
Microarray Analysis with R/ Bioconductor

Jiangwen Zhang, Ph.D.



Outline

- Overview of R and Bioconductor
 - Installation, updating and self learning resources
- Basic commands in R
- Using Bioconductor for various steps of microarray analysis (both 1 & 2 channels)



A quick overview

□ R & Bioconductor

- <http://www.r-project.org>
- A statistical environment which implements a dialect of the S language that was developed at AT&T Bell lab
- Open source, cross platform
- Mainly command-line driven
- Current release: 2.9
- <http://www.bioconductor.org>
- Open source software packages written in R for bioinformatics application
- Mainly for microarray analysis at the moment
- Current release: 2.4



□ Advantages

- Cross platform
 - Linux, windows and MacOS
- Comprehensive and centralized
 - Analyzes both Affymetrix and two color spotted microarrays, and covers various stages of data analysis in a single environment
- Cutting edge analysis methods
 - New methods/functions can easily be incorporated and implemented
- Quality check of data analysis methods
 - Algorithms and methods have undergone evaluation by statisticians and computer scientists before launch. And in many cases there are also literature references
- Good documentations
 - Comprehensive manuals, documentations, course materials, course notes and discussion group are available
- A good chance to learn statistics and programming



□ Limitations

- Not easy to learn
 - Require a substantial effort to learn statistics and programming skills before one can do a meaningful data analysis
- Not intuitive
 - Mainly command-line based analysis
 - There are limited wrapper functions and GUIs for certain basic functions



How to download, install and update the basic packages

- R
 - R website -> download (CRAN) -> select mirror -> select system
- Install and update packages by R-GUI
 - Menu bar -> Packages
 - The installation of a standard Bioconductor suite:
>source("http://www.bioconductor.org/getBioC.R")



download, install R and bioconductor packages

Visit:

<http://cran.r-project.org/>

click windows ->base to install R-2.7.1-win32.exe

launch R

copy and paste after prompt ">"

```
source("http://www.bioconductor.org/getBioC.R")
```

```
getBioC("limma")
```

```
getBioC("affy")
```

```
getBioC("hgu95av2")
```

```
getBioC("estrogen")
```

```
getBioC("hgu95av2cdf")
```

```
getBioC("simpleaffy")
```

```
getBioC("annotate")
```

```
getBioC("XML")
```

```
library(affy)
```

```
library(limma)
```

```
library(simpleaffy)
```



An overview of 5 important types of learning resources

1. Web materials
 - R website -> documentation
 - Bioconductor website -> documentation
2. Books
 - website http://ihome.cuhk.edu.hk/~b400559/book_mray.html
 - The best one to start with:
 - Peter Dalgaard “Introductory Statistics with R”
3. Mailing lists
 - R website -> Project -> Mailing lists
 - Bioconductor website -> Project -> Mailing lists
 - Searchable Bioconductor mailing lists
 - Bioconductor website -> Project -> searching mail archives
4. Help pages
 - Start browser help page
 - > `help.start()`
 - Command line
 - > `help("ls")`
 - > `?ls`



R user interface

- ❑ Create a separate sub-directory, say work, to hold data files on which you will use R for this problem. This will be the working directory whenever you use R for this particular problem.
- ❑ To start
 - Click shortcut of R for window system
 - Unix: `bash$ R` to start
- ❑ `>getwd()`
- ❑ Create one subdirectory "estrogen" in that root directory -> File-> Change Dir...

- ❑ Load library, affy
 - `>library(affy)`

- ❑ to quit
 - `R> q()`



Getting Help

○ Details about a specific command whose name you know (input arguments, options, algorithm):

```
> ? t.test
```

```
> help(t.test)
```

```
> help.start() to launch html web page, then  
use search engine link
```



-
- Simple manipulations numbers and vectors
 - Factors, Arrays and matrices
 - Lists and data frames
 - Reading data from files
 - Probability distributions
 - Loops and conditional execution
 - Writing your own functions
 - Statistical models in R
 - Graphics
 - Packages



R as a Calculator

```
> log2(32)
```

```
[1] 5
```

```
> print(sqrt(2))
```

```
[1] 1.414214
```

```
> pi
```

```
[1] 3.141593
```

```
> seq(0, 5, length=6)
```

```
[1] 0 1 2 3 4 5
```

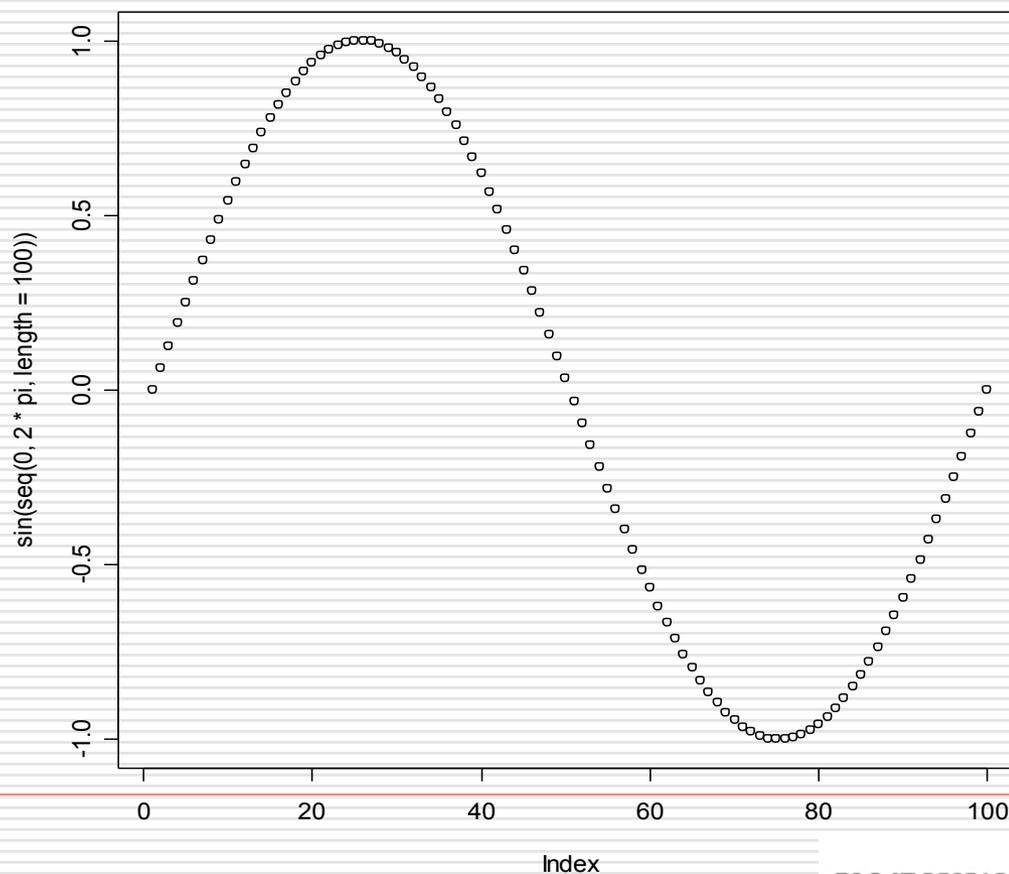
```
> 1+1:10
```

```
[1] 2 3 4 5 6 7 8 9 10 11
```



R as a Graphics Tool

```
> plot(sin(seq(0, 2*pi, length=100)))
```



Variables

```
> a <- 49
```

```
> sqrt(a)
```

```
[1] 7
```

numeric

```
> b <- "The dog ate my homework"
```

```
> sub("dog", "cat", b)
```

```
[1] "The cat ate my homework"
```

**character
string**

```
> c <- (1+1==3)
```

```
> c
```

```
[1] FALSE
```

logical

```
> as.character(b)
```

```
[1] "FALSE"
```



Vectors and assignment

vector: an ordered collection of data of the same type

```
> a <- c(1, 2, 3) #In most contexts the = operator can be used as a alternative for<-  
> a*2  
[1] 2 4 6  
>x<-c(2, 4, 3, 67, 98, 100, 45, 23, 87, 29, 50)  
>sum((x-mean(x))^2)/(length(x)-1)
```

In R, a single number is the special case of a vector with 1 element.

Other vector types: character strings, logical



Matrices and Arrays

matrix: rectangular table of data of the same type

Example: the expression values for 10000 genes for 30 tissue biopsies is a numeric matrix with 10000 rows and 30 columns.

array:

Suppose, for example, z is a vector of 1500 elements. The assignment
`> dim(z) <- c(3,5,100)`

matrices or more generally arrays are multi-dimensional generalizations of vectors.



Lists

list: ordered collection of data of arbitrary types.

lists are a general form of vector in which the various elements need not be of the same type, and are often themselves vectors or lists.

Example:

```
> doe <- list(name="john", age=28, married=F)
> doe$name
[1] "john"
> doe$age
[1] 28
> doe[[3]]
[1] FALSE
```

Typically, vector elements are accessed by their index (an integer) and list elements by \$name (a character string). But both types support both access methods. Slots are accessed by @name.



Data Frames

data frame: rectangular table with rows and columns; data within each column has the same type (e.g. number, text, logical), but different columns may have different types.

Represents the typical data table that researchers come up with – like a spreadsheet.

Example:

```
> a <-data.frame  
(localization,tumorsize,progress,row.names=patients)  
> a
```

	localization	tumorsize	progress
XX348	proximal	6.3	FALSE
XX234	distal	8.0	TRUE
XX987	proximal	10.0	FALSE



Reading data from files

Large data objects will usually be read as values from external files rather than entered during an R session at the keyboard.

Input file form with names and row labels:

	Price	Floor	Area	Rooms	Age	Cent.heat
01	52.00	111.0	830	5	6.2	no
02	54.75	128.0	710	5	7.5	no
03	57.50	101.0	1000	5	4.2	no
04	57.50	131.0	690	6	8.8	no
05	59.75	93.0	900	5	1.9	yes
...						

The data frame may then be read as

```
> HousePrice <- read.table("houses.data", header=TRUE)
```

where the header=TRUE option specifies that the first line is a line of headings, and hence,



Importing and Exporting Data

There are many ways to get data in and out.

Most programs (e.g. Excel), as well as humans, know how to deal with rectangular tables in the form of tab-delimited text files.

```
> x <- read.delim("filename.txt")
```

Also: `read.table`, `read.csv`, `scan`

```
> write.table(x, file="x.txt", sep="\t")
```

Also: `write.matrix`, `write`



Subsetting

Individual elements of a vector, matrix, array or data frame are accessed with “[]” by specifying their index, or their name

```
> a
      localization tumorsize progress
XX348      proximal        6.3         0
XX234       distal         8.0         1
XX987      proximal       10.0         0
> a[3, 2]
[1] 10
> a["XX987", "tumorsize"]
[1] 10
> a["XX987", ]
      localization tumorsize progress
XX987      proximal         10         0
```



Loops

When the same or similar tasks need to be performed multiple times; for all elements of a list; for all columns of an array; etc.

```
for(i in 1:10) {  
  print(i*i)  
}
```

```
i<-1  
while(i<=10) {  
  print(i*i)  
  i<-i+sqrt(i)  
}
```

Also: repeat, break, next



Functions and Operators

Functions do things with data

“Input”: function arguments (0,1,2,...)

“Output”: function result (exactly one)

Example:

```
add <- function(a,b) {  
  result <- a+b  
  return(result)  
}
```



Statistical functions

<code>rnorm, dnorm, pnorm, qnorm</code>	Normal distribution random sample, density, cdf and quantiles
<code>lm, glm, anova</code>	Model fitting
<code>loess, lowess</code>	Smooth curve fitting
<code>sample</code>	Resampling (bootstrap, permutation)
<code>.Random.seed</code>	Random number generation
<code>mean, median</code>	Location statistics
<code>var, cor, cov, mad, range</code>	Scale statistics
<code>svd, qr, chol, eigen</code>	Linear algebra



Distribution	R name	additional arguments
beta	beta	shape1, shape2, ncp
binomial	binom	size, prob
Cauchy	cauchy	location, scale
chi-squared	chisq	df, ncp
exponential	exp	rate
F	f	df1, df2, ncp
gamma	gamma	shape, scale
geometric	geom	prob
hypergeometric	hyper	m, n, k
log-normal	lnorm	meanlog, sdlog
logistic	logis	location, scale
negative binomial	nbinom	size, prob
normal	norm	mean, sd
Poisson	pois	lambda
Student's t	t	df, ncp
uniform	unif	min, max
Weibull	weibull	shape, scale
Wilcoxon	wilcox	m, n



Hypergeometric distribution:

	drawn	not drawn	total
In_pathway	k	$m - k$	m
out_pathway	$n - k$	$N + k - n - m$	$N - m$
Total	n	$N - n$	N

$$P(X = k) = \frac{\binom{m}{k} \binom{N-m}{n-k}}{\binom{N}{n}}$$

```
m <- 10;
N <- 17;
b <- N - m
n <- 8
x <- 0:n
rbind(phyper(x, m, b, n), dhyper(x, m, b, n))
```

```
> rbind(phyper(x, m, b, n), dhyper(x, m, b, n))
  0      1      2      3      4      5      6
[1,] 0      0.0004113534 0.01336898 0.1170300 0.4193747 0.7821884 0.9635952
[2,] 0      0.0004113534 0.01295763 0.1036610 0.3023447 0.3628137 0.1814068
  7      8
[1,] 0.99814891 1.00000000
[2,] 0.03455368 0.00185109
```



Bioconductor & Microarray

Affymetrix: exon, expression, chip-chip, SNP

Agilent:

Illumina: beadarray

Nimblegen:

Solexa: sequencing based chip-seq

microRNA detection

Comparative Genomic Hybridization (CGH)

detects deletions or amplifications of genomic sequence

ChIP on chip

chromatin immunoprecipitation

Single Nucleotide Polymorphism screening (SNP)

measures an individual's genotype at known sites of variance

Resequencing: chip-seq

Cell Arrays

Protein Arrays

Tissue Arrays

Flow-cytometry

Mass data



Challenges in Genomics

- Diverse biological data types: Genotype, Copy Number, Transcription, Methylation,...
- Diverse technologies to measure the above: Affymetrix, Nimblegen,...
- Integration of multiple and diverse data structures
- Large datasets

Functional genomics, gene regulation network, signaling pathway, motif identification



Aims of Bioconductor

- Provide access to powerful statistical and graphical methods for the analysis of genomic data.
- Facilitate the integration of biological metadata (GenBank, GO, LocusLink, PubMed) in the analysis of experimental data.
- Allow the rapid development of extensible, interoperable, and scalable software.
- Promote high-quality documentation and reproducible research.
- Provide training in computational and statistical methods.



Install, start and update bioconductor

- Find `getBioC()`
 - Bioc website -> How To -> `getBioC`
 - > `source("http://www.bioconductor.org/getBioC.R")`
 - > `getBioC.R`
 - to install extra packages not in standard suite
 - Menu -> packages -> install packages from bioconductor

- Start Bioconductor
 - > `library(Biobase)`

- Update Bioconductor packages
 - Menu -> packages -> update packages from bioconductor

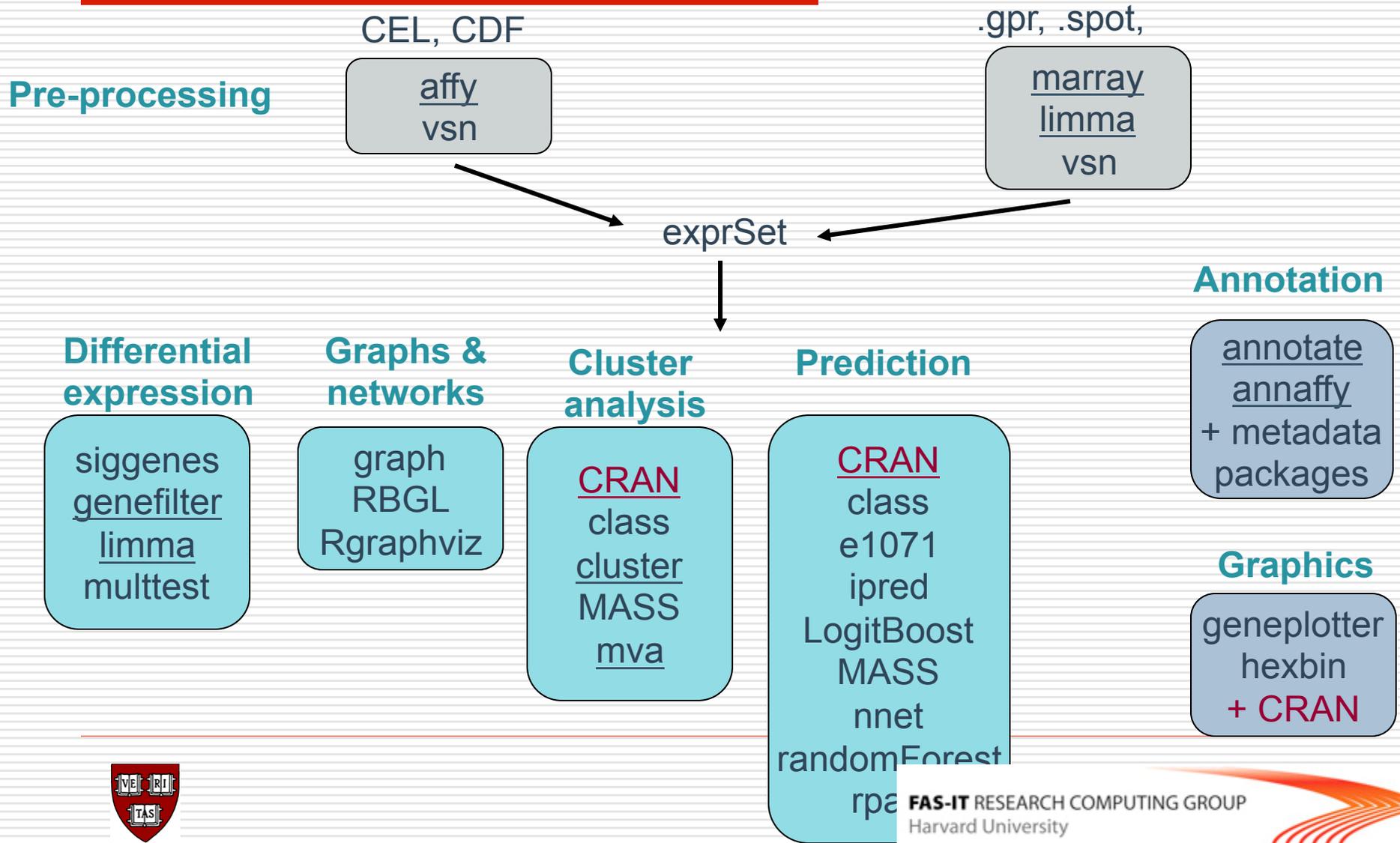


Microarray Data analysis workflow

- Image analysis
- Raw data
- Diagnostic plots
- Normalization
- Filtering
- Estimate missing values
- Differential gene inference
 - Linear modeling and factorial experiment
- Clustering
- Classification
- Annotation
- GO analysis
- Pathway analysis



Microarray data analysis



Useful R/BioC Packages

<code>Marray, limma</code>	Spotted cDNA array analysis
<code>affy</code>	Affymetrix array analysis
<code>vsn</code>	Variance stabilization
<code>annotate</code>	Link microarray data to metadata on the web
<code>ctest</code>	Statistical tests
<code>genefilter, limma, multtest, siggenes</code>	Gene filtering (e.g.: differential expression)
<code>mva, cluster, clust</code>	Clustering
<code>class, rpart, nnet</code>	Classification



What's Your Question?

What are the targets genes for my knock-out gene?

Gene discovery, differential expression

Is a specified group of genes (genes from a pathway) all up-regulated in a specified condition?

Gene set enrichment analysis

Can I use the expression profile of cancer patients to predict chemotherapy outcome?

Class prediction, classification

Pathways/network affected?

Kegg, Biocarta

Considering Pathway/network Topology



❖ Pre-processing microarray data

diagnostic, normalization

❖ Differential Gene Expression

identification of up and down regulated genes

❖ Annotation and metadata

get the DE genes' id, pathway involvement, GO

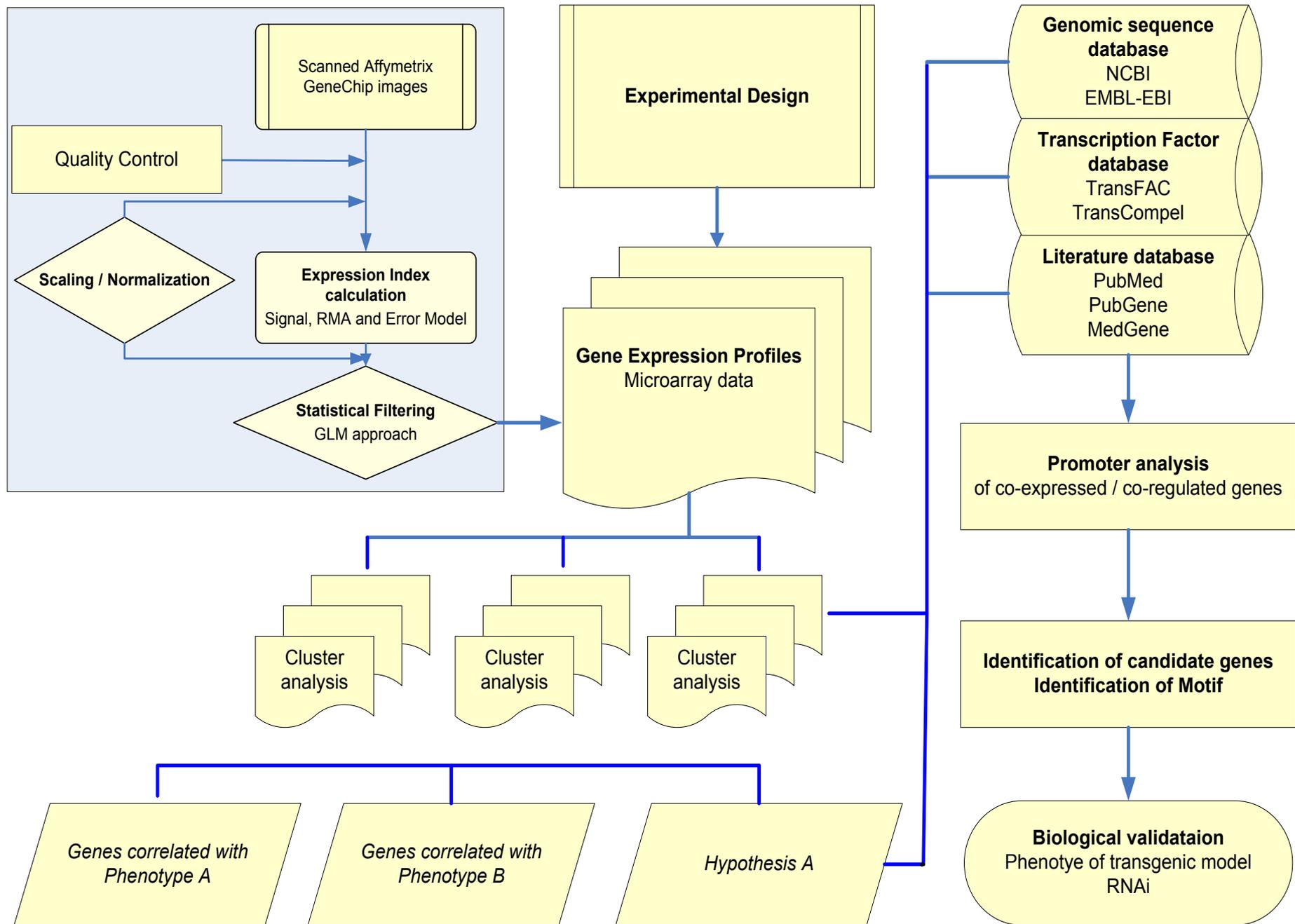
❖ Distances, Prediction, and Cluster Analysis

sample similarity calculation and visualization by heatmap

❖ Class prediction

provide expression profile of type-known samples to computer, train it, and let computer to classify type-unknown samples





LIMMA and LIMMA GUI

- LIMMA is another library to perform basic 2 channels analysis, linear modeling for both single channel and 2 channel
- There is a nice GUI for LIMMA
- `library(limma)`
- `library(limmaGUI)`



Affymetrix chips

- **DAT** file: Image file, $\sim 10^7$ pixels, ~ 50 MB.
- **CEL** file: Cell intensity file, probe level PM and MM values.
- **CDF** (Chip Description File): Describes which probes belong to which probe-pair set and the location of the probes.



GeneChip® Eukaryotic Target Labeling Assays for Expression Analysis

One-Cycle Target Labeling

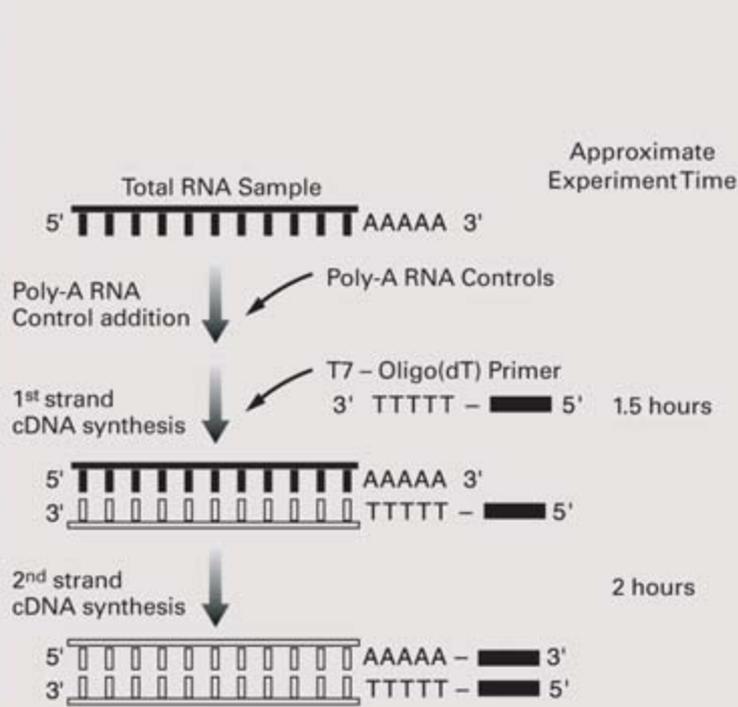
(for 1-15 µg total RNA or 0.2-2 µg mRNA)

Products Used

Poly-A RNA Control Kit

One-Cycle cDNA Synthesis Kit

Approximate Experiment Time



Two-Cycle Target Labeling

(for 10-100 ng total RNA)

Approximate Experiment Time

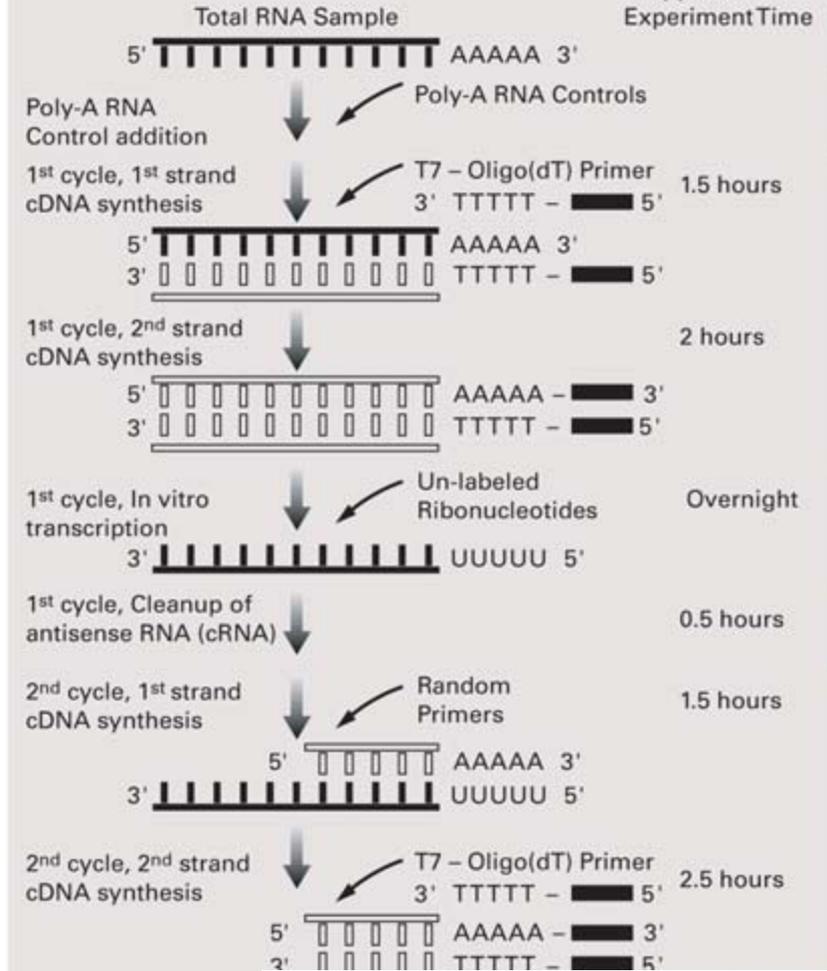
Products Used

Poly-A RNA Control Kit

Two-Cycle cDNA Synthesis Kit

Sample Cleanup Module

Two-Cycle cDNA Synthesis Kit



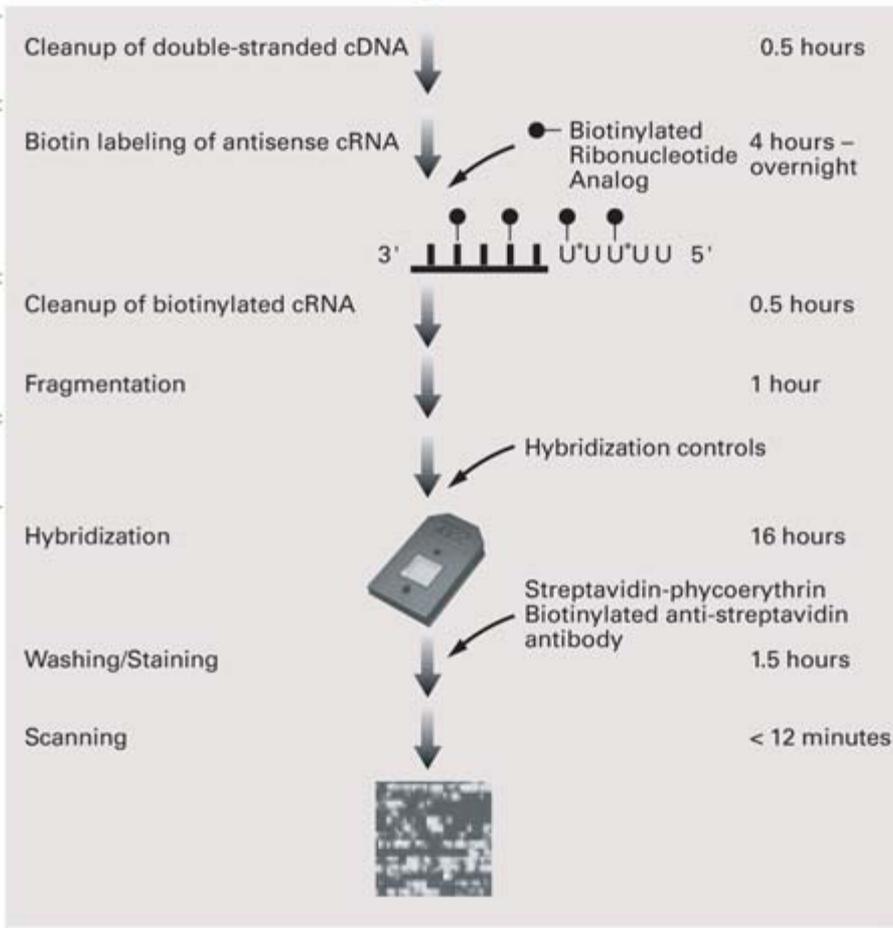
Products Used

Sample Cleanup Module

IVT Labeling Kit

Sample Cleanup Module

Hybridization Control Kit



Legend

-  RNA
-  DNA
-  T7 promoter
-  Biotin
-  U* – Pseudouridine



affy package

CEL and CDF
files



Class `AffyBatch`



`rma` or `mas5`
`expresso`
`express`



Class `exprSet`

Save data to file using `write.exprs` or continue analysis using other Bioconductor and CRAN packages



affy and simpleaffy package

- Class definitions for probe-level data: **AffyBatch**, **ProbSet**, **Cdf**, **Cel**.
- Basic methods for manipulating microarray objects: printing, plotting, subsetting.
- Functions and widgets for data input from **CEL** and **CDF** files, and automatic generation of microarray data objects.
- Diagnostic plots: 2D spatial images, density plots, boxplots, MA-plots.
 - image: 2D spatial color images of log intensities (AffyBatch, Cel).
 - boxplot: boxplots of log intensities (AffyBatch).
 - mva.pairs: scatter-plots with fitted curves (apply exprs, pm, or mm to AffyBatch object).
 - hist: density plots of log intensities (AffyBatch).
- Check RNA degradation and couple control metrics defined by affymetrix company

```
library(simpleaffy)
Data.qc<-qc(data)
avbg(Data.qc)
sfs(Data.qc)
percent.present(Data.qc)
ratios(Data.qc[,1:2])
```



Quality control(2)

QC metrics

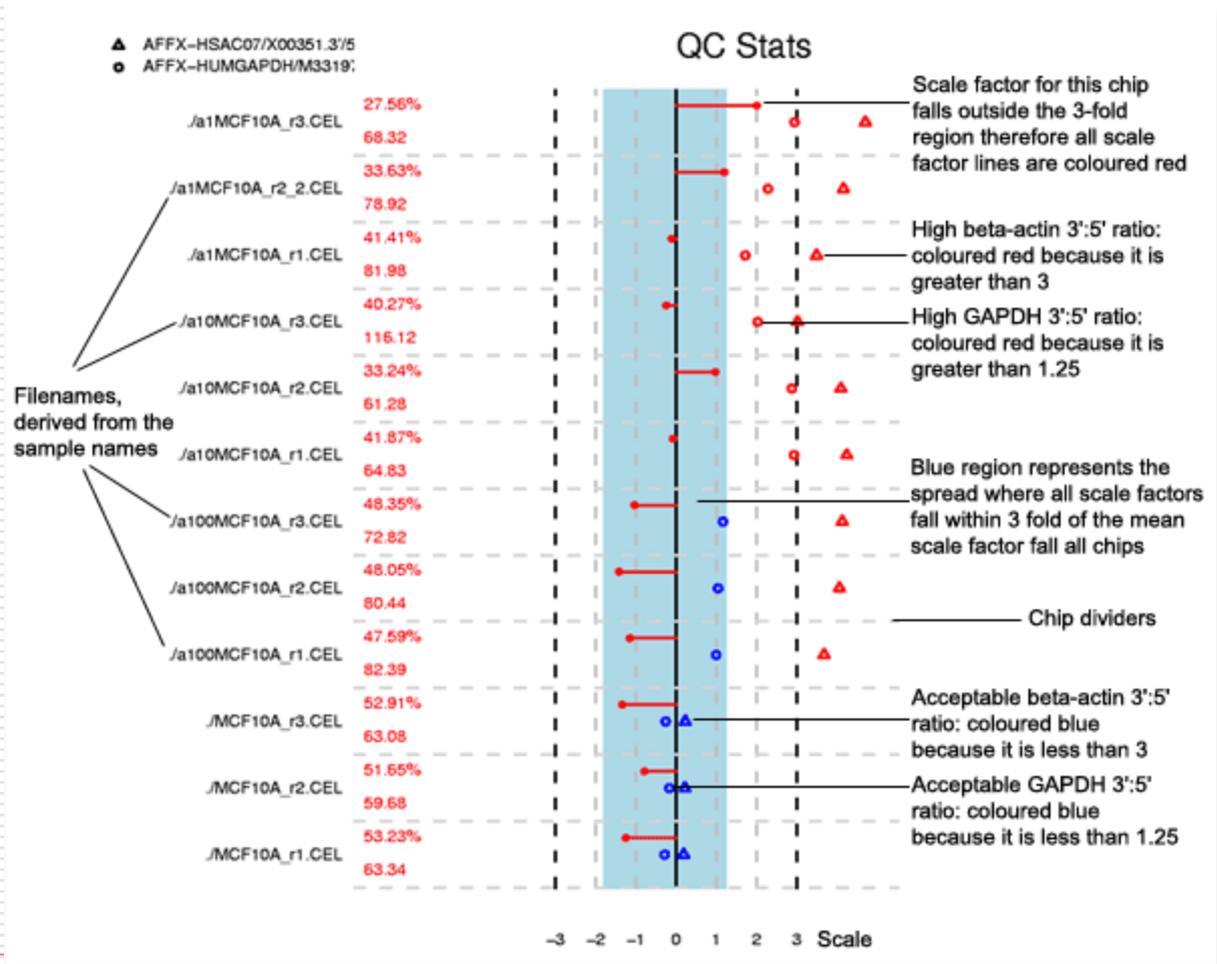
- 1. Average background
- 2. Scale factor
- 3. Number of genes called present
- 4. 3' to 5' ratios of actin and GAPDH
- 5. Uses ordered probes in all probeset to detect possible RNA degradation.



Lab

```
> library(affy)
> library("simpleaffy")
> data <- ReadAffy()
> qc(data) -> data.qc
> avbg(data.qc) # comparable bg expected
> sfs(data.qc) # within 3folds of each other expected
> percent.present(data.qc) # extremely low value is a
  # problem
> ratios(data.qc)
> AffyRNAdeg(data) -> RNAdeg
> plotAffyRNAdeg(RNAdeg)
> summaryAffyRNAdeg(RNAdeg)
```





Affymetrix spike-in controls, ploy-A control and hybridization control

- The ploy-A controls AFFX-r2-Bs-Dap, AFFX-r2-Bs-Thr, AFFXr2-Bs-Phe and AFFX-r2-Bs-Lys (more spikes slot) are modified *B. subtilis* genes and should be called present at a decreasing intensity, to verify that there was no bias during the retro-transcription between highly expressed genes and low expressed genes. Note that the linearity for lys, phe and thr (dap is present at a much higher concentration) is affected by a double amplification.
- hybridization control BioB, BioC, BioD in increasing concentration.



Normalization by affy package

- Background estimation.
- Probe-level normalization: quantile and curve-fitting normalization (Bolstad et al., 2003).
- Expression measures: MAS 4.0 AvDiff, MAS 5.0 Signal, MBEI (Li & Wong, 2001), RMA (Irizarry et al., 2003).
- Main functions: **ReadAffy**, **rma**, **mas5**, **expresso**, **express**.

Pre-processing oligonucleotide chip data:

- diagnostic plots,
- background correction,
- probe-level normalization,
- computation of expression measures.



Low level analysis

□ Normalization

- The main goal is to remove the systematic bias in the data as completely as possible, while preserving the variation in gene expression that occurs because of biologically relevant changes in transcription.
- A basic assumption of most normalization procedures is that the average gene expression level does not change in an experiment.
- Normalization is different in spotted/two-color compared with high-density-oligonucleotides (Affy) technology



Low level analysis

□ *Bioconductor affy R package*

BG correction
None
MAS
RMA
RMA2
<i>gcrma</i>

Normalization
Linear constant
Contrasts
Invariant set
Loess
Cyclic loess
Cubic spline
Quantiles (robust)
VSN

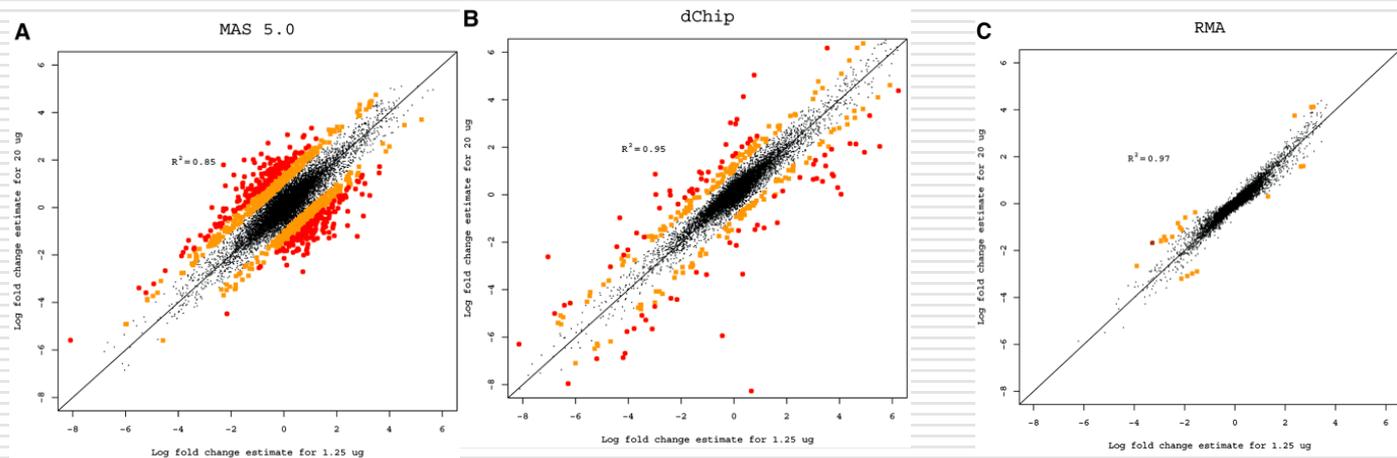
PM correction
none
Subtract MM (MAS 4)
Subtract IM (MAS 5)
PM only

Expression index
Avg Diff (MAS 4)
Signal (MAS 5)
Li.Wong model
RMA (median polish)
GCRMA
<i>Playerout</i>



RMA, MAS5, dCHIP

Folds change consistency comparison
using 1.25 μg of RNA vs using 20 μg of RNA



Code for affy analysis

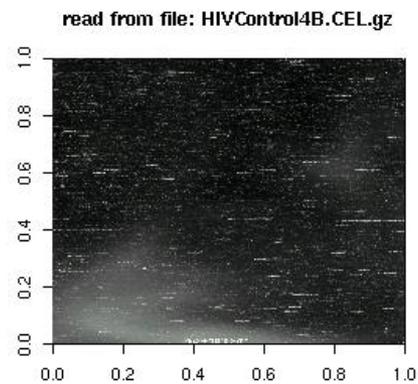
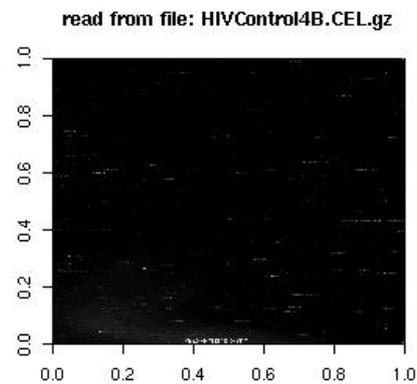
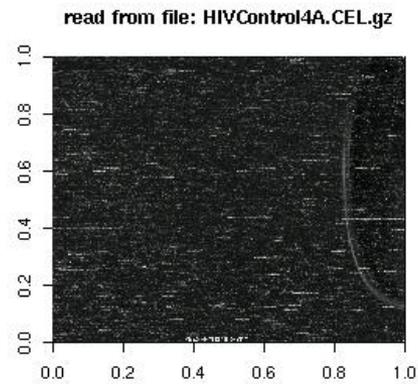
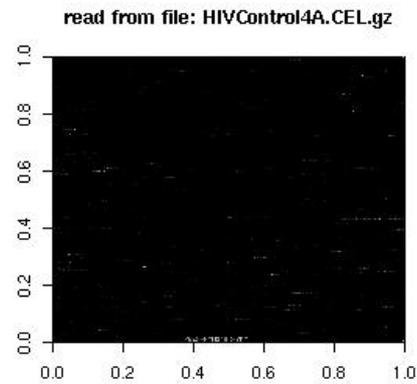
- For Affymetrix data
- Data loading
 - > library(affy)
 - > ReadAffy()->affybatch
- Some diagnostic plots
 - > hist(affybatch)
 - > image(affybatch)
 - > boxplot(affybatch)
 - > RNAdeg<-AffyRNAdeg(affybatch)
 - > plotAffyRNAdeg(RNAdeg(affybatch))
 - > summary(RNAdeg)
 - > mva.pairs(affybatch)

The sequence of affy probe level analysis

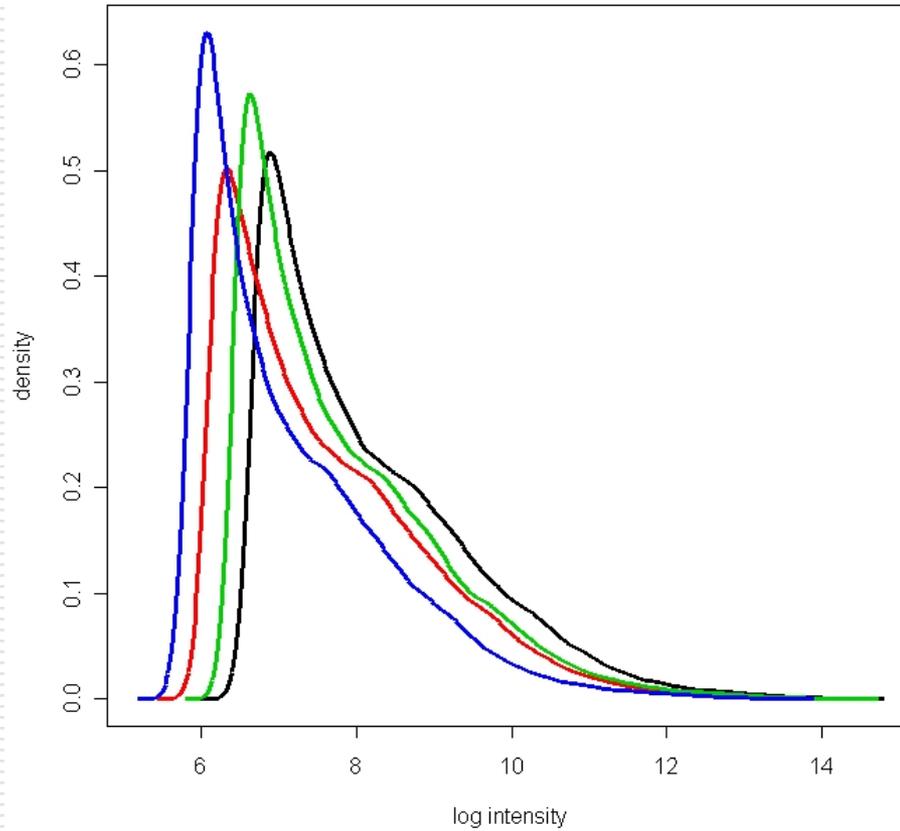
- > background correction -> normalization -> pm correction -> expression summarization
 - > rma(affybatch)
 - > Mas(affybatch)



image



hist

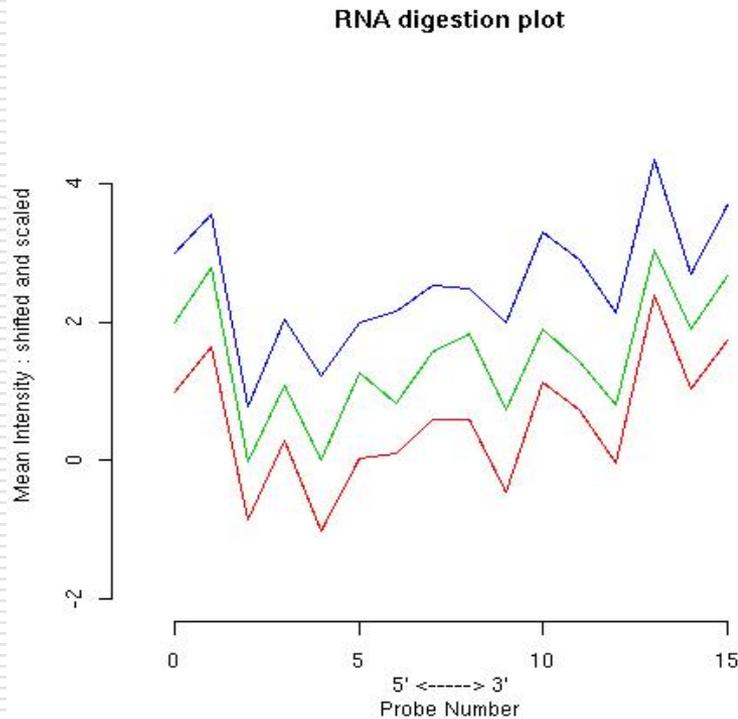


`hist(Dilution,col=1:4,type="l")`



```
>AffyRNADeg(AffyBatch)
```

```
>plotAffyRNADeg()
```



Differential Gene Expression



□ Typical questions

- **Detect genes are differentially expressed between two or more samples. [t-test, F-test and many others]**
- **Identification of groups of genes with characterizing a particular class of tumors. [Discrimination]**
- **Discover at molecular level, potential subclasses of tumors / disease. [Clustering]**
- **Detection of gene regulatory mechanisms. [Network and meta data]**



Significance Inference

1. Calculation of a statistic based on replicate array data for ranking genes according to their possibilities of differential expression
2. Selection of a cut-off value for rejecting the null-hypothesis that the gene is not differentially expressed

- *Underlying distribution*
- *Sample size*
- Parametric test
- Nonparametric test
- Permutation test
- Bayesian mixture model



Significance Inference

- Student's T-test $t_i = (M_i/SE_i)$
 - where SE_i (standard error of M_i) = s_i/\sqrt{n}
- Introduces some conservative protection against outlier M-values and poor quality spots
- Limitations
 - **large t-statistic can be driven by an unrealistically small value for s**
 - Suffers from multiple hypothesis testing problem
 - Suffers from inflated type I error
- Moderated t-statistic from limma package
 - Uses empirical Bayes to estimate a posterior variance of the gene with the information borrow from all genes
 - Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology* **3**, No. 1, Article 3.



Limma package

Ebayes: borrow information from ensemble of genes, a good strategy for small sample project, superior to t-test.

has more functions than just linear modeling, it basically provides another ways for prepossessing, normalizing and plotting data as marray packages



Significance Inference

- Fold change as a threshold cut-off is inadequate, even if it is an average of replicates
- Limitations:
 - Fixed threshold does not account for statistical significance
 - Does not take into account of the variability of the expression levels for each gene
 - Genes with larger variances have a good chance of giving a large fold change even if they are not differentially expressed
 - Inflated type I & type II errors



Statistical filtering

- Ultimately, what matters is biological relevance.
- P-values should help you evaluate the strength of the evidence, rather than being used as an absolute yardstick of significance.
- Statistical significance is not necessarily the same as biological significance.



Filtering before DE study: library (genefilter)

Two filters: gene should be above “100” for 5 times and have a Cox-PH-model p-value <0.01

```
kF <- kOverA(5, 100)
```

Assemble them in a filtering function

```
ff <- filterfun(kF, cF)
```

Apply the filter

```
sel <- genefilter(exprs(DATA), ff)
```

Select the relevant subset of the data

```
mySub <- DATA[sel,]
```



Annotation and metadata



Biological metadata

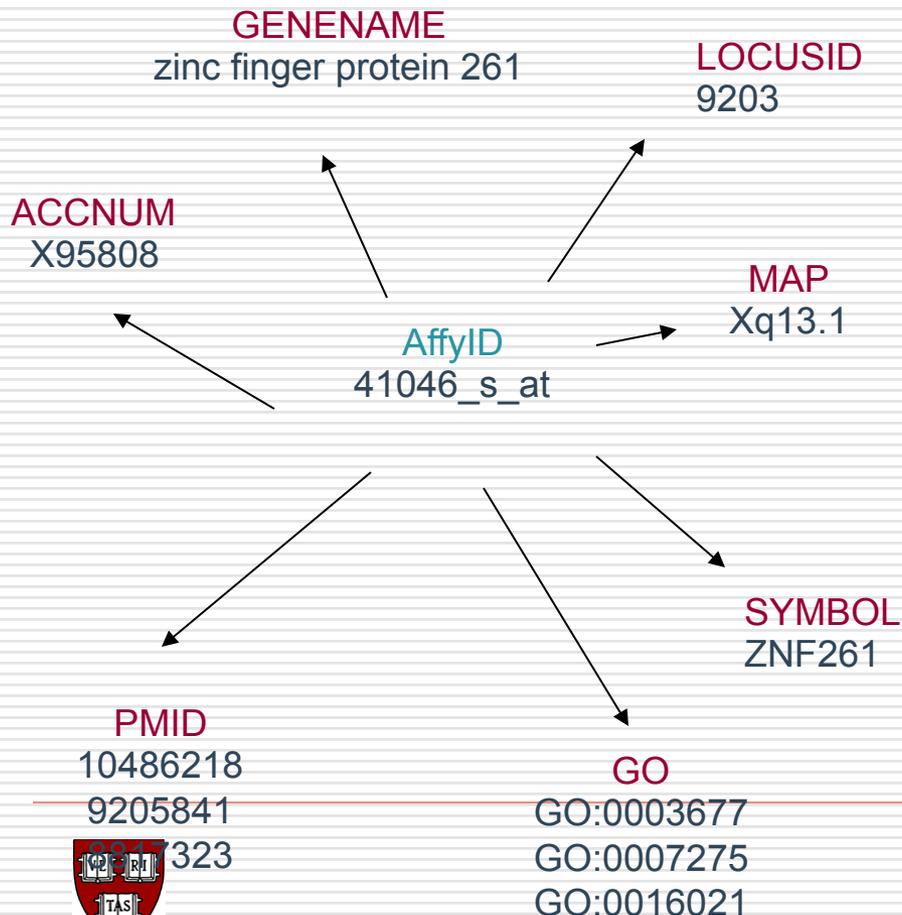
- Biological attributes that can be applied to the experimental data.
- E.g. for genes
 - chromosomal location;
 - gene annotation (LocusLink, GO);
 - relevant literature (PubMed).
- Biological metadata sets are large, evolving rapidly, and typically distributed via the WWW.
- Tools: **annotate**, **annaffy**, and **AnnBuilder** packages, and annotation data packages.



annotate, annafy, and AnnBuilder

Metadata package hgu95av2

mappings between different gene identifiers for hgu95av2 chip.



- Assemble and process genomic annotation data from public repositories.
- Build annotation data packages or XML data documents.
- Associate experimental data in real time to biological metadata from web databases such as GenBank, GO, KEGG, LocusLink, and PubMed.
- Process and store query results: e.g., search PubMed abstracts.
- Generate HTML reports of analyses.

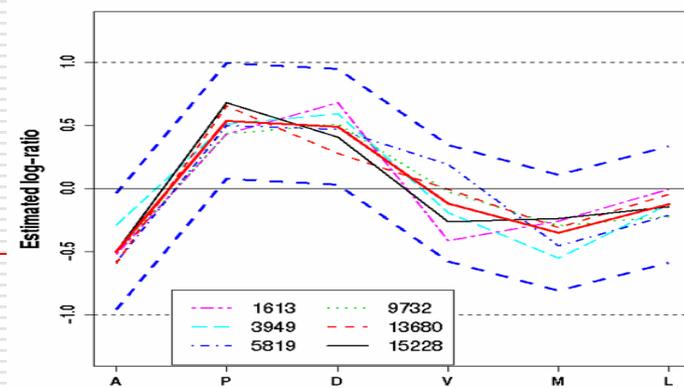


+ many other maps



Distances, Prediction, and Cluster Analysis





clustering

- 1a. Find the genes whose expression fits specific, predefined patterns.
- 1b. Find the genes whose expression follows the pattern of predefined gene or set of genes.
2. Carry out some kind of exploratory analysis to see what expression patterns emerge; **cluster analysis (usually on genes)**.

tumor classification:

1. The identification of new/unknown tumor classes using gene expression profiles; **cluster analysis (usually on samples)**.
2. The classification of malignancies into known classes; **Discrimination**.
3. The identification of “marker” genes that characterize the different tumor classes; **variable selection**.



Cluster analysis for microarray data

- Hierarchical method
 - Bottum-up method
 - Single linkage
 - Complete linkage
 - Average linkage
 - Top-down method
 - TSVQ clustering
 - Macnaughton-Smith clustering
- Partitioning method
 - k-means clustering
 - k-medoids clustering
 - Self-organizing maps (SOMs)



Hierarchical method

Hierarchical clustering methods produce a tree or dendrogram. The tree can be built in two distinct ways

- bottom-up: agglomerative clustering;
- top-down: divisive clustering.

- Merging:
 - Computationally simple
 - Precise at bottom of tree
 - Good for many small clusters
- Divisive
 - More complex, but more precise at the top of the tree
 - Good for looking at large and/or few clusters
- For Gene expression applications, divisive makes more sense.

Applying Euclidean distance to categorical data is invalid
Correlation metric applied to highly skewed data will give misleading results



Partition methods

- Partition the data into a prespecified number k of
- mutually exclusive and exhaustive groups.

- Iteratively reallocate the observations to clusters
- until some criterion is met, e.g. minimize within cluster sums of squares.

- *Examples:*
 - k -means, self-organizing maps (SOM), *PAM*, etc.;
 - Fuzzy: needs stochastic model, e.g. Gaussian mixtures.



Partitioning vs. hierarchical

Partitioning:

Advantages

- Optimal for certain criteria.

Disadvantages

- Need initial k ;
- Often require long computation times.

Hierarchical

Advantages

- Faster computation.

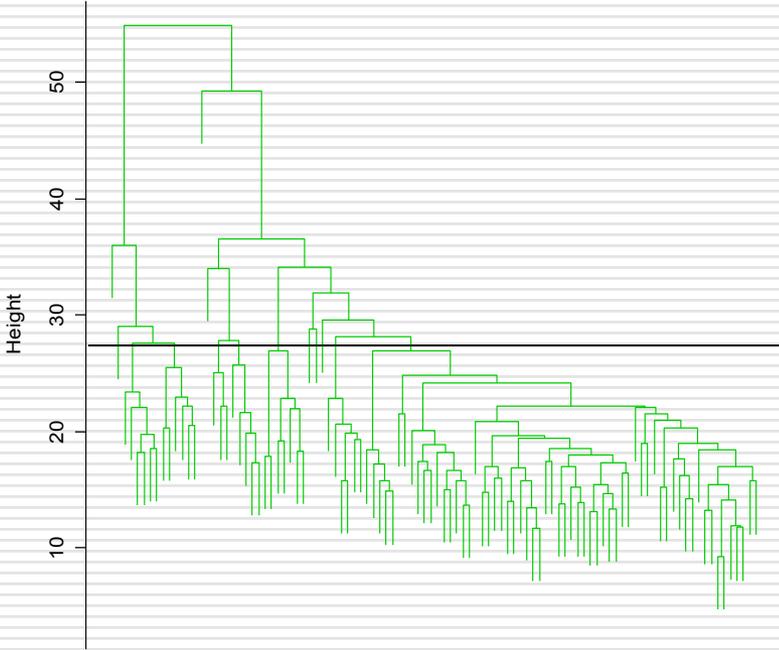
Disadvantages

- Rigid;
- Cannot correct later for erroneous decisions made earlier.



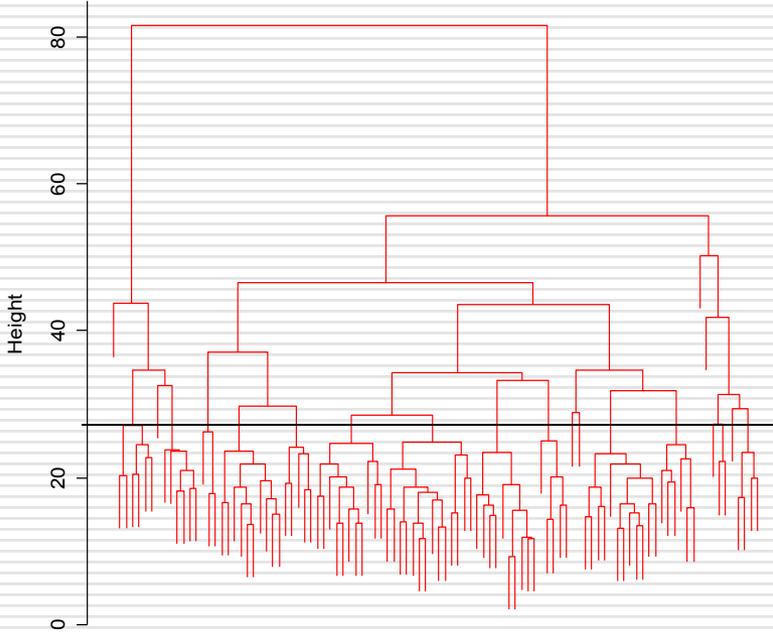
Hierarchical clustering

Average linkage



Cut tree at given height yields
7 clusters

Complete linkage



15 clusters

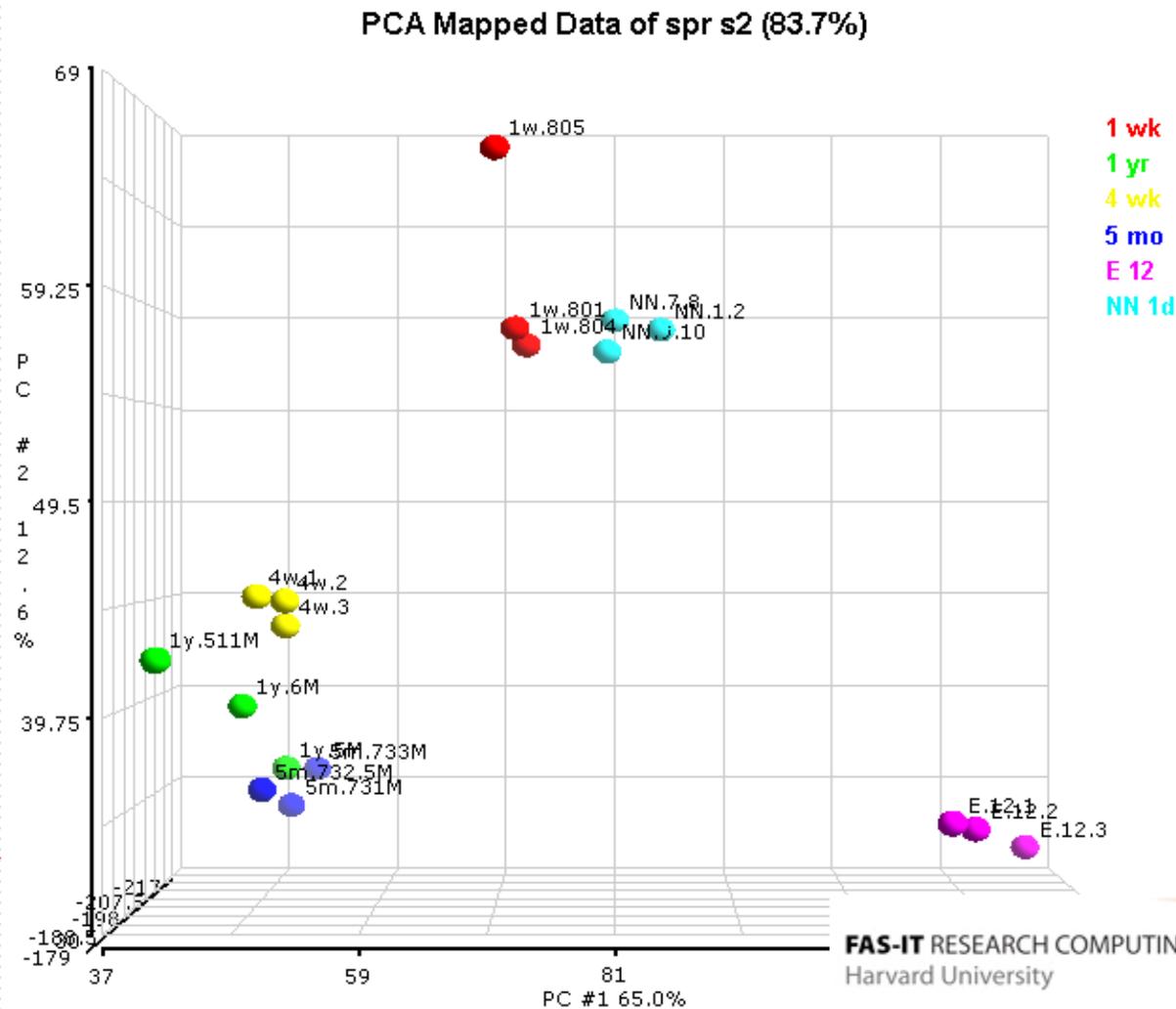


Visualization

- Principal components analysis (PCA), an exploratory technique that reduces data dimensionality to 2 or 3 dimensional space.
- For a matrix of m genes \times n samples, create a new covariance matrix of size $n \times n$
- Thus transform some large number of variables into a smaller number of uncorrelated variables called principal components (PCs).



Visualization



Distances

- Microarray data analysis often involves
 - clustering genes and/or samples;
 - classifying genes and/or samples.
- Both types of analyses are based on a measure of distance (or similarity) between genes or samples.
- R has a number of functions for computing and plotting distance and similarity matrices.



Distances

- Distance functions
 - **dist (mva)**: Euclidean, Manhattan, Canberra, binary;
 - **daisy (cluster)**.
- Correlation functions
 - **cor, cov.wt**.
- Plotting functions
 - **image**;
 - **plotcorr (ellipse)**;
 - **plot.cor, plot.mat (sma)**.



R cluster analysis packages

- **cclust**: convex clustering methods.
- **class**: self-organizing maps (**SOM**).
- **cluster**:
 - AGglomerative NESTing (**agnes**),
 - Clustering LARe Applications (**clara**),
 - **DI**visive **AN**ALysis (**diana**),
 - Fuzzy Analysis (**fanny**),
 - MONothetic Analysis (**mona**),
 - Partitioning Around Medoids (**pam**).
- **e1071**:
 - fuzzy C-means clustering (**cmeans**),
 - bagged clustering (**bclust**).
- **flexmix**: flexible mixture modeling.
- **fpc**: fixed point clusters, clusterwise regression and discriminant plots.
- **GeneSOM**: self-organizing maps.
- **mclust**, **mclust98**: model-based cluster analysis.
- **mva**:
 - hierarchical clustering (**hclust**),
 - k-means (**kmeans**).
- Specialized summary, plot, and print methods for clu

Download
from CRAN



Class prediction

- Old and extensive literature on class prediction, in statistics and machine learning.
- Examples of classifiers
 - nearest neighbor classifiers (k-NN);
 - discriminant analysis: linear, quadratic, logistic;
 - neural networks;
 - classification trees;
 - support vector machines.
- Aggregated classifiers: bagging and boosting



R class prediction packages

Download
from CRAN

- **class:**
 - k-nearest neighbor (**knn**),
 - learning vector quantization (**lvq**).
- **classPP:** projection pursuit.
- **e1071:** support vector machines (**svm**).
- **ipred:** bagging, resampling based estimation of prediction error.
- **knnTree:** k-nn classification with variable selection inside leaves of a tree.
- **LogitBoost:** boosting for tree stumps.
- **MASS:** linear and quadratic discriminant analysis (**lda**, **qda**).
- **mlbench:** machine learning benchmark problems.
- **nnet:** feed-forward neural networks and multinomial log-linear models.
- **pamR:** prediction analysis for microarrays.
- **randomForest:** random forests.
- **rpart:** classification and regression trees.
- **sma:** diagonal linear and quadratic discriminant analysis, naïve Bayes (**stat.diag.da**).



Lab

□ Estrogen dataset at Bioconductor

<http://www.bioconductor.org/data/experimental/html/estrogen.html>

8 arrays (duplicate): ER+ breast cancer cells were exposed to estrogen 10 hours, 48 hours.

E10, E10, C10, C10, E48, E48, C48, C48

□ Create working directory

■ Move .cel files to the directory

□ Create experiment description file

■ Download EstrogenTargets.txt from <http://bioinf.wehi.edu.au/affylmGUI/EstrogenTargets.txt>

A P

10 hours

48 hours

eT2	ET2
et1	Et1

et2	Et2
eT1	ET1



Project

Brief introduction:

data gives results from a 2x2 factorial experiment on MCF7 breast cancer cells using Affymetrix HGU95av2 arrays. The factors in this experiment were estrogen (present or absent) and length of exposure (10 or 48 hours). The aim of the study is the identify genes which respond to estrogen and to classify these into early and late responders. Genes which respond early are putative direct-target genes while those which respond late are probably downstream targets in the molecular pathway

This experiment studied the effect of estrogen on the gene expression in estrogen receptor positive breast cancer cells over time. After serum starvation, samples were exposed to estrogen, and mRNA was harvested at two time points (10 or 48 hours). The control samples were not exposed to estrogen and were harvested at the same time points. Table 1 shows the experimental design, and corresponding samples names. The full data set (12,625 probes, 32 samples) and its analysis are discussed in Scholtens, et al. Analyzing Factorial Designed Microarray Experiments.



Project

Visit:

<http://cran.r-project.org/>

click windows ->base to install R-2.7.1-win32.exe

launch R

copy and paste after prompt ">"

```
source("http://www.bioconductor.org/getBioC.R")
```

```
getBioC("limma")
```

```
getBioC("marray")
```

```
getBioC("affy")
```

```
getBioC("hgu95av2")
```

```
getBioC("estrogen")
```

```
getBioC("hgu95av2cdf")
```

```
getBioC("simpleaffy")
```

```
library(affy)
```

```
library(limma)
```

```
library(simpleaffy)
```



Project

Table 1: Experimental Conditions for .cel Files

time	estrogen	
	absent	present
10 hours	et1	Et1
	et2	Et2
48 hours	eT1	ET1
	eT2	ET2



filename	estrogen	time.h
low10-1.cel	absent	10
low10-2.cel	absent	10
high10-1.cel	present	10
high10-2.cel	present	10
low48-1.cel	absent	48
low48-2.cel	absent	48
high48-1.cel	present	48
high48-2.cel	present	48



gene i in sample j ($j = 1, \dots, 8$).

$$y_{ij} = \mu_i + \beta_{ESi}x_{ESj} + \beta_{TIMEi}x_{TIMEj} + \beta_{ES:TIMEi}x_{ESj}x_{TIMEj} + \epsilon_{ij}$$

$$y_{ij} = \mu_i + \beta_{TIMEi}x_{TIMEj} + \epsilon_i$$

```
>design<-model.matrix(~factor(estrogen)*factor(time.h), pdat)
```

```
> design
```

	Intercept	present	48h	present:48h
low10-1.cel	1	0	0	0
low10-2.cel	1	0	0	0
high10-1.cel	1	1	0	0
high10-2.cel	1	1	0	0
low48-1.cel	1	0	1	0
low48-2.cel	1	0	1	0
high48-1.cel	1	1	1	1
high48-2.cel	1	1	1	1

```
colnames(design)<-c("Intercept", "ES", "T48", "ES:T48")
```

```
fit<-lmFit(esEset, design)
```

```
fit$coefficients[1:3,]
```



Project

```
####lauch necessary bioconductor packages
```

```
library(estrogen)
```

```
library(limma)
```

```
library(hgu95av2cdf)
```

```
####define data directory
```

```
datadir<-system.file(package="estrogen", "extdata")#OR
```

```
datadir <- file.path(.find.package("estrogen"),"extdata")
```

```
datadir
```



Project

```
##read in experiment design information and .cel files
targets <- readTargets("phenoData.txt",
path=datadir,sep="",row.names="filename")
targets
library(affy)
esAB<-ReadAffy(filenames=targets$filename, celfile.path=datadir)
```

```
###quality check by "simpleaffy" package
library(simpleaffy)
qc(esAB)->qc
plot(qc)
```



Project

```
#####normalize#####
```

```
normalize.methods(esAB)
```

```
esEset<-rma(esAB)
```

```
pairs(exprs(esEset))
```

```
boxplot(esAB, col="red")
```

```
boxplot(data.frame(exprs(esEset)))#not code: boxplot(exprs(esEset)), this only  
generate one boxplot
```

```
#####heatmap#####
```

```
##Select the 50 genes with the highest variation (standard deviation) across chips
```

```
rsd <- apply(exprs(esEset), 1, sd)
```

```
sel <- order(rsd, decreasing = TRUE)[1:50]
```

```
heatmap(exprs(esEset)[sel, ], col = gentlecol(256))
```



Project

```
#####limma#####  
library(limma)  
#pdat<-pData(esEset)  
design<-model.matrix(~factor(estrogen)*factor(time.h), targets)  
design  
colnames(design)<-c("Intercept", "ES", "T48", "ES:T48")  
fit<-lmFit(esEset, design)  
#fit$coefficients[1:3,]  
contM<-cbind(es10=c(0,1,0,0), es48=c(0,1,0,1))  
fitC<-contrasts.fit(fit, contM)  
fitC<-eBayes(fitC)  
#glist<-topTable(fitC, n=20,coef=2, adjust="fdr")  
glist<-topTable(fitC, n=20,coef=1, adjust="fdr")  
library(marray)  
table2html(glist, filename="estrogen_file.html", disp="file")  
getwd()  
edit(glist)
```



Project

```
####annotation, query PubMed
getBioC("XML")
getBioC("annotate")
library(annotate)
library(XML)
annotation(esAB)# "hgu95av2"
library("hgu95av2")
absts<-pm.getabst(glist$ID,"hgu95av2") ###pm.getabst query PubMed directly, be careful, too many queries can get you banned
```

```
####to look at second gene's pubmed abstract
titl<-sapply(absts[[2]], articleTitle)
sapply(absts[[2]], pubDate)
sapply(absts[[2]], pmid)
sapply(absts[[2]], journal)
sapply(absts[[2]], abstText)
strwrap(titl, simplify=F) #strwrap to format text to fit the page width
##search abstracts with key word
```



Project

```
sapply(absts[[2]], abstText)->try
grep("estrogen",try)->try1
pmAbst2HTML(absts[[2]][try1],filename="estrogen_pm.html")
#if(!interactive()) file.remove("estrogen_file.html")
ll<-getEG(glist$ID, "hgu95av2")
sym<-getSYMBOL(glist$ID, "hgu95av2")
##htmlpage(genelist, filename, title, othertnames, table.head, table.center = TRUE,
repository = list("en"), ...)
htmlpage(genelist=list(ll, glist$ID),filename="estrogen_ANN.html", title="estrogen
effect", othertnames=data.frame(sym, glist[,-1]),table.head=c
("EntrezID","AffyID", "Symbol", colnames(glist)[-1]), repos=list("en", "affy"))
```



Project

```
#####ANOVA#####
```

```
#Now we can start analysing our data for biological effects. We set up a linear model
```

```
#with main effects for the level of estrogen (estrogen) and the time (time.h). Both are factors
```

```
#with 2 levels.
```

```
lm.coef = function(y) lm(y ~ factor(targets$estrogen) * factor(targets$time.h))
```

```
$coefficients
```

```
eff = esApply(esEset, 1, lm.coef)
```

```
eff[2,order(abs(eff[2,]), decreasing=T)[1:20]]->top20
```

```
topTable(fitC, n=nrow(fitC),coef=1, adjust="fdr")->allLimma
```

```
allLimma[match(names(top20),allLimma$ID),]
```



Project

ID	logFC	AveExpr	t	P.Value	adj.P.Val	B
910_at	3.113733	9.660238	23.59225	4.96E-09	3.13E-05	9.942522
39642_at	2.939428	7.876515	23.71715	4.74E-09	3.13E-05	9.96681
38827_at	2.9322	6.829773	9.110154	1.15E-05	4.26E-03	3.928051
31798_at	2.800195	12.11578	16.38509	1.03E-07	3.51E-04	7.97729
1884_s_at	2.799396	9.034796	12.05054	1.26E-06	1.06E-03	5.949749
1536_at	2.662258	5.937222	13.26247	5.80E-07	7.32E-04	6.610486
40117_at	2.555282	9.676557	15.6807	1.47E-07	3.58E-04	7.705093
1854_at	2.507616	8.532099	15.15848	1.95E-07	3.58E-04	7.490766

910_at	39642_at	38827_at	31798_at	1884_s_at	1536_at	40117_at	1854_at
3.113733	2.939428	2.9322	2.800195	2.799396	2.662258	2.555282	2.507616



result

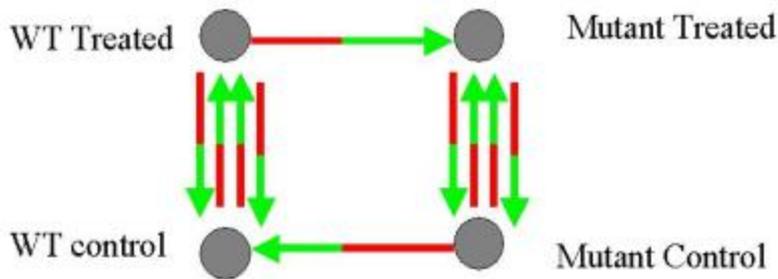
estrogen effect

EntrezID	AffyID	Symbol	logFC	AveExpr	t	P.Value	adj.P.Val	B
7083	910_at	TK1	3.86	9.66	29.21	0.00	0.00	11.61
7031	31798_at	TFF1	3.60	12.12	21.05	0.00	0.00	9.89
4605	1854_at	MYBL2	3.34	8.53	20.20	0.00	0.00	9.64
9768	38116_at	KIAA0101	3.76	9.51	16.86	0.00	0.00	8.48
3148	38065_at	HMGB2	2.99	9.10	16.21	0.00	0.00	8.21
7494	39755_at	XBP1	1.77	12.13	15.83	0.00	0.00	8.05
7153	1592_at	TOP2A	2.30	8.31	15.79	0.00	0.00	8.03
7083	41400_at	TK1	2.24	10.04	15.29	0.00	0.00	7.81
9052	33730_at	GPRC5A	-2.04	8.57	-15.14	0.00	0.00	7.74
11065	1651_at	UBE2C	2.97	10.50	14.78	0.00	0.00	7.57
991	38414_at	CDC20	2.02	9.46	14.59	0.00	0.00	7.48
890	1943_at	CCNA2	2.19	7.60	14.00	0.00	0.00	7.18
4175	40117_at	MCM6	2.28	9.68	13.97	0.00	0.00	7.17
332	40533_at	BIRC5	1.64	8.47	13.53	0.00	0.00	6.93
54898	39642_at	ELOVL2	1.61	7.88	13.03	0.00	0.00	6.65
6790	34851_at	AURKA	1.96	9.96	12.85	0.00	0.00	6.55
5111	1824_s_at	PCNA	1.64	9.24	12.76	0.00	0.00	6.50
11130	35995_at	ZWINT	2.76	8.87	12.68	0.00	0.00	6.46
27338	893_at	UBE2S	1.54	10.95	12.66	0.00	0.00	6.45
	40079_at		-2.41	8.23	-12.63	0.00	0.00	6.42



limma package

- Fitting of gene-wise linear models to estimate log ratios between two or more target samples simultaneously:
- **ebayes**: moderated t-statistics based on empirical Bayes



Chip	WTT	WTC	KOT	KOC
1	1	-1	0	0
2	-1	1	0	0
3	-1	1	0	0
4	1	-1	0	0
5	0	-1	0	1
6	0	0	1	-1
7	0	0	-1	1
8	0	0	-1	1
9	0	0	1	-1
10	1	0	-1	0
11	1	1	1	1

For example, the log ratios in above figure are each the difference of the log abundance in two of the four samples; we may construct a design matrix which specifies how the log ratios in all ten experiments are derived from the log abundances in the four chips; then the best estimate of the log abundances is obtained by solving the least squares problem for this design matrix.



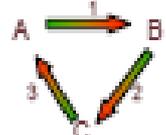
Matrix Multiplication



$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \beta = \begin{pmatrix} \beta \\ -\beta \end{pmatrix} \quad \beta = B - A$$



$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \beta_1 \\ -\beta_1 \\ \beta_1 + \beta_2 \end{pmatrix} \quad \begin{array}{l} \beta_1 = A - \text{Ref} \\ \beta_2 = B - A \end{array}$$



$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \beta_1 \\ -\beta_1 + \beta_2 \\ -\beta_2 \end{pmatrix} \quad \begin{array}{l} \beta_1 = B - A \\ \beta_2 = C - A \end{array}$$

Contrast $\beta_2 - \beta_1 \equiv C - B$

Linear Model Estimates

Obtain a linear model for each gene g

$$E(\underline{y}_g) = X \underline{\beta}_g \quad \text{var}(\underline{y}_g) = W_g^{-1} \sigma_g^2$$

Estimate model by **robust regression**, **least squares** or **generalized least squares** to get

coefficients

$$\hat{\beta}_{gj}$$

standard deviations

$$s_g$$

standard errors

$$\text{se}(\hat{\beta}_{gj})^2 = c_{gj} s_g^2$$

