# SubramanianPastore_3500FinalProject

*load packages*

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v stringr 1.4.0
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   1.4.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(DESeq2)
```

```
## Warning: package 'DESeq2' was built under R version 4.0.3

## Loading required package: S4Vectors

## Warning: package 'S4Vectors' was built under R version 4.0.3

## Loading required package: stats4

## Loading required package: BiocGenerics

## Warning: package 'BiocGenerics' was built under R version 4.0.3
```

```
## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, 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 object is masked from 'package:tidyr':
##
##     expand

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

## Warning: package 'IRanges' was built under R version 4.0.3

##
## Attaching package: 'IRanges'
```

```
## The following object is masked from 'package:purrr':
##
##     reduce

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

## Loading required package: GenomicRanges

## Warning: package 'GenomicRanges' was built under R version 4.0.3

## Loading required package: GenomeInfoDb

## Warning: package 'GenomeInfoDb' was built under R version 4.0.4

## Loading required package: SummarizedExperiment

## Warning: package 'SummarizedExperiment' was built under R version 4.0.3

## Loading required package: MatrixGenerics

## Warning: package 'MatrixGenerics' was built under R version 4.0.3

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':
##
##     count

##
## 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
```

```
## Warning: package 'Biobase' was built under R version 4.0.3
```

```
## 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
```

```
library(pheatmap)
library(ggrepel)
library(NOISeq)
```

```
## Warning: package 'NOISeq' was built under R version 4.0.3
```

```
## Loading required package: splines
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following object is masked from 'package:S4Vectors':
##
##     expand
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
list.files()
```

```
##  [1] "B04_S26.counts"
##  [2] "B08_S56.counts"
##  [3] "C05_S33.counts"
##  [4] "Counts"
##  [5] "E03_S21.counts"
##  [6] "F03_S22.counts"
```

```
##  [7] "G10_S76.counts"
##  [8] "G12_S92.counts"
##  [9] "I05_S126.counts"
## [10] "K12_S181.counts"
## [11] "L03_S113.counts"
## [12] "L12_S182.counts"
## [13] "length.txt"
## [14] "M04_S122.counts"
## [15] "N04_S123.counts"
## [16] "N06_S139.counts"
## [17] "N10_S170.counts"
## [18] "N12_S184.counts"
## [19] "O01_S100.counts"
## [20] "O06_S140.counts"
## [21] "O12_S185.counts"
## [22] "P01_S101.counts"
## [23] "P02_S109.counts"
## [24] "P08_S156.counts"
## [25] "SubramanianPastore_3500FinalProject.Rproj"
## [26] "SubramanianPastore_5300FinalProject_files"
## [27] "SubramanianPastore_5300FinalProject.html"
## [28] "SubramanianPastore_5300FinalProject.pdf"
## [29] "SubramanianPastore_5300FinalProject.Rmd"
## [30] "T02_S198.counts"
```

*read in count data*

```
#create a empty dataframe called m to merge the data into
m = data.frame()
# using for loop read all the count files in the count_dir path
for (i in list.files(pattern = ".counts")) {
  print(paste0("reading file: ", i))
  #read file as a data frame

  f <- read.table(i, sep = "\t", header = TRUE)
  #rename the columns
  colnames(f) <- c("gene_id", substr(i, 1, nchar(i)-7))
  #copy the data to another dataframe called f1
  f1 <- subset(f, select= c("gene_id", substr(i, 1, nchar(i)-7)))

  #if the m is empty just copy the f to m
  if(length(m) == 0){
    m = f1

  } else
  {
    #if the dataframe is not empty then merge the data
    m <- merge(m, f1, by.x = "gene_id", by.y = "gene_id")
  }
  rm(f1)
}
```

```
## [1] "reading file: B04_S26.counts"
## [1] "reading file: B08_S56.counts"
```

```
## [1] "reading file: C05_S33.counts"
## [1] "reading file: E03_S21.counts"
## [1] "reading file: F03_S22.counts"
## [1] "reading file: G10_S76.counts"
## [1] "reading file: G12_S92.counts"
## [1] "reading file: I05_S126.counts"
## [1] "reading file: K12_S181.counts"
## [1] "reading file: L03_S113.counts"
## [1] "reading file: L12_S182.counts"
## [1] "reading file: M04_S122.counts"
## [1] "reading file: N04_S123.counts"
## [1] "reading file: N06_S139.counts"
## [1] "reading file: N10_S170.counts"
## [1] "reading file: N12_S184.counts"
## [1] "reading file: O01_S100.counts"
## [1] "reading file: O06_S140.counts"
## [1] "reading file: O12_S185.counts"
## [1] "reading file: P01_S101.counts"
## [1] "reading file: P02_S109.counts"
## [1] "reading file: P08_S156.counts"
## [1] "reading file: T02_S198.counts"
```

```
#grab the rows from the 1st colum and use it as the row-names in the dataframe
rownames(m) <- m[,1]

# remove the column-1 (gene_ids) from the data frame using dplyr::select function
m <- select(m, "P02_S109", "P08_S156", "N04_S123", "O01_S100", "B08_S56", "N06_S139", "N12_S184", "N10_S
rm(f)
```

*metadata*

```
Sample = c("P02_S109", "P01_S101", "B08_S56", "F03_S22", "N10_S170", "O01_S100", "K12_S181", "N04_S123"
BodyRegion = c("Abd", "Abd", "Abd", "W3", "W3", "W3", "W2", "W2", "W2", "Pro", "Pro", "Pro", "Abd", "Abd
Instar = c("L4", "L4", "L4", "L4", "L4", "L4", "L4", "L4", "L4", "L4", "L4", "L4", "L5", "L5", "L5", "L5
myfactors <- data.frame(Sample, Instar, BodyRegion)
myfactors
```

```
##      Sample Instar BodyRegion
## 1  P02_S109     L4        Abd
## 2  P01_S101     L4        Abd
## 3   B08_S56     L4        Abd
## 4   F03_S22     L4         W3
## 5  N10_S170     L4         W3
## 6  O01_S100     L4         W3
## 7  K12_S181     L4         W2
## 8  N04_S123     L4         W2
## 9  N12_S184     L4         W2
## 10  E03_S21     L4        Pro
## 11 N06_S139     L4        Pro
## 12 P08_S156     L4        Pro
## 13  B04_S26     L5        Abd
## 14 L03_S113     L5        Abd
## 15 L12_S182     L5        Abd
```

```
## 16 I05_S126     L5         W3
## 17 M04_S122     L5         W3
## 18 O12_S185     L5         W3
## 19  C05_S33     L5         W2
## 20 T02_S198     L5         W2
## 21  G10_S76     L5        Pro
## 22  G12_S92     L5        Pro
## 23 O06_S140     L5        Pro
```

*check and order metadata and data*

```
# first check wheter all the columns in dataframe(m) and myfactor is present
all(colnames(m) %in% myfactors$Sample)
```

```
## [1] TRUE
```

```
# Then check the order is correct
all(rownames(m) == myfactors$Sample[1])
```

```
## [1] FALSE
```

```
# Then order them according to the Sample names
m <- m[, myfactors$Sample]

# Now check the order is correct after sorting
all(colnames(m) == myfactors$Sample)
```

```
## [1] TRUE
```

*length information into metadata*

```
# import length information to a dataframe
df_length <- read.table("length.txt", sep = "\t", header = TRUE, row.names = 1)

# create a vector to hold the length information
mylength <- setNames(object = df_length$length, row.names(df_length))
head(mylength)
```

```
##  O12_S185_TRINITY_DN496_c0_g1_i6.p1  P02_S109_TRINITY_DN115_c0_g1_i3.p1
##                               49635                               30330
##   F03_S22_TRINITY_DN693_c0_g1_i7.p1  B04_S26_TRINITY_DN1673_c0_g1_i2.p1
##                               28749                               27624
##  F03_S22_TRINITY_DN471_c0_g3_i13.p1 N12_S184_TRINITY_DN5253_c0_g2_i2.p1
##                               24927                               20889
```

*create NOISeq object*

```
mydata <- readData(data = m, factors = myfactors, length = mylength)
mydata
```

```
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 30231 features, 23 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: P02_S109 P01_S101 ... O06_S140 (23 total)
##   varLabels: Sample Instar BodyRegion
##   varMetadata: labelDescription
## featureData
##   featureNames: B04_S26_TRINITY_DN0_c0_g1_i1.p1
##     B04_S26_TRINITY_DN10003_c0_g1_i1.p1 ...
##     T02_S198_TRINITY_DN9996_c0_g6_i1.p1 (30231 total)
##   fvarLabels: Length
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'
## Annotation:
```

```
head(assayData(mydata)$exprs)
```

```
##                                    P02_S109   P01_S101    B08_S56
## B04_S26_TRINITY_DN0_c0_g1_i1.p1     22507.60000 22852.7000 14429.0000
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1   110.00000    91.0000   101.0000
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1     4.84882     0.0000     0.0000
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1     3.15118     0.0000     2.0000
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1     0.00000     0.0000     0.0000
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1    99.55490    62.9736    84.3052
##                                     F03_S22 N10_S170 O01_S100 K12_S181
## B04_S26_TRINITY_DN0_c0_g1_i1.p1     3.86454e+04  3037.12  4425.86  27021.2
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1 1.63000e+02    12.00    65.00    151.0
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1 1.00000e+00     0.00     0.00      0.0
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1 1.35388e-08     0.00     0.00      0.0
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1 0.00000e+00     0.00     0.00      0.0
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1 3.30000e+01     0.00    11.00      6.0
##                                    N04_S123    N12_S184     E03_S21  N06_S139
## B04_S26_TRINITY_DN0_c0_g1_i1.p1      1148.51 1.22280e+04 3.61043e+04 3321.4200
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1    81.00 1.27000e+02 2.08000e+02   78.0000
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1     1.00 2.00000e+00 2.00000e+00    0.0000
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1     0.00 7.53143e-07 1.47696e-07    0.0000
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1     0.00 0.00000e+00 0.00000e+00    0.0000
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1     5.00 9.00000e+00 5.70566e+01   15.6301
##                                    P08_S156     B04_S26   L03_S113   L12_S182
## B04_S26_TRINITY_DN0_c0_g1_i1.p1      5449.51 15166.80000 23999.1000 40410.5000
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1    57.00    70.00000    56.0000   148.0000
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1     0.00   160.39200     0.0000   151.7620
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1     0.00     3.57241     0.0000    12.0277
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1     0.00    54.03550     3.0000    59.2100
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1    12.00    70.73680    33.3389    90.2680
##                                     I05_S126  M04_S122 O12_S185     C05_S33
## B04_S26_TRINITY_DN0_c0_g1_i1.p1     55811.6000 5377.4800    14261 9.61914e+03
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1   116.0000  113.0000      115 1.10000e+02
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1     1.0000    0.0000        2 1.00000e+00
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1     0.0000    0.0000        0 3.72771e-08
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1     1.0000    0.0000        0 0.00000e+00
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1    15.8189   18.4028       10 1.00000e+00
```

```
##                                    T02_S198 G10_S76     G12_S92 O06_S140
## B04_S26_TRINITY_DN0_c0_g1_i1.p1       790.29 9933.45 6.30637e+03  706.948
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1   138.00  140.00 3.18000e+02   45.000
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1     1.00    1.00 1.00000e+00    0.000
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1     0.00    0.00 8.96331e-08    0.000
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1     1.00    0.00 0.00000e+00    0.000
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1     7.00    5.00 8.94118e+00    3.000
```

```
head(featureData(mydata)@data)
```

```
##                                    Length
## B04_S26_TRINITY_DN0_c0_g1_i1.p1      5904
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1   375
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1   963
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1   993
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1   972
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1  1500
```

```
head(pData(mydata))
```

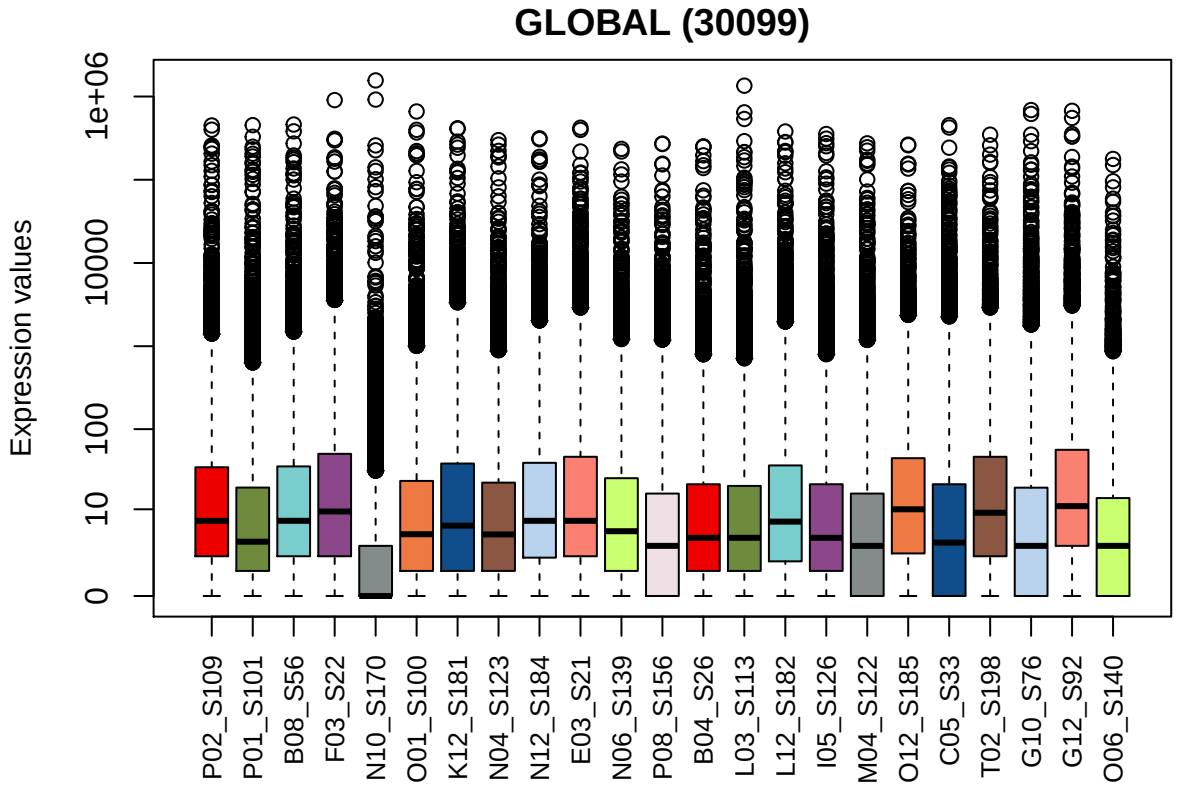
```
##            Sample Instar BodyRegion
## P02_S109 P02_S109     L4        Abd
## P01_S101 P01_S101     L4        Abd
## B08_S56   B08_S56     L4        Abd
## F03_S22   F03_S22     L4         W3
## N10_S170 N10_S170     L4         W3
## O01_S100 O01_S100     L4         W3
```

*Count Distribution Plot*

```
countsbio = dat(mydata, factor = NULL, type = "countsbio")
```

```
## [1] "Warning: 132 features with 0 counts in all samples are to be removed for this analysis."
## [1] "Count distributions are to be computed for:"
##  [1] "P02_S109" "P01_S101" "B08_S56"  "F03_S22"  "N10_S170" "O01_S100"
##  [7] "K12_S181" "N04_S123" "N12_S184" "E03_S21"  "N06_S139" "P08_S156"
## [13] "B04_S26"  "L03_S113" "L12_S182" "I05_S126" "M04_S122" "O12_S185"
## [19] "C05_S33"  "T02_S198" "G10_S76"  "G12_S92"  "O06_S140"
```

```
explo.plot(countsbio, toplot = 1, samples = NULL, plottype = "boxplot")
```



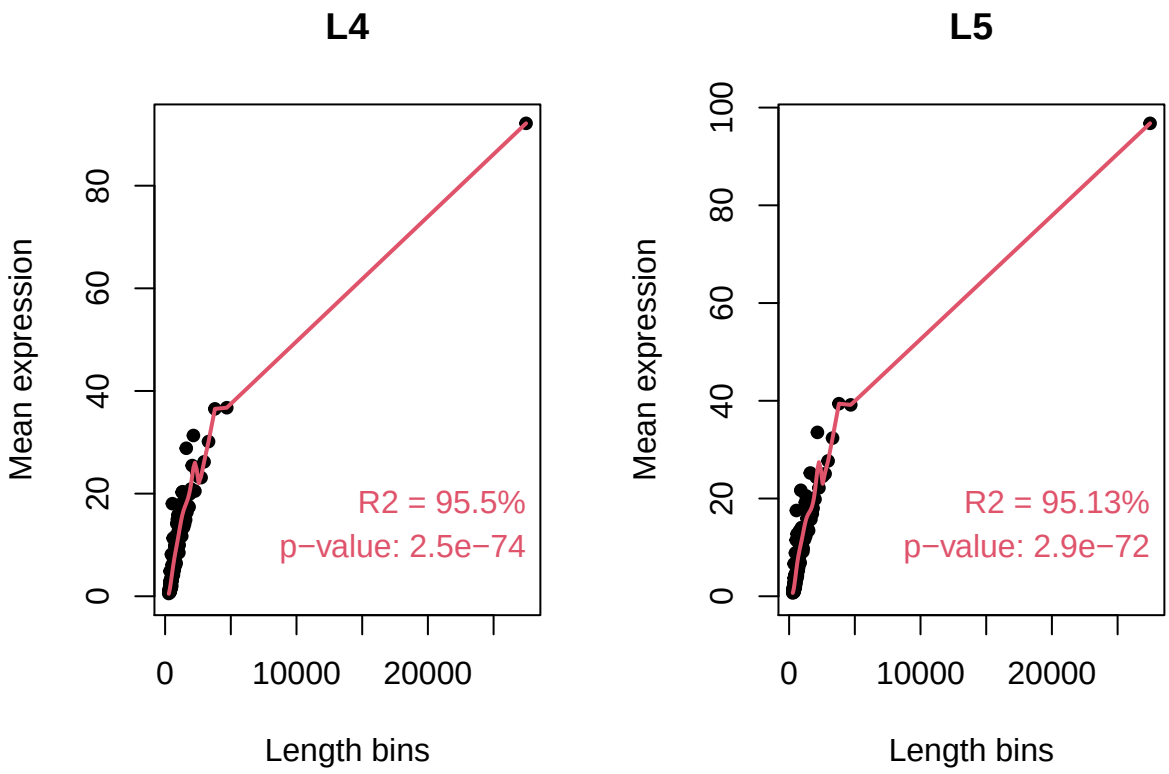

*Length Bias Plot*

```
mylenghtbias = dat(mydata, factor = "Instar", type = "lengthbias")
```

```
## [1] "Warning: 132 features with 0 counts in all samples are to be removed for this analysis."
## [1] "Length bias detection information is to be computed for:"
## [1] "L4" "L5"
## [1] "L4"
##
## Call:
## lm(formula = datos[, i] ~ bx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.5146 -0.9452 -0.2648  0.1605 13.1604
##
## Coefficients: (3 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   92.149      2.446  37.673  < 2e-16 ***
## bx1          -91.668      2.661 -34.452  < 2e-16 ***
## bx2          -90.710      2.703 -33.561  < 2e-16 ***
## bx3          -85.566      2.835 -30.184  < 2e-16 ***
## bx4          -81.492      2.839 -28.705  < 2e-16 ***
## bx5          -75.875      2.991 -25.366  < 2e-16 ***
## bx6          -73.996      3.219 -22.987  < 2e-16 ***
## bx7          -71.258      3.675 -19.390  < 2e-16 ***
## bx8          -62.626      4.530 -13.823  < 2e-16 ***
## bx9          -75.788      6.746 -11.234  < 2e-16 ***
## bx10         -56.343     15.744  -3.579 0.000498 ***
```

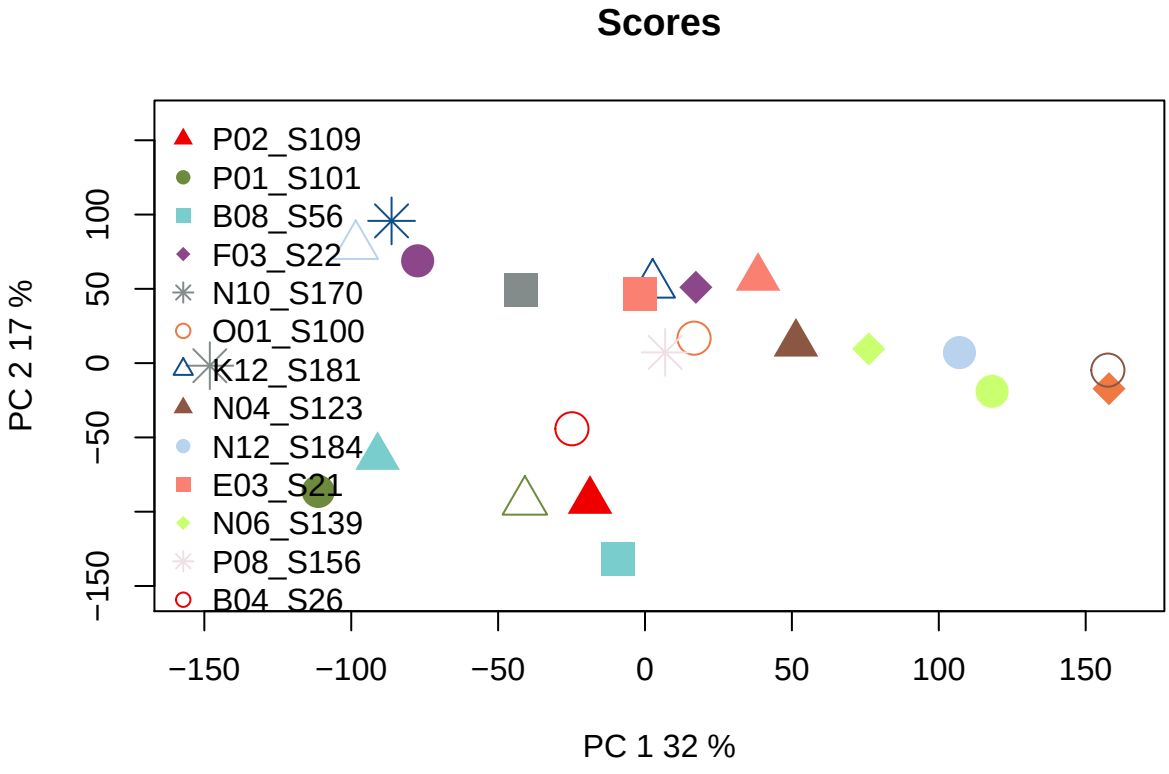
```
## bx11         -98.920     72.751  -1.360 0.176450
## bx12         118.152    349.156   0.338 0.735652
## bx13        -309.437    511.777  -0.605 0.546556
## bx14         -57.914      3.615 -16.018  < 2e-16 ***
## bx15              NA         NA      NA       NA
## bx16              NA         NA      NA       NA
## bx17              NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.446 on 121 degrees of freedom
## Multiple R-squared:  0.955,  Adjusted R-squared:  0.9498
## F-statistic: 183.3 on 14 and 121 DF,  p-value: < 2.2e-16
##
## [1] "L5"
##
## Call:
## lm(formula = datos[, i] ~ bx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2678 -1.2215 -0.3531  0.2808 12.0227
##
## Coefficients: (3 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   96.775      2.640  36.659  < 2e-16 ***
## bx1          -96.108      2.872 -33.468  < 2e-16 ***
## bx2          -95.252      2.917 -32.654  < 2e-16 ***
## bx3          -89.058      3.059 -29.109  < 2e-16 ***
## bx4          -85.637      3.064 -27.950  < 2e-16 ***
## bx5          -80.254      3.228 -24.860  < 2e-16 ***
## bx6          -79.466      3.474 -22.874  < 2e-16 ***
## bx7          -76.561      3.966 -19.303  < 2e-16 ***
## bx8          -64.716      4.889 -13.236  < 2e-16 ***
## bx9          -80.913      7.281 -11.113  < 2e-16 ***
## bx10         -54.029     16.992  -3.180  0.00187 **
## bx11        -126.243     78.516  -1.608  0.11047
## bx12         235.922    376.823   0.626  0.53244
## bx13        -486.224    552.330  -0.880  0.38043
## bx14         -60.211      3.902 -15.431  < 2e-16 ***
## bx15              NA         NA      NA       NA
## bx16              NA         NA      NA       NA
## bx17              NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.64 on 121 degrees of freedom
## Multiple R-squared:  0.9513, Adjusted R-squared:  0.9456
## F-statistic: 168.7 on 14 and 121 DF,  p-value: < 2.2e-16
```

```
explo.plot(mylenghtbias, samples = NULL, toplot = "global")
```

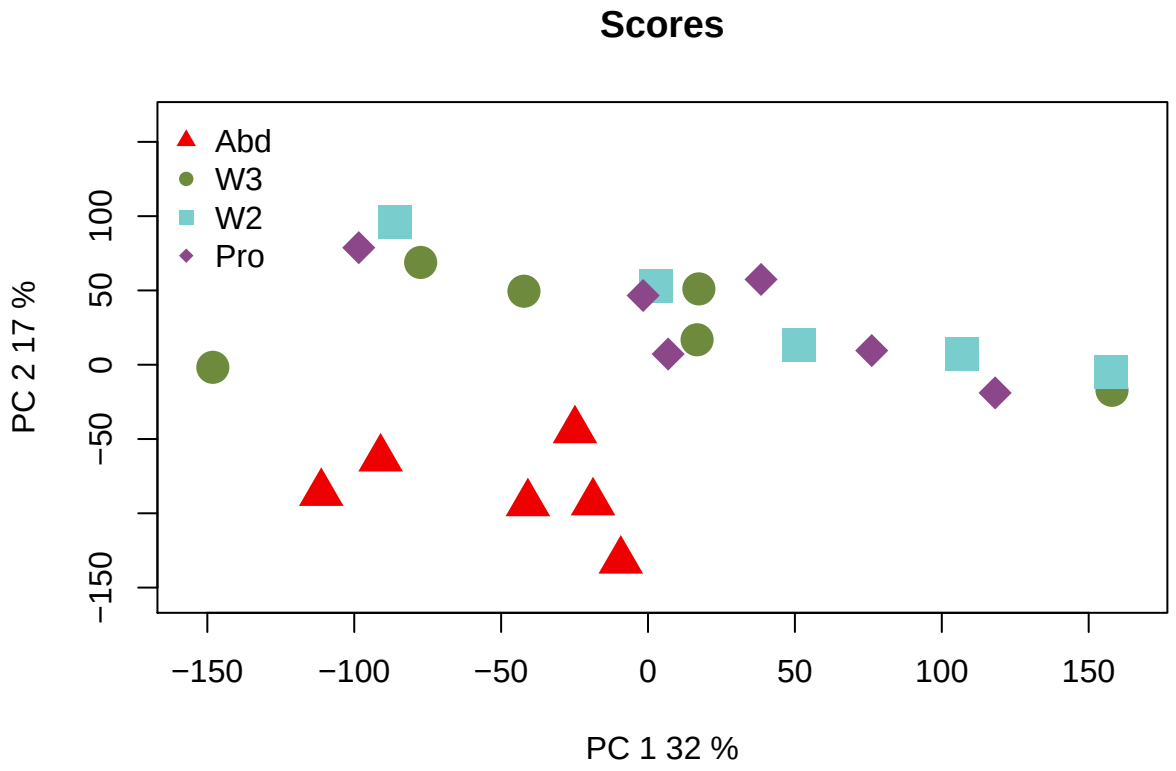



*PCA Plots*

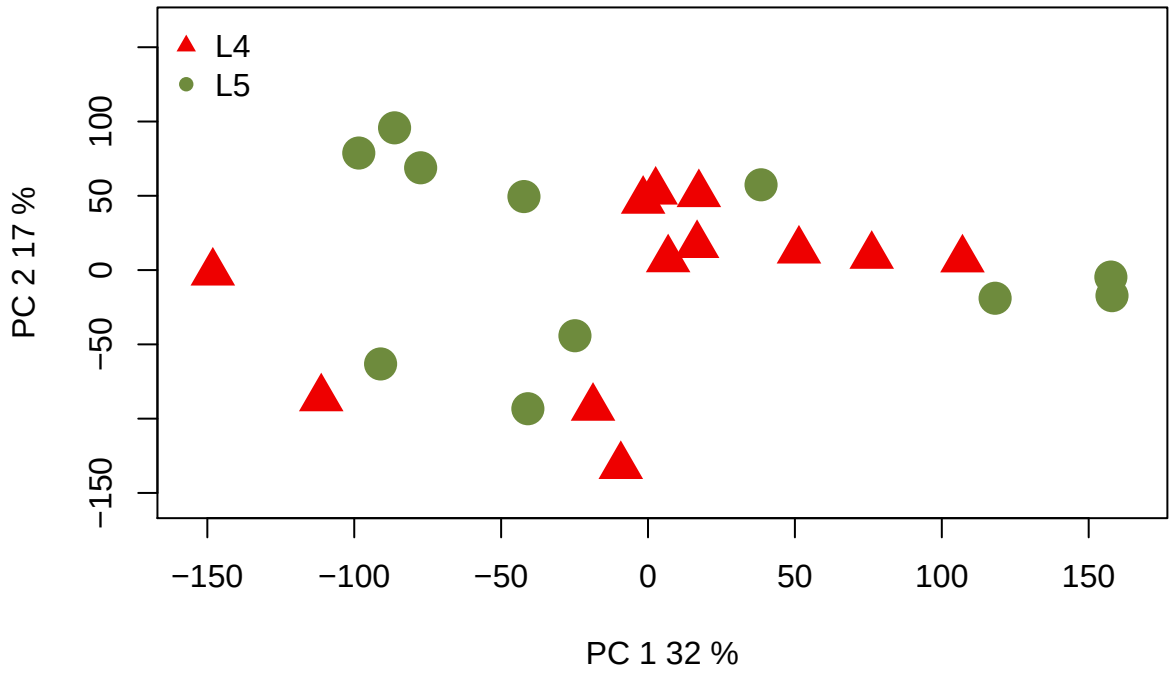
```
myPCA = dat(mydata, type = "PCA")
explo.plot(myPCA, factor= NULL)
```



```
myPCA = dat(mydata, type = "PCA")
explo.plot(myPCA, factor= "BodyRegion")
```




```
myPCA = dat(mydata, type = "PCA")
explo.plot(myPCA, factor= "Instar")
```




```
help(explo.plot)
```

*Differential Expression*

```
##INSTAR
# lc = 1 RPKM values with length correction
# lc = 0 no length correction applied
mynoiseq.bio <- noiseqbio(mydata,
                          factor = c("Instar", "BodyRegion"),
                          norm = "rpkm",
                          random.seed = 12345,
                          lc = 1)
```

```
## Computing Z values...

## Warning in if (factor == i) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (factor == i) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (factor == i) {: the condition has length > 1 and only the first
## element will be used

## Filtering out low count features...
## 25247 features are to be kept for differential expression analysis with filtering method 1
## [1] "r = 1"
## [1] "r = 2"
```

```
## [1] "r = 3"
## [1] "r = 4"
## [1] "r = 5"
## [1] "r = 6"
## [1] "r = 7"
## [1] "r = 8"
## [1] "r = 9"
## [1] "r = 10"
## [1] "r = 11"
## [1] "r = 12"
## [1] "r = 13"
## [1] "r = 14"
## [1] "r = 15"
## [1] "r = 16"
## [1] "r = 17"
## [1] "r = 18"
## [1] "r = 19"
## [1] "r = 20"
## [1] "r = 21"
## [1] "r = 22"
## [1] "r = 23"
## [1] "r = 24"
## [1] "r = 25"
## [1] "r = 26"
## [1] "r = 27"
## [1] "r = 28"
## [1] "r = 29"
## [1] "r = 30"
## [1] "r = 31"
## [1] "r = 32"
## [1] "r = 33"
## [1] "r = 34"
## [1] "r = 35"
## [1] "r = 36"
## [1] "r = 37"
## [1] "r = 38"
## [1] "r = 39"
## [1] "r = 40"
## [1] "r = 41"
## [1] "r = 42"
## [1] "r = 43"
## [1] "r = 44"
## [1] "r = 45"
## [1] "r = 46"
## [1] "r = 47"
## [1] "r = 48"
## [1] "r = 49"
## [1] "r = 50"
## Computing probability of differential expression...
## p0 = 0.522339792908066
## Probability
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   0.352   0.562   0.480   0.636   1.000    4984
```

```
head(mynoiseq.bio@results[[1]])
```

```
##                                         L4_mean    L5_mean      theta      prob
## B04_S26_TRINITY_DN0_c0_g1_i1.p1     463.3159475 520.678831 -0.2066888 0.6612638
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1  52.0856736  61.549930 -0.4054570 0.5848214
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1   0.2333227   6.442942 -1.5091912 0.7613106
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1          NA         NA         NA        NA
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1   0.1068766   2.357195 -1.3767385 0.7127305
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1   4.0176135   3.191645  0.1514477 0.2304759
##                                          log2FC length
## B04_S26_TRINITY_DN0_c0_g1_i1.p1     -0.1683974   5904
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1 -0.2408706    375
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1 -4.7873210    963
## B04_S26_TRINITY_DN10035_c0_g2_i4.p1         NA    993
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1 -4.4630529    972
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1  0.3320387   1500
```

```
##differentially regulated genes
mynoiseq.bio.deg = degenes(mynoiseq.bio, q = 0.9 , M= NULL)
```

```
## [1] "45 differentially expressed features"
```

```
##upregulated genes in instar 4
mynoiseq.bio.deg_up = degenes(mynoiseq.bio, q = 0.9 , M= "up")
```

```
## [1] "11 differentially expressed features (up in first condition)"
```

```
head(mynoiseq.bio.deg_up)
```

```
##                                           L4_mean    L5_mean    theta      prob
## B08_S56_TRINITY_DN676_c0_g1_i1.p1      250.09785  2.5511601 4.000371 0.9999988
## P08_S156_TRINITY_DN127_c0_g1_i1.p1     111.70503  0.3427869 4.114668 0.9820470
## N04_S123_TRINITY_DN50770_c0_g1_i1.p1 1798.32669 92.7203416 3.195376 0.9447717
## B04_S26_TRINITY_DN5869_c0_g1_i1.p1     448.86055 75.5690681 2.448022 0.9304123
## B08_S56_TRINITY_DN291_c0_g1_i1.p1      422.67382 82.5114696 2.453980 0.9287008
## N04_S123_TRINITY_DN18231_c0_g1_i4.p1    23.20452  0.9669220 2.457137 0.9266777
##                                            log2FC length
## B08_S56_TRINITY_DN676_c0_g1_i1.p1     6.615195    330
## P08_S156_TRINITY_DN127_c0_g1_i1.p1    8.348166    831
## N04_S123_TRINITY_DN50770_c0_g1_i1.p1 4.277625    918
## B04_S26_TRINITY_DN5869_c0_g1_i1.p1    2.570400    798
## B08_S56_TRINITY_DN291_c0_g1_i1.p1     2.356878    354
## N04_S123_TRINITY_DN18231_c0_g1_i4.p1 4.584862   1269
```

```
##downregulated genes in instar 4
mynoiseq.bio.deg_down = degenes(mynoiseq.bio, q = 0.9 , M= "down")
```

```
## [1] "34 differentially expressed features (down in first condition)"
```

```
head(mynoiseq.bio.deg_down)
```

```
##                                       L4_mean   L5_mean     theta      prob
## G12_S92_TRINITY_DN100_c0_g1_i1.p1    0.6639226 4413.5986 -4.845284 1.0000000
## G10_S76_TRINITY_DN20_c0_g1_i3.p1     0.3462802 1910.3745 -4.770119 0.9999009
## G10_S76_TRINITY_DN2_c0_g1_i6.p2      0.2421540  441.3725 -4.241185 0.9996487
## P01_S101_TRINITY_DN6936_c0_g1_i1.p1 10.1303053  473.6449 -3.099987 0.9959304
## M04_S122_TRINITY_DN23_c0_g1_i7.p2    0.6974722 1035.7833 -3.809668 0.9958467
## C05_S33_TRINITY_DN47347_c0_g1_i1.p1  0.3698647  154.9577 -3.285747 0.9957601
##                                         log2FC length
## G12_S92_TRINITY_DN100_c0_g1_i1.p1   -12.698653    876
## G10_S76_TRINITY_DN20_c0_g1_i3.p1    -12.429628    300
## G10_S76_TRINITY_DN2_c0_g1_i6.p2     -10.831856    429
## P01_S101_TRINITY_DN6936_c0_g1_i1.p1  -5.547056    792
## M04_S122_TRINITY_DN23_c0_g1_i7.p2   -10.536299    450
## C05_S33_TRINITY_DN47347_c0_g1_i1.p1  -8.710661    834
```

*Try and do reference genome pipeline on this count data*

```
# Load the libraries we'll need in the following code:
library("DESeq2")
library("apeglm")
```

```
## Warning: package 'apeglm' was built under R version 4.0.3
```

```
library("pheatmap")
library("tidyverse")

# create an object with the directory containing your counts:
    # !!edit this to point to your own count file directory!!
directory <- "/Users/swapnasubramanian/Documents/Hell Part 3/Spring 2021/Practical Genomics/Final Proje

# ensure the count files are where you think they are
list.files(directory)
```

```
##  [1] "B04_S26.counts"
##  [2] "B08_S56.counts"
##  [3] "C05_S33.counts"
##  [4] "Counts"
##  [5] "E03_S21.counts"
##  [6] "F03_S22.counts"
##  [7] "G10_S76.counts"
##  [8] "G12_S92.counts"
##  [9] "I05_S126.counts"
## [10] "K12_S181.counts"
## [11] "L03_S113.counts"
## [12] "L12_S182.counts"
## [13] "length.txt"
## [14] "M04_S122.counts"
## [15] "N04_S123.counts"
## [16] "N06_S139.counts"
```

```
## [17] "N10_S170.counts"
## [18] "N12_S184.counts"
## [19] "O01_S100.counts"
## [20] "O06_S140.counts"
## [21] "O12_S185.counts"
## [22] "P01_S101.counts"
## [23] "P02_S109.counts"
## [24] "P08_S156.counts"
## [25] "SubramanianPastore_3500FinalProject.Rproj"
## [26] "SubramanianPastore_5300FinalProject_files"
## [27] "SubramanianPastore_5300FinalProject.html"
## [28] "SubramanianPastore_5300FinalProject.pdf"
## [29] "SubramanianPastore_5300FinalProject.Rmd"
## [30] "T02_S198.counts"
```

```
# look at the data frame to ensure it is what you expect:
myfactors
```

```
##      Sample Instar BodyRegion
## 1  P02_S109     L4        Abd
## 2  P01_S101     L4        Abd
## 3   B08_S56     L4        Abd
## 4   F03_S22     L4         W3
## 5  N10_S170     L4         W3
## 6  O01_S100     L4         W3
## 7  K12_S181     L4         W2
## 8  N04_S123     L4         W2
## 9  N12_S184     L4         W2
## 10  E03_S21     L4        Pro
## 11 N06_S139     L4        Pro
## 12 P08_S156     L4        Pro
## 13  B04_S26     L5        Abd
## 14 L03_S113     L5        Abd
## 15 L12_S182     L5        Abd
## 16 I05_S126     L5         W3
## 17 M04_S122     L5         W3
## 18 O12_S185     L5         W3
## 19  C05_S33     L5         W2
## 20 T02_S198     L5         W2
## 21  G10_S76     L5        Pro
## 22  G12_S92     L5        Pro
## 23 O06_S140     L5        Pro
```

```
##converting all kallisto counts to integers so that DESeq can read it
class(m$P02_S109)
```

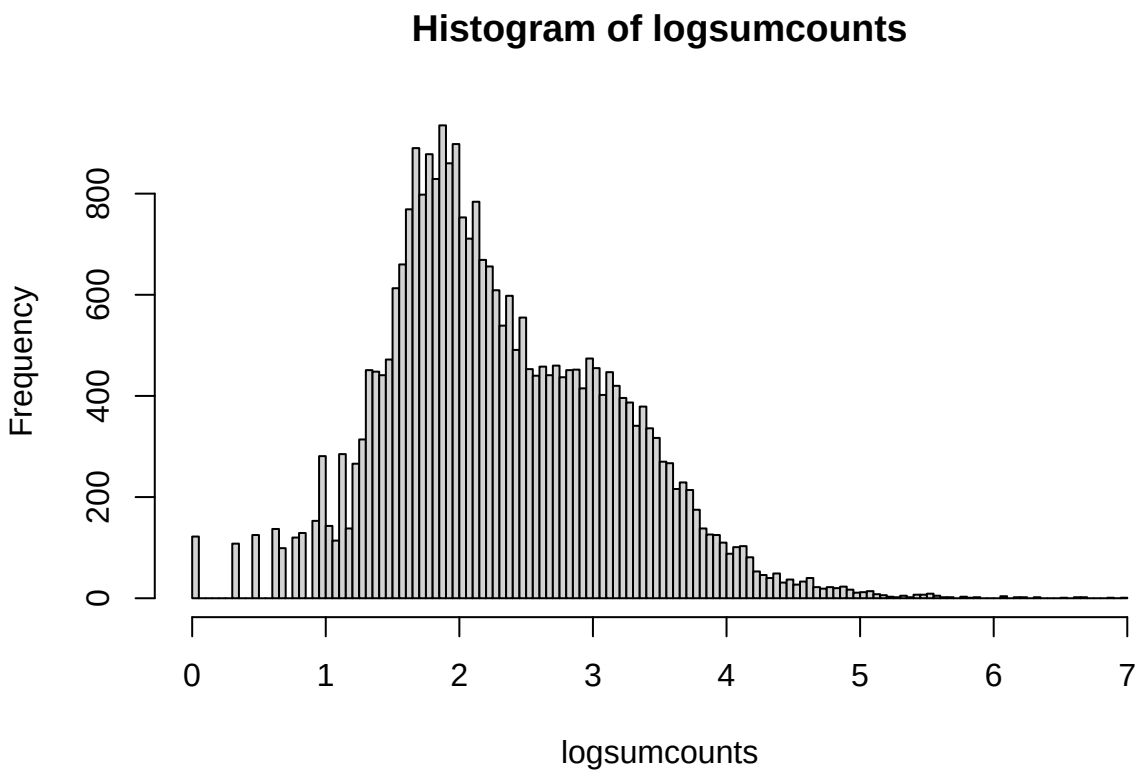
```
## [1] "numeric"
```

```
m$P02_S109 <- as.integer(m$P02_S109)
m$P01_S101 <- as.integer(m$P01_S101)
m$B08_S56 <-as.integer(m$B08_S56)
m$F03_S22 <- as.integer(m$F03_S22)
```

```
m$N10_S170 <- as.integer(m$N10_S170)
m$O01_S100 <- as.integer(m$O01_S100)
m$K12_S181 <- as.integer(m$K12_S181)
m$N04_S123 <- as.integer(m$N04_S123)
m$N12_S184 <- as.integer(m$N12_S184)
m$E03_S21 <- as.integer(m$E03_S21)
m$N06_S139 <- as.integer(m$N06_S139)
m$P08_S156 <- as.integer(m$P08_S156)
m$B04_S26 <- as.integer(m$B04_S26)
m$L03_S113 <- as.integer(m$L03_S113)
m$L12_S182 <- as.integer(m$L12_S182)
m$I05_S126 <- as.integer(m$I05_S126)
m$M04_S122 <- as.integer(m$M04_S122)
m$O12_S185 <- as.integer(m$O12_S185)
m$C05_S33 <- as.integer(m$C05_S33)
m$T02_S198 <- as.integer(m$T02_S198)
m$G10_S76 <- as.integer(m$G10_S76)
m$G12_S92 <- as.integer(m$G12_S92)
m$O06_S140 <- as.integer(m$O06_S140)


# create the DESeq data object from Matrix (I kept the name from the example to make it easier to go th
ddsHTSeq <- DESeqDataSetFromMatrix(
  countData = m,
  colData = myfactors,
        design = ~ Instar + BodyRegion + Instar:BodyRegion)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
# what does expression look like across genes?

# sum counts for each gene across samples
sumcounts <- rowSums(counts(ddsHTSeq))
# take the log
logsumcounts <- log(sumcounts,base=10)
# plot a histogram of the log scaled counts
hist(logsumcounts,breaks=100)
```




```
# you can see the typically high dynamic range of RNA-Seq, with a mode in the distribution around 1000 j

# get genes with summed counts greater than 20
keep <- sumcounts > 20

# keep only the genes for which the vector "keep" is TRUE
ddsHTSeq <- ddsHTSeq[keep,]

dds <- DESeq(ddsHTSeq)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
# get results table
res <- results(dds)

# get a quick summary of the table
summary(res)
```

```
##
## out of 27367 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 3, 0.011%
## LFC < 0 (down)     : 1, 0.0037%
## outliers [1]       : 124, 0.45%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
# check out the first few lines
head(res)
```

```
## log2 fold change (MLE): InstarL5.BodyRegionW3
## Wald test p-value: InstarL5.BodyRegionW3
## DataFrame with 6 rows and 6 columns
##                                      baseMean log2FoldChange     lfcSE
##                                     <numeric>      <numeric> <numeric>
## B04_S26_TRINITY_DN0_c0_g1_i1.p1     15915.23775       0.588496  1.567529
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1   100.25356       0.848187  0.636605
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1    15.62415      -5.881861  2.757454
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1     5.62891      -7.334393  5.491672
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1    28.11418       0.706124  0.874452
## B04_S26_TRINITY_DN1007_c0_g1_i1.p1     47.00228       0.113754  0.952849
##                                          stat    pvalue      padj
##                                     <numeric> <numeric> <numeric>
## B04_S26_TRINITY_DN0_c0_g1_i1.p1      0.375429 0.7073415  0.999979
## B04_S26_TRINITY_DN10003_c0_g1_i1.p1  1.332359 0.1827421  0.999979
## B04_S26_TRINITY_DN10035_c0_g2_i1.p1 -2.133077 0.0329184  0.999979
## B04_S26_TRINITY_DN10035_c0_g2_i5.p1 -1.335548 0.1816970  0.999979
## B04_S26_TRINITY_DN10037_c0_g4_i1.p1  0.807505 0.4193758  0.999979
## B04_S26_TRINITY_DN1007_c0_g1_i1.p1   0.119383 0.9049718  0.999979
```

```
# get shrunken log fold changes
#get the options for coef
resultsNames(dds)
```

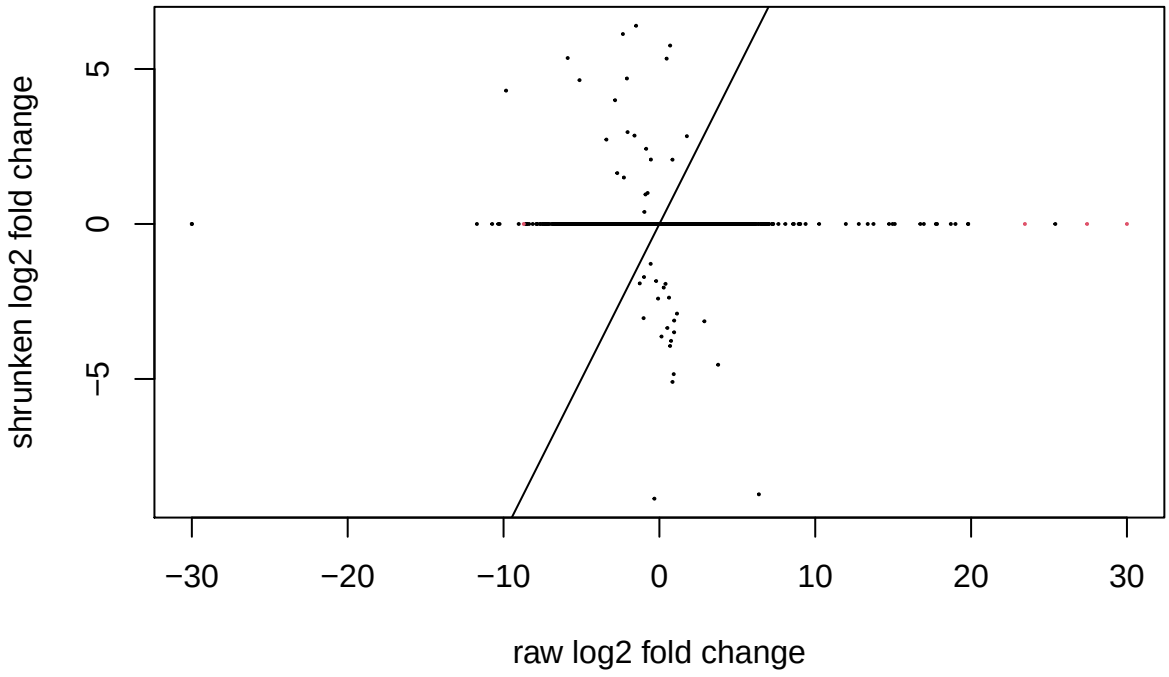
```
## [1] "Intercept"              "Instar_L5_vs_L4"        "BodyRegion_Pro_vs_Abd"
## [4] "BodyRegion_W2_vs_Abd"   "BodyRegion_W3_vs_Abd"   "InstarL5.BodyRegionPro"
## [7] "InstarL5.BodyRegionW2"  "InstarL5.BodyRegionW3"
```

```
res_shrink <- lfcShrink(dds,coef="Instar_L5_vs_L4")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

```
help("lfcShrink")
```

```r
# plot the shrunken log2 fold changes against the raw changes:
plot(
    x=res$log2FoldChange,
    y=res_shrink$log2FoldChange,pch=20,
    cex=.2,
    col=1+(res$padj < 0.05),
    xlab="raw log2 fold change",
    ylab="shrunken log2 fold change"
    )
abline(0,1)
```

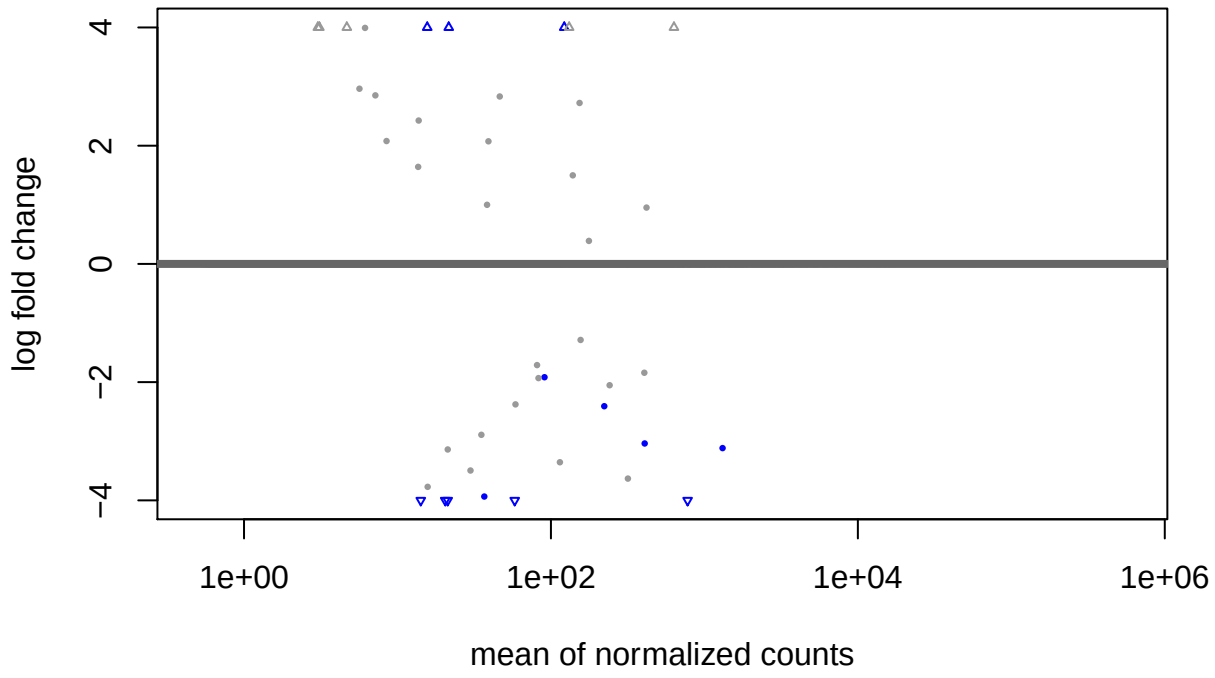



```r
# get the top 20 genes by shrunken log2 fold change
top20 <- order(-abs(res_shrink$log2FoldChange))[1:20]
res_shrink[top20,]
```

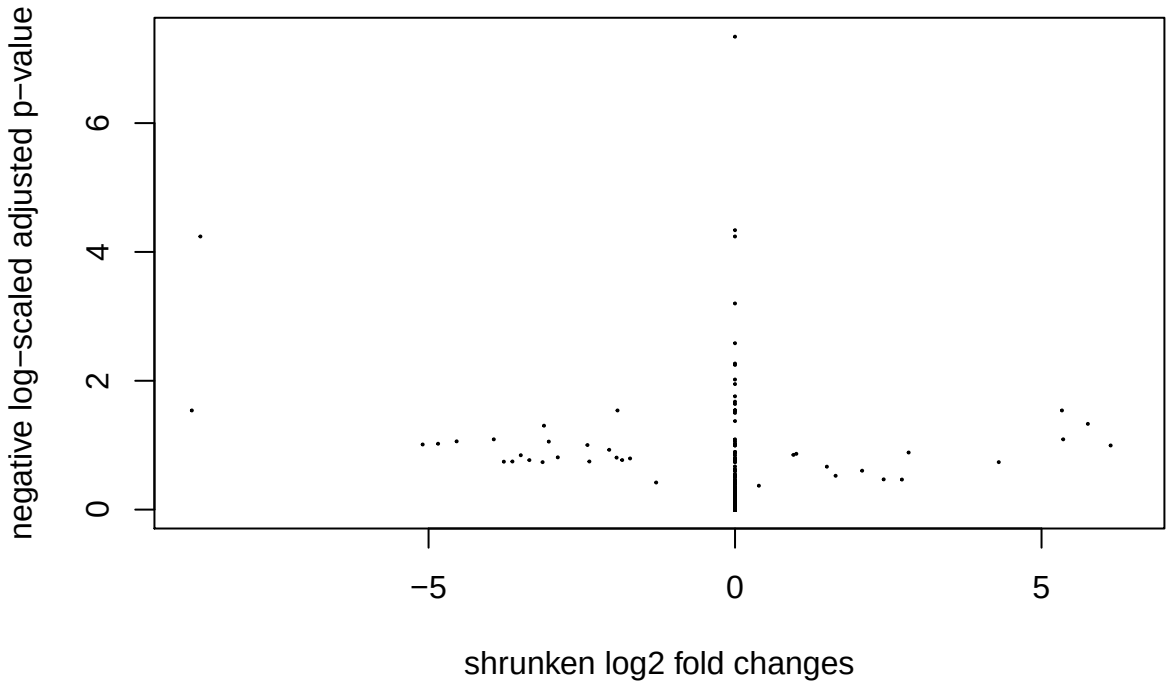
```
## log2 fold change (MAP): Instar L5 vs L4
## Wald test p-value: Instar L5 vs L4
## DataFrame with 20 rows and 5 columns
##                                    baseMean log2FoldChange     lfcSE
##                                   <numeric>      <numeric> <numeric>
## N12_S184_TRINITY_DN29_c0_g1_i13.p1   58.07330       -8.86192   3.50601
## F03_S22_TRINITY_DN244_c1_g1_i1.p1   777.31734       -8.72232   1.76322
## B04_S26_TRINITY_DN1699_c0_g1_i1.p3    4.67673        6.39465   3.17723
## E03_S21_TRINITY_DN9_c0_g2_i4.p1     131.38663        6.12830   3.82964
## B04_S26_TRINITY_DN827_c0_g1_i1.p1   122.07829        5.75641   2.02699
## ...                                       ...            ...       ...
## O01_S100_TRINITY_DN177_c0_g1_i10.p1   15.7276       -3.77137   1.89331
## K12_S181_TRINITY_DN1_c0_g2_i1.p1     317.3700       -3.63168   2.10159
## P02_S109_TRINITY_DN1397_c0_g1_i3.p1   29.9010       -3.49498   1.60719
## P02_S109_TRINITY_DN327_c0_g1_i12.p1  114.3901       -3.35504   1.75125
## B08_S56_TRINITY_DN1271_c0_g2_i1.p1    21.2555       -3.13927   1.81232
```

```
##                                        pvalue        padj
##                                     <numeric>   <numeric>
## N12_S184_TRINITY_DN29_c0_g1_i13.p1  3.95340e-05 2.88974e-02
## F03_S22_TRINITY_DN244_c1_g1_i1.p1   2.02654e-08 5.75487e-05
## B04_S26_TRINITY_DN1699_c0_g1_i1.p3  3.53723e-04          NA
## E03_S21_TRINITY_DN9_c0_g2_i4.p1     3.63312e-04 1.01050e-01
## B04_S26_TRINITY_DN827_c0_g1_i1.p1   9.86206e-05 4.66763e-02
## ...                                        ...         ...
## O01_S100_TRINITY_DN177_c0_g1_i10.p1 0.001031681    0.180290
## K12_S181_TRINITY_DN1_c0_g2_i1.p1    0.001009844    0.179232
## P02_S109_TRINITY_DN1397_c0_g1_i3.p1 0.000642714    0.143149
## P02_S109_TRINITY_DN327_c0_g1_i12.p1 0.000915623    0.170501
## B08_S56_TRINITY_DN1271_c0_g2_i1.p1  0.001102487    0.183381
```

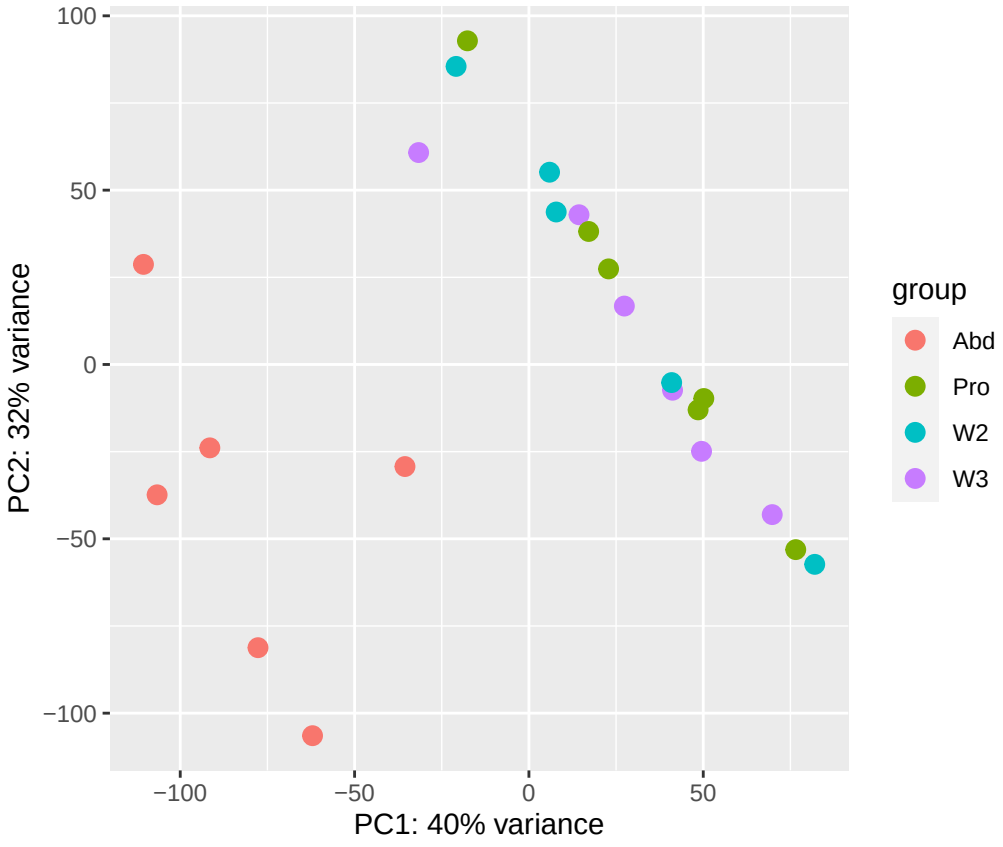
```r
plotMA(res_shrink, ylim=c(-4,4))
```



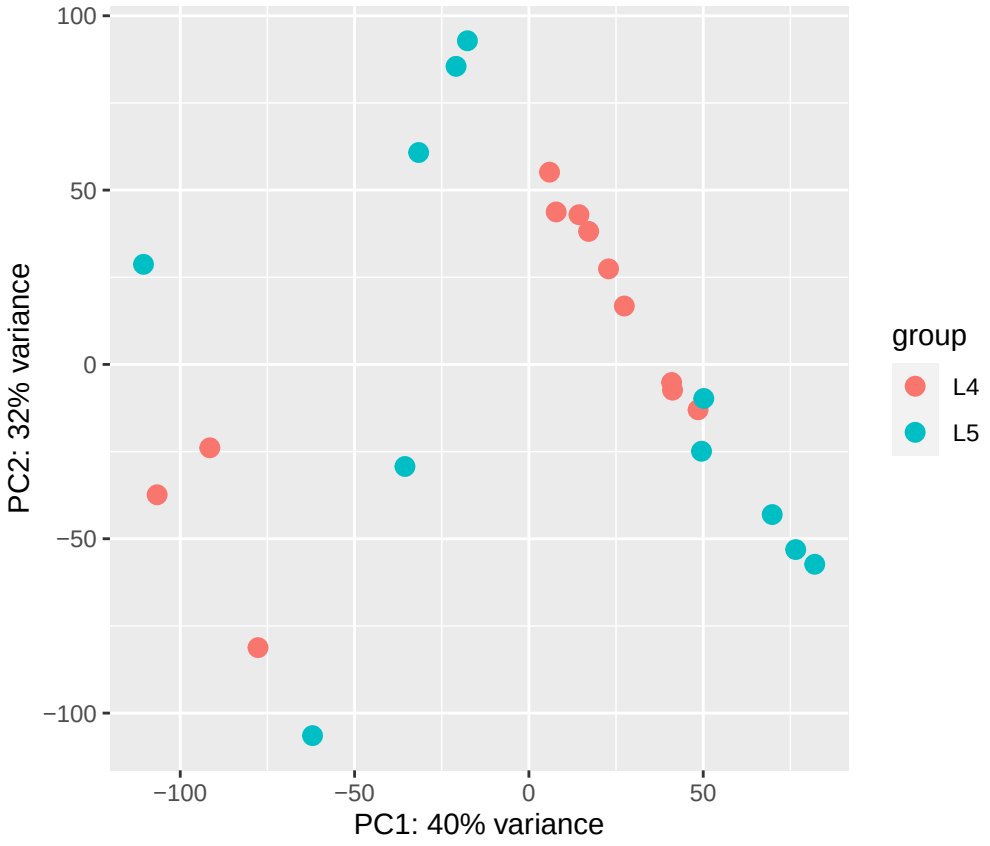

```r
# negative log-scaled adjusted p-values
log_padj <- -log(res_shrink$padj,10)
log_padj[log_padj > 100] <- 100

# plot
plot(x=res_shrink$log2FoldChange,
     y=log_padj,
     pch=20,
     cex=.2,
     col=(log_padj > 10)+1, # color padj < 0.1 red
     ylab="negative log-scaled adjusted p-value",
     xlab="shrunken log2 fold changes")
```



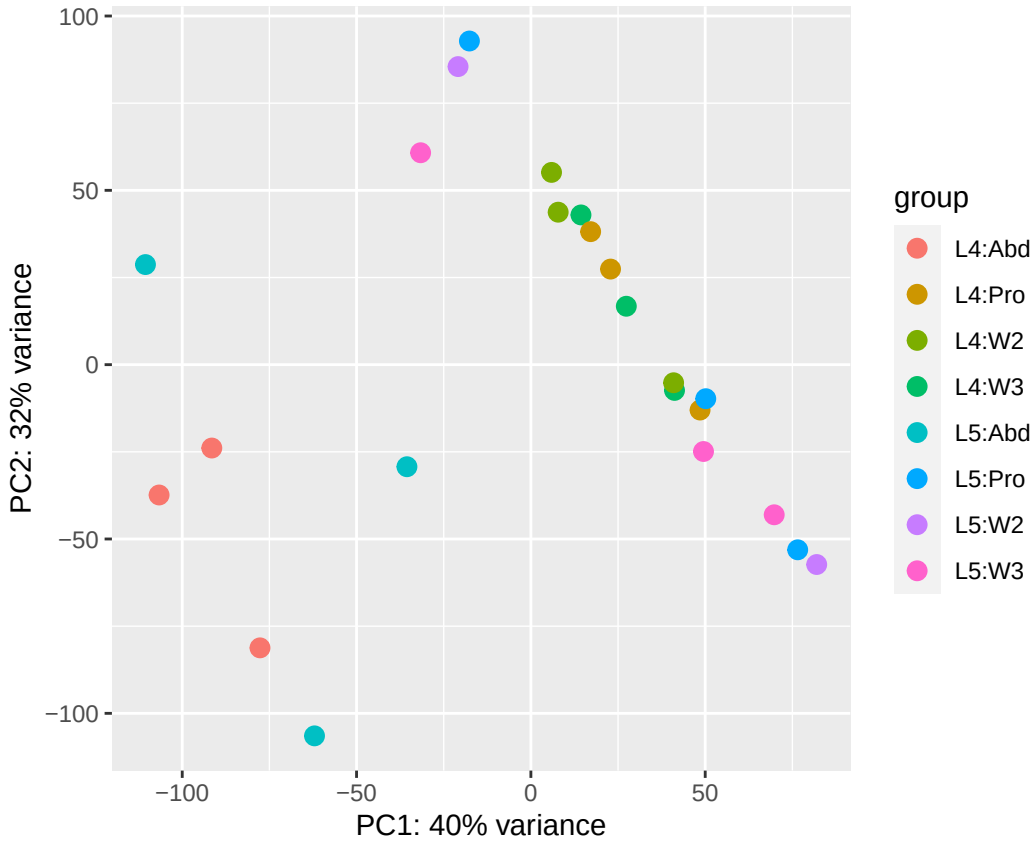

```
# normalized, variance-stabilized transformed counts for visualization
vsd <- vst(dds, blind=FALSE)

plotPCA(vsd, intgroup="BodyRegion")
```



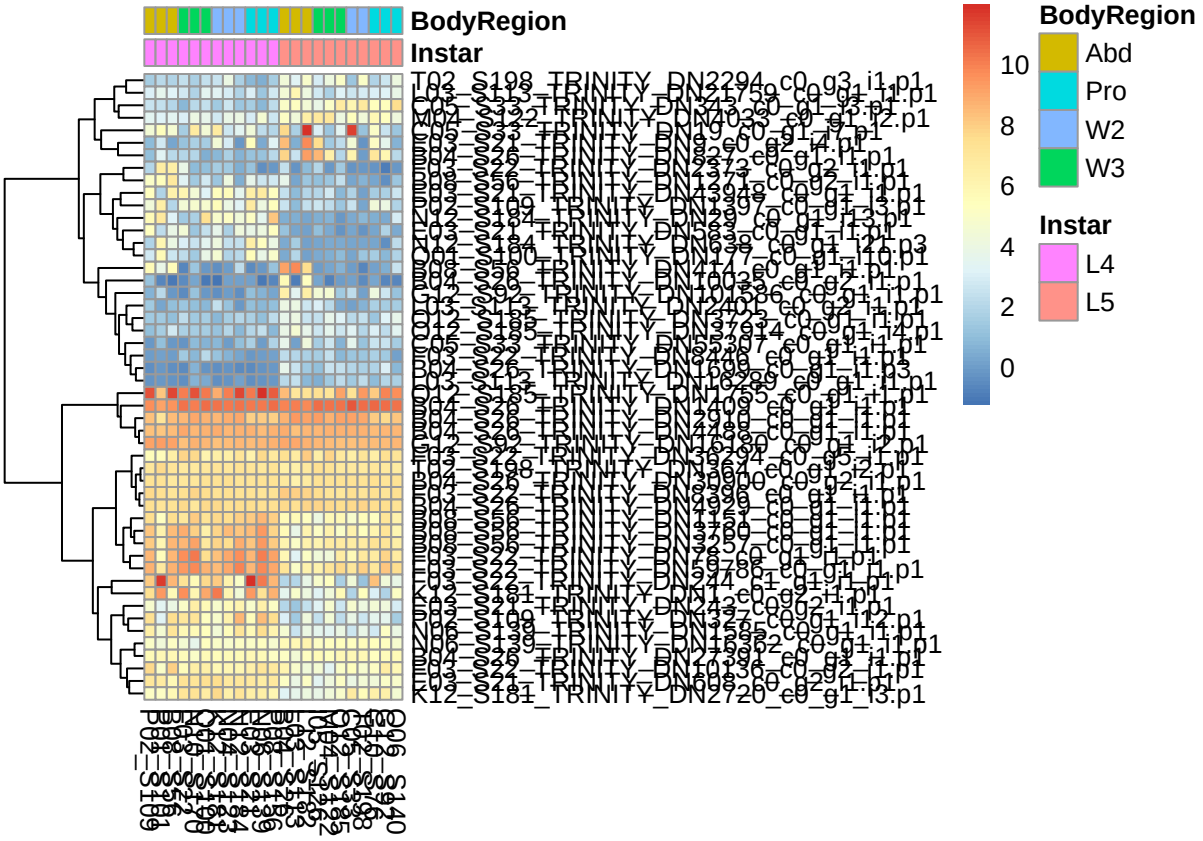
```
plotPCA(vsd, intgroup="Instar")
```




```
#colored by bodyregion and instar
##Abdomen clearly grouped together, L4 seems slightly grouped together but wing genes are grouped with
plotPCA(vsd, intgroup=c("Instar", "BodyRegion"))
```




```
# regularized log transformation of counts
rld <- rlog(dds, blind=FALSE)

# get top 50 log fold change genes of instar vs body region
top50 <- order(-abs(res_shrink$log2FoldChange))[1:50]
df <- data.frame(colData(dds)[,c("Instar", "BodyRegion")])
    rownames(df) <- colnames(dds)
    colnames(df) <- c("Instar", "BodyRegion")
pheatmap(
    assay(rld)[top50,],
    cluster_rows=TRUE,
    show_rownames=TRUE,
    cluster_cols=FALSE,
    annotation_col=df
    )
```




```
l2fc_ord <- order(-abs(res_shrink$log2FoldChange))
plotCounts(dds, gene=l2fc_ord[1], intgroup= c("Instar", "BodyRegion"))
```