
Gene Expression – an Overview of Problems & Solutions: 3&4

Utah State University

Bioinformatics: Problems and Solutions

Summer 2006

Review

- Considering several problems & solutions with gene expression data
 - Previously:
 - 1: Data quality checks
(GIGO; use graphical checks)
 - 2: Preprocessing
(Background, Normalization, Summarization)
 - Here:
 - 3: Differential Expression
 - 4: Effective Visualization
 - Reference: Bioinformatics and Computational Biology Solutions Using R and Bioconductor, edited by Gentleman et al.
-

Problem 3: Testing for Differential Expression

- “Observe” gene expression in different conditions – healthy vs. diseased, e.g.
 - Decide which genes’ expression levels are changing significantly between conditions
 - Target those genes – to halt disease, e.g.
 - Note: there are far too many ways to test for DE to present here – we will just look at major themes of most of them, and focus on implementing one
-

Simple / Naïve test of DE

- “Observe” gene expression levels under two conditions (after preprocessing)

Y_{ijk} = log expr. level of gene k in replicate j of "treatment" i

- Calculate: average log fold change

$\bar{Y}_{i \cdot k}$ = ave. log expr. for gene k in treatment i

$LFC_k = \bar{Y}_{2 \cdot k} - \bar{Y}_{1 \cdot k}$ = ave. log fold change for gene k

- Make a cut-off: R

Gene k is "significant" if $|LFC_k| > R$

What's wrong with the naïve test?

- Summary interpretation okay:
 - LFC > 0 for “up-regulated” genes
 - LFC < 0 for “down-regulated” genes
 - Ignores variability of estimate
 - cannot really test for “significance”
 - what if larger LFC have large variability?
 - then not necessarily significant
-

How to take variability into account?

- Build some test statistic on a per-gene basis
- How do we “usually” test for differences between two groups or samples?

two-sample t-test

- Test statistic:

$$t_k = \frac{\bar{Y}_{2 \cdot k} - \bar{Y}_{1 \cdot k}}{s_k} = \frac{LFC_k}{s_k} \longleftarrow \text{pooled SD}$$

How to use this to “test” for DE?

- What is being tested?

Null: No change for gene k

- Under null, $t_k \sim t$ dist. with n_k d.f.

“parametric” assumption

- But what is needed to do this?
 - “Large” sample size
 - Estimate $\sigma_k =$ “pop. SD” for gene k
-

What if we don't have a large enough sample size?

- Two main problems:
 - 1. Estimate σ_k (especially for small sample size)
 - 2. Appropriate sampling distribution of test stat.
 - Basic solutions:
 - 1. To estimate σ_k : Pool information across genes
 - 2. For comparison:
 - use parametric assumption on “improved” test stat.
 - use non-parametric methods
-

Generalize t-test: linear model (limma)

- For gene k under treatment j on array i :

$$Y_{ijk} = \beta_{k,0} + \beta_{k,1} T_{jk} + \varepsilon_{ijk}, \quad \text{Var}[\varepsilon_{ijk}] = \sigma_k^2$$

expression level (log scale) treatment effect (DE) treatment level (could be more than just 2 levels)

- What if there are more covariates than just treatment? –

use matrix notation for convenience:

$$E[Y_k] = X\beta_k$$

log-scale expression vector design matrix (n x p) covariate effects

Assumptions in linear model (Smyth)

Obtain estimates $\hat{\beta}_k$ and $\hat{\sigma}_k$, and $Var[\hat{\beta}_k] = V_k \hat{\sigma}_k^2$

For covariate w ,

$$\hat{\beta}_{k,w} \mid \beta_{k,w}, \sigma_k^2 \sim N(\beta_{k,w}, V_{k,w} \hat{\sigma}_k^2)$$

$$\hat{\sigma}_k^2 \mid \sigma_k^2 \sim \frac{\sigma_k^2}{d_k} \chi_{d_k}^2, \quad d_k = \text{resid. d.f.} = \underbrace{n_k - p_k}_{\text{k not necessary here}}$$

Then $t_{k,w} = \frac{\hat{\beta}_{k,w}}{\hat{\sigma}_k \sqrt{V_{k,w}}} \sim t_{d_k}$

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

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

$(s_0^2 \text{ and } d_0 \text{ estimated from data using empirical Bayes methods})$

(using all of the genes)

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

Then the "moderated" t - statistic $\tilde{t}_{k,w} = \frac{\hat{\beta}_{k,w}}{\tilde{\sigma}_k \sqrt{V_{k,w,w}}} \sim t_{d_0+d_k}$

↑
represents added information from using all genes

```
# Startup
memory.limit(size=4000)
Sys.putenv("http_proxy"="http://proxy.usu.edu:80/")
Sys.getenv("http_proxy")
source("http://www.stat.usu.edu/~jrstevens/get.packages.R")
library(affy)
windows(record=T)

# Load data
library(ALL); data(ALL)

# define comparison (based on knowledge of samples)
eset <- ALL # these are normalized expression levels
sampleNames(eset); phenoData(eset)
phenoData(eset)$BT
trt <- c(rep(0,95),rep(1,33)) # 0=B, 1=T

# test for differential expression (DE)
library(limma)
design <- cbind(Intercept=1,trt=trt)
fit <- lmFit(eset@exprs,design)
e.fit <- eBayes(fit)
top.all <- topTable(e.fit,n=nrow(eset@exprs),coef=2)
```

Multiple testing

- We are testing many (thousands, often) of genes simultaneously – say m genes

	Fail to Reject Null	Reject Null	Total Count	# of Type I errors: V
Null True	U	V	m_0	
Null False	T	S	$m - m_0$	# of Type II errors: T
	$m - R$	R	m	# of correct “decisions”: $U + S$

Controlling Error Rates

- Family-wise error rate:
 - $\text{FWER} = P(V \geq k)$
 - Usually try to control $\text{FWER} \leq \alpha$
 - Maybe try Bonferroni correction, and reject null for P-values $\leq \alpha/m$; but this is too conservative

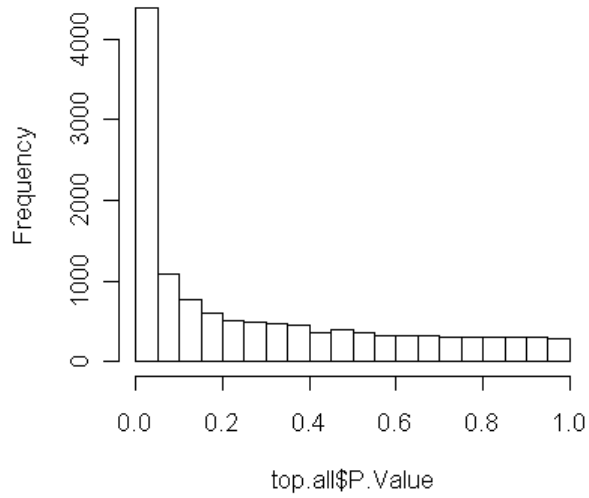
 - False Discovery Rate:
 - $\text{FDR} = E[V/R]$
 - Then control $\text{FDR} \leq \alpha$
 - This uses a “step-wise” Bonferroni adjustment
-

```
# what if there were no trt effect?
set.seed(123)
fake.trt <- sample(trt)
fake.design <- cbind(Intercept=1, fake.trt=fake.trt)
fit <- lmFit(eset@exprs, fake.design)
e.fit <- eBayes(fit)
fake.top.all <- topTable(e.fit, n=nrow(eset@exprs), coef=2)

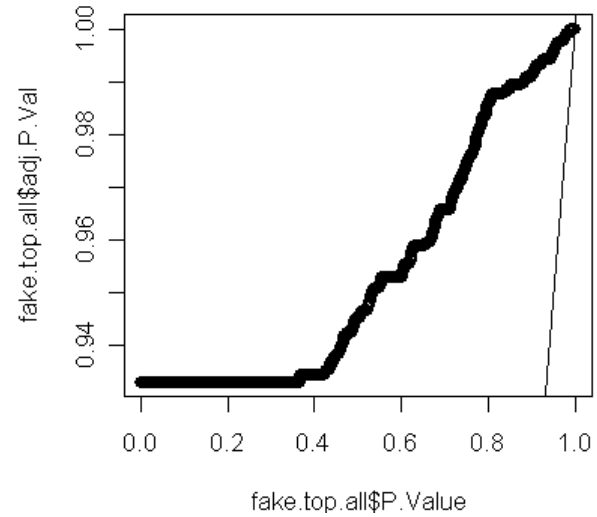
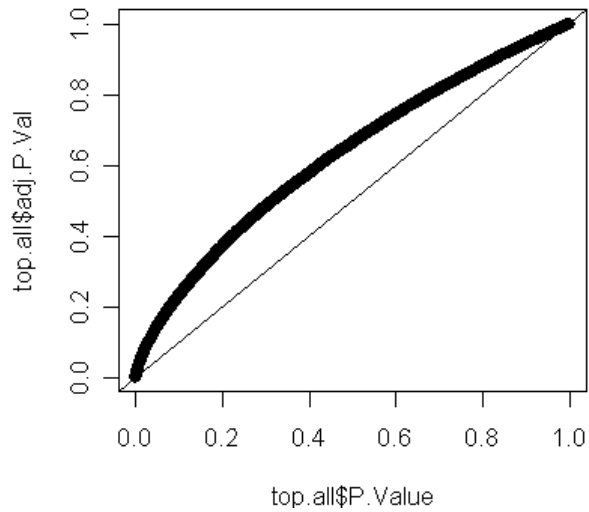
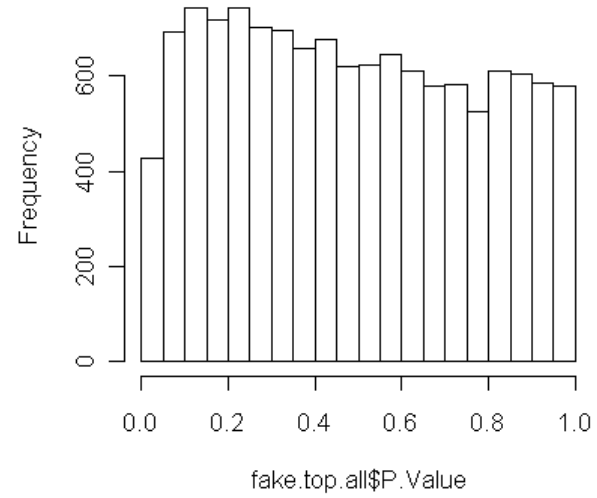
# Compare real & fake
par(mfrow=c(2,2))
hist(top.all$P.Value, main='Real Comparison')
hist(fake.top.all$P.Value, main='Fake Comparison')

# Now return to real comparison
sum(top.all$P.Value<0.05)
# 4385
sum(top.all$adj.P.Val<0.05)
# 3024
plot(top.all$P.Value, top.all$adj.P.Val); abline(0,1)
plot(fake.top.all$P.Value, fake.top.all$adj.P.Val); abline(0,1)
```

Real Comparison



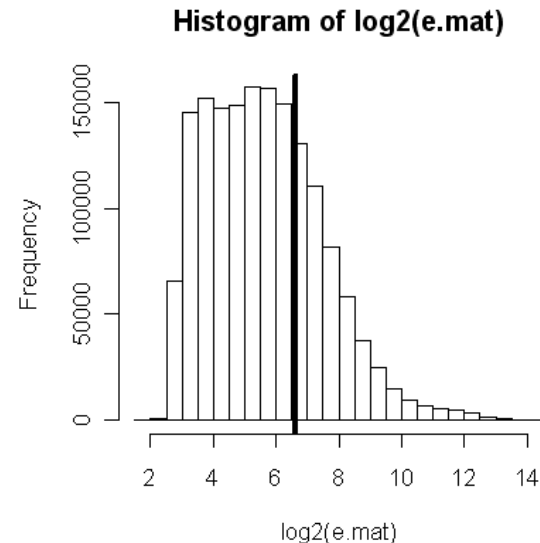
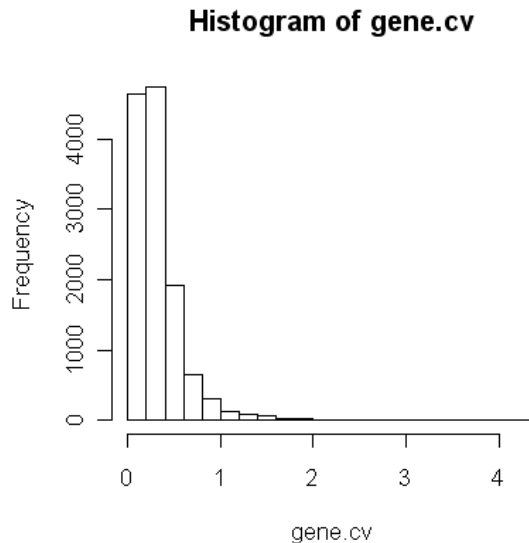
Fake Comparison



Do we need to test every gene?

- Maybe consider only those genes with the greatest relative variability in expression level
 - Or, look more at genes that have high expression values in a certain percentage of arrays
 - Or, look only at genes known to be involved in a certain cellular process (or biological process, etc.)
 - General idea: reduce multiple testing issues by restricting attention to a more “active” set of genes
 - this is called “filtering”
-

```
# Consider filtering:
e.mat <- 2^ALL@exprs # on un-logged scale
gene.mean <- apply(e.mat,1,mean)
gene.sd <- apply(e.mat,1,sd)
gene.cv <- gene.sd/gene.mean
par(mfrow=c(2,2))
hist(gene.cv)
hist(log2(e.mat)); abline(v=log2(100),lwd=3)
# filter: keep genes with cv between .7 and 10,
# and where 20% of samples had exprs. > 100
library(genefilter)
ffun <- filterfun(pOverA(0.20,100), cv(0.7,10))
small.eset <- log2(e.mat[genefilter(e.mat,ffun),])
# find the gene names
gn.keep <- rownames(small.eset)
length(gn.keep)
# 431
# Now reconsider DE test
fit <- lmFit(small.eset,design)
e.fit <- eBayes(fit)
top.small <- topTable(e.fit,n=nrow(small.eset),
  coef=2,adjust="BH")
gn.sig <- top.small$ID[top.small$adj.P.Val<0.05]
length(gn.sig)
```



So after filtering, test for DE using a method that

1. adjusts for small sample size in SD estimation, and
2. makes up for unknown [or sample-size-limited] sampling distribution before
- (3.) making some multiple testing adjustment.

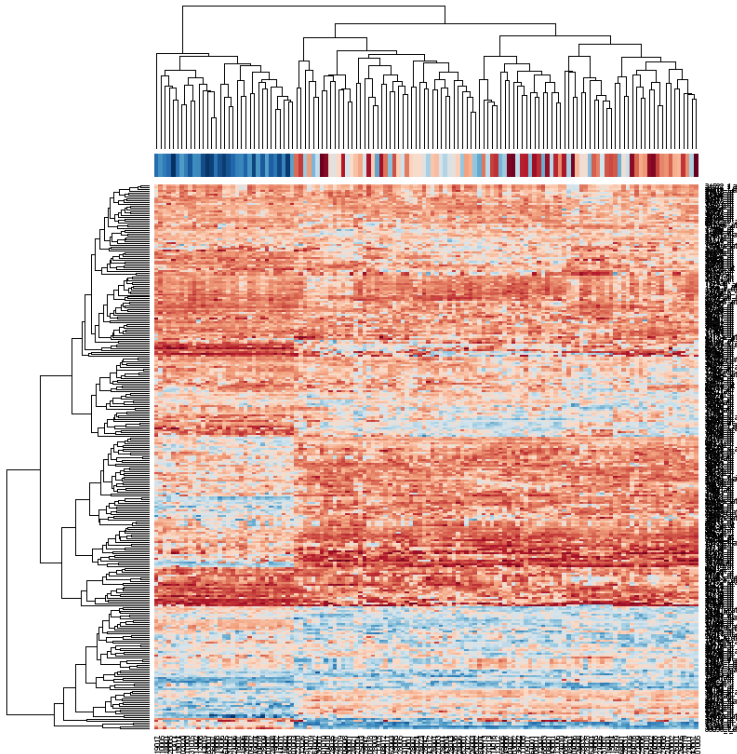
Problem 4: Effective Visualization

- Want to visually display significant results
- Need to be able to “tell a story”
- Focus on similarities and differences

- Color scheme matters

<http://www.vischeck.com/vischeck/vischeckImage.php>

```
# Look at how these significant genes may cluster
t.sig <- is.element(rownames(small.eset), gn.sig)
sig.eset <- small.eset[t.sig,]
library(cluster)
library(RColorBrewer)
hmccl <- colorRampPalette(brewer.pal(10, "RdBu"))(256)
csc <- hmccl[seq(from=1, to=256, length=128)]
heatmap(sig.eset, scale="column", col=hmccl, ColSideColors=csc)
```

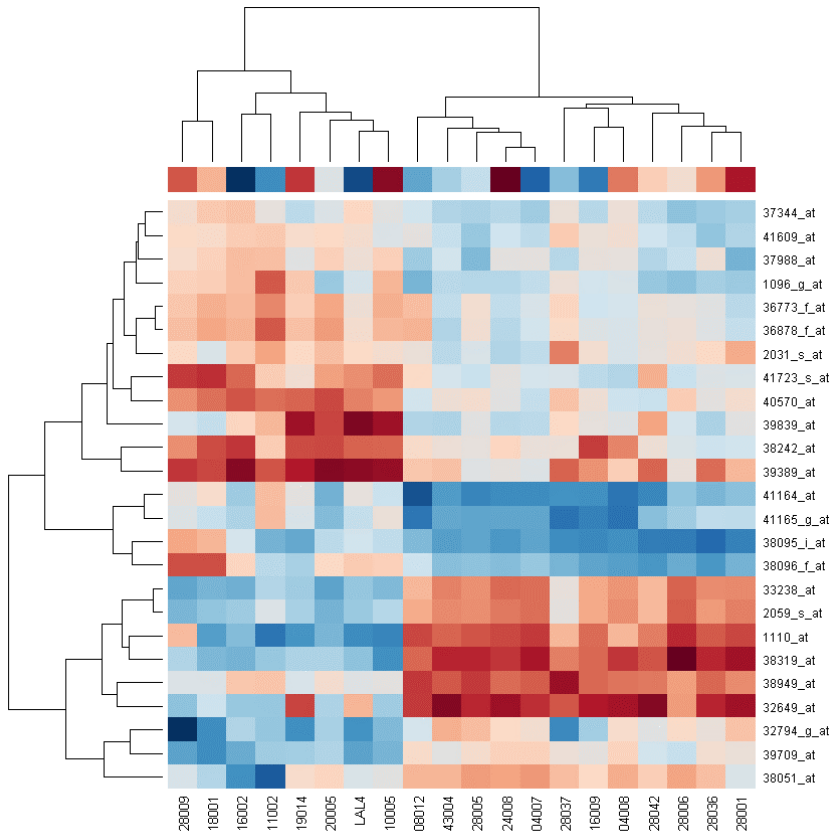


What in the world is this doing?

```

# Zoom in on top 25 genes and 10 randomly-selected arrays
gn.25 <- gn.sig[1:25]
set.seed(123)
sub.arrays <- sample(1:128,20)
sig.eset.25 <- small.eset[is.element(rownames(small.eset),gn.25),sub.arrays]
hmccl <- colorRampPalette(brewer.pal(10,"RdBu"))(256)
csc <- hmccl[seq(from=1,to=256,length=20)]
heatmap(sig.eset.25,scale="column",col=hmccl,ColSideColors=csc)

```

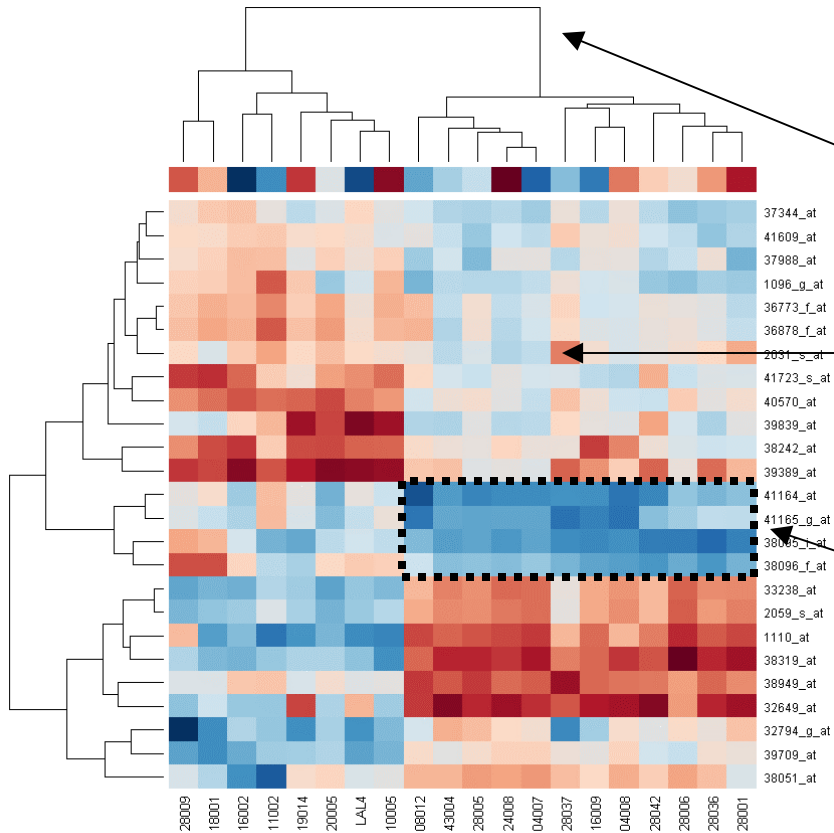


What “story” does this tell?

Divisive Analysis Clustering (diana)

- 1. All genes start out in same cluster
 - 2. Find “best” split to form two new clusters
 - “best” – maximize “distance” between new clusters
 - “distance” between new clusters: linkage - average, single (nearest neighbor), etc.
 - 3. Repeat step 2 until each gene is its own cluster
 - (Same with samples)
 - Visualize with a dendrogram (tree)
-

Interpreting “heat” maps



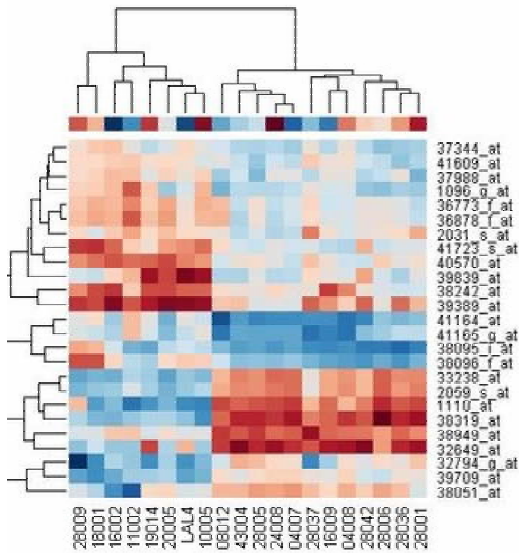
Dendrograms on columns and rows
Here, using divisive analysis clustering

Color represents relative expression level, but on what scale?
Here, scale is red (low) to blue (high)

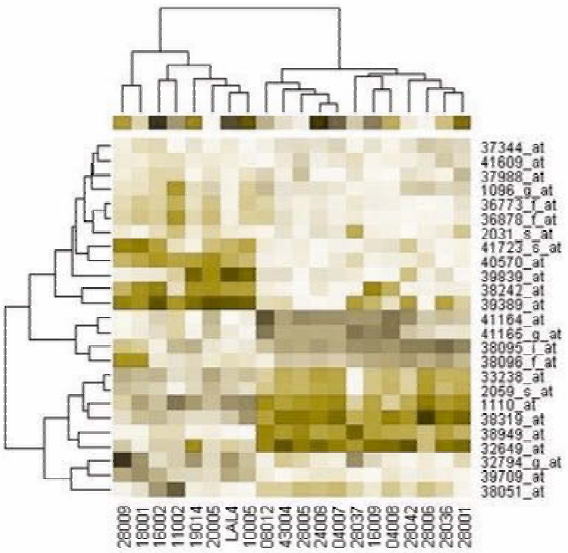
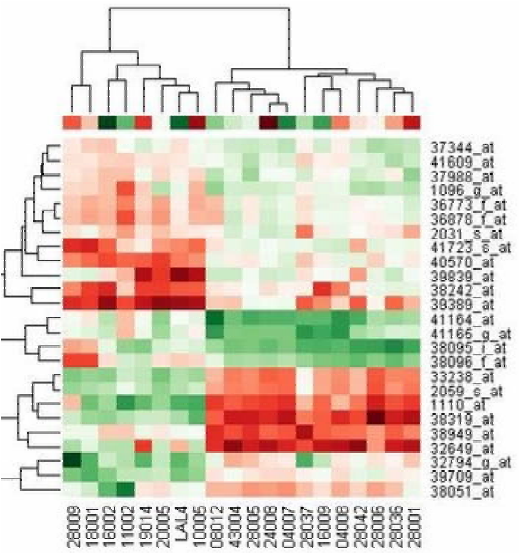
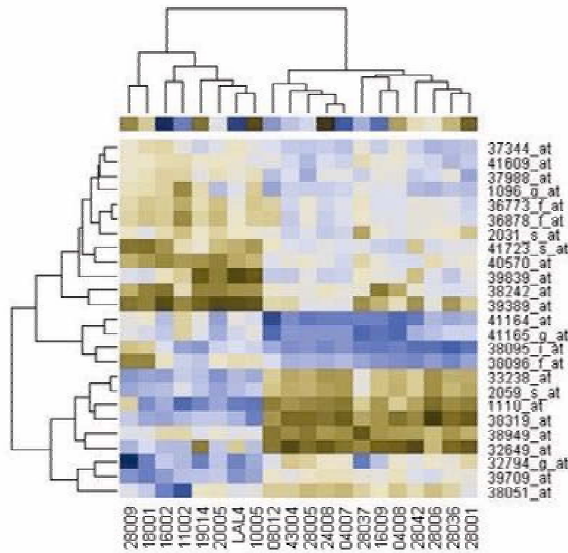
Look for: distinct blocks
- especially meaningful differences

Ask: what distance measure, and what linkage method?
Here, Euclidean distance with average linkage

Original Image



Deuteranope Simulation



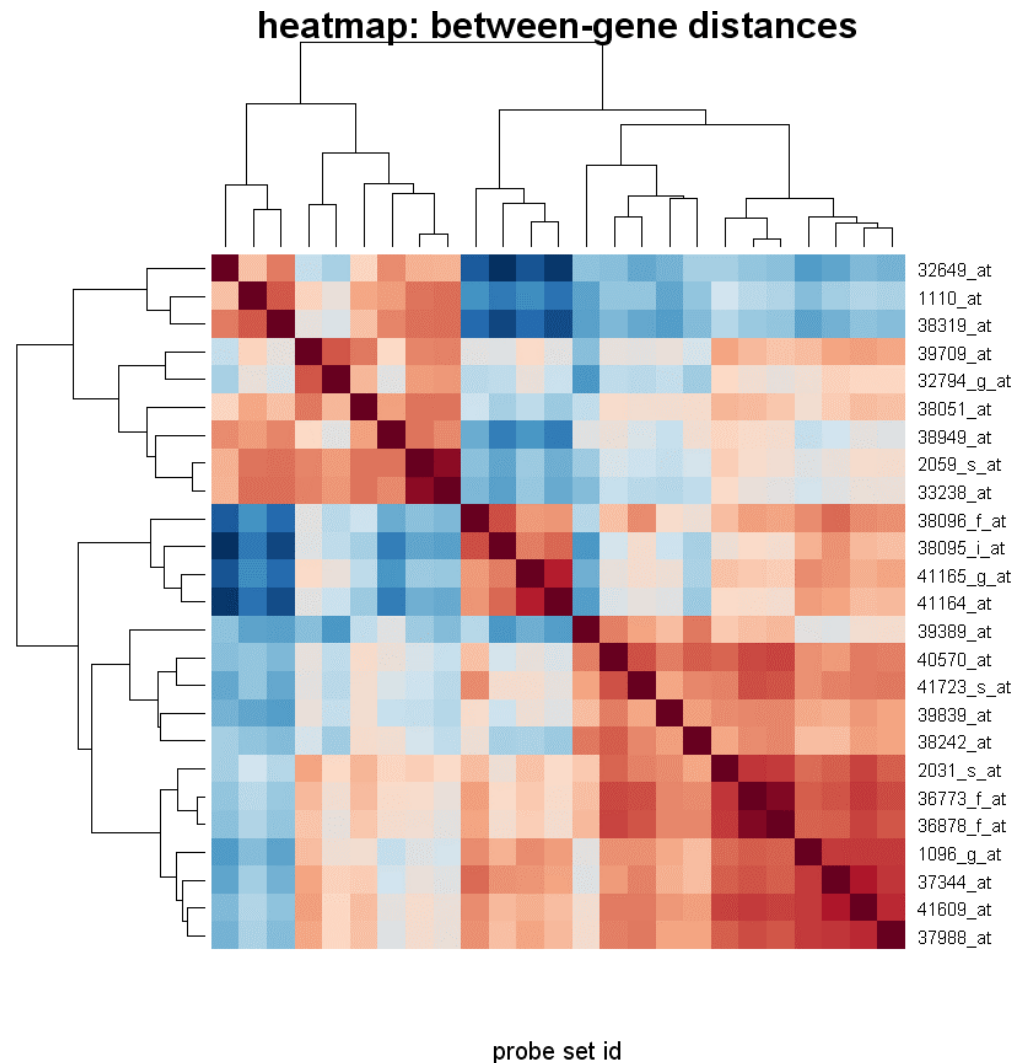
A poor color scheme (like red/green) could prevent a large percentage of the audience from gaining any information from the visualization.

“False Color” images

```
# False color image
library(bioDist)
d.gene.euc <-
  euc(sig.eset.25)
heatmap(
  as.matrix(d.gene.euc),
  sym=TRUE,col=hmcol,
  main='heatmap:
  between-gene distances',
  xlab='probe set id',
  labCol=NA)
```

Visualize pair-wise distances between genes' vectors of expression levels; here, Euclidean distance is used

Could also look at clustering samples instead of genes.

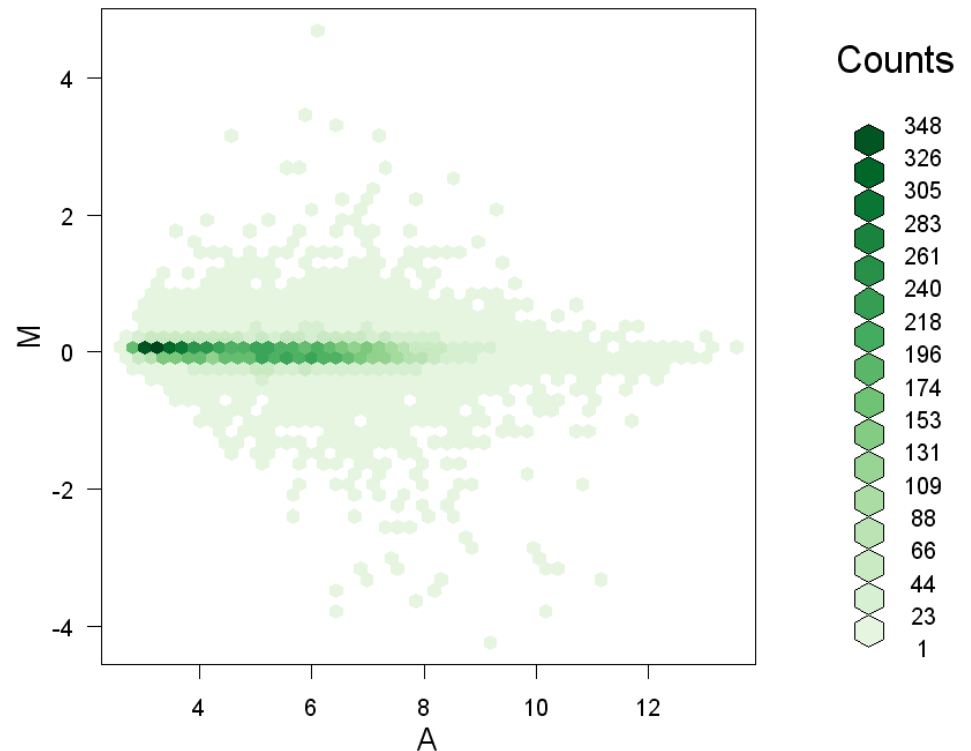


MA-plot

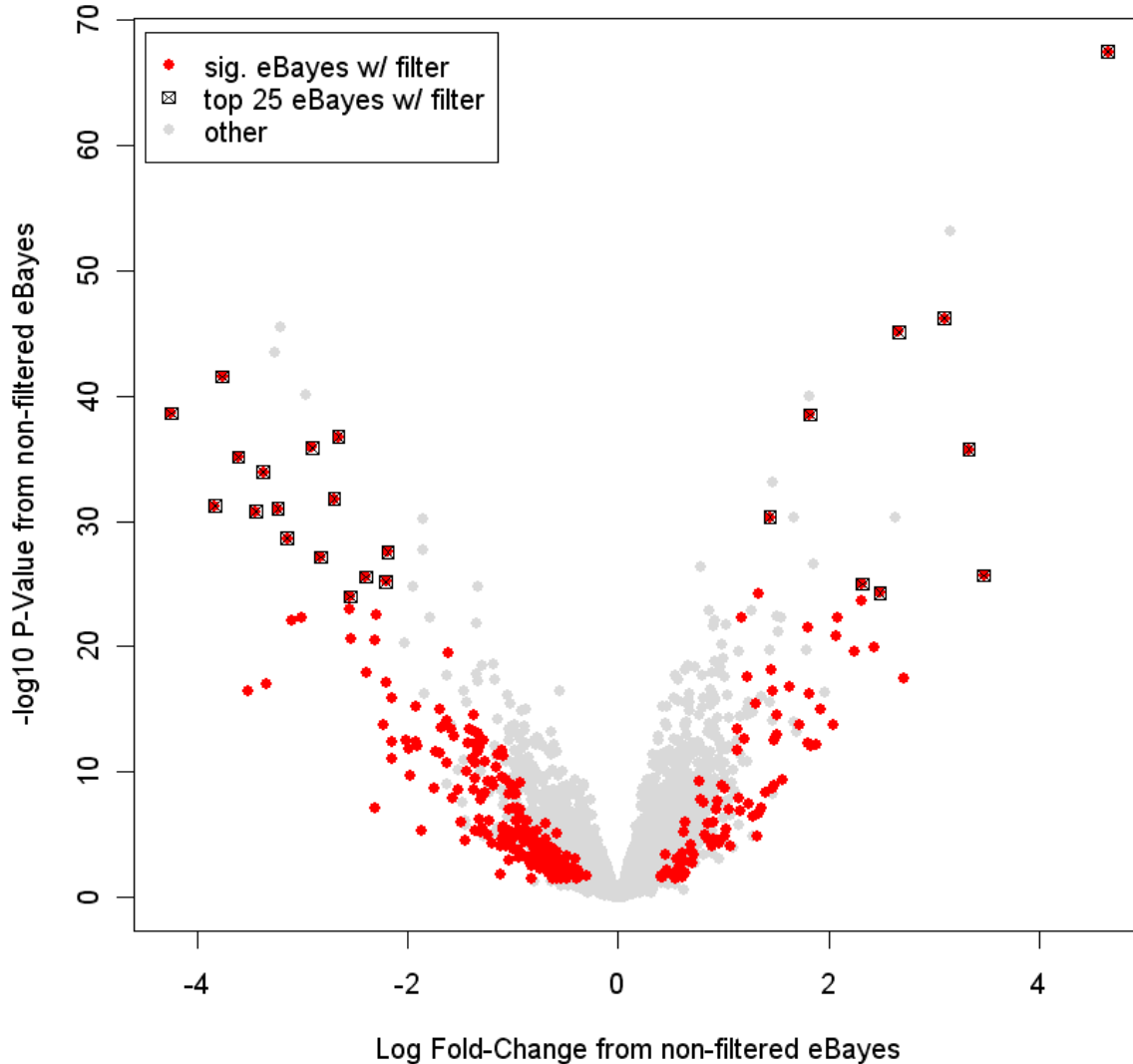
```
M <- top.all$M # M = log fold-change
A <- top.all$A # A = ave. expr. level
# More efficient scatterplots
library(RColorBrewer); library(hexbin)
green.ramp <- colorRampPalette(brewer.pal(9,"Greens")[-1])
mat <- cbind(A,M)
hb <- hexbin(mat,xbins=50)
plot(hb,colramp=green.ramp)
```

Recall:

1. This should be centered at $M=0$
2. There should be no trend to the M vs. A relationship



Volcano Plot



What is the story here?

1. Large Fold-Change is neither necessary nor sufficient for significance (small P-Value)
2. Different methods will “find” different sets of significant genes
3. Filtering has an effect and may “cost” important genes in the final list of significant genes

Volcano Plot

```
# Volcano plot to highlight significant genes after filtering
plot(top.all$M, -log10(top.all$P.Value),
     xlab='Log Fold-Change from non-filtered eBayes',
     ylab='-log10 P-Value from non-filtered eBayes',
     col='grey85', pch=16, main='Volcano Plot')
t <- is.element(top.all$ID, gn.sig)
points(top.all$M[t], -log10(top.all$P.Value[t]), pch=16, col='red')
t <- is.element(top.all$ID, gn.25)
points(top.all$M[t], -log10(top.all$P.Value[t]), pch=7, col='black')
legend(-4.5, 69,
     c('sig. eBayes w/ filter', 'top 25 eBayes w/ filter', 'other'),
     pch=c(16, 7, 16), col=c('red', 'black', 'grey85'), cex=1)
```

Summary

- Testing for Differential Expression
 - Naïve (fold-change cutoff) approach is a poor choice
 - Sample size affects test statistic
 - Estimate of SD
 - Sampling distribution
 - Multiple comparison adjustment (and filtering)
 - Effective Visualization
 - “Heat” maps & clustering
 - MA-plot & Volcano plots
 - Use tools to “tell a story”
 - Need uncluttered final product
 - Need useful color scheme
-