sample1_replicateM]>
   <sample2_replicate1.sam[,...,sample2_replicateM.sam]>
```

Where the input includes a `transcripts.gtf` file, which is an annotation file of the genome of interest, and the aligned reads (either in SAM or BAM format) for the conditions.

Some of the Cufflinks options that we will use to run the program are:

```
-o output directory,
-L labels for the different conditions,
-T tells *Cuffdiff* that the reads are from a time series experiment,
-b, -u, --library-type: same as above in *Cufflinks*.
```

To run cufdiff type on the terminal type:

```
cuffdiff -o cuffdiff/ -L ZV9_2cells,ZV9_6h -T -b
   genome/Danio_rerio.Zv9.66.dna.fa -u --library-type fr-unstranded
   annotation/Danio_rerio.Zv9.66.gtf tophat/ZV9_2cells/accepted_hits.bam
   tophat/ZV9_6h/accepted_hits.bam
```

In the command above we have assumed that the folder where you stored the results of *Tophat* for dataset `6h` was named `ZV9_6h`. If this is not the case please change the previous command accordingly otherwise you will get an error.

We are interested in the differential expression at the gene level. The results are reported by `Cuffdiff` in the file `cuffdiff/gene_exp.diff`.

Look at the first few lines of the file using the following command:

```
head -n 20 cuffdiff/gene_exp.diff
```

We would like to see which are the most significantly differentially expressed genes. Therefore we will sort the above file according to the q value (corrected p value for multiple testing). The result will be stored in a different file called `gene_exp_qval.sorted.diff`.

```
sort -t$'\t' -g -k 13 cuffdiff/gene_exp.diff >
   cuffdiff/gene_exp_qval.sorted.diff
```

Look again at the first few lines of the sorted file by typing:

```
head -n 20 cuffdiff/gene_exp_qval.sorted.diff
```

Copy the Ensembl identifier of one of these genes. Now go back to the *IGV* browser and paste it in the search box. Look at the raw aligned data for the two datasets.

Questions

Do you see any difference in the gene coverage between the two conditions that would justify that this gene has been called as differentially expressed? _____

Note that the coverage on the Ensembl browser is based on raw reads and no normalisation has taken place contrary to the FPKM values.

Functional Annotation of Differentially Expressed genes

After you have performed the differential expression analysis you are interested in identifying if there is any functionality enrichment for your differentially expressed genes.

Open a web browser and go to the following URL http://david.abcc.ncifcrf.gov/

On the left side click on **Functional Annotation**. Then click on the **Upload** tab. Under the section **Choose from File**, click **Choose File** and navigate to the `cuffdiff` folder. Select the file called `globalDiffExprs_Genes_qval.01_top100.tab`. Under *Step 2* select *ENSEMBL_GENE_ID* from the drop-down menu. Finally select *Gene List* and then press *Submit List*.

Click on *Gene Ontology* and then click on the *CHART* button of the *GOTERM_BP_ALL* item.

Questions

Do these categories make sense given the samples we're studying? _____

Browse around DAVID website and check what other information are available.

CONGRATULATIONS! You've made it to the end of the practical.

We hope you enjoyed it!

Don't hesitate to ask any questions and feel free to contact us any time (email addresses on the front page).

Bonus Exercise I

During the alignment step of the practical you set the `-j` parameter to a file that contains all the annotated splice junctions. How can we generate this file for the mouse genome?

- Google `ensembl mouse GTF`, go to `FTP Download - Ensembl` and then download the gene annotation file (GTF format) for the mouse genome. Hint: Please do NOT download the `abinitio` GTF file.
- Store it under the `RNA-seq/annotation` folder
- Decompress the GTF file using `gzip -d` followed by the GTF file

- Use the `gtf_juncs` command to extract the splice junctions from the decompressed GTF file and store the output under the annotation folder in a file called: `mouse.juncs`

Bonus Exercise II - Read mapping with STAR

STAR is a new aligner for RNA-seq, described here: https://github.com/alexdobin/STAR/

The software is already installed on your computers.

The documentation for *STAR* is available here: https://github.com/alexdobin/STAR/raw/master/doc/STARmanual.pdf

As with *Tophat*, we first need to index the reference genome. Have a look at section 1.2 and 2.1 of the manual to see how this is done. You do not need to install *STAR* or provide any advanced options.

Step 1: Prepare the genome index

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

Create a new directory called `STAR_genome` using mkdir. This will be your genome directory.

Then, generate the genome index using the following parameters:

- Number of threads: 4
- Genome dir: The genome directory you just created.
- Genome fasta file: The genome FASTA file (ends in `.fa`, contained in the folder genome)
- Sjdb GTF file: The full genome annotation file from ENSEMBL (ends in `.gtf`, contained in the folder annotation)
- Sjdb overhang: Your read length - 1 (Hint: use *FastQC* to check the read length in one of the fastq files in the `data` folder!)

Step 2: Run the alignment

Now you can align the fastq files to the genome. The commands for this are explained in section 3.1 of the manual.

Now align the pair of files from the 2cells sample to the genome, using the following parameters:

- Number of threads: 4
- Genome dir: `STAR_genome`
- Fastq files: The two fastq files from the 2cells 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!)
- Add the `outSAMtype` parameter to generate a BAM file sorted by coordinate (see section 4.3)

Questions

1. 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 *Tophat*?

   Hint: You can find mapping statistics from Tophat using `samtools flagstat` on the `tophat/ZV9_2cells/accepted_hits.bam` file.

2. Can you think of a reason for the difference in number of aligned reads? How many initial reads does the 2cells dataset contain?

   Hint: Run the following command instead on the `tophat/ZV9_2cells/accepted_hits.bam` file:

   ```
   samtools view tophat/ZV9_2cells/accepted_hits.bam | cut -f 1 | sort -u |
       wc -l
   ```

Bonus Exercise III

In the transcriptome assembly part we discussed how multi-mapping might has an effect on the transcriptome assembly of novel transcripts. Let us re-do this part of the analysis using only uniquely mapped reads. During this exercise we will also learn how to extract splice junctions in a .BED file from a BAM file.

- Filter the `2cells` BAM file to only contain uniquely aligned reads
  - Hint: use `samtools view` to keep only those with a mapping score equal to 255

  - Check the samtools view manual on how to do that.
  - Store the output in a file called: `2cells_unique.bam`

- Remove duplicates from the `2cells_unique.bam` file using the `MarkDuplicates` application from picard tools.
  - Hint: To call picard tools on this computer use `picard-tools MarkDuplicates` and add any options needed for the task you want to perform. By default this function only marks duplicates. However, what we want is to remove them too. Find the right option in the tool manual to do so.

- Store the output in a file called `2cells_unique_rmdup.bam`
- To extract the splice junctions from the `2cells_unique_rmdup.bam` follow the suggested solution from user `brentp` in this thread: https://www.biostars.org/p/12626/
  - Attention: In our case we start from a BAM file rather than a SAM file. Hence the first samtools command suggested needs to be changed so as to convert BAM to SAM. Keep the `-h` option because we also need the header of the file (see below why).
  - Attention: The awk command suggested in his answer should be changed to `awk ($6 ~ /N/ || $0 ~ /^@/)`

    This will ensure that you also keep the header of the BAM file, which is essential when you'd like to convert SAM to BAM. Otherwise samtools will give you the following error.

    [E::sam_parse1] missing SAM header

    [W::sam_read1] parse error at line 1

    [main_samview] truncated file.

- Run the guided Cufflinks transcriptome assembly on this new BAM file
- Compare the two `transcripts.gtf` (the one from `cufflinks/2cells/transcripts.gtf` and the one you just generated) using `cuffcompare`.
- Load them both on IGV and have a look at the results. Do you observe any differences in the transcriptome assembly around the `ENSDART00000082297` transcript?

References

1. Trapnell, C., Pachter, L. & Salzberg, S. L. TopHat: discovering splice junctions with RNA-Seq. Bioinformatics 25, 1105–1111 (2009).

2. Trapnell, C. et al. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat. Biotechnol. 28, 511–515 (2010).
3. Langmead, B., Trapnell, C., Pop, M. & Salzberg, S. L. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol. 10, R25 (2009).
4. Roberts, A., Pimentel, H., Trapnell, C. & Pachter, L. Identification of novel transcripts in annotated genomes using RNA-Seq. Bioinformatics 27, 2325–2329 (2011).
5. Roberts, A., Trapnell, C., Donaghey, J., Rinn, J. L. & Pachter, L. Improving RNA-Seq expression estimates by correcting for fragment bias. Genome Biol. 12, R22 (2011).
6. Young MD, Wakefield MJ, Smyth GK and Oshlack A. "Gene ontology analysis for RNA-seq: accounting for selection bias." Genome Biology, 11, pp. R14 (2010).
7. Dobin A, et al. STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29, 15-21 (2012).