Lab #6

By the beginning of the next lab (March 2 ), send what you have to afodor@uncc.edu

Make sure the text “Lab #6” is in the subject line…

(1) Install DeSeq version 1 in your version of R.

In R type:

source("http://bioconductor.org/biocLite.R")

biocLite("DESeq")

library("DESeq")

See (<http://bioconductor.org/packages/release/bioc/html/DESeq.html> ) for more information.

(2) Generate a simulated dataset in which the variance is a function of the mean but the null hypothesis of differential expression is always true. The following code will accomplish this:

rm(list=ls())

numRows = 3000

numCols = 10

for( i in 1:numCols)

myFrame <- data.frame(1:numRows)

#initiate the data.frame with the correct # of rows to suppress error messages.

#likely, there are much better ways to do this!

names(myFrame)[1] <- "tempColumn"

for( i in 1: numCols)

{

vals <- vector(length=numRows)

for( j in 1:numRows)

{

aMean = j /10

aMean = max( aMean,5)

aVar = aMean+ 5\* aMean

aVal = round( max( rnorm(1,mean=aMean,sd=sqrt(aVar)), 1))

vals[j] = aVal

}

colName <- paste( "sample" , i ,sep="")

myFrame[[colName]] = vals

}

myFrame["tempColumn"] <- NULL

row.names(myFrame) <- paste("Gene\_",1:numRows,sep="")

(3) Following the example in Lecture #11 (or the DeSeq vignette here: https://www.bioconductor.org/packages/devel/bioc/vignettes/DESeq/inst/doc/DESeq.pdf)

run newCountDataSet() and then estimateSizeFactors() and then estimateDispersions() and finally nbinomTest() on the simulated data from step #2. Assume the first 5 columns are from one condition and the second 5 columns are from another condition.

(A) If "cds" is your count dataset from the above step, make a graph showing the relationship between DeSeq's estimated mean and variance for each gene. This can be accomplished with the following code on a CDS object:

means <- apply(counts(cds,normalized=TRUE), 1,mean)

myInfo <- fitInfo( cds )

# DeSeq's estimate of variance calculated for each gene

plot(means, means \* means\* myInfo$perGeneDispEsts)

# the fit that DeSeq uses for the pooled "varince"

lines(means, means\*means\* myInfo$dispFunc(means),col="RED")

# the variance for each gene that DeSeq actually uses for inference

points(means, means \* means\* fData(cds)[,1], col="YELLOW")

What is the relationship between the variance that DeSeq uses for inference (yellow points), the estimate of variance calculated for each gene (black points) and the variance estimated from the mean-variance fit (red line). Why is this default behavior of DeSeq conservative?

For all genes, make a histogram (and/or use qqunif from the "gap" package) of the p-values for the null hypothesis that condition1 has the same distribution as condition2. How close are the p-values to uniform? Would you agree with an assertion that this data analysis path through DeSeq is slightly conservative? How many genes are significantly different at a 10% False Discovery Rate?

(4) Repeat all the steps in (3) except call " estimateDispersions" with the option sharingMode set to "gene-est-only". Now what is the relationship between the perGeneDispEsts (black points) and the data points used for inference ( yellow points). Do the p-values become more or less closer to uniform using this option? How many genes are significantly different at a 10% False Discovery Rate. Is this a more or less conservative option than the default choice?

(5) Now generate p-values from the test (e.g. t.test( myFrame[i,1:5], myFrame[i,6:10] )$p.value where i walks across all the rows of the frame. Are the p-values from the t-test uniform? How do the p-values from the t-test compare to the negative binomial p-values from the "gene-est-only" and default options for estimating dispersions in DeSeq. (That is, plot the p-values from the t-test vs. each of DeSeq’s p-values). Which test is the most conservative? Which test comes closest to a uniform distribution of p-values?