Next Generation Sequencing Workshop

RNA-seq Hands-on Exercise

Myrto Kostadima, EMBL-EBI (kostadim@ebi.ac.uk)
Remco Loos, EMBL-EBI (remco@ebi.ac.uk)

EBI, October 2012

General information

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

 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://tophat.cbcb.umd.edu/

Cufflinks: http://cufflinks.cbcb.umd.edu/

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

BEDTools: http://code.google.com/p/bedtools/

UCSC tools: http://hgdownload.cse.ucsc.edu/admin/exe/

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

DAVID Functional Analysis: http://david.abcc.ncifcrf.gov/

Original Data can be found here:

http://www.ebi.ac.uk/ena/data/view/ERR022484 and

http://www.ebi.ac.uk/ena/data/view/ERR022485

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.

Prepare the 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 polyA 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 fertilization.

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.

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
```

And then 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.
- **`--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.

Due to time constraints the alignment has already been performed for you and the results are stored under the `tophat` subdirectory. If you look in this subdirectory you will find another two folders one for each

sample, called `ZV9_2cells` and `ZV_6h`. Look at the Appendix for further information on the alignment step.

The alignments are stored in the file `tophat/<output dir>/accepted_hits.bam` in BAM (compressed binary version of the SAM format that stands for Sequence Alignment/Map). For more information regarding the SAM format please see: http://samtools.sourceforge.net/SAM1.pdf

We will first look at some of the files produced by *Tophat*. For this please open the RNA-seq folder which can be found on your Desktop. 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 visualize the alignments reported by *Tophat*.

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]
```

These commands have already been performed on the BAM files.

In order to launch IGV type

```
igv.sh &
```

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

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

Can you identify the splice junctions from the BAM file?

Are the junctions annotated for CBY1 consistent with the annotation?

Are all annotated genes (both from RefSeq and Ensembl) expressed?

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.

A reminder from the presentation this morning that FPKM stands for **F**ragments **P**er **K**ilobase of exon per **M**illion fragments mapped.

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.

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.

In the terminal type:

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

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` and `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://cufflinks.cbcb.umd.edu/manual.html#cufflinks_output

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.

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.

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.

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.

We have already performed differential expression analysis genome-wide for you and the results are stored within the cuffdiff folder in the file '`diffExprs_Genes_qval.01.txt`'. This file contains only significantly differential expressed genes using a cutoff of 0.01 for the q-value. From this file we extract only the first column, which contains the Ensembl gene identifiers of the differentially expressed genes, and store the top 100 based on their q-value in a file called `globalDiffExprs_Genes_qval.01_top100.tab` under the `cuffdiff` directory.

```
cut –f 1 cuffdiff/diffExprs_Genes_qval.01.txt \
```

```
| head -n 100 > \
cuffdiff/globalDiffExprs_Genes_qval.01_top100.tab
```

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.

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).

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).

Appendix

Alignment

To perform the alignment of the sequencing reads to the genome we used the following command for each of the different datasets:

```
tophat --solexa-quals \
       -g 2 \
       --library-type fr-unstranded \
       -j annotation/ZV9.spliceSites \
       -o <output dir> \
       genome/ZV9 \
       <fastq file – left mate> \
       <fastq file – right mate>
```