An Introduction to RNA-sequencing

Differential Expression and Pathway Analysis

W. Evan Johnson, Ph.D.
Professor, Division of Infectious Disease
Director, Center for Data Science
Co-Director, Center for Biomedical Informatics and Health AI
Rutgers University – New Jersey Medical School

2025-07-31

Load Packages for Today

We will be using the following packages for today's lecture: We will be using the following packages for our RNA-seq lecture:

```r
library(tidyverse) ## tools for data wranging
library(SummarizedExperiment) ## managing counts data
library(edgeR) ## differential expression
library(DESeq2) ## differential expression
library(ComplexHeatmap) ## Heatmap visualization
library(TBSignatureProfiler) ## TB signature analysis
library(umap) ## dimenstion reduction and plotting data
```

Installing and using the SCTK

```r
install.packages("devtools")
devtools::install_github("wevanjohnson/singleCellTK")
library(singleCellTK)
singleCellTK()

### Example: open example_data/
### features_combined.txt and meta_data.txt
```

Background

- RNA-seq measures gene expression as count data.
- Count data are overdispersed: variance » mean.
- Poisson regression assumes equal mean and variance — often violated.
- Use the **Negative Binomial (NB)** model to allow for overdispersion.

Negative Binomial Model

Let K_{ij} be the count for gene i in sample j.

$$K_{ij} \sim \text{NB}(\mu_{ij}, \alpha_i)$$

- μ_{ij}: expected count
- α_i: dispersion parameter for gene i

Variance:

$$\text{Var}(K_{ij}) = \mu_{ij} + \alpha_i \mu_{ij}^2$$

Link Function and Design Matrix

We model the expected count using a log link:

$$\log(\mu_{ij}) = \log(s_j) + \mathbf{x}_j^\top \boldsymbol{\beta}_i$$

Where: - s_j: size factor (normalizes for sequencing depth)
- $\mathbf{x}_j$: covariates (e.g., condition, batch)
- $\boldsymbol{\beta}_i$: regression coefficients for gene i

Estimating Dispersion

- Dispersion α_i is typically gene-specific.
- Estimated via:
  - Empirical Bayes shrinkage (DESeq2)
  - Tagwise/Moderated dispersion (edgeR)

Goal: Stabilize estimates for low-count genes across the genome.

Hypothesis Testing

For each gene, test:

$$H_0 : \beta_{i1} = 0 \quad \text{(no differential expression)}$$

Common methods: - Wald Test (DESeq2)
- Likelihood Ratio Test (edgeR, DESeq2)

Adjust for multiple testing:

$$\text{FDR control using Benjamini-Hochberg}$$

Summary of RNA-seq Analysis

- Negative Binomial models handle overdispersion in RNA-seq.
- Log-linear link connects expression to experimental design.
- Shrinkage improves dispersion estimation.
- Differential expression is tested gene-wise with multiple testing correction.

Make a SummarizedExperiment

Using an example dataset from: Verma, et al., 2018

```r
## read in data
counts <- read.table(
  "example_data/features_combined.txt",
  sep="\t", header=T, row.names=1)
meta_data <- read.table(
  "example_data/meta_data.txt",
  sep="\t", header=T, row.names=1)
group <- meta_data$Disease
```

Make a SummarizedExperiment

```r
## Make SummarizedExperiment
se_hivtb <- SummarizedExperiment(assays=list(counts=counts),
                    colData = meta_data)

## Make log counts, counts per million (cpm), logcpm
se_hivtb <- mkAssay(se_hivtb, log = TRUE,
                    counts_to_CPM = TRUE)
assays(se_hivtb)
```

```
## List of length 4
## names(4): counts log_counts counts_cpm log_counts_cpm
```

EdgeR Example

Implements statistical methods for DE analysis based on the negative binomial model:

```r
#Gene Filtering
counts<-counts[which(rowSums(counts)>100),]
#Computes library size
dge <- DGEList(counts=counts, group=group)
#TMM normalization
dge <- calcNormFactors(dge)
# Design matrix
design<-model.matrix(~Disease, data=meta_data)
#Estimates common, trended and tagwise dispersion
dge<-estimateDisp(counts,design)
```

EdgeR Example

In negative binomial models, each gene is given a dispersion parameter. Dispersions control the variances of the gene counts and underestimation will lead to false discovery and overestimation may lead to a lower rate of true discovery.

EdgeR Example

```r
# Neg Bin GLM with the dispersion estimates
fit<-glmFit(counts,design,
            dispersion=dge$tagwise.dispersion)
# Performs likelihood ratio test
# Compares full versus reduced model
lrt<-glmLRT(fit, coef=2)
```

EdgeR Example

```r
# Prints the top results
topTags(lrt)
```

```
## Coefficient:  Diseasetb_hiv
##              logFC   logCPM       LR       PValue          FDR
## IL1R2     4.334138 8.208316 93.61044 3.841569e-22 6.150736e-18
## AP3B2     5.751207 2.952926 64.01082 1.237377e-15 6.755923e-12
## FCGR1C    2.818442 4.536519 63.96598 1.265865e-15 6.755923e-12
## VNN1      3.150042 8.072140 60.78606 6.362689e-15 2.546825e-11
## CYP1B1    3.135490 6.873383 59.54714 1.193995e-14 3.823412e-11
## IL18R1    2.726093 6.487874 58.88977 1.667570e-14 4.449912e-11
## SOCS3     2.665152 6.476286 55.84118 7.856849e-14 1.797086e-10
## SLC29A1  -4.135726 3.970313 54.67430 1.422541e-13 2.699678e-10
## CACNG8   -4.134739 3.190544 54.35946 1.669722e-13 2.699678e-10
## ZAK       2.601721 6.127149 54.34023 1.686139e-13 2.699678e-10
```

EdgeR Example

```r
## Use the a quasi-likelihood F-test
fit<-glmQLFit(counts, design,
        dispersion=dge$tagwise.dispersion)
## use for small datasets
qlf<-glmQLFTest(fit, coef=2)
```

EdgeR Example

```r
# Prints the top results
topTags(qlf)
```

```
## Coefficient:  Diseasetb_hiv
##              logFC   logCPM        F       PValue          FDR
## IL1R2     4.334144 8.208316 96.16041 1.411235e-22 2.259528e-18
## FCGR1C    2.818462 4.536519 65.27824 7.434147e-16 4.644916e-12
## AP3B2     5.751790 2.952926 64.96483 8.703233e-16 4.644916e-12
## VNN1      3.150045 8.072140 62.45782 3.077241e-15 1.231742e-11
## CYP1B1    3.135482 6.873383 61.22274 5.734734e-15 1.836376e-11
## IL18R1    2.726091 6.487874 60.59190 7.882229e-15 2.103373e-11
## SOCS3     2.665156 6.476286 57.45227 3.842609e-14 8.789144e-11
## ZAK       2.601720 6.127149 55.89059 8.455732e-14 1.621038e-10
## FKBP5     2.426321 7.528467 55.66226 9.489672e-14 1.621038e-10
## SLC29A1  -4.135688 3.970313 55.53406 1.012453e-13 1.621038e-10
```

EdgeR Example

```r
#For visualization, heatmaps/PCA
Logcpm<-cpm(counts,log=TRUE)
```

DESeq2 Example

```
#colData is a data frame of demographic/phenotypic data
dds <- DESeqDataSetFromMatrix(countData = counts,
                              colData=meta_data,
                              design=~Disease)
#Gene Filtering
dds<-dds[rowSums(counts(dds))>100,]
```

DESeq2 Example

```r
#Performs estimation of size factors,
#dispersion, and negative binomial GLM fitting
dds<-DESeq(dds)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 138 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

DESeq2 Example

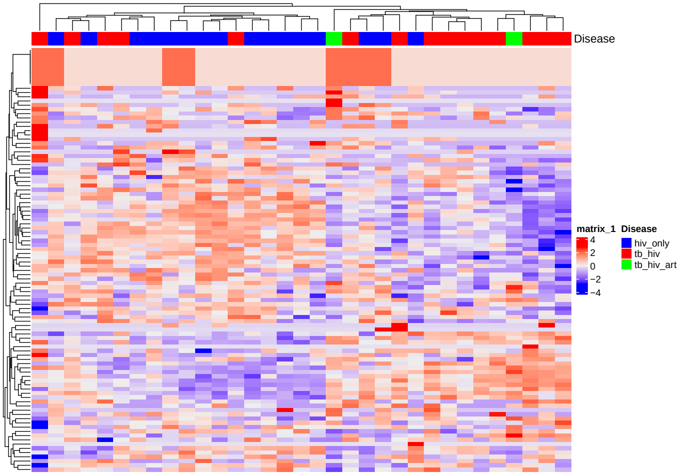
```r
res <- results(dds)[order(results(dds)[,6]),]
res[1:25,]
```

```
## log2 fold change (MLE): Disease tb hiv art vs hiv only
## Wald test p-value: Disease tb hiv art vs hiv only
## DataFrame with 25 rows and 6 columns
##            baseMean log2FoldChange     lfcSE      stat      pvalue        padj
##           <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
## ZEB2      1702.8164        1.75543  0.217589   8.06764 7.16683e-16 1.14748e-11
## DCUN1D3     48.7029        2.52499  0.362090   6.97337 3.09430e-12 2.47714e-08
## TIPARP     724.9429        2.22978  0.333852   6.67894 2.40672e-11 1.28447e-07
## ITPKC      196.8723        3.04144  0.465478   6.53401 6.40298e-11 2.56295e-07
## LAIR1     1528.6852        3.01392  0.471809   6.38800 1.68065e-10 4.25413e-07
## ...             ...            ...       ...       ...         ...         ...
## ATP6V1C1   1006.405        2.50826  0.420512   5.96477 2.44975e-09 1.84789e-06
## TAB3        546.645        1.52153  0.254844   5.97044 2.36615e-09 1.84789e-06
## IL18R1     1777.342        4.19796  0.706351   5.94317 2.79570e-09 1.94617e-06
## NR3C1      2177.200        1.46964  0.247644   5.93451 2.94720e-09 1.96615e-06
## IRAK3      4822.486        4.00584  0.678075   5.90765 3.47012e-09 2.13939e-06
```

Heatmap of DEGs

```r
# Make a Heatmap of DEGs
mat = as.matrix(assay(se_hivtb,"counts")
                )[order(results(dds)[,6])[1:100],]
                # Using first 100 genes to simplify
mat = t(scale(t(mat))) ## scale gene expression
df=data.frame(Disease=colData(se_hivtb)$Disease)
ha = HeatmapAnnotation(df = df,
                       col = list(Disease=c(
                         "tb_hiv"="Red",
                         "hiv_only"="Blue",
                         "tb_hiv_art"="Green")))
Heatmap(mat,show_row_names=F, show_column_names = F,
        top_annotation = ha)
```

Heatmap of DEGs



Limma Example

- Most similar to microarray data flow
- Reads counts are converted to log2 counts per million (logCPM) and the mean-variance relationship is modeled with precision weights (voom transform)

Limma Example

```r
#From edgeR, Computes library size
dge <- DGEList(counts=counts, group=group)
#Gene Filtering
counts<-counts[which(rowSums(cpm(counts))>1),]
dge <- calcNormFactors(dge) #TMM normalization
```

Limma Example

```r
design<-model.matrix(~group)
#voom transform to calculate weights to
#eliminate mean-variance relationship
v<-voom(dge, design)
#use usual limma pipelines
fit<-lmFit(v,design)
fit<-eBayes(fit)
```

Limma Example

```r
topTable(fit, coef=ncol(design))
```

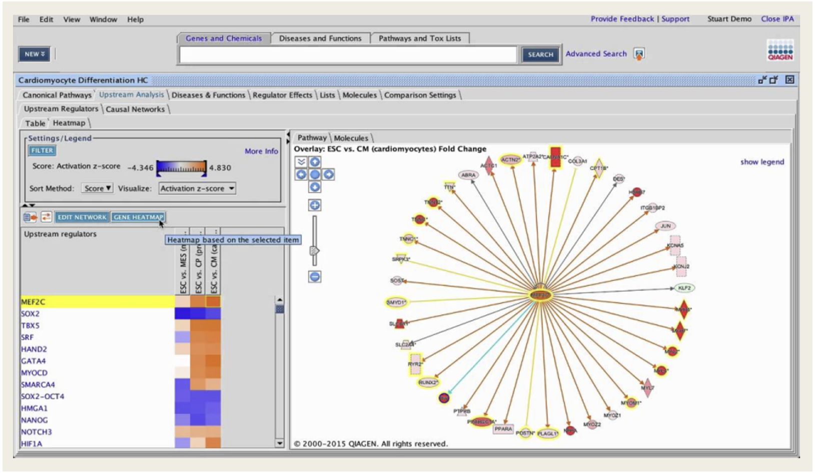
```
##                 logFC   AveExpr        t      P.Value    adj.P.Val         B
## DCUN1D3      2.377176  1.1217001 7.735826 4.125950e-09 3.709568e-05 10.608653
## ZEB2         1.648629  6.4212073 7.535723 7.411363e-09 3.709568e-05 10.235835
## FAM151B      3.664559  1.2897952 7.433182 1.002186e-08 3.709568e-05  9.914427
## C7orf61      2.563702  2.7351541 7.384072 1.158443e-08 3.709568e-05  9.837465
## LINC01093    5.450634  0.8713944 7.395686 1.119395e-08 3.709568e-05  9.782461
## CYP19A1      6.857346 -0.9436098 7.049550 3.127199e-08 6.958151e-05  8.596031
## PGF          5.100817 -3.0391377 7.014062 3.476685e-08 6.958151e-05  7.894619
## IGF2BP3      3.299973  3.6443142 6.650327 1.035781e-07 1.658390e-04  7.749589
## COL4A2-AS1 5.437991 -3.3138293 7.102621 2.669553e-08 6.958151e-05  7.700178
## TIPARP      2.088081  5.0819834 6.585151 1.260852e-07 1.835228e-04  7.561353
```

Pathway analysis

After finding DEGs, look for correlated genes/networks and enriched pathway sets in the gene set using:

- Weighted gene coexpression network analysis (WGCNA)
- GSEA, GSVA, EnrichR, many more!!
- Qiagen Ingenuity Pathway Analysis (IPA)

Pathway analysis



Score: Activation z-score -4.346 ▬ 4.830

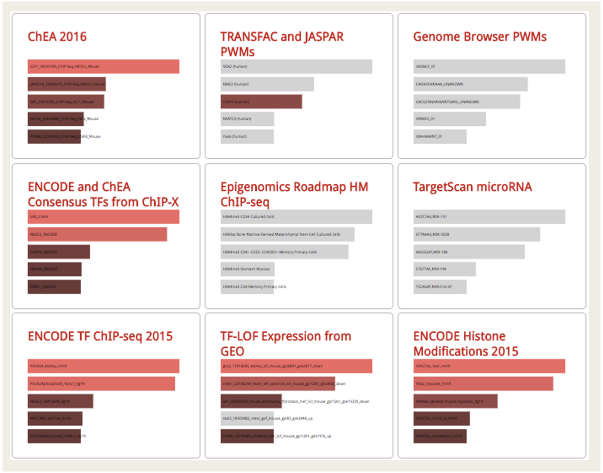
Sort Method: Score Visualize: Activation z-score

Overlay: ESC vs. CM (cardiomyocytes) Fold Change

Upstream regulators	ESC vs. MES	ESC vs. CP	ESC vs. CM
MEF2C			
SOX2			
TBX5			
SRF			
HAND2			
GATA4			
MYOCD			
SMARCA4			
SOX2-OCT4			
HMGA1			
NANOG			
NOTCH3			
HIF1A			

© 2000-2015 QIAGEN. All rights reserved.

Pathway analysis



TBSignatureProfiler Analysis

The TBSignatureProfiler was developed in the Johnson Lab in 2021 to profile new and existing TB gene expression signatures:

https://bmcinfectdis.biomedcentral.com/articles/10.1186/s12879-020-05598-z

TBSignatureProfiler Analysis

```r
se_hivtb_2 <- se_hivtb[,
      colData(se_hivtb)$Disease != "tb_hiv_art"]
TBsigs <- TBsignatures[-12]
ssgsea_res <- runTBsigProfiler(se_hivtb_2,
                  useAssay = "log_counts_cpm",
                  signatures = TBsigs,
                  algorithm = "ssGSEA",
                  combineSigAndAlgorithm = TRUE,
                  parallel.sz = 1)
```

WEJ LAB

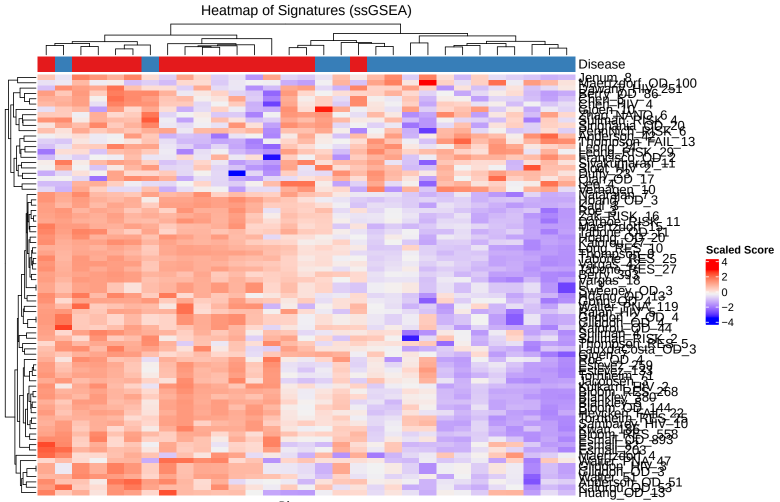
Center for Data Science

Signature Heatmap:

```
# Colors for gradient
signatureHeatmap(ssgsea_res,
        name = "Heatmap of Signatures (ssGSEA)",
        signatureColNames = names(TBsigs),
        annotationColNames = c("Disease"),
        scale = TRUE,
        split_heatmap = "none",
        showColumnNames = FALSE)
```

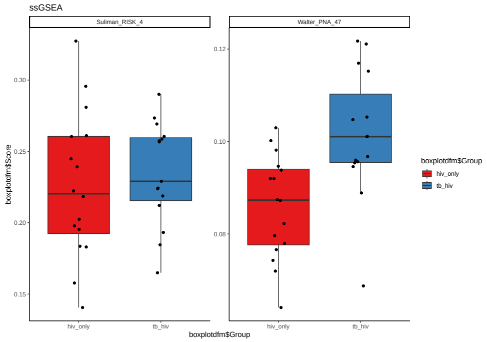
Center for Data Science

Signature Heatmap:



Signature Boxplots

```
signatureBoxplot(ssgsea_res, name="ssGSEA",
              signatureColNames = names(TBsigs)[c(62,77)],
              annotationColName = c("Disease"))
```



Session info

```r
sessionInfo()
```

```
## R version 4.5.1 (2025-06-13)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sequoia 15.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  LAPACK version 3.12.1
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] grid      stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] umap_0.2.10.0               TBSignatureProfiler_1.20.0
##  [3] ComplexHeatmap_2.24.1       DESeq2_1.48.1
##  [5] edgeR_4.6.3                 limma_3.64.1
##  [7] SummarizedExperiment_1.38.1 Biobase_2.68.0
##  [9] GenomicRanges_1.60.0        GenomeInfoDb_1.44.1
## [11] IRanges_2.42.0              S4Vectors_0.46.0
```