Introduction to Filtering with Gene Expression Data

Utah State University – Spring 2014

STAT 5570: Statistical Bioinformatics

Notes 3.2

References

- Chapters 7 & 14 of Bioconductor Monograph (course text)
- Hackstadt and Hess (2009). Filtering for Increased Power for Microarray Data Analysis. BMC Bioinformatics 10:11 http://www.biomedcentral.com/1471-2105/10/11
- Tuglus and van der Laan (2009). Modified FDR Controlling Procedure for Multi-Stage Analyses. SAGMB 8(1):12 http://www.bepress.com/cgi/viewcontent.cgi?article=1397&context=sagmb
- www.geneontology.org

Recall multiple testing issues

- Look for differentially expressed (DE) genes
  - for each gene individually, test null: no change
  - obtain p-value and make decision
  - thousands of genes simultaneously
    → error rate could be misleading

- How to adjust?
  - adjust p-values: Bonferroni, FDR, etc.
  - consider q-values
  - also (here) – restrict attention to fewer genes

Motivation for gene "filtering"

- Relatively few genes should be:
  expressed at any time
- Relatively few genes should be:
  differentially expressed between conditions
- Restrict attention to those genes that are:
  relevant "candidates"

Types of gene filtering

- Specific:

  look only at genes “known” to satisfy some biological constraint (like all genes with a certain function)

  - we will return to this idea in Unit 4

- Non-Specific:

  look only at genes with certain [interesting] properties in “expression values”

Non-specific gene filtering: Variability across all samples

- Genes with large IQR or SD

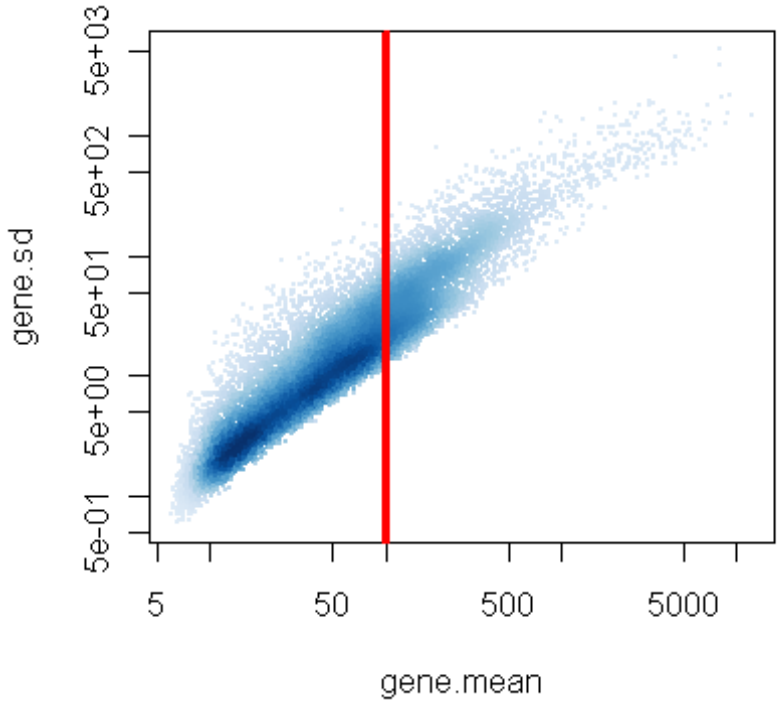
  because: look at genes that actually change expr. levels
  (biological relevance?)

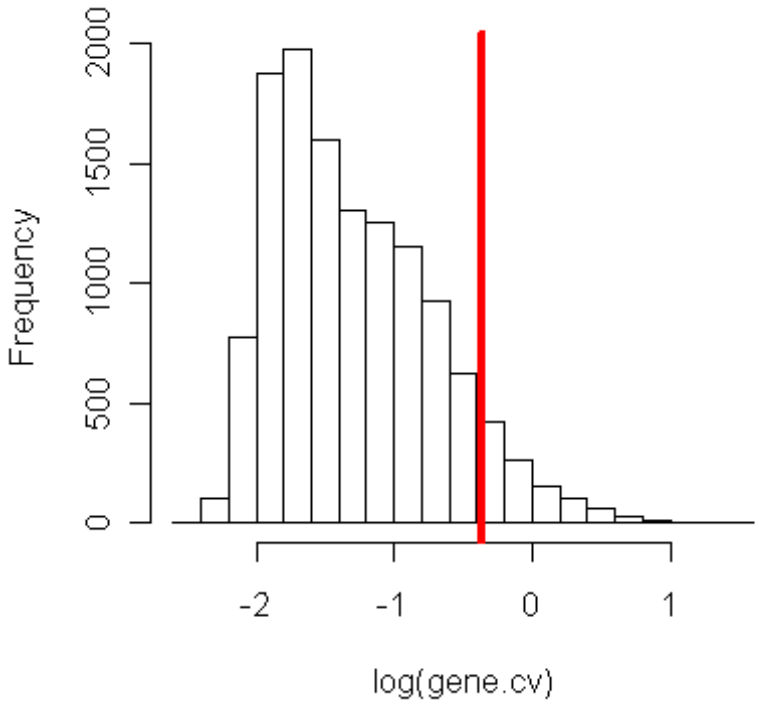
- Compare mean to SD

  because: maybe “tolerate” higher SD for higher expr. levels

- Consider genes with large CV = SD / |mean| = coeff. of var.
  - to balance expr. levels with var. of expr. levels
  - but this can be large just due to - small mean:
    (if gene is “absent” in many samples)
    → look at intensity-based filtering

Example: Look at mean and CV





```
# obtain expression estimates on the UN-LOGGED scale
library(affy); library(ALL); data(ALL)
e.mat <- 2^exprs(ALL)

# look at mean, sd, & cv for each gene across arrays
gene.mean <- apply(e.mat,1,mean)
gene.sd <- apply(e.mat,1,sd)
gene.cv <- gene.sd/gene.mean

# make plots
library(geneplotter); library(RColorBrewer)
blues.ramp <- colorRampPalette(brewer.pal(9,"Blues")[-1])
dCol <- densCols(log(gene.mean),log(gene.sd),
  colramp=blues.ramp)
par(mfrow=c(2,2))
plot(gene.mean,gene.sd,log='xy',col=dCol,pch=16,cex=0.1)
abline(v=100,lwd=3,col='red')
hist(log(gene.cv),main=NA)
abline(v=log(.7),lwd=3,col='red')
```

Non-specific filtering: Intensity-based

- Expression above some threshold in a certain:
  % of cases (pOverA)

- Expression above some threshold in a certain:
  # of cases (kOverA)

- dChip defaults often used (but should rely on data):
  - intensity above 100 in at least: 20% of samples

  and
  - CV between: 0.7 and 10

Non-specific filtering and scale

- Non-specific filtering is intensity-based
- Most pre-processing methods return expression estimates on the log-scale
- Before applying non-specific filtering, expression level estimates need to be –
  "un-logged"

Example: Filter based on CV and pOverA

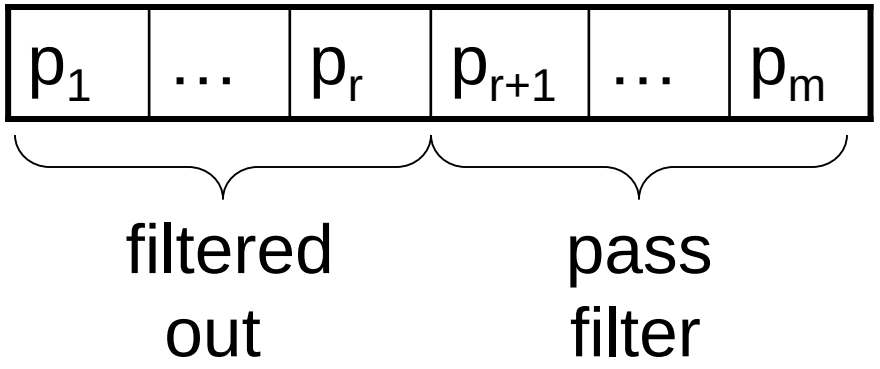
```
# filter: keep genes with cv between .7 and 10,
#         and where 20% of samples had exprs. > 100
library(genefilter)
ffun <- filterfun(pOverA(0.20,100), cv(0.7,10))
t.fil <- genefilter(e.mat,ffun)
# apply filter, and put expression back on log scale
small.eset <- log2(e.mat[t.fil,])

dim(e.mat)
# 12625 128
dim(small.eset)
# 431 128

# find the gene names
gn.keep <- rownames(small.eset)
```

Then test for DE using only the genes in this small.eset object. (Here, 431 genes instead of all 12,625.)

NOTE: This filter is very severe and only used for in-class simplicity.

A multi-stage analysis (MSA)

- Stage 1: Apply filter; partition set of raw P-values

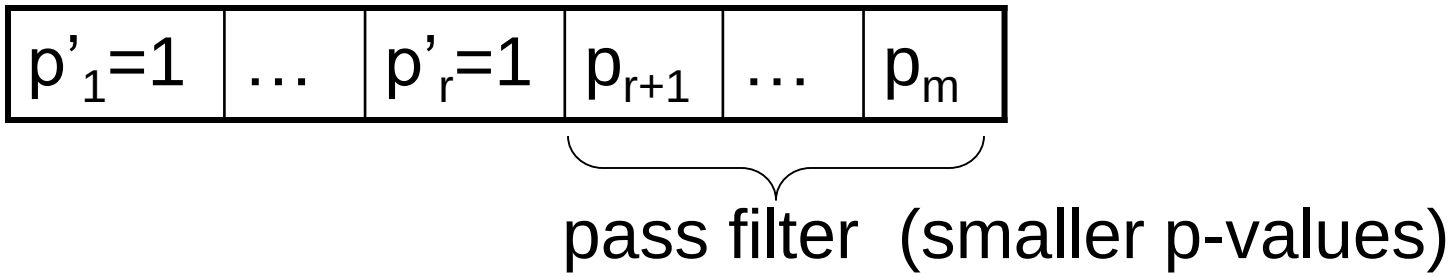


- Commonly-used Stage 2: Apply MCP (like FDR or q-value) adjustment to p-values of tests that pass filter

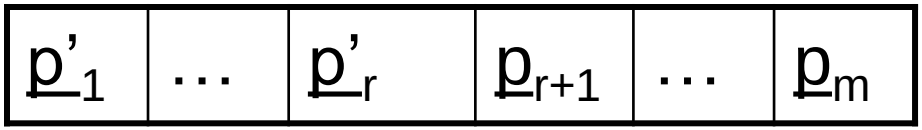
NA	…	NA	p_{r+1}	…	p_m

What if filter is based on rank of p-values? Control error rate in multi-stage: FDR-MSA

- Stage 2a: Reset filtered-out p-values to 1



- Stage 2b: Apply FDR adjustment to [reset] p-values of all tests



- When the filter preserves the rank of the p-values, this will appropriately control Type I and II rates

When will filter preserve rank of p-values?

- Example: only consider genes whose [raw] p-value is < .10
- Example: only consider genes with smallest 40% of p-values
- Example: apply randomForest (coming up in Notes 3.4) and only consider genes with non-zero ‘variable importance’

Notes / concerns regarding filtering

- Can be a bit subjective

  - Which threshold and why?

- Can erroneously eliminate important genes

  not all genes well-studied
  (recall specific filtering and prior knowledge)

- Can ignore: experimental design

  Maybe more important than high CV:

  - var. low within groups, but high between

  - but – this could “double dip” test statistic

- Can help reduce: multiple testing issues

Summary

- Filtering helps:
  - reduce the problems with multiple testing
    - Restrict attention to “active” genes
      → less of a correction is necessary
    - Expect a higher “concentration” of DE genes
  - Increase statistical power
- Types of filtering:
  - Specific
    based on “known” biology – return to this later
  - Non-specific
    based on properties of expression estimates
- Avoid filtering on anything related to test statistic
  - Double-dipping changes meaning of α
  - Otherwise, need to control error rate in multi-stage analysis