Lab: Differential Expression via RNA-Seq Analysis

Genomic Technologies Workshop 2024 (PLPTH885)

Sanzhen Liu

6/5/2024

Outline

- Differential expression test using DESeq2
- Result visualization

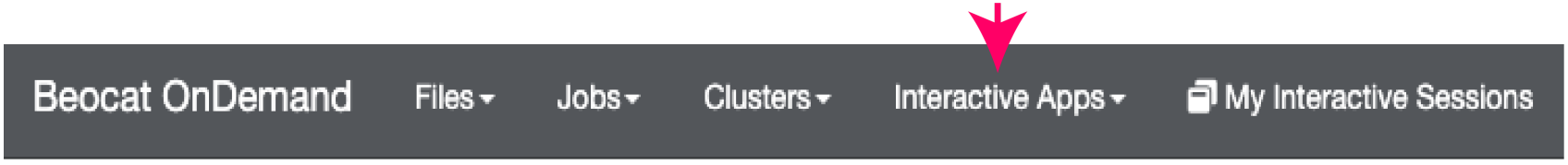
Course webpage

RNA-seq DE analysis

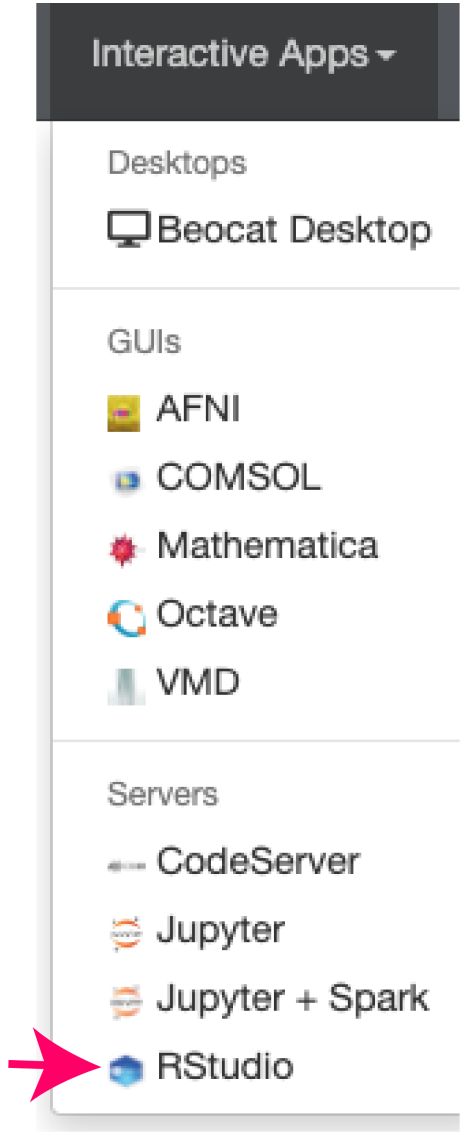
OnDemand at Beocat

ondemand

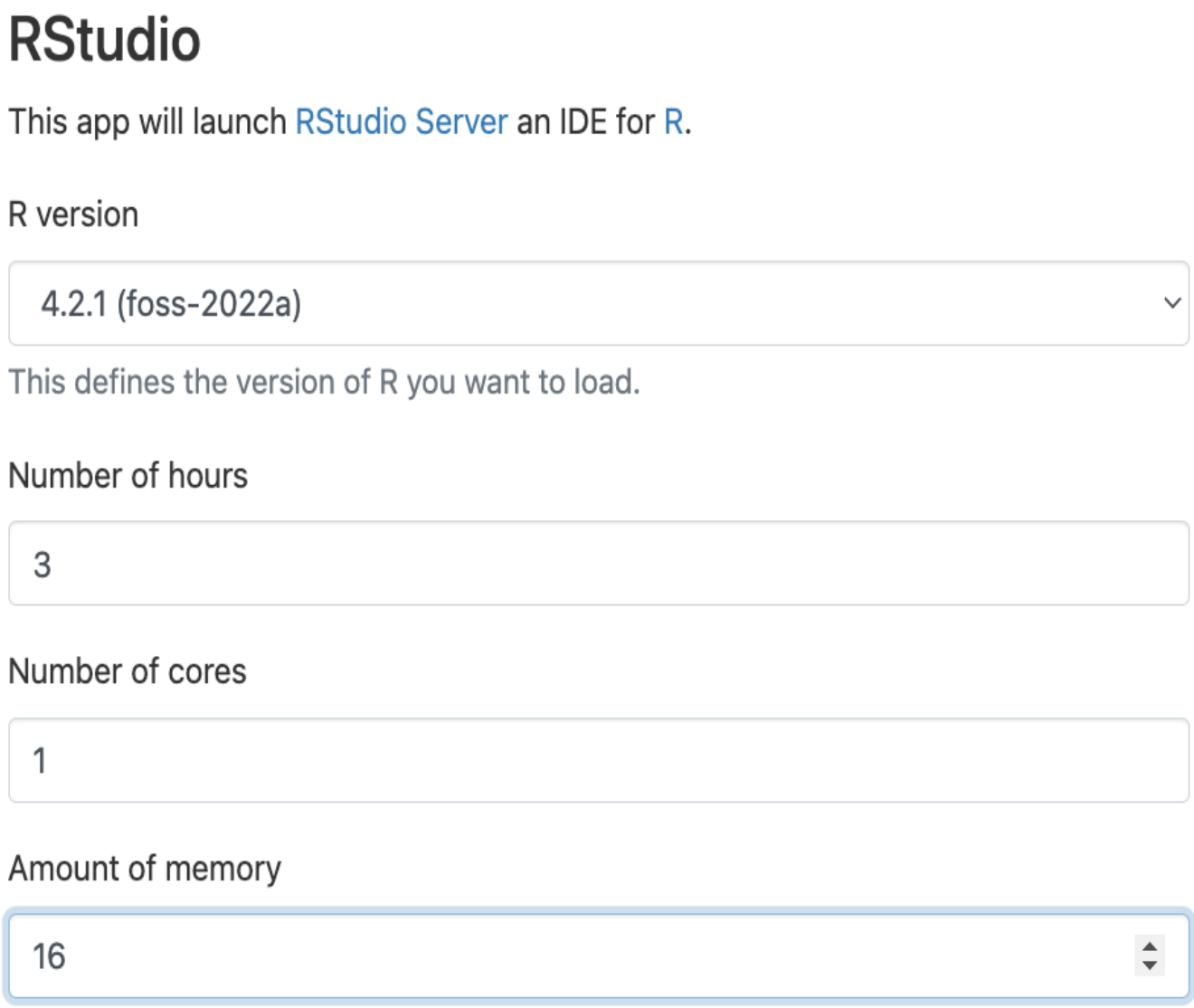
login with eID



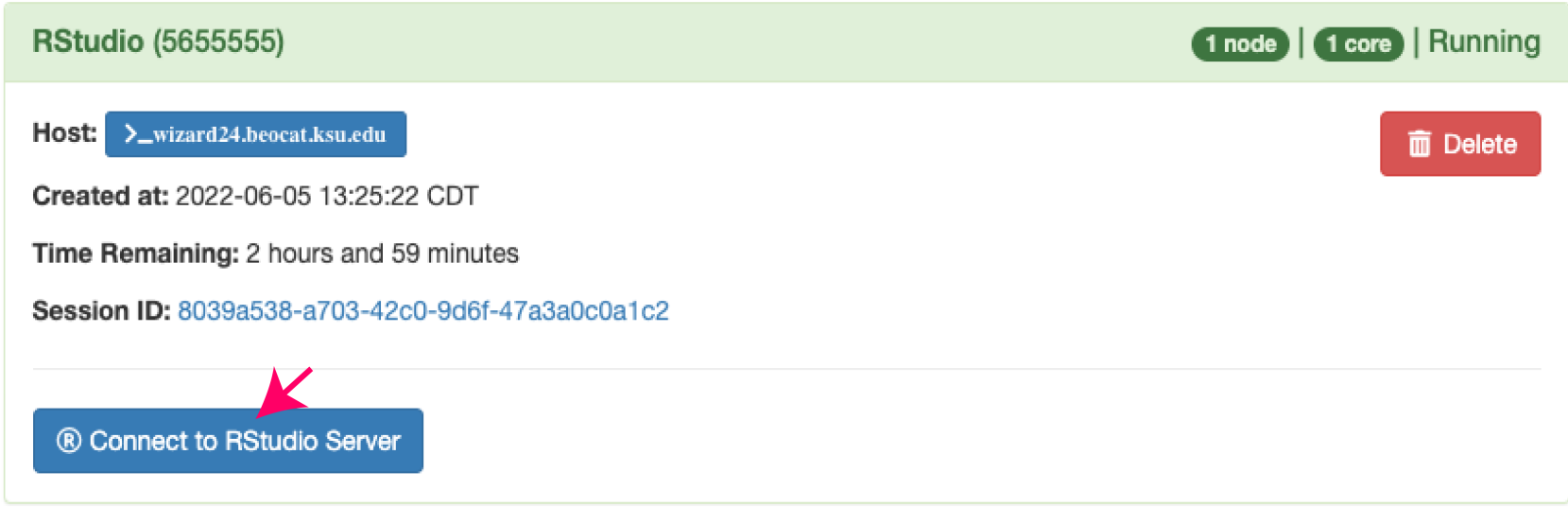
Select RStudio



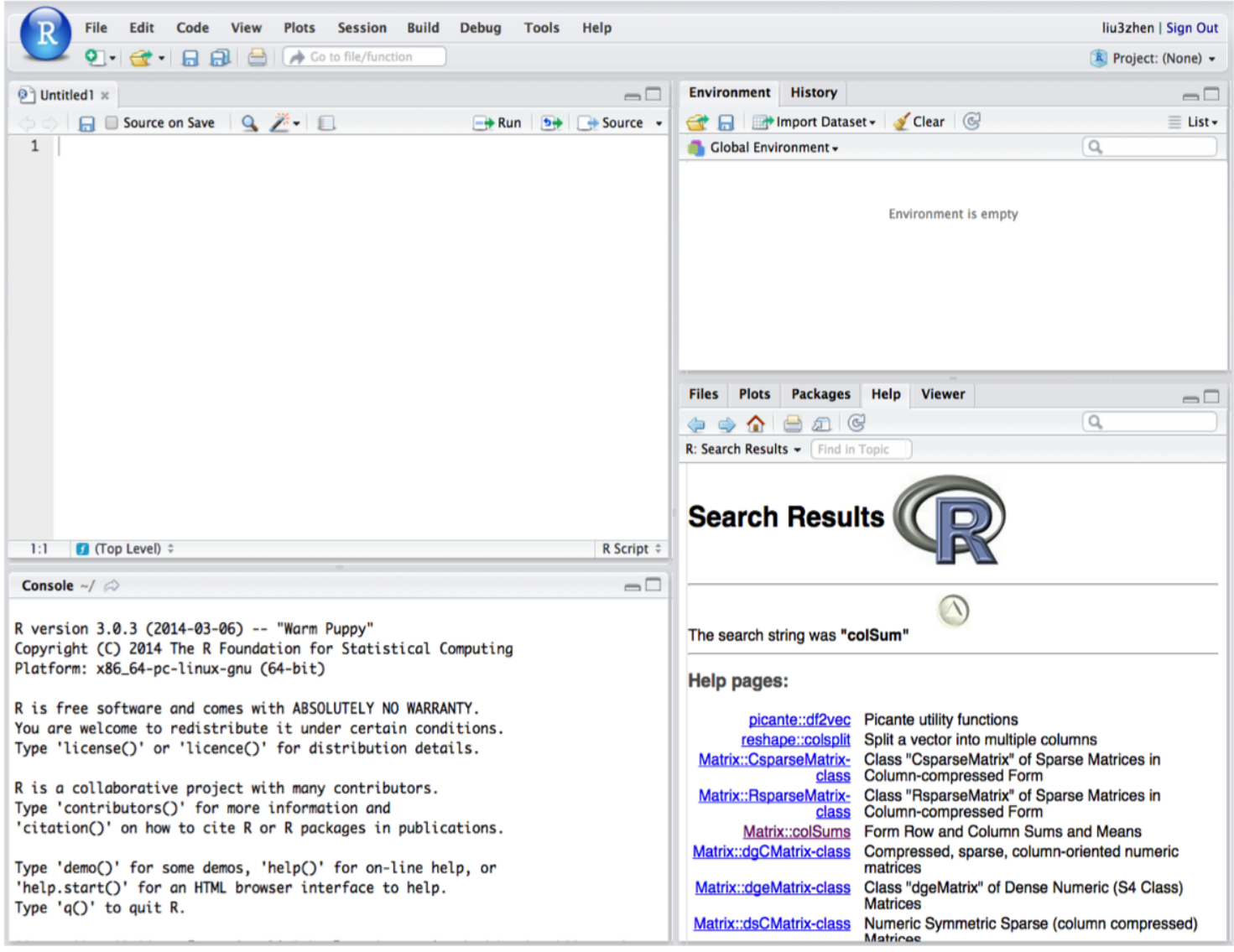
Request resources



Connect to RStudio



Rstudio interface



Package installation

```
if (!require("BiocManager", quietly=T))
  install.packages("BiocManager")

# to solve a version issue, install matrixStats to update the versio
# suggested by Adam Tygart from Beocat

install.packages("matrixStats", repos="http://cran.us.r-project.org")

if (!require("DESeq2", quietly=T))
  BiocManager::install("DESeq2") # DESeq2
```

preload modules

```
data_url <- "https://raw.githubusercontent.com/liu3zhenlab/teaching/r
pls <- paste0(data_url, "/utils/load.R")
source(pls)
```

- panel.cor2
- rnaseq.pca
- normalization

codes

Read expression data (Read counts per gene)

```
rc_file <- paste0(data_url, "/data/rc.txt")
grc <- read.delim(rc_file)
nrow(grc)  # the number of rows/lines
```

```
[1] 22697
```

first entry:

Gene	ExonSize	ck_rep1	ck_rep2	ck_rep3	trt_rep1	trt_rep2	trt_rep
AC147602.5_FG004	483	5480	6075	5934	3370	5784	643

last entry:

	Gene	ExonSize	ck_rep1	ck_rep2	ck_rep3	trt_rep1	trt_rep2	t
22697	GRMZM5G899985	615	267	327	348	83	342	

RPKM normalization

```
datacols <- c("ck_rep1", "ck_rep2", "ck_rep3",
              "trt_rep1", "trt_rep2", "trt_rep3")
lib.sizes <- colSums(grc[, datacols]) # library sizes
exon.sizes <- grc$ExonSize
grcn <- normalization(counts = grc,
                      normcolname = datacols,
                      libsize = lib.sizes,
                      exonsize = exon.sizes,
                      methods = "RPKM")
```

Gene	ExonSize	ck_rep1	ck_rep2	ck_rep3	trt_rep1	trt_rep2	trt_rep
AC147602.5_FG004	483	5480	6075	5934	3370	5784	643
AC148152.3_FG005	1422	187	295	377	169	158	56

data organization for DESeq2

- count information

```
geneid <- grc$Gene
in.data <- as.matrix(grc[, 3:8])
```

ck_rep1	ck_rep2	ck_rep3	trt_rep1	trt_rep2	trt_rep3
5480	6075	5934	3370	5784	6432

sample names and grouping information (treatment)

```
sample.ids <- colnames(in.data)
treatment <- c("ck", "ck", "ck", "trt", "trt", "trt")
sample.info <- data.frame(row.names=sample.ids, trt=treatment)
```

	trt
ck_rep1	ck
ck_rep2	ck
ck_rep3	ck
trt_rep1	trt
trt_rep2	trt
trt_rep3	trt

Differential expression test

```
dds <- DESeqDataSetFromMatrix(countData=in.data,
                              colData=sample.info,
                              formula(~trt))
dds <- DESeq(dds)
```

DE output

```
res <- results(object = dds)
res <- data.frame(res)
res$Gene <- geneid
res <- res[,c("Gene","baseMean","log2FoldChange","pvalue","padj")]
nrow(res)
```

```
[1] 22697
```

DE + normalized data

```
### Merge the normalized result with the DE result
out <- merge(grcn, res, by = "Gene")
out <- data.frame(out)
```

Gene	ExonSize	ck_rep1	ck_rep2	ck_rep3	trt_rep1	trt_rep2
AC147602.5_FG004	483	5480	6075	5934	3370	5784
AC148152.3_FG005	1422	187	295	377	169	158

trt_rep3	ck_rep1.RPKM	ck_rep2.RPKM	ck_rep3.RPKM	trt_rep1.RPKM
6432	854.123	895.760	904.373	567.493
563	9.900	14.775	19.516	9.666

trt_rep2.RPKM	trt_rep3.RPKM	baseMean	log2FoldChange	pvalue	padj
915.326	916.971	5441.6579	-0.1490702	0.4642180	0.8049379
8.493	27.262	285.5493	0.0431574	0.9258171	0.9845702

significant gene sets at different FDRs

```
sum(!is.na(out$padj) & out$padj < 0.05)
```

```
[1] 1261
```

problem

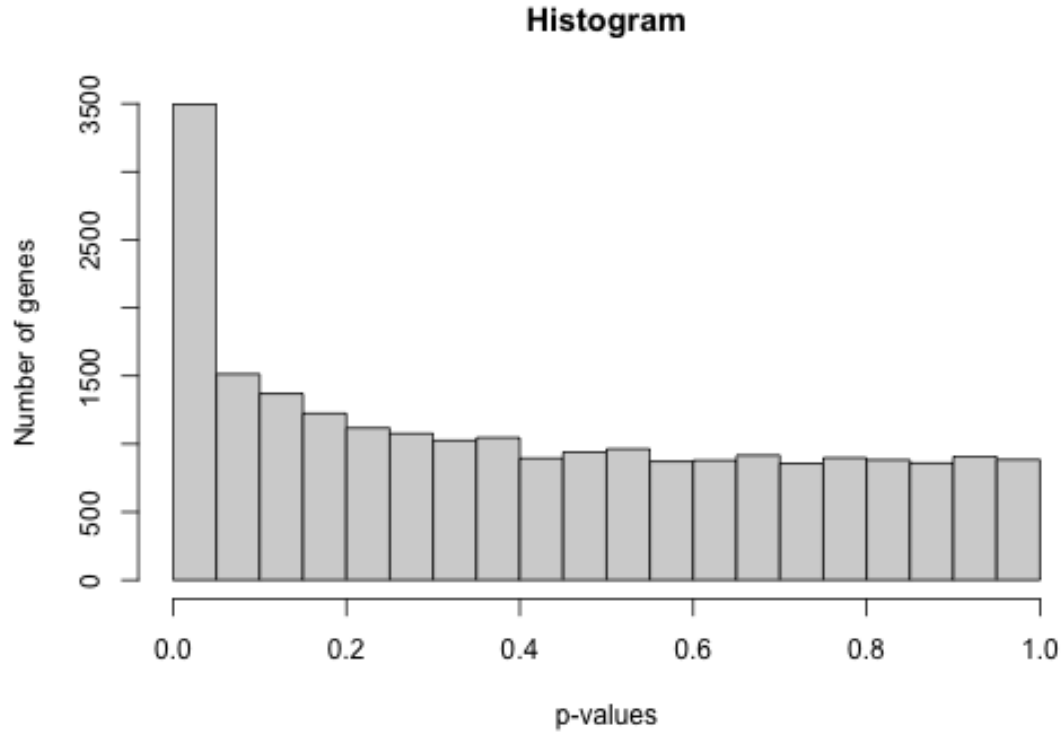
Please revise the code to calculate the number of significant genes with the FDR smaller than 10% and 15%?

significantly DEG

```
sig <- out[!is.na(out$padj) & out$padj < 0.05, ]
```

p-value histogram

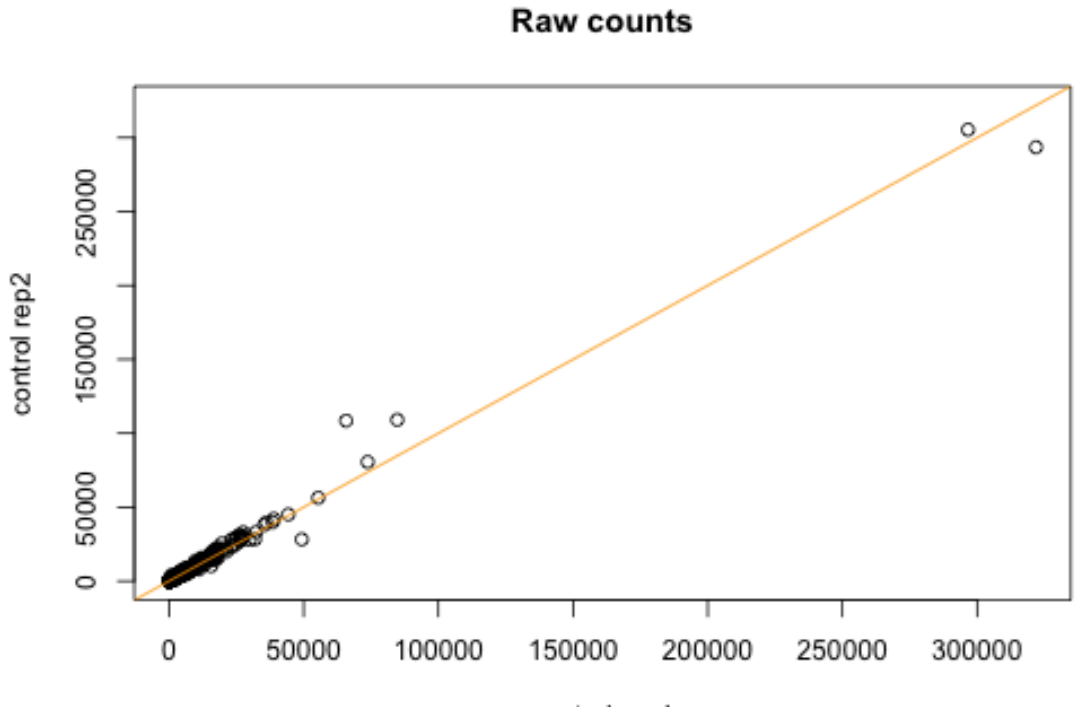
```
pvals <- out$pvalue
hist(pvals, main="Histogram",xlab="p-values",ylab="Number of genes")
```



problem
Please modify the plot code to change the figure title to "DE"

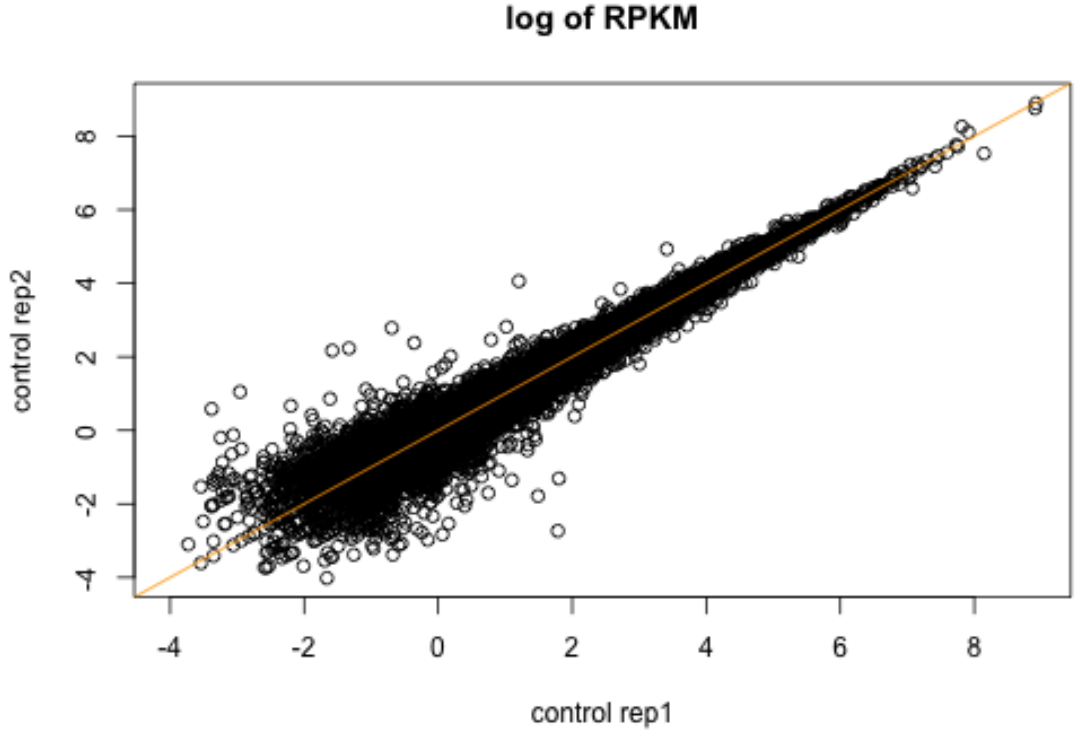
scatter plot - raw counts

```
x.raw <- out$ck_rep1
y.raw <- out$ck_rep2
drange <- range(x.raw, y.raw, finite=T)
# plot a scatter plot between two samples
plot(x.raw, y.raw, main = "Raw counts",
     xlab="control rep1", ylab="control rep2",
     xlim = drange, ylim = drange)
abline(a = 0, b = 1, col = "orange")
```



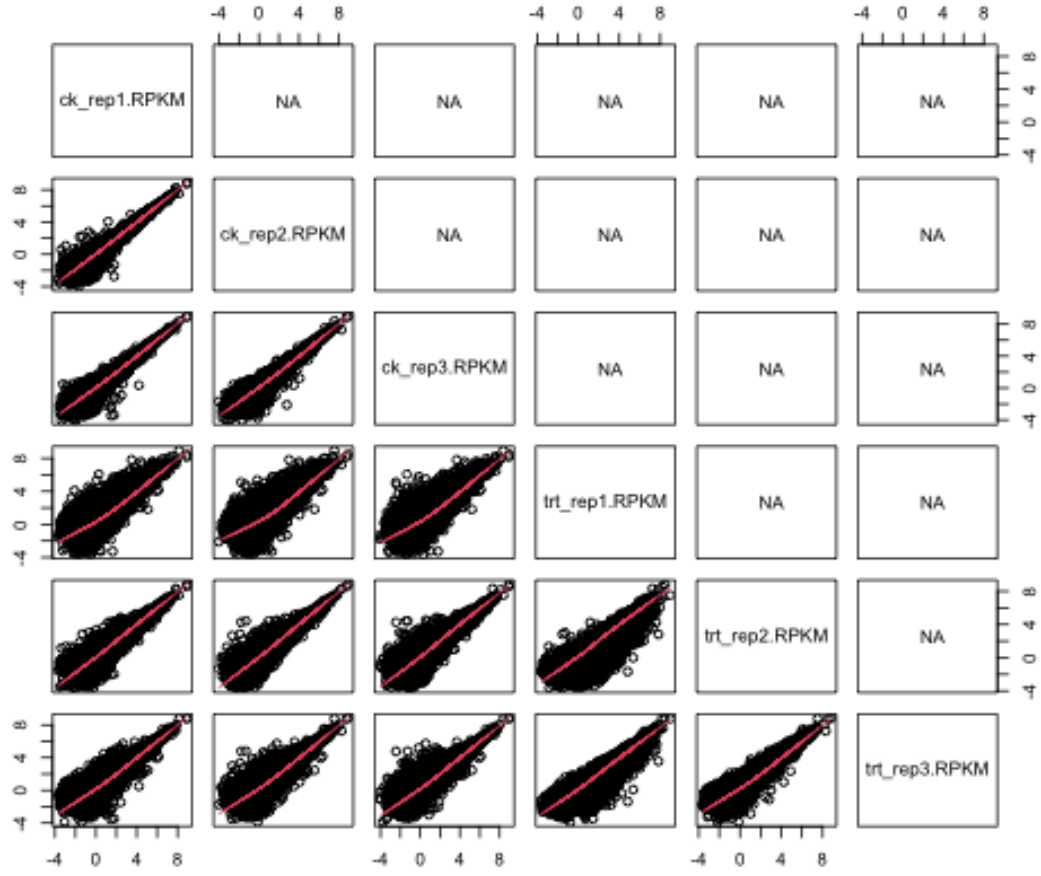
scatter plot - RPKM

```
x <- log(out$ck_rep1.RPKM)
y <- log(out$ck_rep2.RPKM)
drange <- range(x, y, finite=T)
plot(x, y, main = "log of RPKM",
     xlab = "control rep1", ylab = "control rep2",
     xlim = drange, ylim = drange)
abline(a = 0, b = 1, col = "orange")
```



pair-wise scatter plots

```
logrpkm <- log(out[, 9:14])
pairs(logrpkm, lower.panel=panel.smooth, upper.panel=panel.cor2)
```

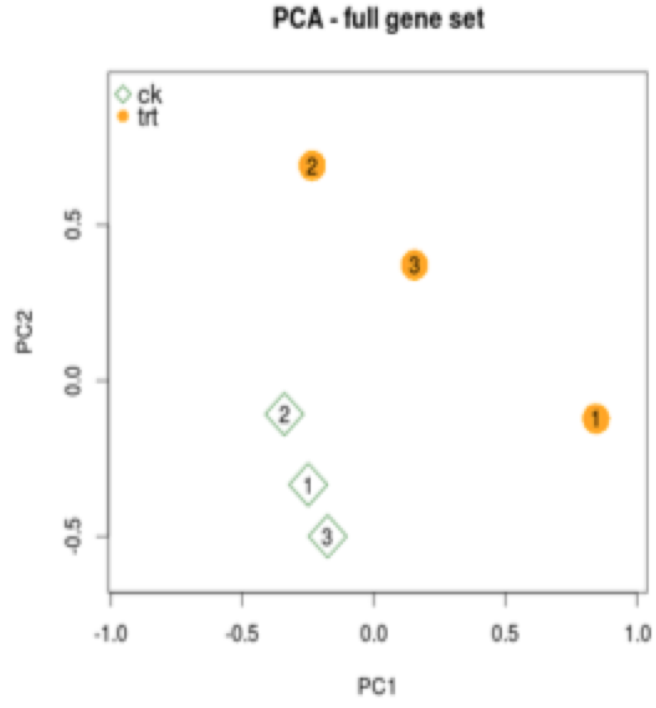


Principal Component Analysis (PCA)

PCA is a mathematical algorithm that reduces the dimensionality of the data while retaining most of the variation in the data set.

	Control			Treatment		
Gene	Rep1	Rep2	Rep3	Rep1	Rep2	Rep3
1	2679	2360	2573	2563	3398	3012
2	177	161	171	154	137	152
3	381	371	397	541	723	635
...						
20000	990	1073	1236	850	672	859

Normalized and standardized data



function / module

You can write your own function:
fun_name <- function (...) { ... }

```
gpa_improve <- function(gpa, rate) {
### gpa: a numeric vector for GPAs
### rate: percentage for the improvement
    new.gpa <- gpa * (1 + rate)
    new.gpa[new.gpa > 4] <- 4
    return(new.gpa)
}

### running the function
our.gpa <- c(3.8, 3.3, 2.8, 3.1)
gpa_improve(our.gpa, 0.1)
```

```
[1] 4.00 3.63 3.08 3.41
```

```
gpa_improve(our.gpa, 0.2)
```

```
[1] 4.00 3.96 3.36 3.72
```

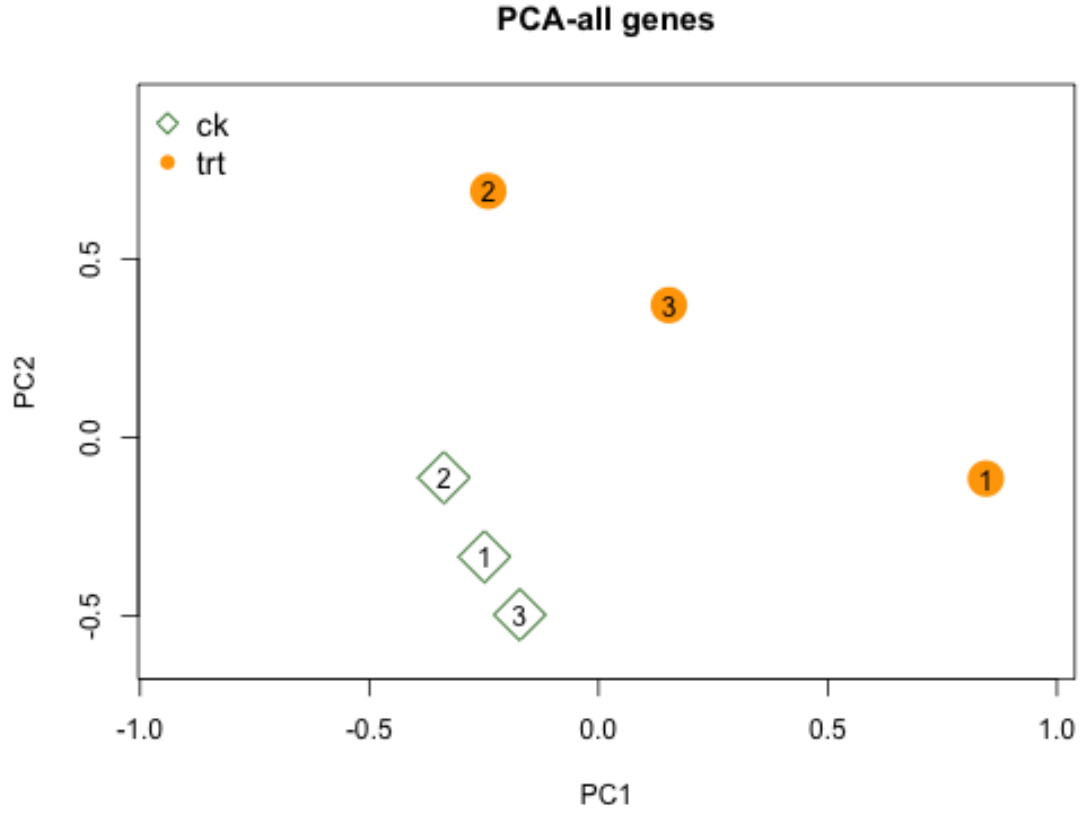
PCA function

principal component analysis and ploting

```
rnaseq.pca <- function(norm.data,
  norm.feature="RPKM",
    group.feature,
    title="",
  shape.code=NULL,
    mean.cutoff=0.1,
    colors=NULL,
    scaling=T, ...) {
    …
}
```

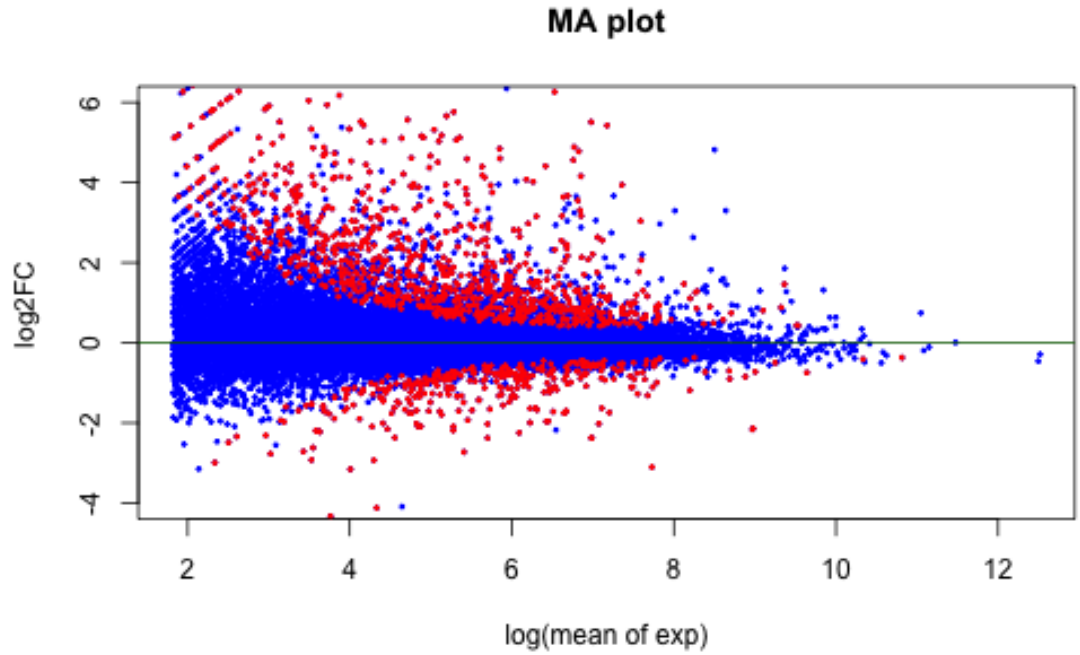
PCA plotting

```
rnaseq.pca(out, norm.feature="RPKM", group.feature=c("ck", "trt"),
           cex=3, shape.code=c(5, 16), mean.cutoff=0.1, scaling = T,
           title="PCA-all genes", colors=c("dark green","orange"))
```



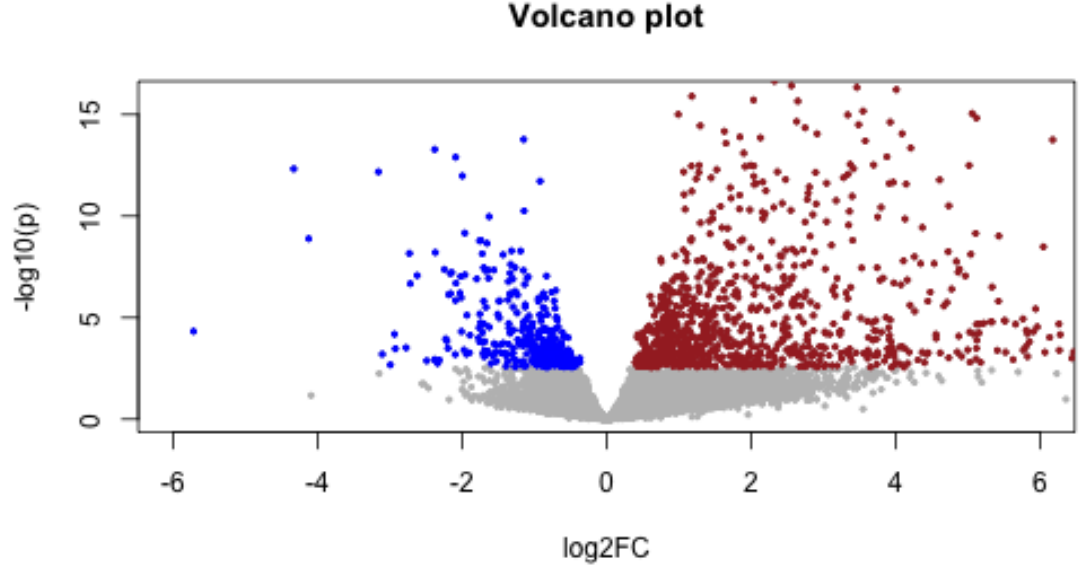
MA plot

```
aval <- log(out$baseMean)
mval <- out$log2FoldChange
sig.set <- which(out$padj < 0.05)
plot(aval, mval, main="MA plot",
     xlab="log(mean of exp)", ylab="log2FC",
     ylim=c(-4, 6), pch=19, cex=0.3, col="blue")
# add points and a horizontal line
points(aval[sig.set], mval[sig.set], pch=19,cex=0.3,col="red")
abline(h = 0, cex = 1, col = "dark green")
```



Volcano plot

```
log2val <- out$log2FoldChange  # log2 fold change
mlogP <- -log10(out$pvalue) # transformed p-values
### plot
plot(log2val, mlogP, main="Volcano plot", xlab="log2FC", ylab="-log10
     pch=19, cex=0.4, col="grey", xlim=c(-6,6), ylim=c(0,16))
### highlight
up <- which(out$padj<0.05 & log2val > 0) # up
down <- which(out$padj<0.05 & log2val < 0)  # down
points(log2val[up], mlogP[up], pch=19,cex=0.4,col="brown")
points(log2val[down], mlogP[down],pch=19,cex=0.4,col="blue")
```



Summary of the analyzing procedure

1. Read counts per gene
2. DE analysis based on the experimental design
3. Examine results (p-value distribution, number of significant genes)
4. Gene Ontology enrichment test

DESeq2 tutorial

Contact information

Sanzhen Liu
Plant Pathology
4729 Throckmorton Plant Sciences Center
Manhattan, KS 66506-5502
phone: 785-532-1379

liu3zhen@ksu.edu
twitter (X): liu3zhen

Bioinformatics Applications

PLPTH813, Spring 2025