Table of contents

expected learning outcome

To understand the basics of RNA-Seq data, how to use RNA-Seq for different objectives and to familiarize yourself with some standard software packages for such analysis.

getting started

Sample pooling has revolutionized sequencing. It is now possible to sequence 10s of samples together. Different objectives require different sequencing depths. Doing differential gene expression analysis requires less sequencing depth than transcript reconstruction so when pooling samples it is critical to keep the objective of the experiment in mind.

In this activity we will use subsets of experimentally generated datasets. One dataset was generated for differential gene expression analysis while the other towards transcript annotation.

For quantification we will use a set of data generated from the same strain as the genome reference mouse (C57BL/6J). We selected three replicates from control (wild type) and three from a knock-out strain. The idea is to find genes that are in the same pathway as the gene that were knock out. We will use a reduced genome consisting of the first 9.5 million bases of mouse chromosome 16 and the first 50.5 million bases of chromosome 7.

Quantification and differential gene expression analysis

The main goal of this activity is to go through a standard method to obtain gene expression values and perform differential gene expression analysis from an RNA-Seq experiment.

We will start by alignment and visualizing the data using the TopHat2 spliced aligner. We will then perform gene quantification using the RSEM program and finally differential gene expression analysis of the estimated counts using DESeq.

This activity should also serve as a review of the previous two classes as you will work on the hpcc

Before you start

1. Connect to cluster using the command below with your username and password

```
ssh –X your_user@ghpcc06.umassrc.org
```

2. Please start an interactive job in the cluster using the command below for any of your operations in the cluster. Head node is used only for job submissions. An interactive session is a way to log in to a node and use it as if it were your local workstation for the length of the session. You may use this same script in the future whenever you want to start an work on a node and run programs in *interactive session*

```
/project/umw_biocore/bin/qlogin
```

3. Prepare your working directory

```
mkdir ~/RNASeqWS
```

4. Create transcriptomics bin directory

```
mkdir -p ~/RNASeqWS/transcriptomics/bin
```

Q. What happens if you ommit the -p in the command above? Try for example

```
mkdir ~/RNASeqWS/testwithoutp/otherdir
```

cd ~/RNASeqWS/transcriptomics/bin

5. Download the following two scripts and save it in the ~/RNASeqWS/transcriptomics/bin directory:

```
 wget http://bioinfo.umassmed.edu/pub/rsem.to.table.perl
```

```
 wget http://bioinfo.umassmed.edu/pub/rsem.quant.r
```

6. Create transcriptomics genome.quantification directory

```
mkdir -p ~/RNASeqWS/transcriptomics/genome.quantification

cd ~/RNASeqWS/transcriptomics/genome.quantification
```

7. Download IGV friendly annotations for each of the two activities:

```
wget http://garberlab.umassmed.edu/data/transcriptomics/genome.quantification/ucsc.corrected.gtf
```

8. Create a symoblic link for the files below to reduce used space for your trials.

A symbolic link (also known as a soft link or symlink) consists of a special type of file that serves as a reference to another file or directory. Unix/Linux like operating systems often uses symbolic links. When you don't want to copy big files like genomes. You can create a symbolic link like below.

```
ln -s /share/data/umw_biocore/genome_data/mousetest/mm10/mm10.fa ~/RNASeqWS/transcriptomics/genome.quantification/mm10.fa
```

```
ln -s /share/data/umw_biocore/genome_data/mousetest/mm10/mm10t.fa ~/RNASeqWS/transcriptomics/genome.quantification/mm10t.fa
```

```
ln -s /share/data/umw_biocore/genome_data/mousetest/mm10/ucsc.gtf ~/RNASeqWS/transcriptomics/genome.quantification/ucsc.gtf
```

```
ln -s /share/data/umw_biocore/genome_data/mousetest/mm10/ucsc_into_genesymbol.rsem ~/RNASeqWS/transcriptomics/genome.quantification/ucsc_i
nto_genesymbol.rsem
```

```
ln -s /share/data/umw_biocore/genome_data/mousetest/mm10/fastq.quantification  ~/RNASeqWS/transcriptomics/fastq.quantification
```

Exercise 1: prepare for genomic alignment

Both TopHat and RSEM rely on bowtie to perform read alignment (similar to the BWA aligner you used in genome assembly tutorial). Bowtie like BWA uses very efficient genome compression algorithm (Burrows-Wheeler transform) that allows for quick matching of sequences with less than 3 missmatches. To use these alignments it is necessary to create the BW transform of our genome before mapping reads. The bowtie-build2 program in the Bowtie distribution creates the BW index. Change your directory to genome.quantification. Invoke the BW transform on the mm10.fa file found in the directory genome.quantification:

1. First load necessary modules

```
module load RSEM/1.2.11
```

```
module load tophat/2.0.9
```

```
module load IGV/2.3.15
```

```
module load igvtools/2.3.25
```

```
module load samtools/0.0.19
```

```
module load java/1.7.0_25
```

2. Build bowtie2 index files

```
cd ~/RNASeqWS/transcriptomics/genome.quantification
```

```
bowtie2-build -f mm10.fa mm10
```

We named it mm10 (following the UCSC genome browser naming convention (http://genome.ucsc.edu/cgi-bin/hgGateway). Although we named it similarly to the full genome, this sequence file only contains a very small region of the mouse genome. Our alignment database will be called mm10, and will include partial sequences for chromosomes 7 and 16.

3. In addition we also will prepare the transcriptome for RSEM (http://deweylab.biostat.wisc.edu/rsem/README.html) alignment. RSEM will align directly to the set of transcripts included (*ucsc.gtf* file). The transcript file was downloaded directly from the UCSC table browser. The file does not contain the information necessary to map isoforms to genes, we therefore compiled a table, *ucsc_into_genesymbol.rsem* that contains this information. To generate the necessary index files use the command below. Please not that the \ is used to span multiple lines:

```
rsem-prepare-reference \
--gtf ucsc.gtf --transcript-to-gene-map ucsc_into_genesymbol.rsem \
mm10.fa mm10.rsem
```

Expected result: Your *genome.quantification* directory now should contain the following files (Tip: use ls -1):

Bowtie2 indeces:

```
mm10.1.bt2
mm10.2.bt2
mm10.3.bt2
mm10.4.bt2
mm10.rev.1.bt2
mm10.rev.2.bt2
```

Bowtie indeces:

```
mm10.rsem.ti
mm10.rsem.grp
mm10.rsem.chrlist
mm10.rsem.transcripts.fa
mm10.rsem.seq
mm10.rsem.idx.fa
mm10.rsem.1.ebwt
mm10.rsem.2.ebwt
mm10.rsem.3.ebwt
mm10.rsem.4.ebwt
mm10.rsem.rev.1.ebwt
mm10.rsem.rev.2.ebwt
```

Can you tell what are these indeces for? What is the difference between the fasta files used for each index set?

exercise 2: Genome alignment of RNA-seq reads

The *fastq.quantification* folder contains a relative small set of illumina sequencing reads. We will examine this set by first directly mapping to the reduced mouse genome.

Make sure you are in the transcriptomics directory for this activity. *genome.quantification*, *genome.reconstruction*, *fastq. quantification* and *fastq.reconstruction* should be subdirectories. Check this before you proceed.

1. To avoid cluttering the workspace we will direct the output of each exercise to its own directory. In this case for example:

```
mkdir ~/RNASeqWS/transcriptomics/tophat
```

2. Then align each of the libraries to the genome. The *fastq.quantification* subdirectory contains six different libraries, three for a control experiment from wild type mouse liver and from mouse that are deficient in two different proteins. Each genotype was sequenced in triplicates using paired-end 50 base paired reads.

To first explore the data visually in IGV, we'll use the TopHat2 aligner to map these reads to our reduced genome:

You have to be in this directory to run tophat alignment commands

```
cd ~/RNASeqWS/transcriptomics
```

```
tophat2 --library-type fr-firststrand --segment-length 20 -G  genome.quantification/ucsc.gtf \
-o  tophat/th.quant.ctrl1 genome.quantification/mm10 fastq.quantification/control_rep1.1.fq \
fastq.quantification/control_rep1.2.fq
```

And using this command as a template, align the other three different libraries

```
tophat2 --library-type fr-firststrand --segment-length 20 -G genome.quantification/ucsc.gtf \
-o tophat/th.quant.ctrl2 genome.quantification/mm10 fastq.quantification/control_rep2.1.fq \
fastq.quantification/control_rep2.2.fq
```

```
tophat2 --library-type fr-firststrand --segment-length 20 -G genome.quantification/ucsc.gtf \
-o tophat/th.quant.ctrl3 genome.quantification/mm10 fastq.quantification/control_rep3.1.fq \
fastq.quantification/control_rep3.2.fq
```

```
tophat2 --library-type fr-firststrand --segment-length 20 -G genome.quantification/ucsc.gtf \
-o tophat/th.quant.expr1 genome.quantification/mm10 fastq.quantification/exper_rep1.1.fq \
fastq.quantification/exper_rep1.2.fq
```

```
tophat2 --library-type fr-firststrand --segment-length 20 -G genome.quantification/ucsc.gtf \
-o tophat/th.quant.expr2 genome.quantification/mm10 fastq.quantification/exper_rep2.1.fq \
fastq.quantification/exper_rep2.2.fq
```

```
tophat2 --library-type fr-firststrand --segment-length 20 -G genome.quantification/ucsc.gtf \
-o tophat/th.quant.expr3 genome.quantification/mm10 fastq.quantification/exper_rep3.1.fq \
fastq.quantification/exper_rep3.2.fq
```

Question: What percent of reads were mapped for each library?

Check the tophat reports of tophat for each of the six libraries, in particular the align_summary.txt file
4. Tophat always creates reports its alignment in a file named “accepted_hits.bam”. To make things clear we’ll move this files onto a clean directory. Move the files by, for example, doing

```
mv tophat/th.quant.ctrl1/accepted_hits.bam tophat/th.quant.ctrl1.bam
mv tophat/th.quant.ctrl2/accepted_hits.bam tophat/th.quant.ctrl2.bam
mv tophat/th.quant.ctrl3/accepted_hits.bam tophat/th.quant.ctrl3.bam
mv tophat/th.quant.expr1/accepted_hits.bam tophat/th.quant.expr1.bam
mv tophat/th.quant.expr2/accepted_hits.bam tophat/th.quant.expr2.bam
mv tophat/th.quant.expr3/accepted_hits.bam tophat/th.quant.expr3.bam
```

To visualize the alignments we generate indexes (for rapid data access) and compressed read density plots:

```
java -Xmx5g -jar /share/pkg/picard/1.96/BuildBamIndex.jar I=tophat/th.quant.ctrl1.bam
```

and with all the other libraries:

```
java -Xmx5g -jar /share/pkg/picard/1.96/BuildBamIndex.jar I=tophat/th.quant.ctrl2.bam
java -Xmx5g -jar /share/pkg/picard/1.96/BuildBamIndex.jar I=tophat/th.quant.ctrl3.bam
java -Xmx5g -jar /share/pkg/picard/1.96/BuildBamIndex.jar I=tophat/th.quant.expr1.bam
java -Xmx5g -jar /share/pkg/picard/1.96/BuildBamIndex.jar I=tophat/th.quant.expr2.bam
java -Xmx5g -jar /share/pkg/picard/1.96/BuildBamIndex.jar I=tophat/th.quant.expr3.bam
```

Finally create read density files to be able to look at all libraries together

```
/project/umw_biocore/bin/igvtools.sh count -w 5 tophat/th.quant.ctrl1.bam tophat/th.quant.ctrl1.bam.tdf genome.quantification/mm10.fa
```

Similarly create density files for all other libraries.

```
/project/umw_biocore/bin/igvtools.sh count -w 5 tophat/th.quant.ctrl2.bam tophat/th.quant.ctrl2.bam.tdf genome.quantification/mm10.fa
/project/umw_biocore/bin/igvtools.sh count -w 5 tophat/th.quant.ctrl3.bam tophat/th.quant.ctrl3.bam.tdf genome.quantification/mm10.fa
/project/umw_biocore/bin/igvtools.sh count -w 5 tophat/th.quant.expr1.bam tophat/th.quant.expr1.bam.tdf genome.quantification/mm10.fa
/project/umw_biocore/bin/igvtools.sh count -w 5 tophat/th.quant.expr2.bam tophat/th.quant.expr2.bam.tdf genome.quantification/mm10.fa
/project/umw_biocore/bin/igvtools.sh count -w 5 tophat/th.quant.expr3.bam tophat/th.quant.expr3.bam.tdf genome.quantification/mm10.fa
```

We are now ready to look at the data. Download the IGV program from the IGV site (http://www.broadinstitute.org/igv) and transfer your files from the hpcc to your laptop. In general you will not transfer the files. There are ways to access the files from your dekstop or laptop directly at the HPCC, we'll cover this later. Since the files are very small transfer time will not be a problem. Use Filezila or the scp command if you are on a mac, to copy the files onto your laptop/desktop, please ensure you note to which directory you transfer the files as later you will need to load them into IGV.

Launch the IGV browser, and use the *File* -> *load* to load the files onto the browser. Load all the .tdf files and only one or two .bam files to begin with

A few genes are good examples of differentially expressed genes. For example the whole region around the key *Fgf21* gene is upregulated in experiment vs controls, while the gene *Crebbp* is downregulated in experiments vs controls. To point your browser to either gene just type or copy the name of the gene in the location box at the top.

We will revisit these genes below when we do differential gene expression.

exercise 3: Quantify with the RSEM program

RSEM (http://deweylab.biostat.wisc.edu/rsem/) depends on an existing annotation and will only scores transcripts that are present in the given annotation file. We will compare the alignments produced by RSEM and tophat and this will become clear.

The first step is to prepare the transcript set that we will quantify. We selected the UCSC genes (http://genome.ucsc.edu/cgi-bin/hgTables?hgsid=359712943&clade=mammal&org=Mouse&db=mm10&hgta_group=genes&hgta_track=knownGene&hgta_table=0&hgta_regionType=genome&position=chr6%3A113001853-113229210&hgta_outputType=bed&hgta_outFileName=hg19.ensembl.bed) which is a very comprehensive, albeit a bit noisy dataset. As with all the data in this activity we will only use the subset of the genes that map to the genome regions we are using.

3.1 Prepare transcripts for alignment

Given an annotation list in GTF file format and the genome sequence, RSEM is capable of extracting the sequence for each transcript, keep the gene/isoform relationship and invoke the bowtie-build program to create BW indeces.

First change you working directory to the genome.quantification subdirectory and then use this command to create the necessary indeces and maps:

```
cd ~/RNASeqWS/transcriptomics/genome.quantification
```

If you haven't prepared rsem reference files. Run the command below

```
rsem-prepare-reference --gtf ucsc.gtf --transcript-to-gene-map ucsc_into_genesymbol.rsem mm10.fa mm10.rsem
```

Question: Why do we need the –transcript-to-gene-map? What is the information in that table? Look at the GTF file, what is missing?

Question: What files were created? What do you think each one is?

3.2 Calculate expression

RSEM now is ready to align and then attempt to perform read assignment and counting for each isoform in the file provided above. You must process each one of the 6 libraries:

```
cd ~/RNASeqWS/transcriptomics
```

```
mkdir rsem

rsem-calculate-expression --paired-end --strand-specific -p 2 \
 --output-genome-bam fastq.quantification/control_rep1.1.fq \
fastq.quantification/control_rep1.2.fq genome.quantification/mm10.rsem rsem/ctrl1.rsem
```

And similarly for each of the other 5 libraries

```
rsem-calculate-expression --paired-end -p 2 --output-genome-bam \
fastq.quantification/control_rep2.1.fq fastq.quantification/control_rep2.2.fq \
genome.quantification/mm10.rsem rsem/ctrl2.rsem
```

```
rsem-calculate-expression --paired-end -p 2 --output-genome-bam \
fastq.quantification/control_rep3.1.fq fastq.quantification/control_rep3.2.fq \
genome.quantification/mm10.rsem rsem/ctrl3.rsem
```

```
rsem-calculate-expression --paired-end -p 2 --output-genome-bam \
fastq.quantification/exper_rep1.1.fq fastq.quantification/exper_rep1.2.fq \
genome.quantification/mm10.rsem rsem/expr1.rsem
```

```
rsem-calculate-expression --paired-end -p 2 --output-genome-bam \
fastq.quantification/exper_rep2.1.fq fastq.quantification/exper_rep2.2.fq \
genome.quantification/mm10.rsem rsem/expr2.rsem
```

```
rsem-calculate-expression --paired-end -p 2 --output-genome-bam \
fastq.quantification/exper_rep3.1.fq fastq.quantification/exper_rep3.2.fq \
genome.quantification/mm10.rsem rsem/expr3.rsem
```

You should take the time to familiarize yourself with the output

3.3 Create consolidated table

In the bin directory (we provide a simple script to take all the independent RSEM output and combine it into a single table, which is then useful for inspection and ready for differential gene expression analysis.

To find out what the script does you may type the following command in the transcriptomics directory

```
cd ~/RNASeqWS/transcriptomics
```

```
perl bin/rsem.to.table.perl –help
```

We will generate two tables with isoform and gene level expression:

```
perl bin/rsem.to.table.perl -out rsem.gene.summary.count.txt -indir rsem -gene_iso genes -quantType expected_count
perl bin/rsem.to.table.perl -out rsem.isoforms.summary.count.txt -indir rsem -gene_iso isoforms -quantType expected_count
```

You should now take the time to inspect these tables, find genes that look affected by the experiment.

3.3 Visualize the raw data: Make a IGV genome with the transcriptome (NOT YET TESTED, Try at your own risk)

We will need to create bam index files for each of the two alignments generated by RSEM.

As we have done before we need to create TDF files for both the genome and transcript alignment e.g.:

```
/project/umw_biocore/bin/igvtools.sh count -w 5 rsem/ctrl1.rsem.transcript.sorted.bam rsem/ctrl1.rsem.transcript.sorted.bam.tdf \
genome.quantification/mm10t.fa
```

and

```
/project/umw_biocore/bin/igvtools.sh count -w 5 rsem/ctrl1.rsem.genome.sorted.bam  rsem/ctrl1.rsem.genome.sorted.bam.tdf \
genome.quantification/mm10t.fa
```

Now we can compare the later with the tophat alignments and the former to the table we built.

exercise 4: Differential gene expression analysis with DESeq

DESeq is an R package available via Bioconductor and is designed to normalise count data from high-throughput sequencing assays such as RNA-Seq and test for differential expression.

Please finish the tutorial to learn how to run R commands
tryr.codeschool.com (tryr.codeschool.com)

To run R you just need to type the following in the command line:

```
module load R/3.0.1
Rscript  bin/rsem.quant.r
```

We will revisit this R section in the 6th week.

Note:In the R code change the naming of the control and expr according to the order of colnames(d).

```
colData = as.data.frame((colnames(d)));
colData[1,2] = "control";
colData[2,2] = "expr";
colData[3,2] = "expr";
colData[4,2] = "control";
colData[5,2] = "expr";
colData[6,2] = "control";
```
