**Reference guided RNA-seq data analysis using BioHPC Lab computers**

- This document assumes that you already know some basics of how to use a Linux computer.
- Some of the command lines in this document are too long to fit in one single line. "\" characters are used between the broken lines.
- This documents give you the basic commands to run the software. For more advanced users, please refer to the software manual:
  Tophat: http://tophat.cbcb.umd.edu/manual.shtml
  Cufflinks: http://cufflinks.cbcb.umd.edu/manual.html
  EdgeR: http://www.bioconductor.org/packages/release/bioc/vignettes/edgeR/inst/doc/edgeRUsersGuide.pdf

## Part 1. Prepare the files.

1. Create a directory /workdir/<UserName>/fastq, and copy all your fastq files into this directory. You can keep the files in compressed .gz format.

Note:

If you have many files that need to be downloaded from a sequencing facility, you might want to write a shell script to download the files in batch. First, you need to gather all the files names and their corresponding URL. Then use a text editor to make a file "download.sh" using following commands, you will need to write one command for each file.

```
wget -q -O filename1.fastq.gz "http://FileURL1"

wget -q -O filename2.fastq.gz "http://FileURL2"
```

Once you have the file ready, execute the following command.

```
sh download.sh
```

2. Check the sequencing quality (optional)

This step is optional. It tells you the sequencing quality of the data. It is recommended to run this step if you got unexpected RNA-seq results.

```
cd /workdir/<UserName>/fastq

fastqc -o qcreport sample1.fastq.gz sample2.fastq.gz …
```

- -o qcreport : specify the output directory for the QC report.
- Sample1.fastq.gz sample2.fastq.gz …: All the fastq files separated by space " "
- After it is done, you can use FileZilla to copy the qcreport directory to you laptop computer, and open the index.html file with a web browser.

3. Prepare the genome database.

In most cases, you will need a genome sequence fasta file and an annotation file either in gff3 or gtf formats.

If your data is from a model organism, including Human, mouse, yeast, Arabidopsis, C.elegans, D.melanogaster, maize, we have the databases on the local drive on each BioHPC computer, under the directory: /local_data. You can use them directly.

For certain species, or older versions of model species, we keep the genome database on the shared file server, /shared_data/genome_db/. You will need to copy the files to your /workdir/<UserName>. The genome files are under the bowtie2 directory, the annotation file is a file with name extension like gff3 or gtf.

If your genome is not in our server, you will need to download the genome FASTA file and annotation file, keep them in the /workdir/< UserName >/genome directory. Then index the genome using the following command. Before you build index, change the fasta file name to end with ".fa". As this step takes a while, it is a good idea to wrap your commands between "nohup" and ">& logfile &". You can check whether the step is finished or not using the "top" command (press q to exit "top").

```
cd /workdir/<UserName>/genome

nohup bowtie2-build mygenome.fa mygenome >& build.log &
```

- bowtie2-build: The software that is used to build bowtie2 index that can be used by the followup alignment software tophat.
- mygenome.fa:The name of the genome fasta file.
- mygenome: the name of the database.

4. Prepare the tophat transcript index (optional).

In order to map reads that are from the splicing junctions, the tophat software align the reads not only to the genome database, also to a transcriptome database. You can either supply tophat with a pre-built transcriptome database, or let tophat to build a new transcriptome database every time you run the software. If you need to run tophat on many fastq files, it is worthwhile to pre-build the transcriptome database. That will save ~30 minutes for each tophat run.

```
tophat -G myAnnotation.gtf --transcriptome-index transcriptome \
mygenome
```

- -G myAnnotation.gtf : specify the genome annotation file. Tophat can accept either gff or gtf file.
- --transcriptome-index transcriptome: specify the output directory for the indexed transcriptome database.
- mygenome: The genome database created by the bowtie2-build.

After this step, you will see a new directory called "transcriptome", with file names derived from your gff file, e.g. myAnnotation.*. You will refer to your transcriptome database as genome/transcriptome/myAnnotation.

## Part 2. Align the reads to the genome

1. Alignment with tophat.
   Practically, you will write the tophat commands in a batch script. Here are the templates for the batch script. You can use a text editor to create the batch script file. Mac user can use the built-in Text Editor. Windows user can download the free Notepad+.

Template 1 (You DID NOT pre-build the optional transcriptome database as described in step 3 of Part 1)

```
tophat -p 8 -o sample1 -G myAnnotation.gtf --no-novel-juncs genome/mygenome \
fastq/sample1.fastq.gz
```

- -p 8 : use 8 processors of the computer.
- -o sample: write the output file into a directory named sample1
- -G myAnnotation.gtf: use the annotation file myAnnotation.gtf to construct the transcriptome database

- --no-novel-juncs: Do not detect novel splicing junctions beyond what are listed in the annotation file. You need to skip this option if you are interested in finding new gene models.
- Genome/mygenome: genome database as created by bowtie2-build.
- Fastq/sample1.fastq.gz: your data file.
- Write one line for each one of your fastq files.

Template 2 (You pre-built the optional transcriptome database as described in step 3 of Part 1)

```
tophat -p 8 -o sample1 -G myAnnotation.gtf --transcriptome-index \
genome/transcriptome/myAnnotation --no-novel-juncs genome/mygenome \
fastq/sample1.fastq.gz
```

- -p 8 : use 8 processors of the computer.
- -o sample: write the output file into a directory named sample1
- `transcriptome-index genome/transcriptome/myAnnotation`: use the prebuild transcriptome database `genome/transcriptome/myAnnotation.`
- --no-novel-juncs: Do not detect novel splicing junctions beyond what are listed in the annotation file. You need to skip this option if you are interested in finding new gene models.
- Genome/mygenome: genome database as created by bowtie2-build.
- Fastq/sample1.fastq.gz: your data file.
- Write one line for each one of your fastq files.

After you finish the batch script, you can use FileZilla to copy the script to your BioHPC home directory.

If the file is created on Windows, make sure to execute the following command before you run the batch script (myscript.sh is the name of your script)

```
cd /home/<UserName>
dos2unix myscript.sh
```

To execute the script:

```
cd /workdir/<UserName>/
sh /home/<UserName>/myscript.sh
```

## Part 3. Count the number of reads mapped to each gene

1. Tophat created an output directory for each one of the fastq files. There is a BAM file named "accepted_hits.bam" in each of the output directories.
   There is also a file "align_summary.txt". This file tells you what percentage of your reads can be aligned to your reference genome uniquely or ambiguously. These are important QC measures you need to keep for your records. To see what are in the align_summary.txt, you can use the command : "more align_summary.txt".

   Before you move on to next step, you need to rename these accepted_hits.bam files and move them into the same directories. First, create a new directory called "bam".

```
mkdir /workdir/<UserName>/bam
```

   If you like, you can use a shell script to move the files.

```
mv sample1/accepted_hits.bam bam/sample1.bam
mv sample2/accepted_hits.bam bam/sample2.bam
mv sample3/accepted_hits.bam bam/sample3.bam
 …
```

2. Use the cuffdiff software to count the reads.

```
cd /workdir/<UserName>/bam

cuffdiff -p 8 -o cuffdiffout  genome/myAnnotation.gtf  sample1_rep1.bam,sample1_rep2.bam \
sample2_rep1.bam, sample2_rep2.bam  …
```

- -p 8 : use 8 processors of the computer.
- -o cuffdiffout : write the output file into a directory named cuffdiffout .
- genome/myAnnotation.gtf : the genome annotation file
- sample1_rep1.bam,sample1_rep2.bam sample2_rep1.bam, sample2_rep2.bam : all the bam files. The bam files from biological replicates should be separated by comma ",", while bam files from independent samples should be separated by space " ".

3. After you run cuffdiff, one of the output files "gene_exp.diff" give you the list of differentially expressed genes.

## Part 3. Generate MDS plot of the libraries.

MDS plot can be used to evaluate the variance between biological replicates, and identify sample outliers and mislabeled samples. We are going to use edgeR package to run MDS.

1. First, you need to convert the genes.read_group_tracking file into a tab-delimited text file that can be loaded into an R data table. You will see two new files created after this step: edgeR_count.xls and edgeR_fpkm.xls.

```
cd cuffdiffout

parse_cuffdiff_readgroup.pl
```

2. From the ssh terminal, type “R” and press return. Now, you are in R console. Use the following steps to generate a PDF file with MDS plot of the samples.

```
library("edgeR")

x <- read.delim("edgeR_count.xls", row.names='Gene')

group <- factor(c(1,1,1,2,2,2,3,3,3))

y <- DGEList(counts=x,group=group)

y <- calcNormFactors(y)

pdf("myplot.pdf")

plotMDS(y, col=c(rep("black",3), rep("red",3), rep(“blue”, 3)))

dev.off()

quit()
```

- In this example, there are three biological conditions with three replicates for each condition. If your data set is different, you need to modify the c(1,1,1,2,2,2,3,3,3) and `c(rep("black",3), rep("red",3), rep("blue", 3))`.

3. Transfer the myplot.pdf file back to your computer.

Part 4. Get the differentially expressed genes using EdgeR

Bioconductor is an R package for analyzing high-throughput genomic data. You can run Bioconductor tools either on the BioHPC computer or on your own computer. The edgeR package is a popular tool for RNA-seq data analysis to detect differentially expressed genes.

You will need to start R and load the data into R. For this, you can use the file edgeR_count.xls generated by parse_cuffdiff_readgroup.pl as described in Part3 of this workflow.

A very comprehensive tutorial of edgeR can be found in:

http://cgrlucb.wikispaces.com/file/view/edgeR_Tutorial.pdf.

1. Testing for differential expressed (DE) genes with edgeR's generalized linear models (glm)

```
library("edgeR")

x <- read.delim("edgeR_count.xls", row.names='Gene')

x <- round(x, 0)

group <- factor(c(1,1,1,2,2,2,3,3,3))

y <- DGEList(counts=x,group=group)

# only keep genes with cpm value greater than 1 in at least 3 samples

keep <-rowSums(cpm(y)>=1) >=3

y<-y[keep,]

y <- calcNormFactors(y)

# write the CPM values into a tab-delimited text file

# cpm stands for counts per million.

d <- cpm(y)

write.table(d, "CPM.txt", sep="\t")
```

```
design<-model.matrix(~0+group)
y <- estimateGLMCommonDisp(y,design)
y <- estimateGLMTrendedDisp(y,design)
y <- estimateGLMTagwiseDisp(y,design)
fit<-glmFit(y,design)
```

This fit has three parameters. The first is the baseline level of group 1. The second and the third are the 2 vs 1, and 3 vs 1 differences.

To compare 2 vs 1

```
lrt.2vs1<-glmLRT(fit, contrast=c(-1,1,0))
top2v1 <-  topTags(lrt.2vs1, n=2000)
write.table(top2v1, "diff2-1.txt", sep="\t")
```

To compare 3 vs 1

```
lrt.3vs1<-glmLRT(fit, contrast=c(-1,0,1))
top3v1 <-  topTags(lrt.3vs1, n=2000)
write.table(top3v1, "diff3-1.txt", sep="\t")
```

To compare 3 vs 2

```
lrt.3vs2<-glmLRT(fit, contrast=c(0,-1,1))
top3v2 <-  topTags(lrt.3vs2, n=2000)
write.table(top3v2, "diff3-2.txt", sep="\t")
```