Integrating BioConductor Packages in the Analysis of Affymetrix Data

James W. MacDonald

UMCCC Affymetrix and cDNA

Microarray Core Facility

Overview

- Analysis of Affymetrix GeneChip® Data
- Using/writing ‘wrapper’ functions
  - Emphasis on *using*
  - *affycoretools* package
- Literate Programming/Reproducible Research

Analysis of Affy data

- CEL files → Finished output
  - CEL files contain raw Affymetrix data
  - Finished output
    - Some sort of data presentation (HTML/text tables)
    - Description of analysis

Wrapper functions

- Write functions that ‘wrap’ existing functions to perform common tasks.
  - Analyses use multiple packages
    - *affy, limma, annaffy, GOstats, biomaRt, annotate*, etc.
    - Data structures may be similar, but packages are not explicitly designed to work together.
  - Relatively similar analyses result in lots of replicated R code.

Wrapper functions

EXAMPLES*:*

Create a density plot with a legend

```
> hist(dat, lty=c(rep(1,8), rep(2,7)), lwd=2, col=1:length(filenames))
> legend(12, 0.25, legend=filenames, lty=c(rep(1,8), rep(2,7)), lwd=2,
    col=1:length(filenames))
```

-or-

```
> plotHist(dat)
```

Create a 'degradation' plot with a legend

```
> plotAffyRNAdeg(AffyRNAdeg(dat), col=1:length(filenames))
> legend(0,50, legend=filenames, lty=1, col=1:length(filenames), cex=0.7)
```

-or-

```
> plotDeg(dat)
```

Wrapper functions

Basic idea: If you think you might do the same thing more than say, five times, write a wrapper function.

Literate programming

- Donald Knuth
  - Program should be combination of programming language and documentation language
- In R
  - .Rnw file
  - Sweave() – *utils* package (part of base R)

An extended example

- Getting started
- Model data/make comparisons
- Create output/documentation

Getting started

- Read data into R
- Check quality of raw data
- Compute expression values
- Check quality of expression values

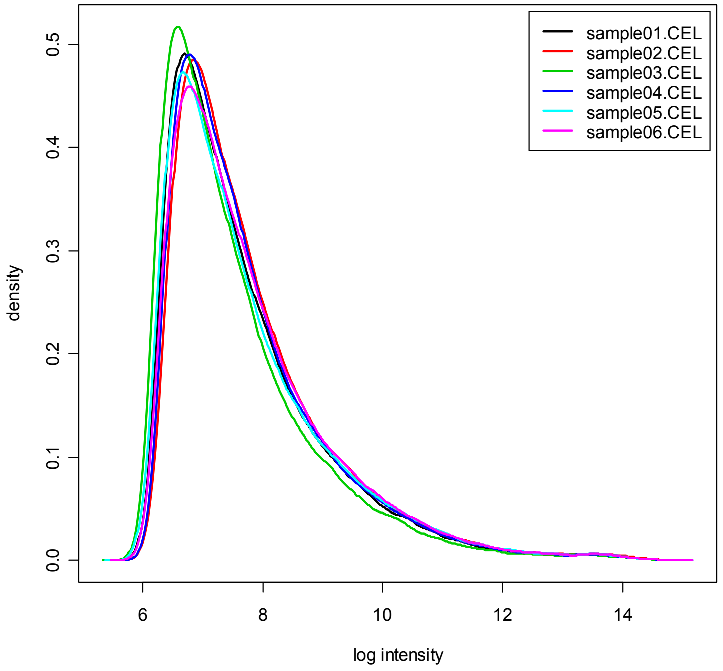
Read data into R

- ReadAffy() – *affy* package

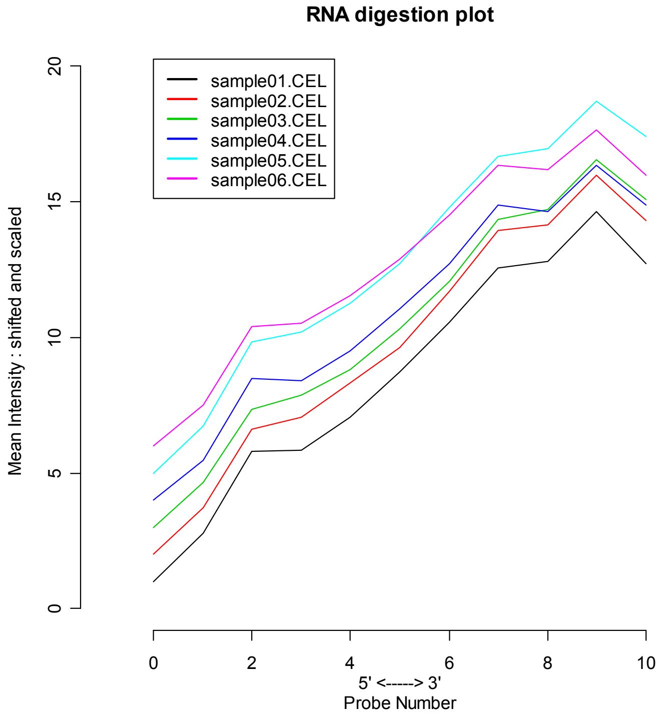
- Read in Cel files
  - R_HOME/library/affycoretools/examples
- Twelve samples, three replicates, four sample types (A, B, C, D)

Check quality of raw data

plotHist(dat[,1:6])



plotDeg(dat[,1:6])



Compute expression values

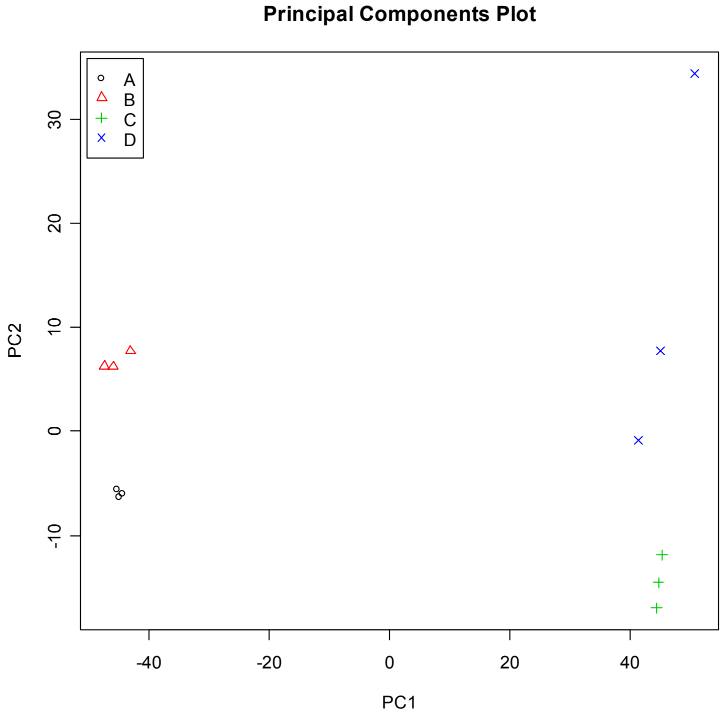
- Various methods
  - – rma() – *affy* package
  - – gcrma() – *gcrma* package
  - – mas5() – *affy* package
  - – affystart() – *affycoretools* package

Check quality of expression values

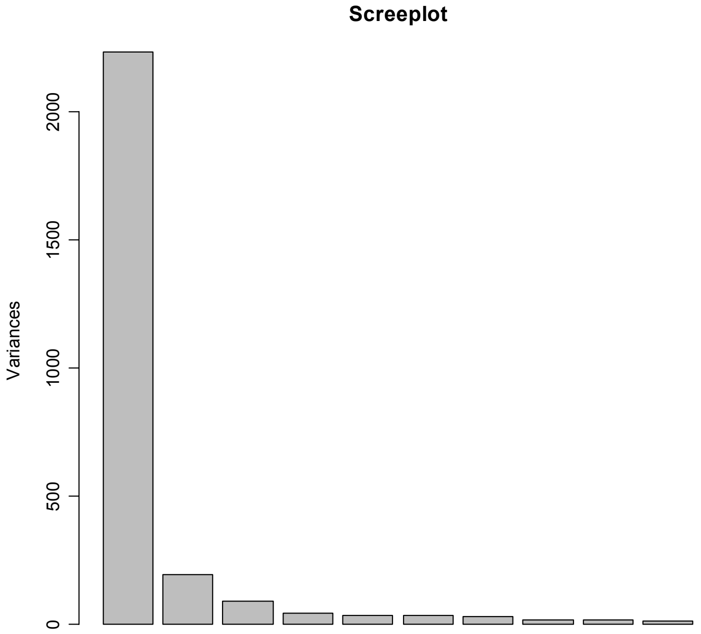
- plotPCA() – *affycoretools* package
- image() – *affyPLM* package
  - rmaPLM() is *affyPLM* equivalent of rma()

plotPCA()

plotPCA(eset, groups = rep(1:4, each = 3), groupnames = LETTERS[1:4])

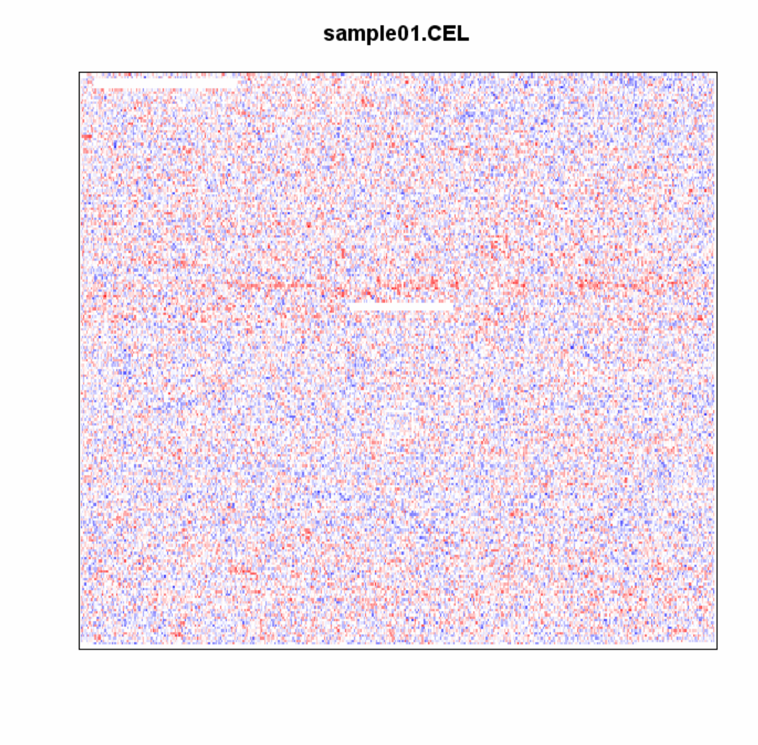


plotPCA(eset, screeplot = TRUE)

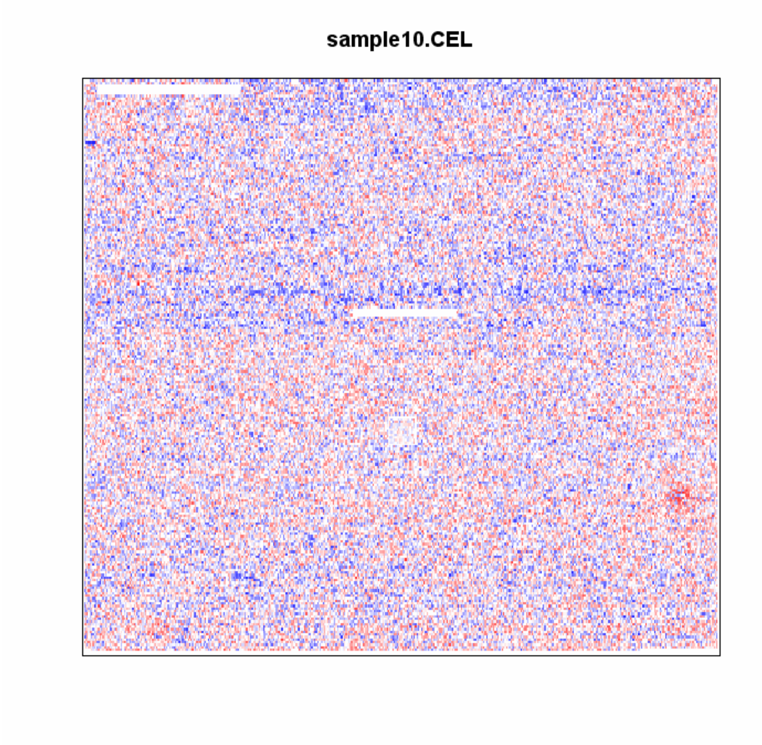


image()

image(pset, type = "resid", which = 1)



image(pset, type = "resid", which = 10)



Practice

- Compute expression values
- Try plotting a PCA plot
- *affyPLM*/residual plots
- affystart() – *affycoretools* package

Model data/make comparisons

- *limma* package
  - Why *limma*?
- Three step process
  - Design matrix
  - Contrasts matrix
  - Empirical Bayes adjustment

Design matrix

- Matrix of (usually) 0, 1 used to specify model
- Usually easiest to use model.matrix()
- Two models
  - – Factor effects
  - – Cell means

Factor effects model

$$y_{ij} = \mu + \tau_i x_i + \varepsilon_{ij}$$

$i = 1, 2, 3, 4$ (Samples)
$j = 1, 2, 3$ (Replicates)

In this parameterization:

μ represents a baseline level (Sample A)

τ represents the *difference* between the baseline and a given sample type

$\boldsymbol{\varepsilon}$ represents the *error*

$$
\begin{pmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ y_{33} \\ y_{41} \\ y_{42} \\ y_{43} \end{pmatrix}
=
\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}
\text{x}
\begin{pmatrix} \mu \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{pmatrix}
+
\begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{12} \\ \varepsilon_{13} \\ \varepsilon_{21} \\ \varepsilon_{22} \\ \varepsilon_{23} \\ \varepsilon_{31} \\ \varepsilon_{32} \\ \varepsilon_{33} \\ \varepsilon_{41} \\ \varepsilon_{42} \\ \varepsilon_{43} \end{pmatrix}
$$

Factor effects model

$$y_{11} = \mu \cdot 1 + \tau_1 \cdot 0 + \tau_2 \cdot 0 + \tau_3 \cdot 0 + \varepsilon_{11}$$

$$\Downarrow$$

$$y_{11} = \mu + \varepsilon_{11} \longleftarrow \text{No interesting parameters}$$

$$y_{21} = \mu \cdot 1 + \tau_1 \cdot 1 + \tau_2 \cdot 0 + \tau_3 \cdot 0 + \varepsilon_{21}$$

$$\Downarrow$$

$$y_{21} = \mu + \tau_1 + \varepsilon_{21} \longleftarrow \text{Here } \tau_1 = \text{B - A}$$

Factor effects design matrix

```
> design <- model.matrix(~ factor(rep(1:4, each = 3)))
> colnames(design) <- c("Intercept",“DifBA",“DifCA",“DifDA")
> design

   Intercept   DifBA   DifCA   DifDA
1      1         0       0       0
2      1         0       0       0
3      1         0       0       0
4      1         1       0       0
5      1         1       0       0
6      1         1       0       0
7      1         0       1       0
8      1         0       1       0
9      1         0       1       0
10     1         0       0       1
11     1         0       0       1
12     1         0       0       1
```

Practice

- Make a factor effects design matrix for our data

Cell means model

$$y_{ij} = \mu_i x_i + \varepsilon_{ij}$$

$i = 1, 2, 3, 4$ (Samples)
$j = 1, 2, 3$ (Replicates)

In this parameterization:

- μ represents the sample mean (hence cell means model)
- $\boldsymbol{\varepsilon}$ represents the *error*

$$
\begin{pmatrix} y_{11} \\ y_{12} \\ y_{13} \\ y_{21} \\ y_{22} \\ y_{23} \\ y_{31} \\ y_{32} \\ y_{33} \\ y_{41} \\ y_{42} \\ y_{43} \end{pmatrix}
=
\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}
\text{ x }
\begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{pmatrix}
+
\begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{12} \\ \varepsilon_{13} \\ \varepsilon_{21} \\ \varepsilon_{22} \\ \varepsilon_{23} \\ \varepsilon_{31} \\ \varepsilon_{32} \\ \varepsilon_{33} \\ \varepsilon_{41} \\ \varepsilon_{42} \\ \varepsilon_{43} \end{pmatrix}
$$

Cell means model

$$y_{11} = \mu_1 \cdot 1 + \mu_2 \cdot 0 + \mu_3 \cdot 0 + \mu_4 \cdot 0 + \varepsilon_{11}$$

$$\Downarrow$$

$$y_{11} = \mu_1 + \varepsilon_{11} \longleftarrow$$

Here μ_1 estimates the mean expression for A samples.

$$y_{21} = \mu_1 \cdot 0 + \mu_2 \cdot 1 + \mu_3 \cdot 0 + \mu_4 \cdot 0 + \varepsilon_{21}$$

$$\Downarrow$$

$$y_{21} = \mu_2 + \varepsilon_{21} \longleftarrow$$

Here μ_2 estimates the mean expression for B samples.

Cell means design matrix

```
> design <- model.matrix(~ 0 +  factor(rep(1:4, each = 3)))
> colnames(design) <- LETTERS[1:4]
> design

        A       B       C       D
1       1       0       0       0
2       1       0       0       0
3       1       0       0       0
4       0       1       0       0
5       0       1       0       0
6       0       1       0       0
7       0       0       1       0
8       0       0       1       0
9       0       0       1       0
10      0       0       0       1
11      0       0       0       1
12      0       0       0       1
```

Practice

- Make a cell means design matrix for our data

Which model is ‘better’?

- Factor effects
  - If properly constructed, all comparisons are implicit
  - More complicated analysis can be confusing
- Cell means
  - Extra steps required to make comparisons
  - Less confusing for complicated model
  - Most *affycoretools* functions expect a cell means model

Contrasts matrix

- A contrast is a comparison between parameter estimates
- *limma* requires a matrix that specifies the requested comparisons (contrasts matrix)

What is a contrasts matrix?

- Matrix of (usually) 0, 1, -1 used to make comparisons
  - Can use decimal values to compare means of groups
- Best visualized with example

Parameter Estimates

$$\left(\begin{array}{cccc} \textbf{A} & \textbf{B} & \textbf{C} & \textbf{D} \\ 7.11 & 10.94 & 3.16 & 12.93 \\ 7.19 & 15.05 & 16.71 & 4.55 \\ 3.4 & 16.71 & 13.2 & 13.09 \\ 11.21 & 2.97 & 7.33 & 10.45 \\ 9.72 & 13.05 & 15.41 & 3.42 \\ 5.38 & 9.55 & 3.43 & 10.62 \\ 3.36 & 10.73 & 15.49 & 10.67 \\ 13.51 & 9.15 & 3.01 & 5.37 \\ 5.71 & 9.16 & 5.28 & 8.08 \\ 6.26 & 1.94 & 2.27 & 9.1 \\ 1.96 & 6.69 & 4.11 & 4.46 \\ 4.49 & 1.6 & 6.63 & 6.45 \\ 10.17 & 5 & 16.43 & 14.19 \\ 12.81 & 14.77 & 13.77 & 12.18 \\ 8.32 & 14.45 & 11.97 & 7.55 \\ 5.07 & 13.2 & 3.77 & 7.19 \end{array}\right) \times \overset{\text{Contrasts Matrix}}{\begin{pmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{pmatrix}} = \begin{pmatrix} -3.83 & -9.77 \\ -7.86 & 12.16 \\ -13.31 & 0.11 \\ 8.24 & -3.12 \\ -3.33 & 11.99 \\ -4.17 & -7.19 \\ -7.37 & 4.82 \\ 4.36 & -2.36 \\ -3.45 & -2.8 \\ 4.32 & -6.83 \\ -4.73 & -0.35 \\ 2.89 & 0.18 \\ 5.17 & 2.24 \\ -1.96 & 1.59 \\ -6.13 & 4.42 \\ -8.13 & -3.42 \end{pmatrix}$$

Simplification

Parameter estimates:

$$A \times 1 \quad B \times -1 \quad C \times 0 \quad D \times 0 \longrightarrow A - B \longrightarrow \begin{pmatrix} 1 \\ -1 \\ 0 \\ 0 \end{pmatrix}$$

More complex contrasts

Factor effects model
we want C – D

Parameters are
Int, B-A, C-A, D-A

$$\begin{pmatrix}0\\0\\1\\-1\end{pmatrix}$$

C-A - D-A = C - D

Cell means model
we want (A + B)/2 - C

Parameters are
A B C D

$$\begin{pmatrix}0.5\\0.5\\-1\\0\end{pmatrix}$$

0.5A + 0.5B - C

makeContrasts()

Design matrix

	A	**B**	**C**	**D**
1	1	0	0	0
2	1	0	0	0
3	1	0	0	0
4	0	1	0	0
5	0	1	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	1	0
9	0	0	1	0
10	0	0	0	1
11	0	0	0	1
12	0	0	0	1

```
> makeContrasts(A – B, C - D, levels = design)

      A – B  C - D
 A      1       0
 B     -1       0
 C      0       1
 D      0      -1
```

matrix()

```
> makeContrasts('Standard/sensitive – Standard/insensitive', levels=design)
Error in eval(expr, envir, enclos) : object "Standard" not found
```

Doesn’t work!

```
> matrix(c(1,-1,0,0,0,0,1,-1), nc = 2, dimnames = list(colnames(design),
                         paste(colnames(design)[c(1,3)], colnames(design)[c(2,4)], sep = " vs ")))

                Standard/sensitive vs Standard/insensitive     Amplified/sensitive vs Amplified/insensitive
Standard/sensitive                                       1                                            0
Standard/insensitive                                    -1                                            0
Amplified/sensitive                                      0                                            1
Amplified/insensitive                                    0                                           -1
```

Practice

- Make a contrasts matrix for our data, assuming a cell means model.
  - Use makeContrasts()
  - Try using matrix(), using more descriptive column names

More complex example

- Four sample types, in duplicate
  - Wild type (WT) untreated
  - WT treated
  - Knock out (KO) untreated
  - KO treated
- Two questions
  - Does the treatment differ between WT and KO?
    - This is known as an *interaction*
  - What are the coefficients in our model?

Design matrix

```
> trt <- factor(rep(c("Treated","Untreated"), 4))
> typ <- factor(rep(c("WT","KO"), each=4), levels = c("WT","KO"))
> model.matrix(~trt*typ)

  (Intercept) trtUntreated typKO trtUntreated:typKO
1           1            0     0                  0
2           1            1     0                  0
3           1            0     0                  0
4           1            1     0                  0
5           1            0     1                  0
6           1            1     1                  1
7           1            0     1                  0
8           1            1     1                  1
```

What are the coefficients?

	(Intercept)	trtUntreated	typKO	trtUntreated:typKO
Trt.WT	1	0	0	0
Untr.WT	1	1	0	0
Trt.WT	1	0	0	0
Untr.WT	1	1	0	0
Trt.KO	1	0	1	0
Untr.KO	1	1	1	1
Trt.KO	1	0	1	0
Untr.KO	1	1	1	1

Baseline = Trt.WT

trtUntreated = Untr.WT – Trt.WT

typKO = Trt.KO – Trt.WT

trtUntreated:typKO = (Untr.KO – Trt.KO) – (Untr.WT – Trt.WT)

Coefficients (another way)

Note how coefficients are calculated:

$$(X'X)^{-1}X'Y$$

We can do this in R!

Compute using R

```
> b <- model.matrix(~trt*typ)
> d <- unique(b)
> solve(t(d) %*% d) %*% t(d)
```

$$\equiv (X'X)^{-1}X'$$

```
                        1    2    5    6
(Intercept)             1    0    0    0
trtUntreated           -1    1    0    0
typKO                  -1    0    1    0
trtUntreated:typKO      1   -1   -1    1
```

Where:
1 = Trt.WT
2 = Untr.WT
5 = Trt.KO
6 = Untr.KO

Practice

- Create the preceding model matrix and determine what the coefficients are
- Create a cell means model for the same data, and determine coefficients.

Empirical Bayes Adjustment

- Why do we need this?

$$statistic = \frac{\text{difference of means}}{\text{some measure of intra - group variability}}$$

- Mean is efficient
- Variance is not
  - Borrow strength

Model data/make comparisons in R

```
> design <- model.matrix(~ 0 + factor(rep(1:4, each = 3)))
> colnames(design) <- LETTERS[1:4]
> contrast <- makeContrasts(A - B, C - D, levels = design)
> fit <- lmFit(eset, design)
> fit2 <- contrasts.fit(fit, contrast)
> fit2 <- eBayes(fit2)
> topTable(fit2)
            ID        M     A       t    P.Value   adj.P.Val     B
2356  204582_s_at  3.47 10.15   39.05  1.96e-14   1.72e-10  19.86
6051  211548_s_at -2.33  7.18  -22.73  1.53e-11   6.76e-08  15.89
6756  216598_s_at  1.94  7.69   21.74  2.66e-11   7.80e-08  15.48
5961  211110_s_at  3.16  7.91   21.19  3.62e-11   7.96e-08  15.25
3299  206001_at   -1.59 12.40  -18.65  1.71e-10   3.01e-07  14.02
```

Practice

- Fit a cell means model to our data (you should already have a design matrix and contrasts matrix), and look at the top genes for each coefficient.

Create output/documentation

- Output
  - – HTML tables
  - – text tables
  - – graphics
- Documentation
  - – Written record of the analysis
  - – graphics

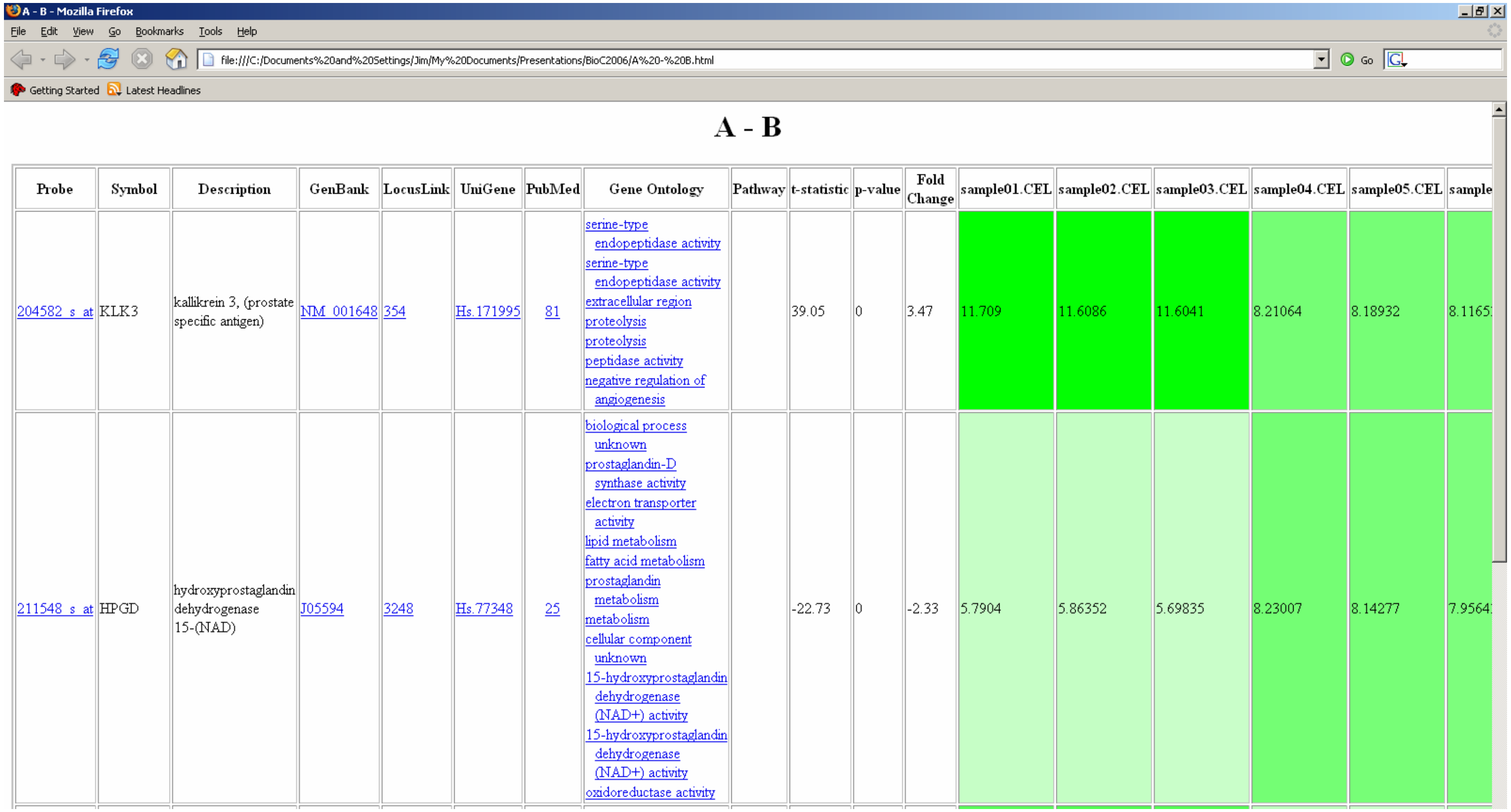
HTML/text tables

- HTML tables
  - interactive exploration of results
  - links to databases
- Text tables
  - easier to manipulate

HTML/text tables

- *annaffy* package
  - both HTML and text
- *annotate* package/*biomaRt* package
  - currently HTML only

HTML tables


A - B - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/Jim/My%20Documents/Presentations/BioC2006/A%20-%20B.html
Go
Getting Started
Latest Headlines

A - B

Probe	Symbol	Description	GenBank	LocusLink	UniGene	PubMed	Gene Ontology	Pathway	t-statistic	p-value	Fold Change	sample01.CEL	sample02.CEL	sample03.CEL	sample04.CEL	sample05.CEL	sample
204582_s_at	KLK3	kallikrein 3, (prostate specific antigen)	NM_001648	354	Hs.171995	81	serine-type endopeptidase activity serine-type endopeptidase activity extracellular region proteolysis proteolysis peptidase activity negative regulation of angiogenesis		39.05	0	3.47	11.709	11.6086	11.6041	8.21064	8.18932	8.1165
211548_s_at	HPGD	hydroxyprostaglandin dehydrogenase 15-(NAD)	J05594	3248	Hs.77348	25	biological process unknown prostaglandin-D synthase activity electron transporter activity lipid metabolism fatty acid metabolism prostaglandin metabolism metabolism cellular component unknown 15-hydroxyprostaglandin dehydrogenase (NAD+) activity 15-hydroxyprostaglandin dehydrogenase (NAD+) activity oxidoreductase activity		-22.73	0	-2.33	5.7904	5.86352	5.69835	8.23007	8.14277	7.9564

Building HTML tables (*annaffy*)

- Select probesets (genes) for a comparison
- Create a table containing annotation links
- Create a table containing the statistics
- Merge these two tables
- Create a table containing the expression values
- Merge these two tables
- Output the table as HTML
- Output the table as text
- Select next set of probesets and repeat above steps

Can’t we simplify this process?

- Answer, of course, is yes!
- limma2annaffy() will output HTML and/or text tables for *all* contrasts automatically.
- Caution; filenames are based on column names of contrasts matrix.

Practice

- Use limma2annaffy() to output HTML tables for the two comparisons we made with our data (A – B, C – D)

annotate/biomaRt

- Useful when no annotation package exists
  - Newer/less common chips
  - MBNI re-mapped chips
- limma2biomaRt()
  - Very similar to limma2annaffy()
  - Uses *biomaRt* package to annotate
  - Uses htmlpage() from *annotate* package for HTML table
  - ENSEMBL

Graphical output

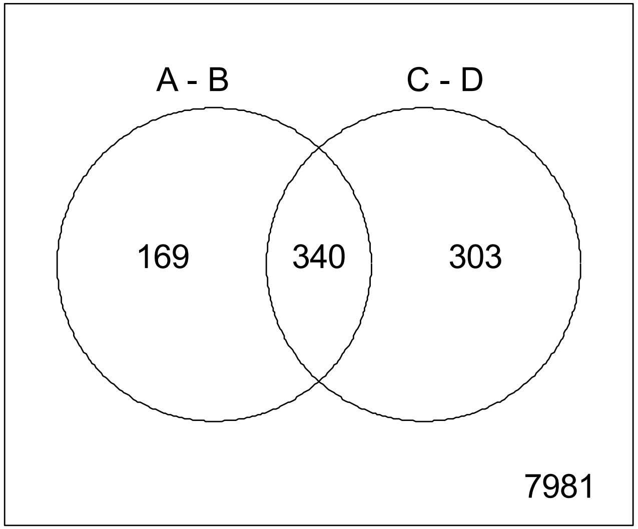
- Quality control plots
- Venn Diagrams

Venn Diagrams

- Common/unique to different comparisons
- decideTests() – *limma* package
- vennCounts2() – *affycoretools* package
  - Select common genes going in same direction
- vennDiagram() – *limma* package

Venn Diagrams

- Nice visual representation
- Great for reports
- But which genes?
- vennSelect() – *affycoretools*
- vennSelectBM() - *affycoretools*



Documentation

- Really two ways to do this
  - Write up something in Word
    - Simple, fast
    - Easiest short term solution
    - Requires boss/client to have Word too
    - Separate analysis/documentation
  - Put analysis/documentation in .Rnw file and use Sweave()
    - Less simple
    - Not a short term solution
    - Requires boss/client to have Acrobat/pdf reader
    - Single analysis/documentation file
    - This is literate programming

What is an .Rnw file?

- Mixture of LaTeX and R code
  - Examples are BioC vignettes
  - Another example in /examples directory of affycoretools package (Statistical_analysis.Rnw)
- Sweave() processes R code and outputs remainder as LaTeX

Why bother?

- Faster in long term
- Consistency in analysis/documentation
- Nicer/more professional looking documentation

Practice

- Run Sweave() on Statistical_analysis.Rnw file
- Assume samples are Trt.WT, Untr.WT, Trt.KO, Untr.KO, modify this file to fit a model that compares Trt.WT vs Untr.WT, Trt.KO vs Untr.KO and the interaction