hw16

Meha Thakur (A16020450)

Table of contents


```
library(BiocManager)
```

```
Bioconductor version '3.16' is out-of-date; the current release version '3.22'
  is available with R version '4.5'; see https://bioconductor.org/install
```

```
#BiocManager::install("tximport")
library(tximport)
```

Downstream analysis

since each kallisto has its own output, need to import using tximport

```
# setup the folder and filenames to read
folders <- dir(pattern="SRR21568*")
samples <- sub("_quant", "", folders)
files <- file.path( folders, "abundance.h5" )
names(files) <- samples

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

```
1 2 3 4
```

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

```
                SRR2156849 SRR2156850 SRR2156851 SRR2156852
ENST00000539570          0    0.00000          0          0
ENST00000576455          0    2.62037          0          0
ENST00000510508          0    0.00000          0          0
ENST00000474471          1    1.00000          0          1
ENST00000381700          0    0.00000          0          0
ENST00000445946          0    0.00000          0          1
```

txi.kallisto imports the each gene (rows) and each dataset(columns?)

how many counts for each sample (how many transcripts). We can also detect how many transcripts are detected in at least one sample

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

```
SRR2156849 SRR2156850 SRR2156851 SRR2156852
   2600800    2372309    2111474    3144476
```

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

```
[1] 95382
```

now we filter out annotated samples with no reads, and no change

```
#no reads
to.keep <- rowSums(txi.kallisto$counts) > 0
kset.nonzero <- txi.kallisto$counts[to.keep,]

#no change
keep2 <- apply(kset.nonzero,1,sd)>0 #row-wise standard deviations only keeps rows with nonze
x <- kset.nonzero[keep2,]
```

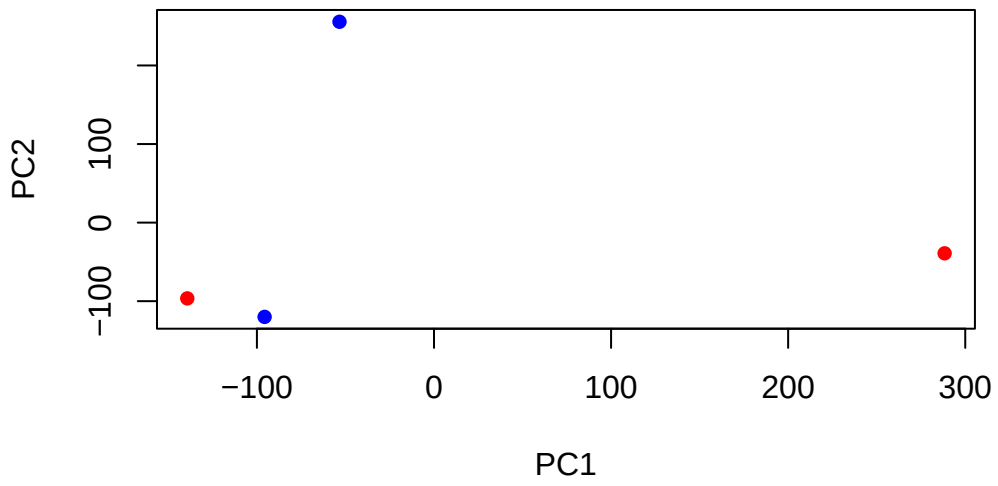
PCA analysis

```
pca <- prcomp(t(x), scale=TRUE)
summary(pca)
```

```
Importance of components:
                            PC1      PC2      PC3     PC4
Standard deviation     195.4181 173.7535 164.2414 1.80269
Proportion of Variance   0.4005   0.3166   0.2829 0.00003
Cumulative Proportion    0.4005   0.7171   1.0000 1.00000
```

use the first two PCs to visualse transcriptomic profiles

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



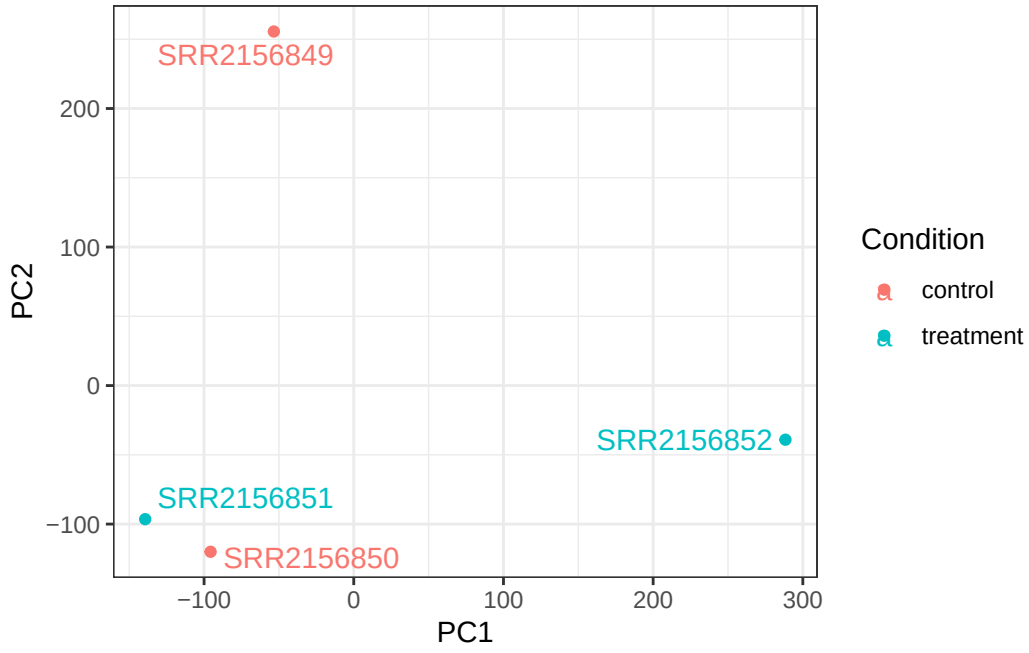
using ggplot for PC1 vs PC2, PC1 vs. PC3

```
library(ggplot2)
library(ggrepel)

# Make metadata object for the samples
colData <- data.frame(condition = factor(rep(c("control", "treatment"), each = 2)))
rownames(colData) <- colnames(txi.kallisto$counts)

# Make the data.frame for ggplot
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)) +
  theme_bw()
```



DESeq analysis

```
library(DESeq2)
```

```
Loading required package: S4Vectors
```

```
Loading required package: stats4

Loading required package: BiocGenerics


Attaching package: '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


Attaching package: 'S4Vectors'

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

    expand.grid, I, unname

Loading required package: IRanges

Loading required package: GenomicRanges

Loading required package: GenomeInfoDb

Loading required package: SummarizedExperiment

Loading required package: MatrixGenerics

Loading required package: matrixStats
```

```
Attaching package: '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

Loading required package: Biobase

Welcome to Bioconductor

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


Attaching package: 'Biobase'

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

    rowMedians

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

    anyMissing, rowMedians
```

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

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

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

DESeq

```
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
```

```
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  0.798927       -3.14432   4.85393 -0.647789  0.517121
ENST00000510508  0.000000             NA        NA        NA        NA
ENST00000474471  0.710994       -1.13012   3.70581 -0.304958  0.760398
ENST00000381700  0.000000             NA        NA        NA        NA
ENST00000445946  0.200530        1.31201   4.98300  0.263298  0.792321
                     padj
                <numeric>
ENST00000539570        NA
ENST00000576455        NA
ENST00000510508        NA
ENST00000474471        NA
ENST00000381700        NA
ENST00000445946        NA
```