Intro. to Tests for Differential Expression (Part 2)

Utah State University – Spring 2014

STAT 5570: Statistical Bioinformatics

Notes 3.4

```
### First prepare objects for DE test
### (as on slide 13 of Notes 3.3)

# load data
library(affy); library(ALL); data(ALL)

# obtain relevant subset of data; similar Notes 3.2 p. 11
# (filter genes on raw scale, then return to log scale)
library(genefilter); e.mat <- 2^exprs(ALL)
ffun <- filterfun(pOverA(0.20,100))
t.fil <- genefilter(e.mat,ffun)
small.eset <- log2(e.mat[t.fil,])
dim(small.eset) # 4305 genes, 128 arrays

# define comparison to be tested
# first 95 are B-cell, then last 33 are T-cell
T.cell <- c(rep(0,95),rep(1,33))
#  0=B-cell, 1=T-cell
```

Ex 3: limma / eBayes

- (Smyth) Linear models for microarray data
- A very flexible family of models, for multiple technologies
- Two stages:
  - First, fit linear model (ANOVA framework)
  - Second, “moderate” test statistic by borrowing information across genes

Recall one-way ANOVA model (here, for each gene k separately)

- Consider a single factor (treatment) with # levels $\alpha \geq 2$, and Y is log-scale expr.

- Means model: $Y_{ijk} = \mu_{ik} + \varepsilon_{ijk}$

- Effects model: $Y_{ijk} = \mu_k + A_{ik} + \varepsilon_{ijk}$, $\Sigma_i A_{ik} = 0$

- General assumption: ε_{ijk} iid $N(0,\sigma_k^2)$

Matrix representation (per gene, subscript k suppressed)

- $\mathbb{Y}$ = vector of Y_{ij}'s
- $\boldsymbol{\beta}$ = vector of parameters
- $\boldsymbol{\varepsilon}$ = vector of error terms ε_{ij} iid $N(0,\sigma^2)$
- $\mathbb{X}$ = design matrix (with row for each obs. and column for each parameter) such that:

  $$\mathbb{Y} = \mathbb{X}\,\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

- Obtain parameter estimates: $\hat{\beta} = (\mathbb{X}'\mathbb{X})^{-1}\mathbb{X}'\mathbb{Y}$

β vector and X matrix depend on parameterization

- Means model (~0+A)

$$\beta = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \end{bmatrix} \qquad X = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 0 & & 0 \\ \vdots & & & \\ 0 & 1 & & \\ 0 & 1 & & \\ \vdots & & & 1 \end{bmatrix}$$

β vector and $\mathbb{X}$ matrix depend on parameterization

- Effects model (~A)

$$\beta = \begin{bmatrix} \mu \\ A_1 \\ \vdots \\ A_{a-1} \end{bmatrix} \qquad X = \begin{bmatrix} 1 & 1 & 0 & \dots \\ 1 & 1 & 0 & \\ \vdots & & & \\ 1 & & 1 & \\ 1 & & 1 & \\ \vdots & & & \end{bmatrix}$$

Simple two-group comparison

```
# prepare
library(limma)

# Define factors in model
Cell <- as.factor(c(rep('B',95),rep('T',33)))

# Define Design matrix (X)
design <- model.matrix(~0+Cell)
design
```

	CellB	CellT
1	1	0
2	1	0
3	1	0
4	1	0
...		
126	0	1
127	0	1
128	0	1

Contrasts

- Given a design matrix $\mathbf{X}$, test nulls of interest using contrasts:
  - Linear combination of parameters where sum of coefficients is zero:

$$\Psi = \Sigma_i w_i \beta_i \qquad \Sigma_i w_i = 0$$

- In our two-group comparison, test H_0: $\mu_B = \mu_T$ using contrast $\Psi = \mu_B - \mu_T$
  - H_0: $\Psi = 0$
  - One contrast → 1 d.f.

Contrasts with # groups > 2

- Example: H_0: $\mu_1=\mu_2=\mu_3$ using two contrasts:
  - $\Psi_1 = \mu_1 - \mu_2$
  - $\Psi_2 = \mu_2 - \mu_3$

- H_0: $\mu_1=\mu_2=\mu_3$ is equivalent to:

  H_{01}: $\Psi_1 = 0$ <u>and</u> H_{02}: $\Psi_2 = 0$

- Two contrasts → 2 D.F.

Testing Contrasts: H_0: $\Psi=0$ (here, bringing back subscript k)

$$\Psi = \sum_i w_i \beta_i \qquad \hat{\Psi} = \sum_i w_i \hat{\beta}_i = w' \hat{\beta}$$

$$Var\left(\hat{\Psi}\right) = w' Cov\left(\hat{\beta}\right) w = w' V w \cdot \hat{\sigma}_k^2$$

$$F = \frac{1}{\hat{\sigma}_k^2} \cdot \left(\frac{\hat{\Psi}}{w' V w} \right)^2 \sim F_{1,d_k}$$

Hierarchical model to borrow information across genes (Smyth): eBayes

Assume prior distribution $\frac{1}{\sigma_k^2} \sim \frac{1}{d_0 s_0^2} \chi^2_{d_0}$

(s_0^2 and d_0 estimated from data using empirical Bayes methods)

(using all of the genes)

Consider the posterior mean $\tilde{\sigma}_k^2 = E\left[\sigma_k^2 \mid \hat{\sigma}_k^2\right] = \frac{d_0 s_0^2 + d_k \hat{\sigma}_k^2}{d_0 + d_k}$

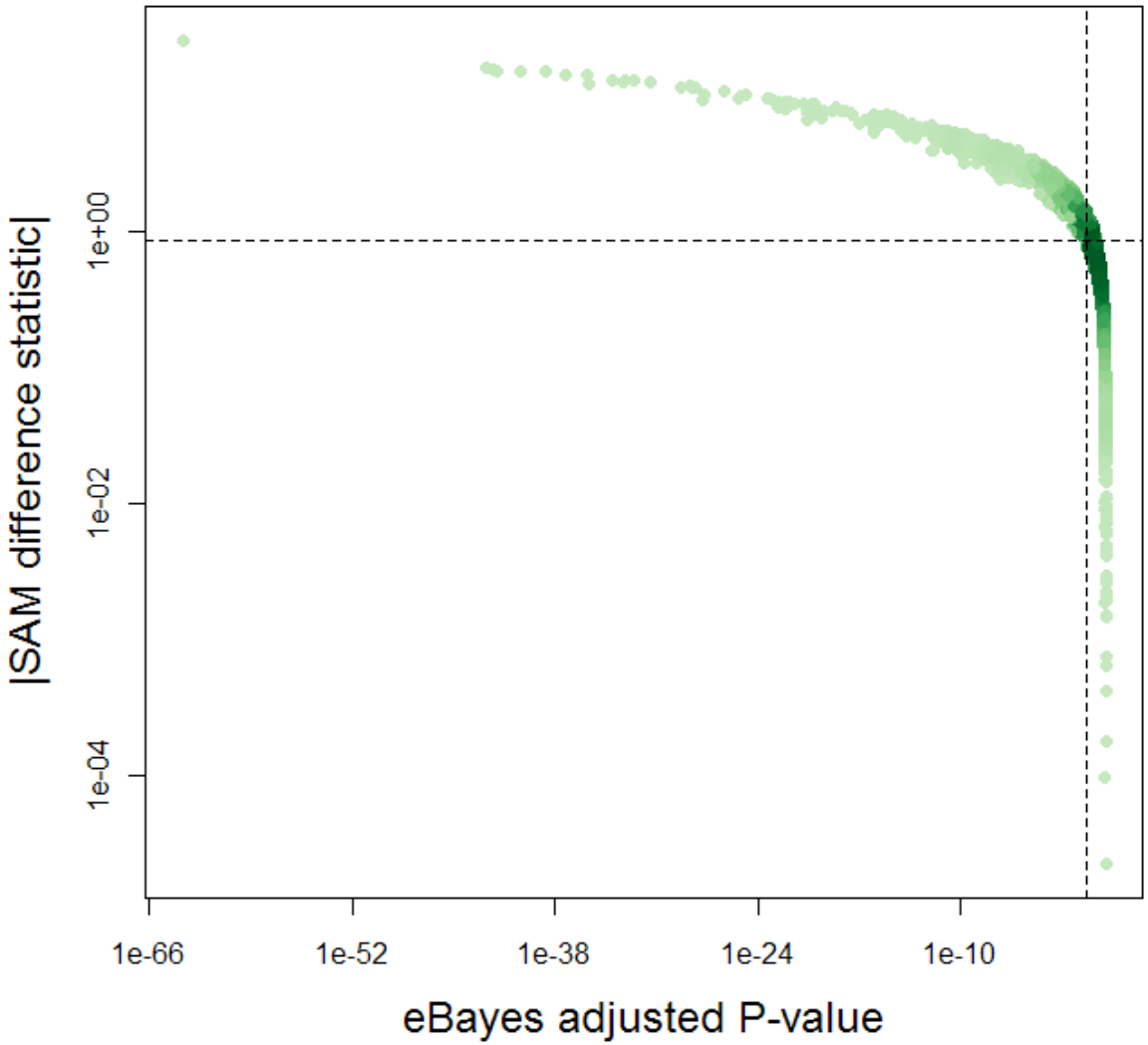
Then the "moderated" F-statistic $\tilde{F} = \frac{1}{\tilde{\sigma}_k^2} \cdot \left(\frac{\hat{\Psi}}{w' V w} \right) \sim F_{1,(d_0+d_k)}$

represents: added information
(from using all genes)

```
# Fit linear model
colnames(design) <- c('B','T')
fit <- lmFit(small.eset, design)

# Test Cell effect after eBayes moderation
contrast.Cell <- makeContrasts(B-T, levels=design)
fit.Cell <- contrasts.fit(fit, contrast.Cell)
final.fit.Cell <- eBayes(fit.Cell)
top.Cell <- topTableF(final.fit.Cell, n=nrow(small.eset))
head(top.Cell)

# Get significant genes
gn.eB <- rownames(top.Cell)[top.Cell$adj.P.Val<.05]
length(gn.eB) # 1443 genes
```

	B...T	AveExpr	F	P.Value	adj.P.Val
38319_at	-4.655042	6.041217	1242.0961	4.901575e-68	2.110128e-64
33238_at	-3.102294	7.292159	514.1754	8.067444e-47	1.736517e-43
35016_at	3.214222	10.337892	497.9320	4.209382e-46	6.040463e-43
2059_s_at	-2.668482	7.232735	489.0459	1.058404e-45	1.139107e-42
37039_at	3.265990	11.072596	454.2931	4.448245e-44	3.829939e-41
38095_i_at	3.762299	10.228156	416.3136	3.446245e-42	2.472681e-39

Compare limma & SAM results



```
# Get all results in single data.frame
# (First run Notes 3.3 code to create comb object)
eB.frame <- data.frame(gn=rownames(top.Cell),
   eB.p=top.Cell$adj.P.Val)
newcomb <- merge(comb,eB.frame)
head(newcomb)
#         gn  SAM.diff      maxT.p          eB.p
#1  100_g_at 0.6015435 0.154559598 1.979672e-01
#2   1000_at 1.4462143 0.004837079 1.791954e-03
#3   1005_at 1.1961784 0.075836759 7.841016e-02
#4 1007_s_at 3.2327910 0.004837079 1.042560e-06
#5 1008_f_at 0.1260003 0.878172179 9.043473e-01

# Visualize comparison
dCol <- densCols(log(newcomb$eB.p), log(newcomb$SAM.diff),
  colramp=green.ramp)
plot(newcomb$eB.p,  newcomb$SAM.diff, col=dCol,
  log='xy', pch=16, cex.lab=1.5,
  xlab='eBayes adjusted P-value',
  ylab='|SAM difference statistic|')
abline(v=.05,lty=2); abline(h=.87,lty=2)
```

Multiple Factor Levels

- limma framework based on design of experiments
  - design matrices (multiple factors and factor levels)
  - contrasts (special comparisons of interest)

- Example: ALL data (128 patients)
  - Before: 95 B-cell vs. 33 T-cell
  - Now: Stage (1-4), Sex (M/F)

```
# Define factors in model
Cell <- as.factor(c(rep('B',95),rep('T',33)))

Stage <- as.factor( c(
2,2,4,1,2,1,1,1,2,2,3,3,3,2,3,1,2,3,2,3,2,2,2,1,
1,2,1,2,1,2,1,1,2,2,2,1,2,2,2,2,2,4,4,2,2,2,4,2,
1,2,2,3,4,3,3,3,4,3,3,1,1,1,1,3,3,3,3,3,3,3,3,1,
3,1,4,2,2,1,3,4,4,2,2,3,4,4,4,1,2,2,2,1,2,1,1,1,
3,2,2,3,2,1,4,2,3,3,1,2,3,2,2,2,1,4,1,2,3,2,2,2,
2,3,3,3,2,3,2,1) )

Sex <- as.factor( c(
"M","M","F","M","M","M","F","M","M","M","M","M","M","M",
"M","F","M","M","M","F","M","M","M","M","M","M","F","F",
"F","F","F","F","M","M","F","F","F","F","F","M","F","M",
"F","M","M","M","F","M","F","F","F","M","M","M","M","F",
"M","F","M","F","M","M","F","M","M","M","M","M","M","M",
"F","M","M","F","F","M","M","M","F","M","M","M","F","F",
"F","M","M","M","M","F","M","M","F","M","F","M","F","F",
"M","F","M","M","M","M","M","M","F","F","M","F","M","M",
"M","F","M","M","F","M","M","M","M","M","M","M","M","M",
"M","F") )
```

```
# Check factor level counts
table(Sex:Stage)
#
# F:1 F:2 F:3 F:4 M:1 M:2 M:3 M:4
#  14  15   9   6  16  36  24   8

# Define response variable (Y) and Design matrix (X)
eset <- exprs(ALL)
design <- model.matrix(~0+Sex:Stage)
head(design)
```

	SexF:Stage1	SexM:Stage1	SexF:Stage2	SexM:Stage2	SexF:Stage3	SexM:Stage3	SexF:Stage4	SexM:Stage4
1	0	0	0	1	0	0	0	0
2	0	0	0	1	0	0	0	0
3	0	0	0	0	0	0	1	0
4	0	1	0	0	0	0	0	0
5	0	0	0	1	0	0	0	0
6	0	1	0	0	0	0	0	0

```
colnames(design) <- c('F1','M1','F2','M2','F3','M3','F4','M4')
fit <- lmFit(eset, design)
```

```
# Test Sex effect (after controlling for Stage)
contrast.Sex <- makeContrasts(F1+F2+F3+F4-M1-M2-M3-M4,
   levels=design)
fit.Sex <- contrasts.fit(fit, contrast.Sex)
final.fit.Sex <- eBayes(fit.Sex)
top.Sex <- topTableF(final.fit.Sex, n=nrow(eset))
head(top.Sex)
```

ID	F1...F2...F3...F4...M1...M2...M3...M4	AveExpr	F	P.Value	adj.P.Val
41214_at	-16.916692	8.018510	399.5523	1.957628e-40	2.471506e-36
38355_at	-18.995516	6.963830	322.9287	3.435485e-36	2.168650e-32
37583_at	-6.615164	6.831782	209.5935	2.474373e-28	1.041299e-24
38446_at	9.135225	4.245985	194.2986	4.532476e-27	1.430563e-23
34477_at	-4.519017	5.108099	106.1668	2.526096e-18	6.378392e-15
35885_at	-5.333148	4.676934	103.4995	5.222972e-18	1.099000e-14

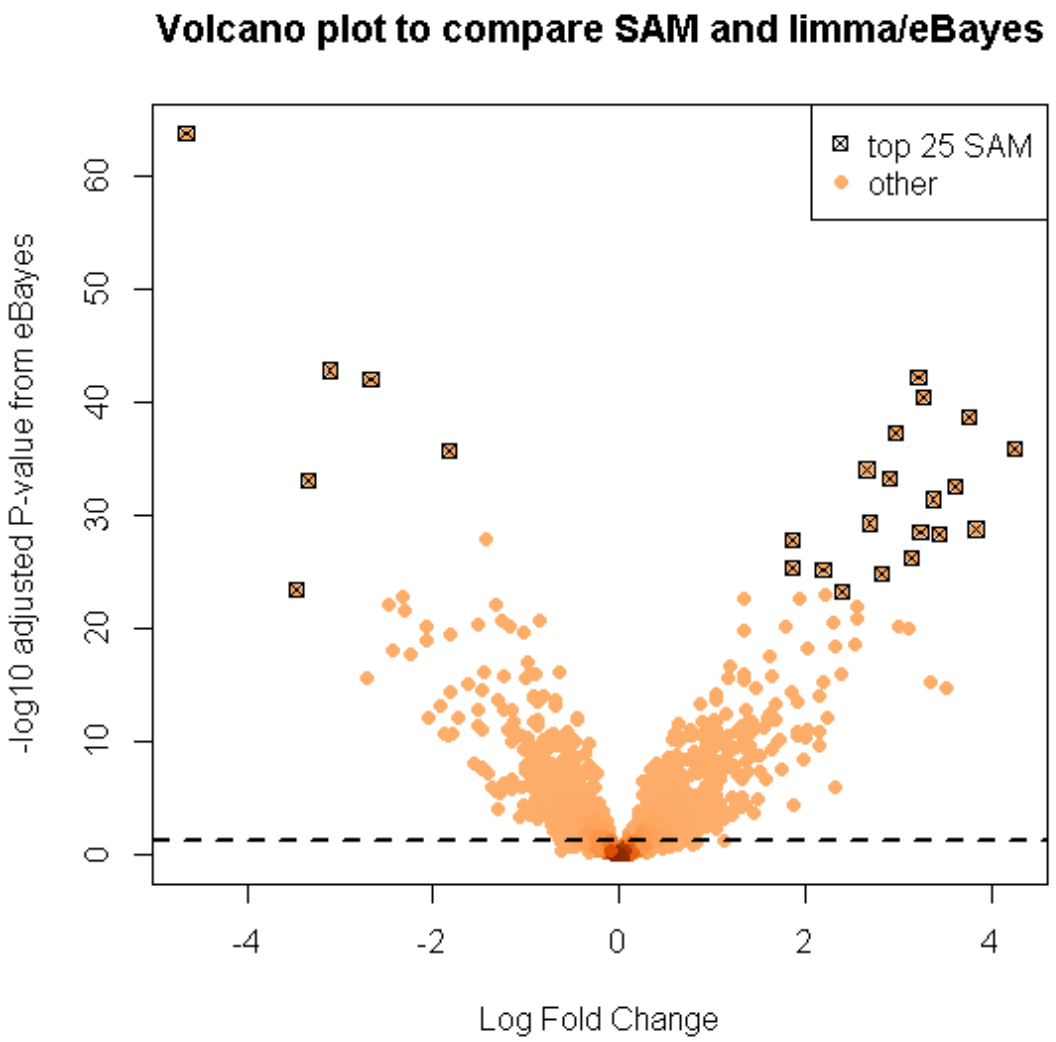
```
# Test Stage effect (after controlling for Sex)
contrast.Stage <- makeContrasts(F1+M1-F2-M2, F1+M1-F3-M3,
   F1+M1-F4-M4, levels=design)
fit.Stage <- contrasts.fit(fit, contrast.Stage)
final.fit.Stage <- eBayes(fit.Stage)
top.Stage <- topTableF(final.fit.Stage, n=nrow(eset))
head(top.Stage)
```

	F1...M1...F2...M2	F1...M1...F3...M3	F1...M1...F4...M4	AveExpr	F	P.Value	adj.P.Val
1914_at	2.5675495	3.258069	3.2932071	4.439241	18.44363	6.065565e-10	7.657776e-06
1389_at	-2.1549694	-3.000142	-3.4004355	9.468680	13.46102	1.185497e-07	7.483450e-04
39716_at	0.8615624	1.011228	0.9968166	3.517922	12.24857	4.576102e-07	1.925776e-03
38032_at	-1.6112750	-1.551330	-1.8252913	7.872982	11.55082	1.008126e-06	3.181897e-03
35614_at	0.1686151	-2.616389	-1.7489307	5.788071	10.75792	2.501554e-06	6.316425e-03
40268_at	0.6599309	1.515524	1.5172557	8.058545	10.51488	3.313193e-06	6.971510e-03

Useful visualization technique: the volcano plot

Good for:

1. comparing:
   DE methods
2. visualizing
   importance of:
   variability



```
# Volcano plot - start
v.frame <- data.frame(gn = rownames(top.Cell),
   LFC = top.Cell$B...T,
   LOD = -log10(top.Cell$adj.P.Val))
vp <- merge(newcomb, v.frame)

# make main plot
o.ramp <- colorRampPalette(brewer.pal(9,"Oranges")[4:9])
dCol <- densCols(vp$LFC, vp$LOD, colramp=o.ramp)
plot(vp$LFC,vp$LOD,col=dCol,pch=16,xlab='Log Fold Change',
     ylab='-log10 adjusted P-value from eBayes',
     main='Volcano plot to compare SAM and limma/eBayes')

# highlight top 25 genes in SAM
t.25 <- rev(order(vp$SAM.diff))[1:25]
points(vp$LFC[t.25],vp$LOD[t.25],pch=7,col='black')

# Add reference line for FDR and a legend
abline(h=-log10(0.05),lwd=2,lty=2)
legend('topright',c('top 25 SAM','other'),
       pch=c(7,16),col=c('black',o.ramp(1)),cex=1)
```

Sample size concerns

- What if have few arrays?
  - 1. Try to: get more arrays
  - 2. Somehow: “pool” info. across genes

- What if just one array per sample type?
  - Could use “housekeeping” or “control” genes as benchmarks
  - Affymetrix software uses SLR: signal log ratio
    - Based on: probe-level differences between arrays
    - Uses one-step Tukey’s biweight to estimate: change in location for each gene
  - (Dangerous without reasonable estimate of within-group variability)

affyNFM: DE test with small sample sizes

- Rather than lose probe-level information by summarizing, fit (per-gene) model on bg-corrected and quantile-normalized intensities:

$$Y_{ijl} = \mu + T_i + S_{j(i)} + P_l + (TP)_{il} + (SP)_{jl(i)}$$

$$H_0 : T_i = 0 \ \ \forall i$$

- Get F-statistic for each gene by (iterative) REML estimation
- Get p-value for each gene by permutation
  - enumerate all permutations → empirical sampling distribution of F (using all genes)
  - compare gene's original F to empirical sampling distribution
- Very powerful for small-sample studies

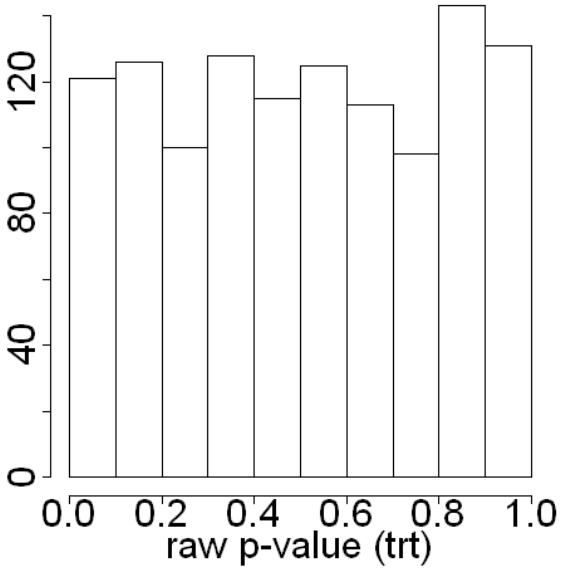
Stevens, Bell, Aston, & White (2010)

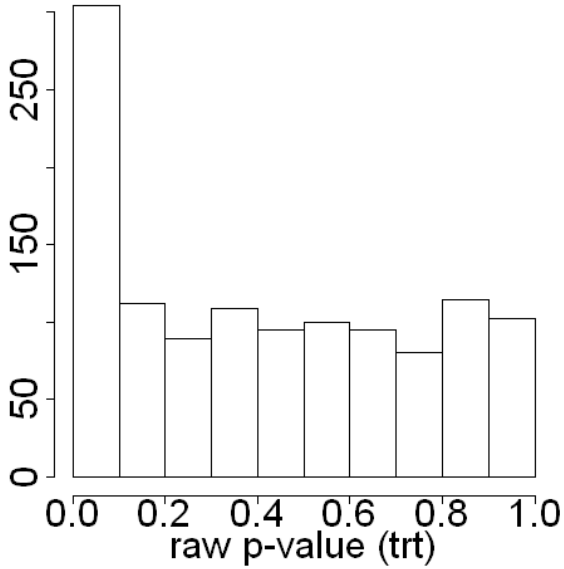
```
# Load necessary functions
library(nlme); library(perm)
source("http://www.stat.usu.edu/jrstevens/affyNFM.R")

# Create AffyBatch object
library(affydata); data(Dilution)
use.abatch <- Dilution  # 4 arrays
# Define subset of genes (severe filter to save class time)
library(genefilter); e.mat <- 2^exprs(rma(use.abatch))
ffun <- filterfun(pOverA(0.20,2^9),cv(.07))
t.fil <- genefilter(e.mat,ffun)
use.gn <- geneNames(use.abatch)[t.fil]  # 108 genes, 4 arrays

# Define comparison of interest – indices for ctl and trt
use.t1 <- c(1,2); use.t2 <- c(3,4)

# Perform NFM model fit; about 30 seconds
use.frame <- affyNFM(abatch = use.abatch,
   t1 = use.t1, t2 = use.t2,
   gn = use.gn, verbose = TRUE)
pframe <- nfm.pvals(use.frame)
head(pframe)
```

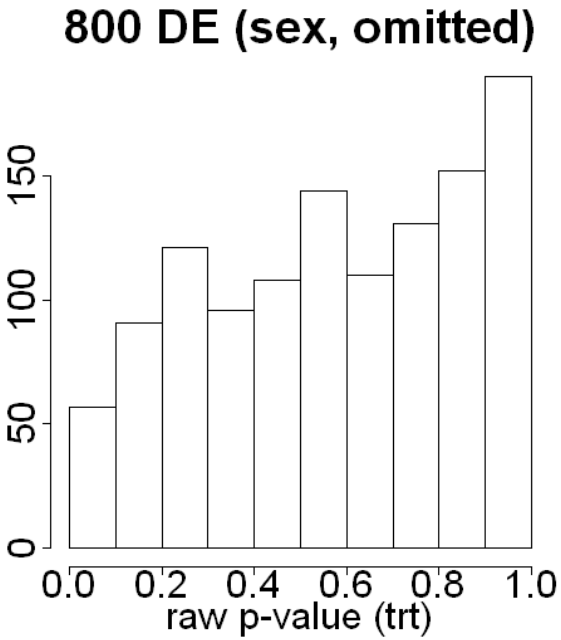
	gn	F	p
1	1077_at	6.419	0.0617
2	1173_g_at	3.186	0.1913
3	1180_g_at	0.6479	0.4814
4	1366_i_at	8.012	0.0154
5	1449_at	1.178	0.3858
6	1653_at	6.136	0.0802

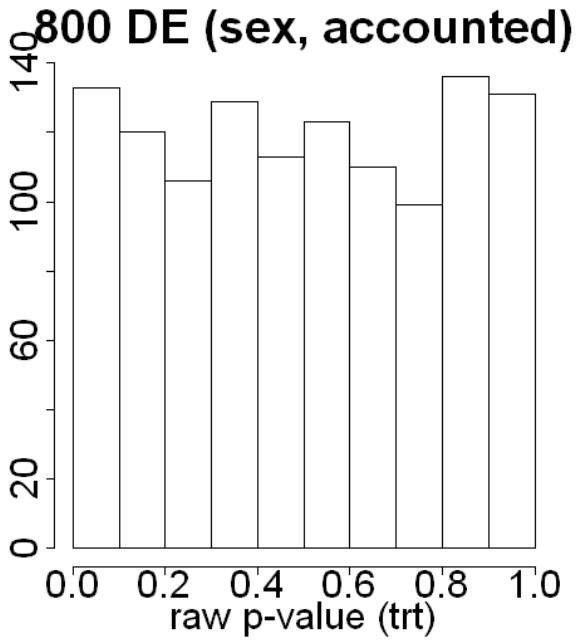
A final note on [raw] p-value histograms

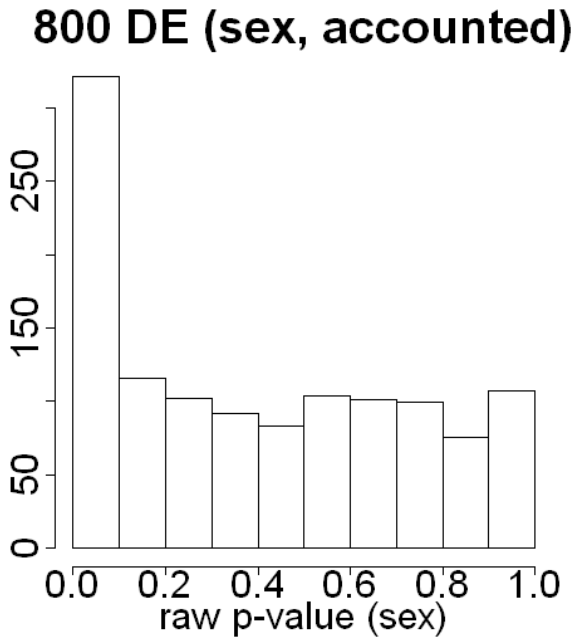
- Recall:
  - Flat histogram suggests: no DE genes
  - Peak near zero suggests: some DE genes
- What about a ‘dip’ near zero?
  - This suggests there is DE based on an omitted covariate
- [Simulated] example: 4800 genes on 4 arrays
  - Two factors: trt (C/T) and sex (M/F)
    - 1: C-M 2: C-F 3: T-M 4: T-F
  - Testing for trt when only sex is causing DE …


No DE
0
40
80
120
0.0
0.2
0.4
0.6
0.8
1.0
raw p-value (trt)


800 DE (trt)
0
50
150
250
0.0
0.2
0.4
0.6
0.8
1.0
raw p-value (trt)


800 DE (sex, omitted)
0
50
100
150
0.0
0.2
0.4
0.6
0.8
1.0
raw p-value (trt)


800 DE (sex, accounted)
0
20
60
100
140
0.0
0.2
0.4
0.6
0.8
1.0
raw p-value (trt)

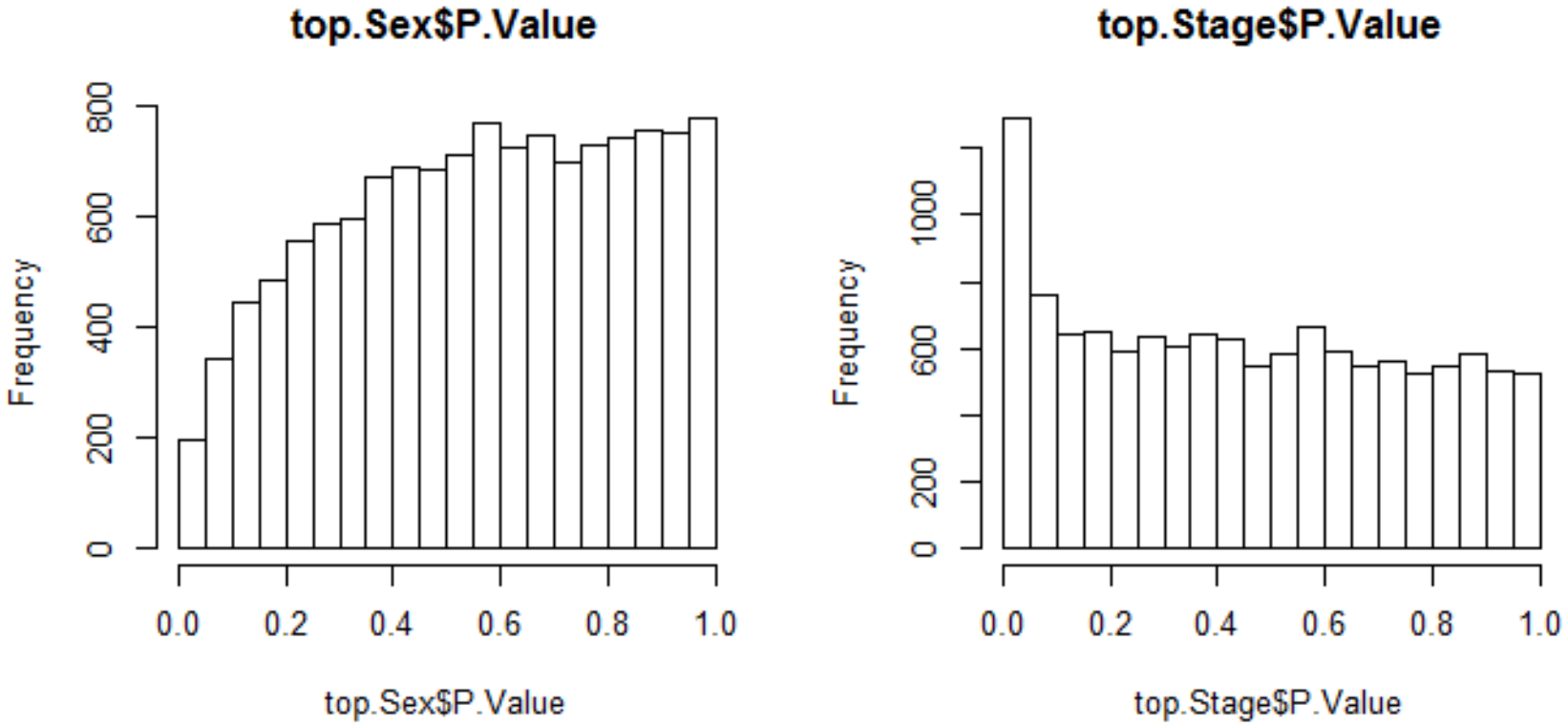

800 DE (sex, accounted)
0
50
150
250
0.0
0.2
0.4
0.6
0.8
1.0
raw p-value (sex)

```
library(limma)
design <- cbind(Intercept=1,trt=c(0,0,1,1))
par(mfrow=c(2,3),ps=35); use.seed <- 12

# First simulate 4000 noisy (non-DE) genes on 4 arrays
set.seed(use.seed)
n4000 <- cbind(
   rnorm( 1000, 20, 4 ),    # C M
   rnorm( 1000, 20, 4 ),    # C F
   rnorm( 1000, 20, 4 ),    # T M
   rnorm( 1000, 20, 4 ) )   # T F

# Now simulate 800 genes (non-DE)
set.seed(use.seed)
g.nonDE <- cbind(
   rnorm( 200, 20, 4 ),    # C M
   rnorm( 200, 20, 4 ),    # C F
   rnorm( 200, 20, 4 ),    # T M
   rnorm( 200, 20, 4 ) )   # T F

# Test
data <- rbind(n4000,g.nonDE)
p <- topTable(eBayes(lmFit(data,design)),n=4800,coef=2)$P.Value
hist(p, main='No DE',xlab='raw p-value (trt)',ylab=NA)
```

```
# Replace those with 800 genes (DE based on trt)
set.seed(use.seed)
g.DE <- cbind(
   rnorm( 200, 20, 4 ),    # C M
   rnorm( 200, 20, 4 ),    # C F
   rnorm( 200, 40, 4 ),    # T M
   rnorm( 200, 40, 4 ) )   # T F
data <- rbind(n4000,g.DE)
p <- topTable(eBayes(lmFit(data,design)),n=4800,coef=2)$P.Value
hist(p, main='800 DE (trt)',xlab='raw p-value (trt)',ylab=NA)

# Replace those with 800 genes (DE based on sex, but sex omitted)
par(mfg=c(2,1))
set.seed(use.seed)
g.DE.sex <- cbind(
   rnorm( 200, 20, 4 ),    # C M
   rnorm( 200, 40, 4 ),    # C F
   rnorm( 200, 20, 4 ),    # T M
   rnorm( 200, 40, 4 ) )   # T F
data <- rbind(n4000,g.DE.sex)
p <- topTable(eBayes(lmFit(data,design)),n=4800,coef=2)$P.Value
hist(p, main='800 DE (sex, omitted)',
     xlab='raw p-value (trt)',ylab=NA)
```

```
# Now account for the omitted covariate
design <- cbind(Intercept=1,trt=c(0,0,1,1),sex=c(0,1,0,1))
design
#     Intercept trt sex
#[1,]         1   0   0
#[2,]         1   0   1
#[3,]         1   1   0
#[4,]         1   1   1

# Test for DE based on trt
p.trt <- topTable(eBayes(lmFit(data,design)),n=4800,coef=2)$P.Value
hist(p.trt, main='800 DE (sex, accounted)',
     xlab='raw p-value (trt)',ylab=NA)

# Test for DE based on sex
p.sex <- topTable(eBayes(lmFit(data,design)),n=4800,coef=3)$P.Value
hist(p.sex, main='800 DE (sex, accounted)',
     xlab='raw p-value (sex)',ylab=NA)
```

Example (ALL, omitting Cell)




```
par(mfrow=c(2,2))
hist(top.Sex$P.Value, main='top.Sex$P.Value')      # slide 19
hist(top.Stage$P.Value, main='top.Stage$P.Value') # slide 20
```

Summary

- Common themes of DE tests
  - Pool information to estimate: population SD
  - Distribution of test statistic
    - Parametric: assume distribution
    - Nonparametric: generate distribution with - permutations
- Methods usually based on probeset-level data:
  - SAM
  - maxT
  - limma / eBayes
- Others exist: probe-level (affyNFM for small sample size), weighted, poe, …
- Visualization
  - Volcano plot
  - Histogram of raw p-values (and accounting for relevant covariates)