1) Launch interactive job

Modules needed

module load tophat/2.0.9
module load bioinfo/bowtie/2.1.0
module load bioinfo/samtools/0.1.18
module load bioinfo/cufflinks/2.2.1
module load bioinfo/R/3.1.0

2) Make genome index

bowtie2-build genome.fa genome

3) Tophat on first library

tophat -I 1000 -i 20 --library-type fr-firststrand -o tophat.Sp_ds.dir genome Sp_ds.left.fq Sp_ds.right.fq

4) Rename output

mv tophat.Sp_ds.dir/accepted_hits.bam tophat.Sp_ds.dir/Sp_ds.bam

5) Index Output

samtools index tophat.Sp_ds.dir/Sp_ds.bam

6) Cufflinks

cufflinks --overlap-radius 1 --library-type fr-firststrand -o cufflinks.Sp_ds.dir tophat.Sp_ds.dir/Sp_ds.bam

7) Rename Cufflinks

mv cufflinks.Sp_ds.dir/transcripts.gtf cufflinks.Sp_ds.dir/Sp_ds.transcripts.gtf

8) Exit interactive job

9) Repeat steps 3-7 on the other 3 samples (make a batch script for each sample)

10) When all jobs have run then you can move on to the next steps.

Launch another interactive job
Modules you’ll need

11) Create merged assemblies

```bash
    echo cufflinks.Sp_ds.dir/Sp_ds.transcripts.gtf >> assemblies.txt
    echo cufflinks.Sp_hs.dir/Sp_hs.transcripts.gtf >> assemblies.txt
    echo cufflinks.Sp_log.dir/Sp_log.transcripts.gtf >> assemblies.txt
    echo cufflinks.Sp_plat.dir/Sp_plat.transcripts.gtf >> assemblies.txt
```

12) check assemblies.txt file

```bash
    cat assemblies.txt
```

13) Merge all assemblies together

```bash
    cuffmerge -s genome.fa assemblies.txt
```

14) We will now look at these files in IGV on your local machine

```bash
    mkdir IGV
    sh IGV_move.sh
```

Use filezilla to move that directory to your current machine.

15) run cuffdiff on these datasets

```bash
```

16) Inspect cuffdiff

```bash
    head diff_out/gene_exp.diff
```

Use cummeRbund to produce useful graphs and figures

**Study transcript expression and analyze DE using CummeRbund:**

Use ‘cummeRbund’ to analyze the results from cuffdiff:

```bash
    % R
    (note, to exit R, type cntrl-D, or type “q()”).
```
# load the cummerbund library into the R session
> library(cummeRbund)

# import the cuffdiff results
>cuff = readCufflinks('diff_out')
>cuff

# examine the distribution of expression values for the reconstructed transcripts
>csDensity(genes(cuff))
# Examine transcript expression values in a scatter plot

Expression values are typically log-normally distributed. This is just a sanity check.

```r
> csScatter(genes(cuff), 'Sp_log', 'Sp_plat')
```

Strongly differentially expressed transcripts should fall far from the linear regression line.
# Examine individual sample distributions of gene expression values and the pairwise scatterplots together in a single plot.

> cs ScatterMatrix(genes(cuff))
# Volcano plots are useful for identifying genes most significantly differentially expressed.

> csVolcanoMatrix(genes(cuff))
## Extract the ‘genes’ that are significantly differentially expressed (red points above)

# retrieve the gene-level differential expression data
> gene_diff_data = diffData(genes(cuff))

# how many ‘genes’ are there?
> nrow(gene_diff_data)

# from the gene-level differential expression data, extract those that
# are labeled as significantly different.
# note, normally just set criteria as “significant='yes'”, but we’re adding an
# additional p_value filter just to capture some additional transcripts for
# demonstration purposes only. This simulated data is overly sparse and actually
# suboptimal for this demonstration (in hindsight).

> sig_gene_data = subset(gene_diff_data,(significant=='yes' | p_value < 0.01))

# how many genes are significantly DE according to these criteria?
> nrow(sig_gene_data)

# Examine the entries at the top of the unsorted data table:

> head(sig_gene_data)

<table>
<thead>
<tr>
<th>gene_id</th>
<th>sample_1</th>
<th>sample_2</th>
<th>status</th>
<th>value_1</th>
<th>value_2</th>
<th>log2_fold_change</th>
<th>test_stat</th>
<th>p_value</th>
<th>q_value</th>
<th>significant</th>
</tr>
</thead>
<tbody>
<tr>
<td>XLOC_000056</td>
<td>Sp_ds</td>
<td>Sp_hs</td>
<td>OK</td>
<td>33560.500</td>
<td>117.6900</td>
<td>-8.15563</td>
<td>-4.86421</td>
<td>0.00360</td>
<td>0.25792</td>
<td>no</td>
</tr>
<tr>
<td>XLOC_000136</td>
<td>Sp_ds</td>
<td>Sp_hs</td>
<td>OK</td>
<td>30094.800</td>
<td>246.0650</td>
<td>-6.93433</td>
<td>-4.49008</td>
<td>0.00795</td>
<td>0.29173</td>
<td>no</td>
</tr>
<tr>
<td>XLOC_000146</td>
<td>Sp_ds</td>
<td>Sp_hs</td>
<td>OK</td>
<td>1125.700</td>
<td>70.9957</td>
<td>-3.98694</td>
<td>-3.99117</td>
<td>0.00640</td>
<td>0.29173</td>
<td>no</td>
</tr>
<tr>
<td>XLOC_000056</td>
<td>Sp_ds</td>
<td>Sp_log</td>
<td>OK</td>
<td>33560.500</td>
<td>108.8130</td>
<td>-8.26877</td>
<td>-4.76096</td>
<td>0.00450</td>
<td>0.29173</td>
<td>no</td>
</tr>
<tr>
<td>XLOC_000102</td>
<td>Sp_ds</td>
<td>Sp_log</td>
<td>OK</td>
<td>0.0000</td>
<td>44.0000</td>
<td>Inf</td>
<td>NA</td>
<td>0.00160</td>
<td>0.14355</td>
<td>no</td>
</tr>
<tr>
<td>XLOC_000123</td>
<td>Sp_ds</td>
<td>Sp_log</td>
<td>OK</td>
<td>753.187</td>
<td>0.0000</td>
<td>Inf</td>
<td>NA</td>
<td>0.00005</td>
<td>0.00870</td>
<td>yes</td>
</tr>
</tbody>
</table>

# You can write the list of significantly differentially expressed genes to a file like so:

> write.table(sig_gene_data, 'sig_diff_genes.txt', sep = '\t', quote = F)
# examine the expression values for one of your genes that’s diff. expressed:

# select expression info for the one gene by its gene identifier:
# let’s take the first gene identifier in our sig_gene_data table:

# first, get its gene_id
> ex_gene_id = sig_gene_data$gene_id[1]

# print its value to the screen:
> ex_gene_id

# get that gene ‘object’ from cummeRbund and assign it to variable ‘ex_gene’
> ex_gene = getGene(cuff, ex_gene_id)

# now plot the expression values for the gene under each condition
# (error bars are only turned off here because this data set is both simulated
# and hugely underpowered to have reasonable confidence levels)

> expressionBarplot(ex_gene, logMode=T, showErrorbars=F)
## Draw a heatmap showing the differentially expressed genes

# first retrieve the ‘genes’ from the ‘cuff’ data set by providing a
# a list of gene identifiers like so:
> sig_genes = getGenes(cuff, sig_gene_data$gene_id)

# now draw the heatmap
> csHeatmap(sig_genes, cluster='both')