class17

Jaewon Kim

#SRA - First, create directory for today’s work (In terminal), $cd ~/Downloads/bimm143/R code/class17 $mkdir class17

- Next, move .pem file into work directory and log into remote laptop $ cp ~Desktop/class16/bimm143_louis.pem . $ chmod 400 “bimm143_louis.pem” $ ssh -i “bimm143_louis.pem” ubuntu@ec2-44-234-145-108.us-west-2.compute.amazonaws.com
- Get SRA toolkit $ curl -O https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/current/sratoolkit.current-ubuntu64.tar.gz $ tar -zxvf sratoolkit.current-ubuntu64.tar.gz (Or use unzip to remove .gz first and then use tar)
- Now, move to SRA toolkit folder $ ls $ cd sratoolkit.3.0.10-ubuntu64 $ pwd
- Export path as variable to shortcut full path. Check if pathway is registered well $ export PATH=$PATH:/home/ubuntu/sratoolkit.3.0.10-ubuntu64/bin $ prefetch –version

#Working with RNA-Seq data - Download data of record in interest $ prefetch SRR600956

-Run fastq $ fastq-dump SRR600956

- Print first few lines and select lines of SRR600956 result $ head SRR600956.fastq $ grep -c “@SRR600956” SRR600956.fastq
- Download another fastq file we’ll work with $ prefetch SRR2156848
- Run data extract, but per mate-pairs $ fastq-dump –split-3 SRR2156848
- Print first few lines and check number of sequences in the file

> `#Q. How would you check that these files with extension ‘.fastq’ actually look like what w`

$ head -3 SRR2156848_1.fastq

> `#Q. How could you check the number of sequences in each file?`

$ grep -c “@SRR2156848” *fastq

- Now download file to analyze and run fastq-dump. Then, check how many sequences are there in each files $ prefetch SRR2156849 SRR2156850 SRR2156851 $ fastq-dump –split-3 SRR2156849 SRR2156850 SRR2156851 $ grep -c “@SRR21568” *fastq

#Pseudoalignment - Download Kallisto $ wget https://github.com/pachterlab/kallisto/releases/download/v0.44 v0.44.0.tar.gz $ gunzip kallisto_linux-v0.44.0.tar.gz $ tar -xvf kallisto kallisto_linux-v0.44.0.tar

- Export pathway $ cd kallisto_linux-v0.44.0/ $ pwd $ export PATH=$PATH:/home/ubuntu/sratoolkit.3.0. ubuntu64/bin/kallisto_linux-v0.44.0

```
#Q. Can you run kallisto to print out it’s citation information?
```

$ kalisto cite

- Get human transcriptome $ wget ftp://ftp.ensembl.org/pub/release-67/fasta/homo_sapiens/cdna/Homo_ $ gunzip Homo_sapiens.GRCh37.67.cdna.all.fa.gz $ grep -c “>” Homo_sapiens.GRCh37.67.cdna.all.fa.gz (How many genes in the .fa?)
- Build transcript index $ kallisto index -i hg19.ensembl Homo_sapiens.GRCh37.67.cdna.all.fa

#Quantifying transcripts - Create file and run transcript qualification $ nano run.me.sh kallisto quant -i hg19.ensembl -o SRR2156848_quant SRR2156848_1.fastq SRR2156848_2.fastq kallisto quant -i hg19.ensembl -o SRR2156849_quant SRR2156849_1.fastq SRR2156849_2.fastq kallisto quant -i hg19.ensembl -o SRR2156850_quant SRR2156850_1.fastq SRR2156850_2.fastq kallisto quant -i hg19.ensembl -o SRR2156851_quant SRR2156851_1.fastq SRR2156851_2.fastq

```
#Q. Have a look at the TSV format versions of these files to understand their structure. W
```

They’re text files with read counts. Gene names follows Ensembl format.

- Transfer results to local folder (Move to local terminal for this) $ scp -r -i “bimm143_louis.pem” ubuntu@ec2-44-234-145-108.us-west-2.compute.amazonaws.com:~/SRR*_quant .

#Downstream analysis - Install tximport() BiocManager::install(“tximport”)

- Setup folder and files to read

```
library(tximport)

folders <- dir(pattern = "SRR21568*")
samples <- sub("_quant", "", folders)
files <- file.path(folders, "abundance.h5")
names(files) <- samples
```

```r
txi.kallisto <- tximport(files, type = "kallisto", txOut = TRUE)
```

```
1 2 3 4
```

```r
head(txi.kallisto$counts)
```

```
                SRR2156848 SRR2156849 SRR2156850 SRR2156851
ENST00000539570          0          0    0.00000          0
ENST00000576455          0          0    2.62037          0
ENST00000510508          0          0    0.00000          0
ENST00000474471          0          1    1.00000          0
ENST00000381700          0          0    0.00000          0
ENST00000445946          0          0    0.00000          0
```

- Check how many transcripts are in each samples. How many transcripts are detected in at least one sample?

```r
colSums(txi.kallisto$counts)
```

```
SRR2156848 SRR2156849 SRR2156850 SRR2156851
   2563611    2600800    2372309    2111474
```

```r
sum(rowSums(txi.kallisto$counts)>0)
```

```
[1] 94561
```

- Exclude zero reads from the data

```r
nonzero <- rowSums(txi.kallisto$counts) > 0
set.nonzero <- txi.kallisto$counts[nonzero, ]
head(set.nonzero)
```

```
                SRR2156848 SRR2156849 SRR2156850 SRR2156851
ENST00000576455    0.00000  0.0000000    2.62037   0.000000
ENST00000474471    0.00000  1.0000000    1.00000   0.000000
ENST00000420022    0.00000  2.0000000    4.00000   4.000000
ENST00000553856   10.96649  5.2579568   13.11994   2.720173
ENST00000556126    0.00000  0.0000000    4.00000   0.000000
ENST00000483851    0.00000  0.6084246    0.00000   0.000000
```

- Filter out genes where transcript level stays constant

```r
nonconstant <- apply(set.nonzero, 1, sd) > 0 #Find rows with sd >0
set.nonconstant <- set.nonzero[nonconstant, ] #seperate out those rows
head(set.nonconstant)
```

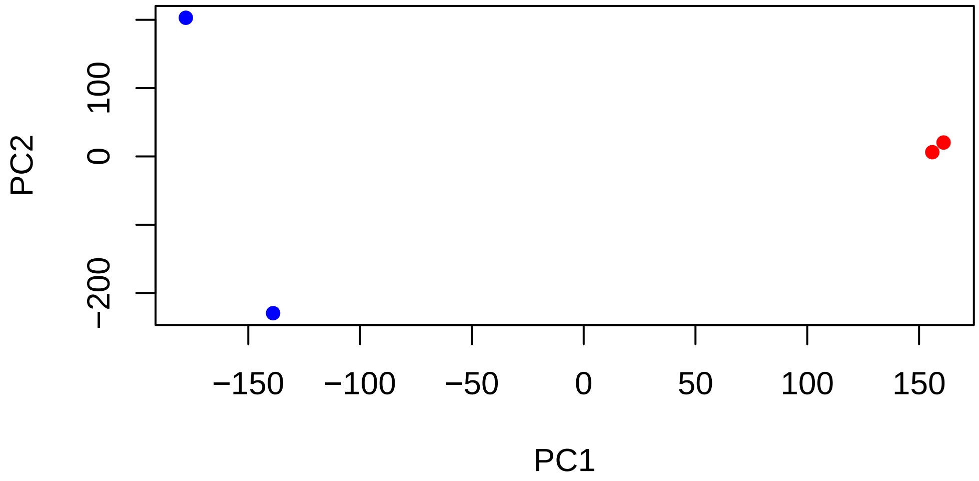
```
                SRR2156848 SRR2156849 SRR2156850 SRR2156851
ENST00000576455    0.00000  0.0000000    2.62037   0.000000
ENST00000474471    0.00000  1.0000000    1.00000   0.000000
ENST00000420022    0.00000  2.0000000    4.00000   4.000000
ENST00000553856   10.96649  5.2579568   13.11994   2.720173
ENST00000556126    0.00000  0.0000000    4.00000   0.000000
ENST00000483851    0.00000  0.6084246    0.00000   0.000000
```

#PCA

```r
pca <- prcomp(t(set.nonconstant), scale = TRUE)
summary(pca)
```

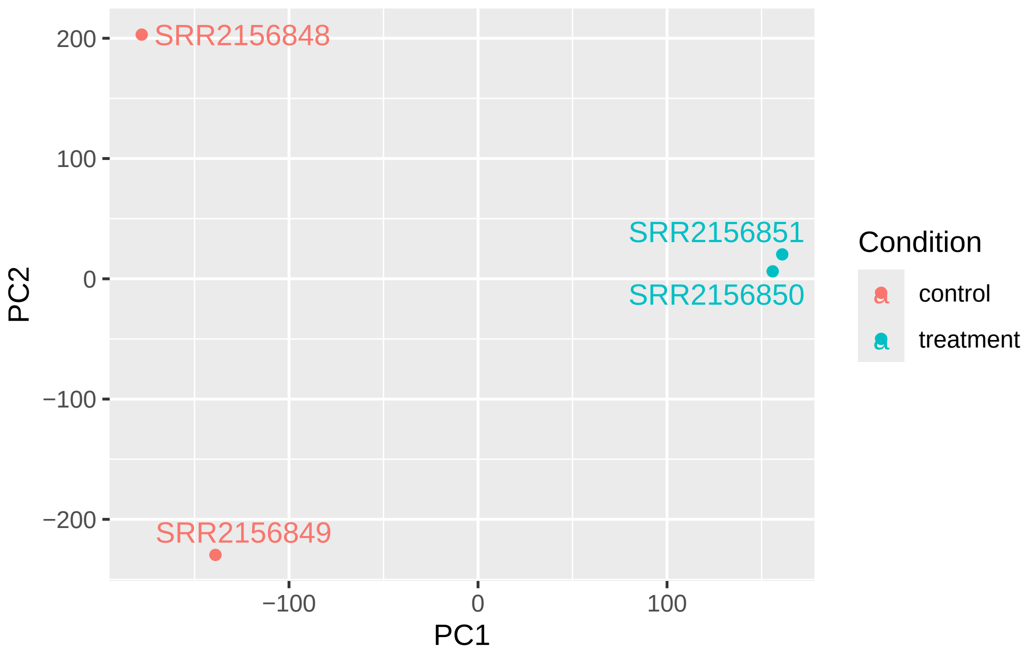
```
Importance of components:
                            PC1      PC2      PC3   PC4
Standard deviation     183.6379 177.3605 171.3020 1e+00
Proportion of Variance   0.3568   0.3328   0.3104 1e-05
Cumulative Proportion    0.3568   0.6895   1.0000 1e+00
```

```r
plot(pca$x[,1], pca$x[,2],
     col=c("blue","blue","red","red"),
     xlab = "PC1", ylab = "PC2", pch = 16)
```

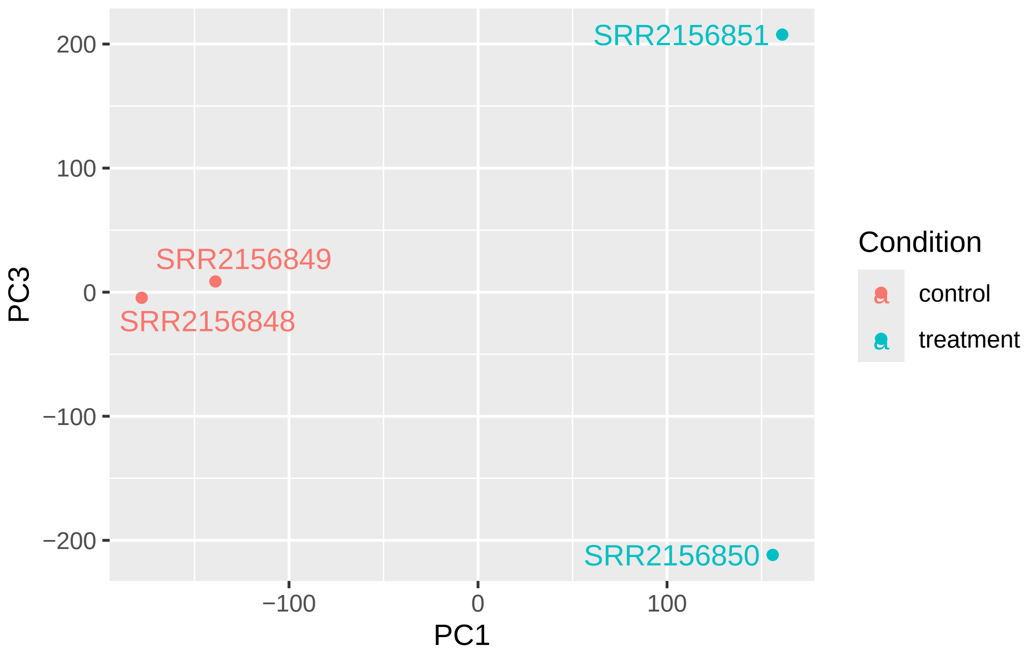


Q. Use ggplot to make a similar figure of PC1 vs PC2 and a seperate figure PC1 vs PC3 and PC2 vs PC3.

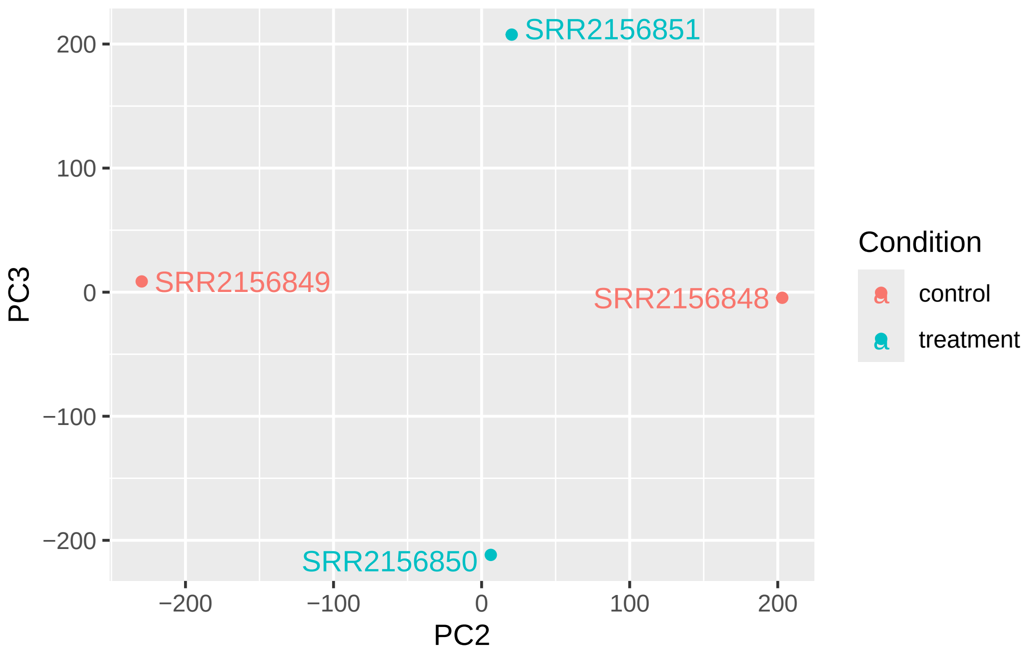
```r
library(ggplot2)
library(ggrepel)

colData <- data.frame(condition = factor(rep(c("control", "treatment"), each = 2)))
rownames(colData) <- colnames(txi.kallisto$counts)

y <- as.data.frame(pca$x)
y$Condition <- as.factor(colData$condition)

ggplot(y) +
  aes(PC1, PC2, col = Condition) +
  geom_point() +
  geom_text_repel(label = rownames(y))
```



```
ggplot(y) +
  aes(PC1, PC3, col = Condition) +
  geom_point() +
  geom_text_repel(label = rownames(y))
```



```
ggplot(y) +
  aes(PC2, PC3, col = Condition) +
  geom_point() +
  geom_text_repel(label = rownames(y))
```



#Differential expression analysis

```
#|message: FALSE
library(DESeq2)
```

```
        : S4Vectors


        : stats4


        : BiocGenerics



        : 'BiocGenerics'


The following objects are masked from 'package:stats':

    IQR, mad, sd, var, xtabs
```

```
The following objects are masked from 'package:base':

    anyDuplicated, aperm, append, as.data.frame, basename, cbind,
    colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
    get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
    match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
    Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
    table, tapply, union, unique, unsplit, which.max, which.min


        : 'S4Vectors'

The following object is masked from 'package:utils':

    findMatches

The following objects are masked from 'package:base':

    expand.grid, I, unname

         : IRanges


        : 'IRanges'

The following object is masked from 'package:grDevices':

    windows

         : GenomicRanges

         : GenomeInfoDb

         : SummarizedExperiment

         : MatrixGenerics

         : matrixStats
        : 'MatrixGenerics'


The following objects are masked from 'package:matrixStats':

    colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
    colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
    colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
    colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
    colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
    colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
    colWeightedMeans, colWeightedMedians, colWeightedSds,
    colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
    rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
    rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
    rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
    rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
    rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
    rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
    rowWeightedSds, rowWeightedVars

         : Biobase

Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages 'citation("pkgname")'.


        : 'Biobase'


The following object is masked from 'package:MatrixGenerics':

    rowMedians


The following objects are masked from 'package:matrixStats':

    anyMissing, rowMedians
```

```r
Table <- data.frame(condition = factor(rep(c("control", "treatment"), each = 2)))
rownames(Table) <- colnames(txi.kallisto$counts)

dds <- DESeqDataSetFromTximport(txi.kallisto,
                                Table,
                                ~condition)
```

```
using counts and average transcript lengths from tximport
```

```r
dds <- DESeq(dds)
```

```
estimating size factors
```

```
using 'avgTxLength' from assays(dds), correcting for library size
```

```
estimating dispersions
```

```
gene-wise dispersion estimates
```

```
mean-dispersion relationship
```

```
-- note: fitType='parametric', but the dispersion trend was not well captured by the
   function: y = a/x + b, and a local regression fit was automatically substituted.
   specify fitType='local' or 'mean' to avoid this message next time.
```

```
final dispersion estimates
```

```
fitting model and testing
```

```r
res <- results(dds)
head(res)
```

```
log2 fold change (MLE): condition treatment vs control
Wald test p-value: condition treatment vs control
DataFrame with 6 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat    pvalue
                <numeric>      <numeric> <numeric> <numeric> <numeric>
ENST00000539570  0.000000             NA        NA        NA        NA
ENST00000576455  0.761453       3.155061   4.86052 0.6491203  0.516261
ENST00000510508  0.000000             NA        NA        NA        NA
ENST00000474471  0.484938       0.181923   4.24871 0.0428185  0.965846
ENST00000381700  0.000000             NA        NA        NA        NA
ENST00000445946  0.000000             NA        NA        NA        NA
                     padj
                <numeric>
ENST00000539570        NA
ENST00000576455        NA
ENST00000510508        NA
ENST00000474471        NA
ENST00000381700        NA
ENST00000445946        NA
```