Exercise 2. RNA-seq

Part 1. Prepare the working directory.

1. Use the same computer as the exercise 1.
2. Connect to your assigned computer. If you do not know how, follow the instruction at http://cbsu.tc.cornell.edu/lab/doc/Remote_access.pdf (Read the section under "Connection by ssh". There are separate instructions for Windows and Mac users)
3. Use the following commands to create a working directory, and copy Arabidopsis genome FASTA file (TAIR10.fa) and rice annotation file (TAIR10.gff3) for this homework to the working directory.

```
mkdir /workdir/my_user_ID/exercise2

cd /workdir/my_user_ID/exercise2

cp /shared_data/RNAseq/exercise2/* ./

ls
```

- Replace my_user_ID with your netID.

Part 2. Index the genome file

1. We are going to use STAR to align RNA-seq reads to the genome. First, we will need to index the reference genome.
   Add STAR to the current path, so that you can run "STAR" without full path.

```
export PATH=/programs/STAR:$PATH
```

   Index the reference genome.

```
mkdir genomedb

nohup  STAR --runMode genomeGenerate --runThreadN 2 \

--genomeDir genomedb \

--genomeFastaFiles genome.fa --sjdbGTFfile genome.gtf --sjdbOverhang 49 \

>& mystarlog &
```

- This process might take 20 minutes. With “nohup” “>&mylogfile &” before and after the command, you would be running the software in background. This way, even if you shut down your laptop, the job would continue to run on the remote server.
- Sometime, you might want to terminate a program that is running in the background (e.g. you realized that you set the parameter wrong, need to restart the job). First use “top” or “ps -u my_user_ID” to find out the process ID (PID) of your program, it should be an integer number. Then use the command “kill PID” to terminate the program. Make sure you use “top” command again to confirm that the program is actually killed.
- If you need to terminate a program that is NOT running in the background, press “Ctrl-C” to stop it. If this does not work, you can open ssh connection in a new terminal window, and use “kill PID” to kill a program that is running.
- When you use a shell script to run a batch of commands, you will need to use “kill PID” to kill both the PID of the shell script, and the any of the commands that are still running.

Part 3. Run STAR to align reads and count number of reads per gene.

1. We have files from 4 RNA-seq libraries. Running STAR could take up to an hour. We will prepare a shell script to process all 4 fastq files. Create a text file with a Text editor (We recommend Text Wrangler for mac users, Notepad+ or Edit+ for Windows users). Using the content in the black box below. Name the file rnaseq.sh. Upload the file to your home directory.
   If the file is made on a Windows computer, you need to make sure to save the file as a LINUX style text file. From NotePad++, used the "Edit -> EOL Conversion -> UNIX" option.

```
STAR --quantMode GeneCounts --genomeDir genomedb --runThreadN 2 --outFilterMismatchNmax 2
--readFilesIn WTa.fastq.gz --readFilesCommand zcat --outFileNamePrefix WTa --
outFilterMultimapNmax 1 --outSAMtype BAM SortedByCoordinate

STAR --quantMode GeneCounts --genomeDir genomedb --runThreadN 2 --outFilterMismatchNmax 2
--readFilesIn WTb.fastq.gz --readFilesCommand zcat --outFileNamePrefix WTb --
outFilterMultimapNmax 1 --outSAMtype BAM SortedByCoordinate

STAR --quantMode GeneCounts --genomeDir genomedb --runThreadN 2 --outFilterMismatchNmax 2
--readFilesIn MUa.fastq.gz --readFilesCommand zcat --outFileNamePrefix MUa --
outFilterMultimapNmax 1 --outSAMtype BAM SortedByCoordinate

STAR --quantMode GeneCounts --genomeDir genomedb --runThreadN 2 --outFilterMismatchNmax 2
--readFilesIn MUb.fastq.gz --readFilesCommand zcat --outFileNamePrefix MUb --
outFilterMultimapNmax 1 --outSAMtype BAM SortedByCoordinate
```

Now, run the shell script:

```
cd /workdir/my_user_ID/exercise2

nohup sh ~/rnaseq.sh >& runlog &
```

- In Linux, "~" is a shortcut for your home directory. Use "~/rnaseq.sh" if your script is located in the home directory. If you keep the script in the working directory, then use "./rnaseq.sh".
- This process would take up to an hour. You need to use "nohup" to run the shell script.

Use "top" command to check whether the job is finished or not. If the job is finished, you should not see any jobs running under your user name. You should see "cuffdiffout" directory should have been corrected.

After running STAR, you will find many new files are created. You need to keep the following files for each sample:

- *Aligned.sortedByCoord.out.bam: Alignment file.
- *Log.final.out: Alignment statistics. This file tells you what % of reads can be mapped. You will need this information for QC and for publication.
- *ReadsPerGene.out.tab: read count per gene.

We will need the ReadsPerGene.out.tab file for next step to identify differentially expressed genes. It is a tab delimited text file. The columns are:

column 1: gene ID;

column 2: counts for unstranded RNA-seq;

column 3: counts for reads from forward strand of RNA;

column 4: counts for reads from reverse strand of RNA;

As this data set is strand specific, we will use the values in column 4 as read count per gene.

Part 3. Integrate read count from all 4 samples into one table.

STAR would produce one gene count file per sample. Here we will use Linux tools “paste” and “cut” to combine them into one single table. If you do not feel comfortable with the Linux commands, you can also use Excel to combine them into one table. In the following command line, I am using Linux pipe “|” to connect Linux functions into one single command.

```
paste WTaReadsPerGene.out.tab WTbReadsPerGene.out.tab MUaReadsPerGene.out.tab
MUaReadsPerGene.out.tab | \

cut -f1,4,8,12,16 | \

tail -n +5 > tmpfile

cat tmpfile | sed "s/^gene://" >gene_count.txt
```

paste: merge files by columns;
cut: extract columns in the merged file;
tail -n +5: discard the first 4 lines of statistics summary and start from line 5;
`sed "s/^gene://"`: remove “gene:” from each line;
`>gene_count.txt`: Write the result into a file gene_count.txt

Part 4. 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 generate the plot.

1. 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("gene_count.txt", header=F, row.names=1)
colnames(x)<-c("WTa","WTb","MUa","MUb")
group <- factor(c(1,1,2,2))
y <- DGEList(counts=x,group=group)
y <- calcNormFactors(y)
pdf("myplot.pdf")
plotMDS(y, col=c(rep("red",2), rep("blue", 2)))
dev.off()
quit()
```

Part 5. Using EdgeR to identify differentially expressed genes.

DESeq and EdgeR are two commonly used statistics packages for analyzing RNA-seq data. In this section, we will use DESeq to detect differentially expressed genes.

From the ssh terminal, type “R” and press return. Now, you are in R console.

```
library("edgeR")

x <- read.delim("gene_count.txt", header=F, row.names=1)

colnames(x)<-c("WTa","WTb","MUa","MUb")

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

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

y <- calcNormFactors(y)

# only keep genes with cpm value greater than 1 in at least 2
samples

keep <-rowSums(cpm(y)>=1) >=2

y<-y[keep,]

# 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)

lrt.2vs1 <- glmLRT(fit, contrast=c(-1,1))

top2v1 <- topTags(lrt.2vs1, n=5000)

write.table(top2v1, "diff2-1.txt", sep="\t")
```

The Bioinformatics Facility is maintaining a workflow documentation for RNA-seq. Go to http://cbsu.tc.cornell.edu/lab/userguide.aspx, click “workflows”.