

Bioconductor Tutorial

Statistical Methods and Software for the Analysis of DNA Microarray Data

Katie Pollard & Todd Lowe
University of California, Santa Cruz

Sandrine Dudoit
University of California, Berkeley

August 15, 2003

© Copyright 2003, all rights reserved



Acknowledgments

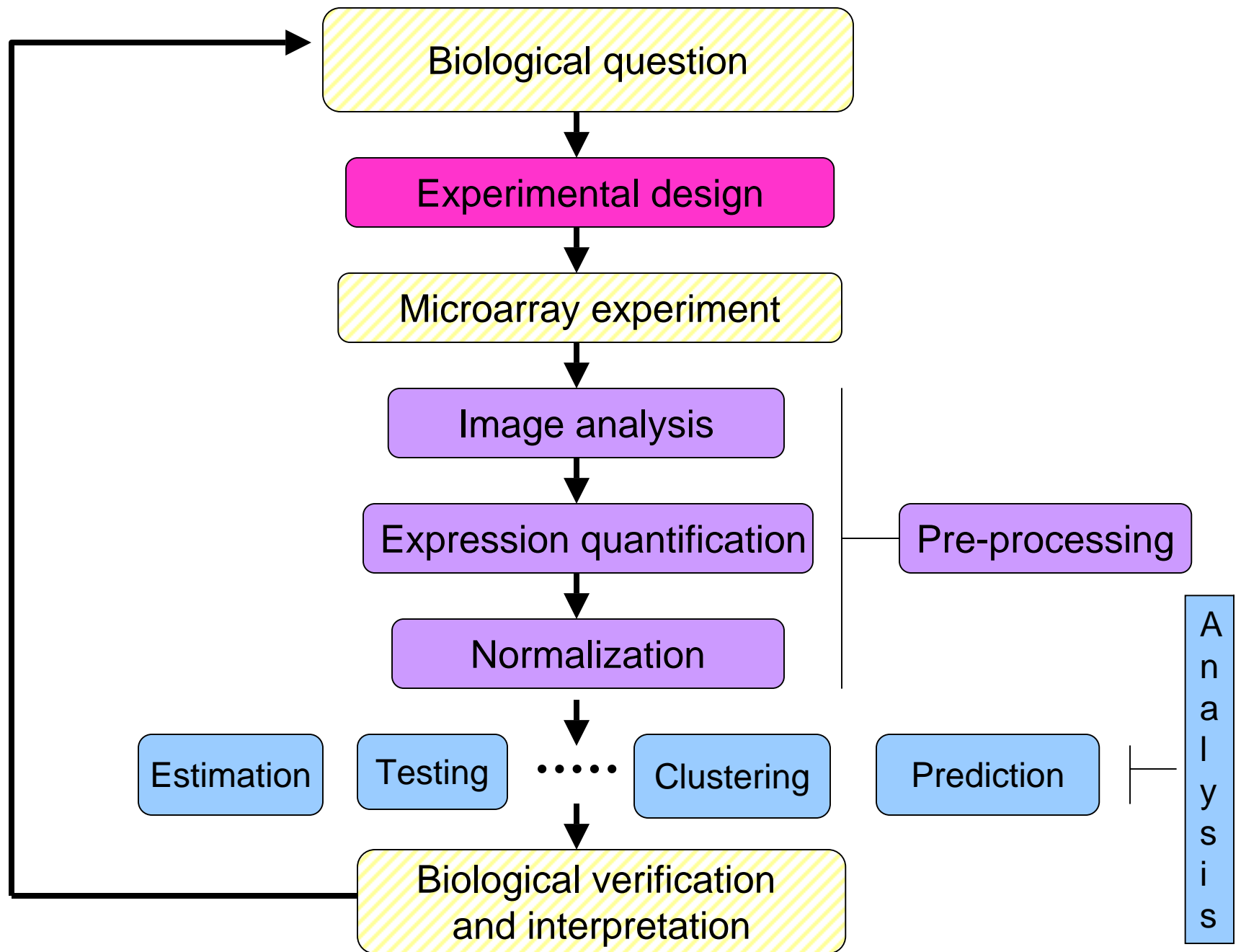
Course materials developed with

- **Robert Gentleman**, Biostatistics, Harvard.
- **Yee Hwa (Jean) Yang**, Biostatistics, UCSF.
- **Wolfgang Huber**, Department of Molecular Genome Analysis, German Cancer Research Center.

Acknowledgments

Bioconductor Core Team

- **Vince Carey**, Biostatistics, Harvard.
- **Yongchao Ge**, Statistics, UC Berkeley.
- **Robert Gentleman**, Biostatistics, Harvard.
- **Jeff Gentry**, Dana-Farber Cancer Institute.
- **Rafael Irizarry**, Biostatistics, Johns Hopkins.
- **Yee Hwa (Jean) Yang**, Biostatistics, UCSF.
- **Jianhua (John) Zhang**, Dana-Farber Cancer Institute.



Statistical computing

Everywhere ...

- Statistical design and analysis:
 - image analysis, normalization, estimation, testing, clustering, prediction, etc.
- Integration of experimental data with biological metadata from WWW-resources
 - gene annotation (GenBank, LocusLink);
 - literature (PubMed);
 - graphical (pathways, chromosome maps).

Outline

- Overview of the Bioconductor Project.
- Introduction to R Programming.
- Pre-processing two-color spotted microarray data
 - image analysis,
 - normalization.
- Differential gene expression.
- Clustering and classification.
- Annotation.
- Visualization.

References

- **Bioconductor** www.bioconductor.org
 - software, data, and documentation (vignettes);
 - training materials from short courses;
 - mailing list.
- **R** www.r-project.org
 - software; documentation; RNews.

Overview of the Bioconductor Project

Bioconductor

- **Bioconductor** is an open source and open development software project for the analysis and comprehension of biomedical and genomic data.
- Software, data, and documentation are available from www.bioconductor.org.

Bioconductor

- The project was started in the Fall of 2001 by Robert Gentleman, at the Biostatistics Unit of the Dana Farber Cancer Institute.
- R and the R package system are used to design and distribute software (www.r-project.org).
- There are currently 22 core developers, at various institutions in the US and Europe.
- Releases:
 - v 1.0: May 2nd, 2002, 15 packages.
 - v 1.1: November 18th, 2002, 20 packages.
 - v 1.2: May 28th, 2003, 30 packages.
- ArrayAnalyzer: Commercial port of Bioconductor packages in S-Plus.

Bioconductor

- Mechanisms for facilitating the design and deployment of **portable**, **extensible**, and **scalable** software.
- Support for **interoperability** with software written in other languages.
- Tools for integrating **biological metadata** from the **WWW** in the analysis of **experimental metadata**.
- Access to a broad range of **statistical and numerical methods**.
- High-quality **visualization** and **graphics** tools that support interactivity.
- An effective, extensible **user interface**.
- Tools for producing innovative, high-quality **documentation** and **training** materials.
- Methodology that supports the **creation**, **testing**, and **distribution** of software and data modules.

Bioconductor

There are two main classes of packages

- **End-user packages:**
 - aimed at users unfamiliar with R or computer programming;
 - polished and easy to use interfaces to a wide variety of computational and statistical methods for the analysis of genomic data.
- **Developer packages:** aimed at software developers, in the sense that they provide *software to write software*.

Bioconductor packages

Release 1.2, May 28th, 2003

- General infrastructure:
`Biobase`, `DynDoc`, `reposTools`, `rhdf5`, `ruuid`, `tkWidgets`,
`widgetTools`,.
- Annotation:
`annotate`, `AnnBuilder` → data packages.
- Graphics:
`geneplotter`, `hexbin`.
- Pre-processing for Affymetrix oligonucleotide chip data:
`affy`, `affycomp`, `affydata`, `makecdfenv`, `vsn`.
- Pre-processing for spotted DNA microarray data:
`limma`, `marrayClasses`, `marrayInput`, `marrayNorm`, `marrayPlots`,
`marrayTools`, `vsn`.
- Differential gene expression:
`eddi`, `genefilter`, `limma`, `multtest`, `ROC`.
- Graphs:
`graph`, `RBGL`, `Rgraphviz`.
- SAGE: `SAGElyzer`.

Ongoing efforts

- Variable (feature) selection;
 - Prediction;
 - Cluster analysis;
 - Cross-validation;
 - Multiple testing;
 - Quality measures for microarray data;
 - Interactions with MAGE-ML: new **MAGEML** package (Durinck & Allemeersch);
 - Biological sequence analysis;
 - Etc.
- Many methods already available in R.

Bioconductor

- Scenario:
 - Pre-processing of spotted array data with `marrayNorm`.
 - List of differentially expressed genes from `multtest`, `limma`, or `genefilter`.
 - Use the `annotate` package
 - to retrieve and search [PubMed abstracts](#) for these genes;
 - to generate an [HTML report](#) with links to [LocusLink](#) for each gene.

Widgets

- **Widgets**. Small-scale graphical user interfaces (GUI), providing point & click access for specific tasks.
- Packages: **tkWidgets**, **widgetTools**.
- E.g. File browsing and selection for data input, basic analyses:
tkWidgets: `dataViewer`, `fileBrowser`, `fileWizard`, `importWizard`, `objectBrowser`.

Data

- Issues:
 - complexity;
 - size;
 - evolution.
- We distinguish between **biological metadata** and **experimental metadata**.

Experimental metadata

- Gene expression measures
 - scanned images, i.e., raw data;
 - image quantitation data, i.e., output from image analysis;
 - normalized expression measures, i.e., log ratios M or Affy measures.
- Reliability information for the expression measures.
- Information on the probe sequences printed on the arrays (array layout).
- Information on the target samples hybridized to the arrays.
- See [Minimum Information About a Microarray Experiment – MIAME](#) – standards and new **MAGEML** package.

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 sources are large, complex, evolving rapidly, and typically distributed via the WWW.
- Cf. **annotate** and **AnnBuilder** packages.

Object-oriented programming

- The Bioconductor project has adopted the **object-oriented programming – OOP** – paradigm presented in J. M. Chambers (1998). *Programming with Data*.
- This object-oriented **class/method design** allows efficient representation and manipulation of large and complex biological datasets of multiple types (cf. MIAME standards).
- Tools for programming using the class/method mechanism are provided in the R **methods** package.
- Tutorial: www.omegahat.org/RSMMethods/index.html

OOP

- A **class** provides a software abstraction of a real world object. It reflects how we think of certain objects and what information these objects should contain.
- Classes are defined in terms of **slots** which contain the relevant data.
- An object is an **instance** of a class.
- A class defines the structure, inheritance, and initialization of objects.

OOP

- A **method** is a function that performs an action on data (objects).
- Methods define how a particular function should behave depending on the class of its arguments.
- Methods allow computations to be adapted to particular data types, i.e., classes.
- A **generic function** is a dispatcher, it examines its arguments and determines the appropriate method to invoke.
- Examples of generic functions include `plot`, `summary`, `print`.

marrayRaw class

Pre-normalization intensity data for a batch of arrays

maRf

maGf

Matrix of red and green foreground intensities

maRb

maGb

Matrix of red and green background intensities

maW

Matrix of spot quality weights

maLayout

Array layout parameters - `marrayLayout`

maGnames

Description of spotted probe sequences
- `marrayInfo`

maTargets

Description of target samples - `marrayInfo`

maNotes

Any notes

AffyBatch class

Probe-level intensity data for a batch of arrays (same CDF)

`cdfName`

Name of CDF file for arrays in the batch

`nrow`

`ncol`

Dimensions of the array

`exprs`

`se.exprs`

Matrices of probe-level intensities and SEs
rows → probe cells, columns → arrays.

`phenoData`

Sample level covariates, instance of class `phenoData`

`annotation`

Name of annotation data

`description`

MIAME information

`notes`

Any notes

exprSet class

exprs

Matrix of expression measures, genes x samples

se.exprs

Matrix of SEs for expression measures, genes x samples

phenoData

Sample level covariates, instance of class **phenoData**

annotation

Name of annotation data

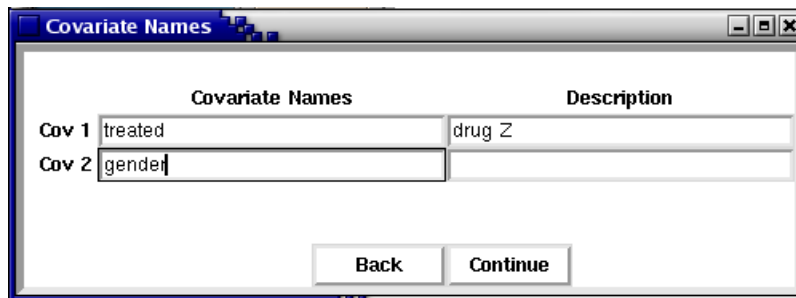
description

MIAME information

notes

Any notes

Reading in phenoData

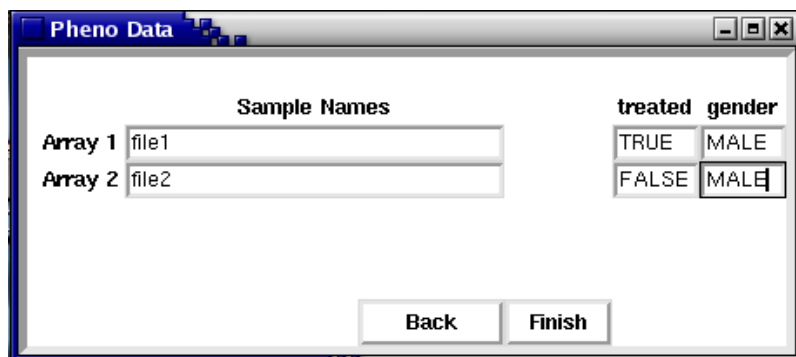


A dialog box titled "Covariate Names" with a table for entering covariate information. The table has two columns: "Covariate Names" and "Description".

	Covariate Names	Description
Cov 1	treated	drug Z
Cov 2	gender	

Buttons: Back, Continue

tkSampleNames

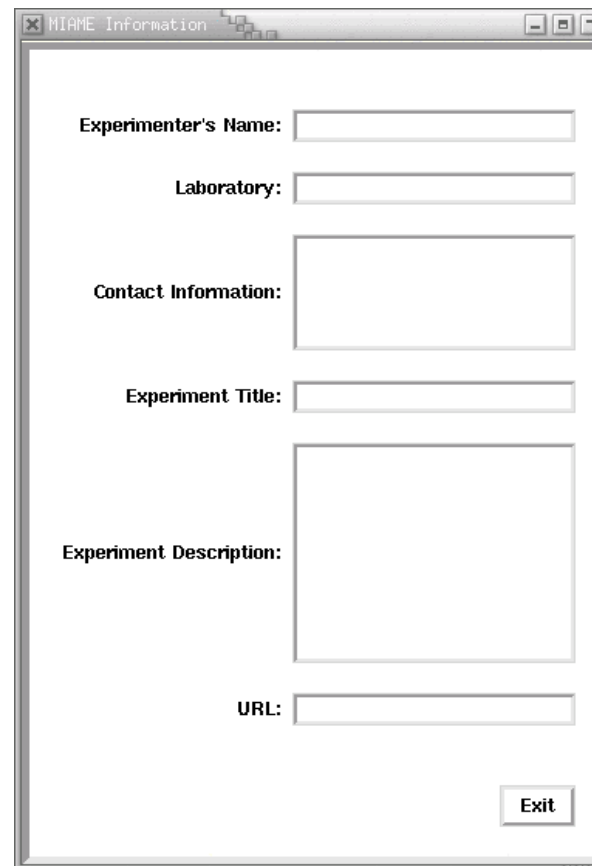


A dialog box titled "Pheno Data" with a table for entering sample information. The table has two columns: "Sample Names" and two sub-columns for "treated" and "gender".

	Sample Names	treated	gender
Array 1	file1	TRUE	MALE
Array 2	file2	FALSE	MALE

Buttons: Back, Finish

tkphenoData



A dialog box titled "MIAME Information" with several input fields for experimental metadata.

Fields:

- Experimenter's Name:
- Laboratory:
- Contact Information:
- Experiment Title:
- Experiment Description:
- URL:

Buttons: Exit

tkMIAME

Documentation

Extensive documentation and training resources for R and Bioconductor are available on the WWW.

- [R manuals and tutorials](#) are available from the R website.
- [R help system](#)
 - detailed on-line documentation, available in text, HTML, PDF, and LaTeX formats;
 - e.g. `help(genefilter)`, `?pubmed`.
- [R demo system](#)
 - user-friendly interface for running demonstrations of R scripts;
 - e.g. `demo(marrayPlots)`, `demo(affy)`.
- [Bioconductor short courses](#)
 - modular training segments on software and statistical methodology;
 - lectures and computer labs available on WWW for self-instruction.

Vignettes

- Bioconductor has adopted a new documentation paradigm, the vignette.
- A **vignette** is an **executable document** consisting of a collection of documentation text and code chunks.
- Vignettes form **dynamic, integrated, and reproducible statistical documents** that can be automatically updated if either data or analyses are changed.
- Vignettes can be generated using the **Sweave** function from the R **tools** package.

Vignettes

- Each Bioconductor package contains at least one vignette, located in the `doc` subdirectory of an installed package and accessible from the help browser.
- Vignettes provide task-oriented descriptions of the package's functionality and can be used interactively.
- Vignettes are available separately from the Bioconductor website or as part of the packages.

Vignettes

- Tools are being developed for managing and using this repository of step-by-step tutorials
 - **Biobase**: `openVignette` – Menu of available vignettes and interface for viewing vignettes (PDF).
 - **tkWidgets**: `vExplorer` – Interactive use of vignettes.
 - **reposTools**.

An Introduction to Programming in the R Language

adapted from

**Course in Practical Microarray
Analysis**

Heidelberg 23.-27.9.2002

Wolfgang Huber

R as a Calculator

```
> log2(32)
```

```
[1] 5
```

```
> sqrt(2)
```

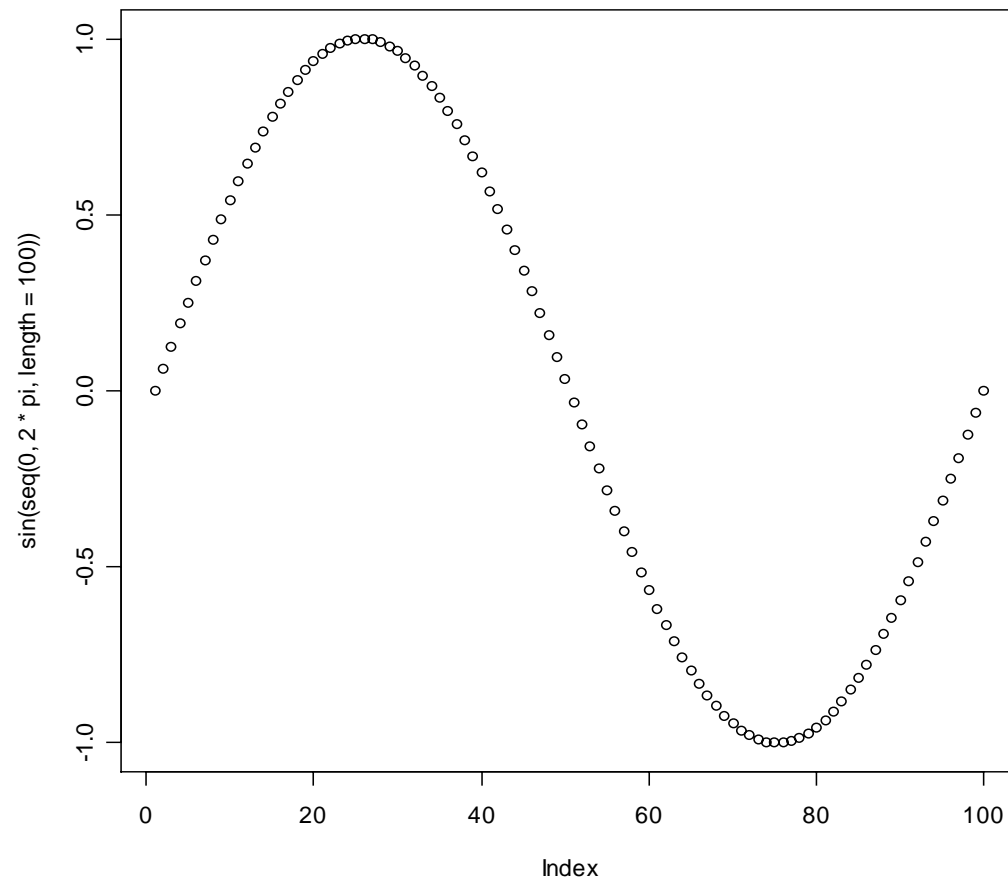
```
[1] 1.414214
```

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

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


R as a Graphics Tool

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



Variables

```
> a = 49
```

numeric

```
> sqrt(a)
```

```
[1] 7
```

```
> a = "The dog ate my homework"
```

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

**character
string**

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

```
> a = (1+1==3)
```

```
> a
```

logical

```
[1] FALSE
```

Missing Values

Variables of each data type (numeric, character, logical) can also take the value **NA**: not available.

- NA is not the same as 0
- NA is not the same as ""
- NA is not the same as FALSE

Any operations (calculations, comparisons) that involve NA may or may not produce NA:

```
> NA==1
[1] NA
> 1+NA
[1] NA
> max(c(NA, 4, 7))
[1] NA
> max(c(NA, 4, 7), na.rm=T)
[1] 7
```

```
> NA | TRUE
[1] TRUE
> NA & TRUE
[1] NA
```

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)  
}
```

Operators: Short-cut writing for frequently used functions of one or two arguments.

E.g.: + - * / ! & | %%

Vectors

vector: an ordered collection of data of the same type

```
> a = c(1, 2, 3)
```

```
> a*2
```

```
[1] 2 4 6
```

Example: the mean spot intensities of all 15488 spots on a chip: a vector of 15488 numbers

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: a rectangular table of data of the same type

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

array: 3-,4-,...dimensional matrix

Example: the red and green foreground and background values for 20000 spots on 120 chips: a 4 x 20000 x 120 (3D) array.

Lists

list: an ordered collection of data of arbitrary types.

Example:

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

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

Data Frames

data frame: is supposed to represent the typical data table that researchers come up with – like a spreadsheet.

It is a 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.

Example:

```
> a
```

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

Subsetting

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

```
> a
      localisation 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", ]
      localisation tumorsize progress
XX987      proximal      10        0
```

Example:

```
> a
      localisation tumorsize progress
XX348      proximal      6.3         0
XX234      distal       8.0         1
XX987      proximal     10.0         0
```

subset rows by a
vector of indices

```
> a[c(1,3),]
      localisation tumorsize progress
XX348      proximal      6.3         0
XX987      proximal     10.0         0
```

subset rows by a
logical vector

```
> a[c(T,F,T),]
      localisation tumorsize progress
XX348      proximal      6.3         0
XX987      proximal     10.0         0
```

subset a column

```
> a$localisation
[1] "proximal" "distal"  "proximal"
```

comparison resulting
in logical vector

```
> a$localisation=="proximal"
[1] TRUE FALSE TRUE
```

subset the
selected rows

```
> a[ a$localisation=="proximal", ]
      localisation tumorsize progress
XX348      proximal      6.3         0
XX987      proximal     10.0         0
```

Branching

```
if (logical expression) {  
    statements  
}  
else {  
    alternative statements  
}
```

else branch is optional

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)  
}
```

Regular Expressions

A tool for text matching and replacement which is available in similar forms in many programming languages (Perl, Unix shells, Java)

```
> a = c("CENP-F", "Ly-9", "MLN50", "ZNF191", "CLH-17")
```

```
> grep("L", a)
```

```
[1] 2 3 5
```

```
> grep("L", a, value=T)
```

```
[1] "Ly-9"      "MLN50"     "CLH-17"
```

```
> grep("^L", a, value=T)
```

```
[1] "Ly-9"
```

```
> grep("[0-9]", a, value=T)
```

```
[1] "Ly-9"      "MLN50"     "ZNF191"    "CLH-17"
```

```
> gsub("[0-9]", "X", a)
```

```
[1] "CENP-F"    "Ly-X"      "MLNXX"     "ZNFXXX"    "CLH-XX"
```

Storing Data

Every R object can be stored into and restored from a file with the commands “save” and “load”.

This uses the XDR (external data representation) standard of Sun Microsystems and others, and is portable between MS-Windows, Unix, Mac.

```
> save(x, file="x.Rdata")  
> load("x.Rdata")
```

Importing and Exporting Data

There are many ways to get data into R and out of R.

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
```

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

Importing Data: caveats

Type conversions: by default, the read functions try to guess and autoconvert the data types of the different columns (e.g. number, factor, character). There are options `as.is` and `colClasses` to control this – *read the online help*

Special characters: the delimiter character (space, comma, tabulator) and the end-of-line character cannot be part of a data field. To circumvent this, text may be “quoted”. However, if this option is used (the default), then the quote characters themselves cannot be part of a data field. Except if they themselves are within quotes...

Understand the conventions your input files use and set the quote options accordingly.

Getting Help

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

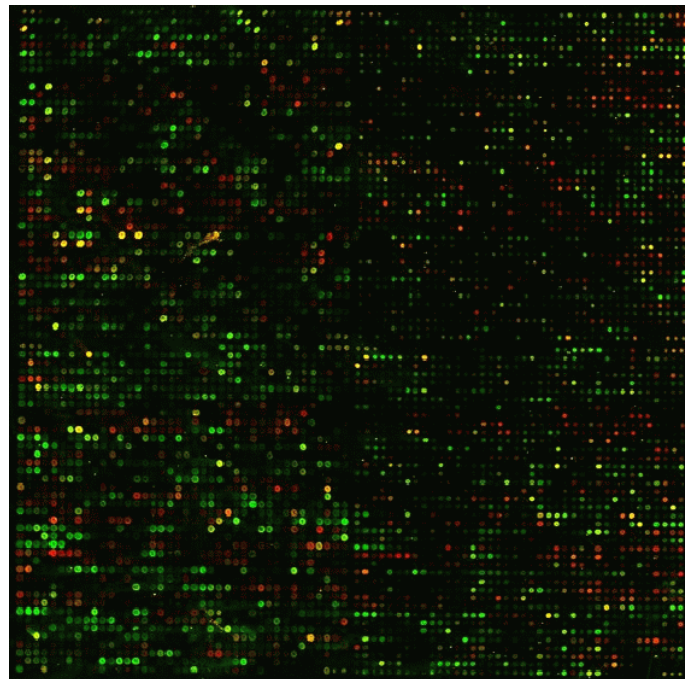
```
>? t.test
```

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

- **HTML search engine lets you search for topics with regular expressions:**

```
>help.search
```

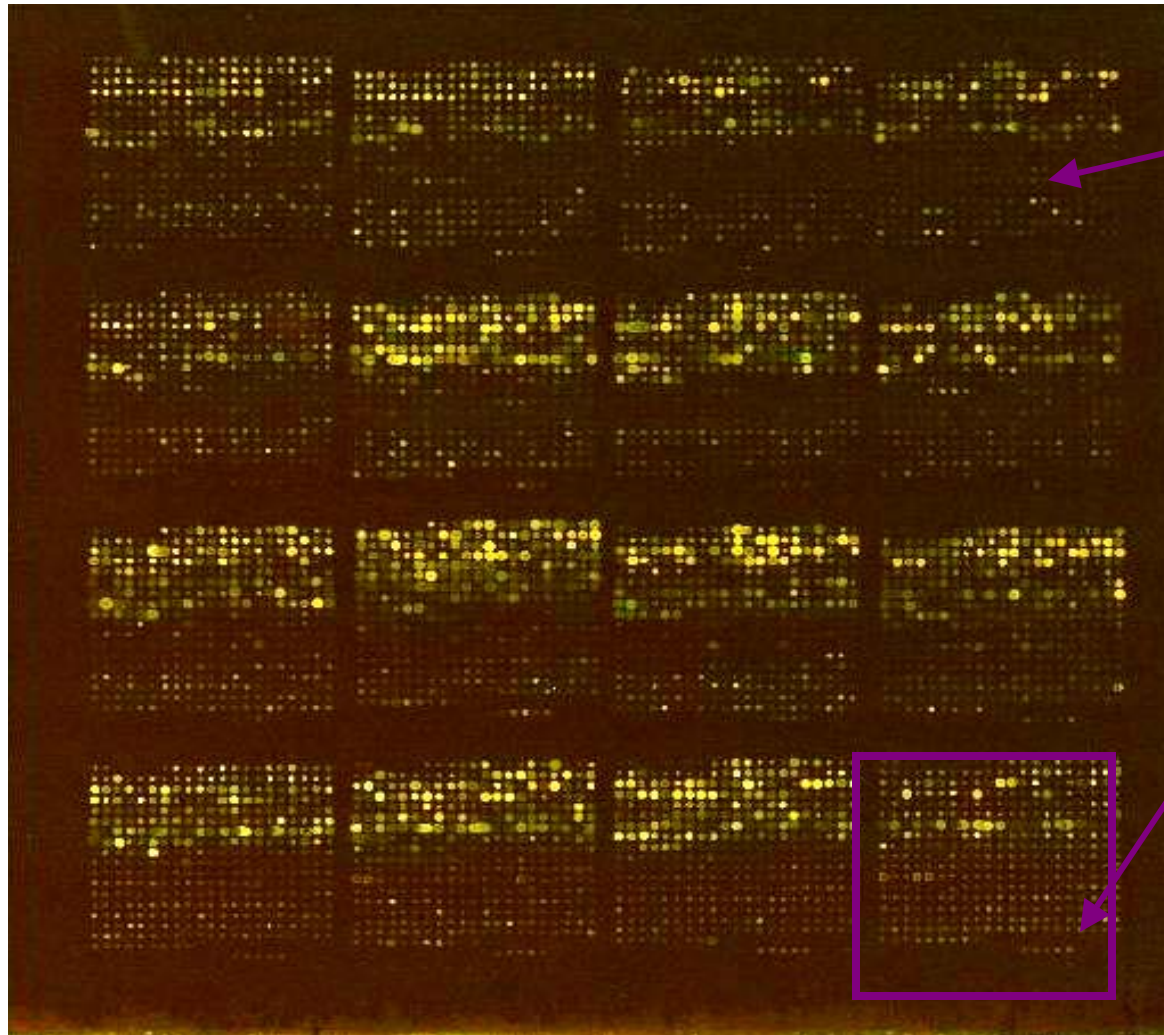
Pre-processing Two-color Spotted Microarray Data



Terminology

- **Target:** DNA hybridized to the array, mobile substrate.
- **Probe:** DNA spotted on the array, aka. spot, immobile substrate.
- **Sector:** collection of spots printed using the same print-tip (or pin), aka. **print-tip-group**, pin-group, spot matrix, grid.
- The terms **slide** and **array** are often used to refer to the printed microarray.
- **Batch:** collection of microarrays with the same probe layout.
- **Cy3 = Cyanine 3 = green dye.**
- **Cy5 = Cyanine 5 = red dye.**

RGB overlay of Cy3 and Cy5 images



Probe

4 x 4 sectors
19 x 21 probes/sector
6,384 probes/array

Sector

Raw data

- Pairs of 16-bit TIFFs, one for each dye.
- E.g. Human cDNA arrays:
 - ~43K spots;
 - ~ 20Mb per channel;
 - ~ 2,000 x 5,500 pixels per image;
 - spot separation: ~ 136um.
- For a “typical” array, the spot area has
 - mean = 43 pixels,
 - med = 32 pixels,
 - SD = 26 pixels.

Image analysis

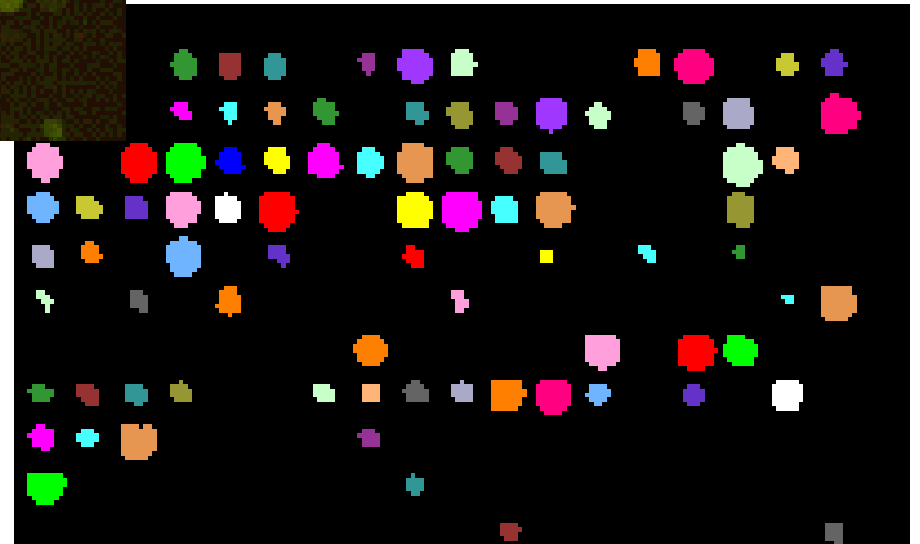
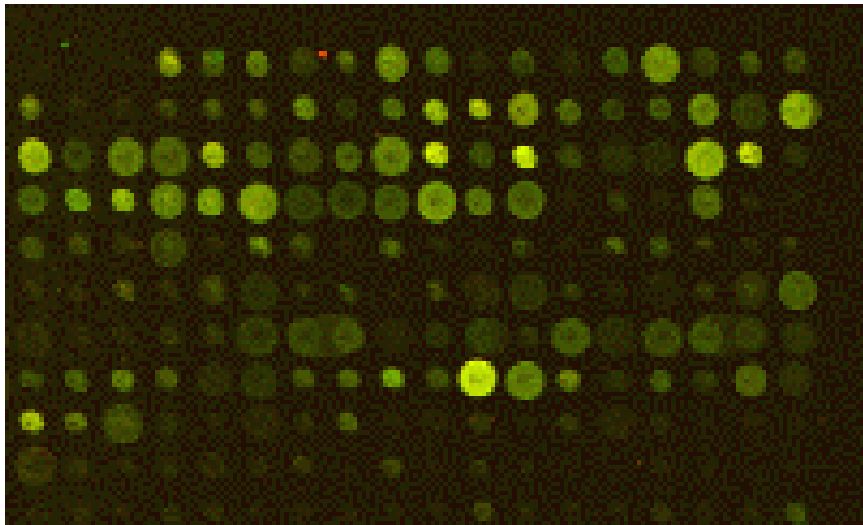
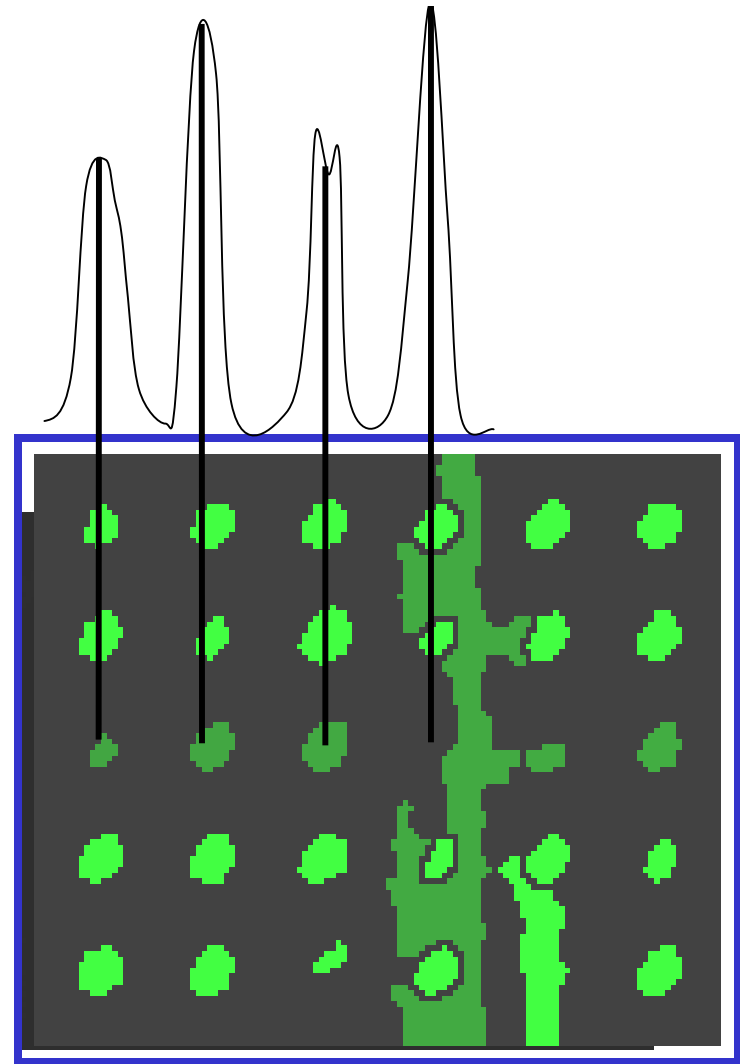


Image analysis

- The **raw data** from a cDNA microarray experiment consist of pairs of **image files**, 16-bit TIFFs, one for each of the dyes.
- **Image analysis** is required to extract measures of the red and green fluorescence intensities, **R** and **G**, for each spot on the array.

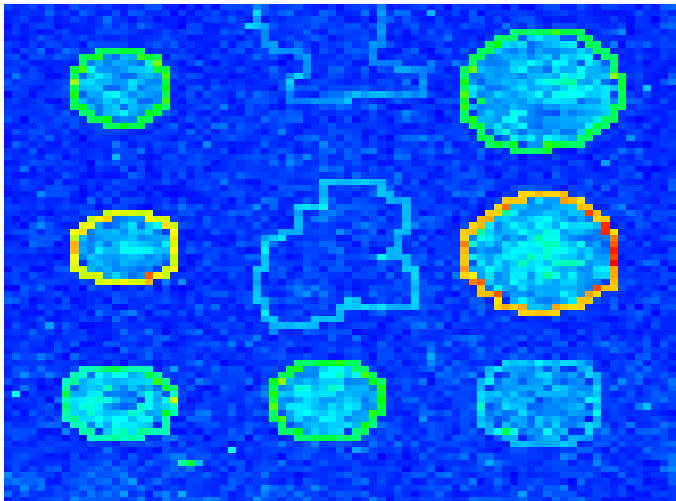
Image analysis

- 1. Addressing.** Estimate location of spot centers.
- 2. Segmentation.** Classify pixels as foreground (signal) or background.
- 3. Information extraction.** For each spot on the array and each dye
 - foreground intensities;
 - background intensities;
 - quality measures.

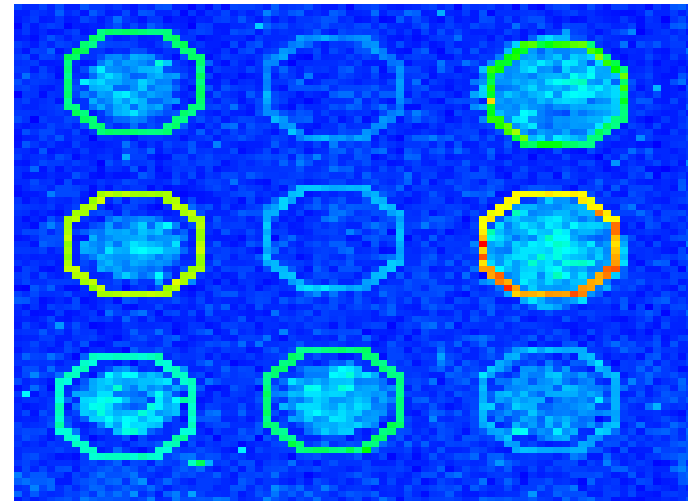


R and G for each spot on the array.

Segmentation



Adaptive segmentation, SRG



Fixed circle segmentation

Spots usually vary in size and shape.

Seeded region growing

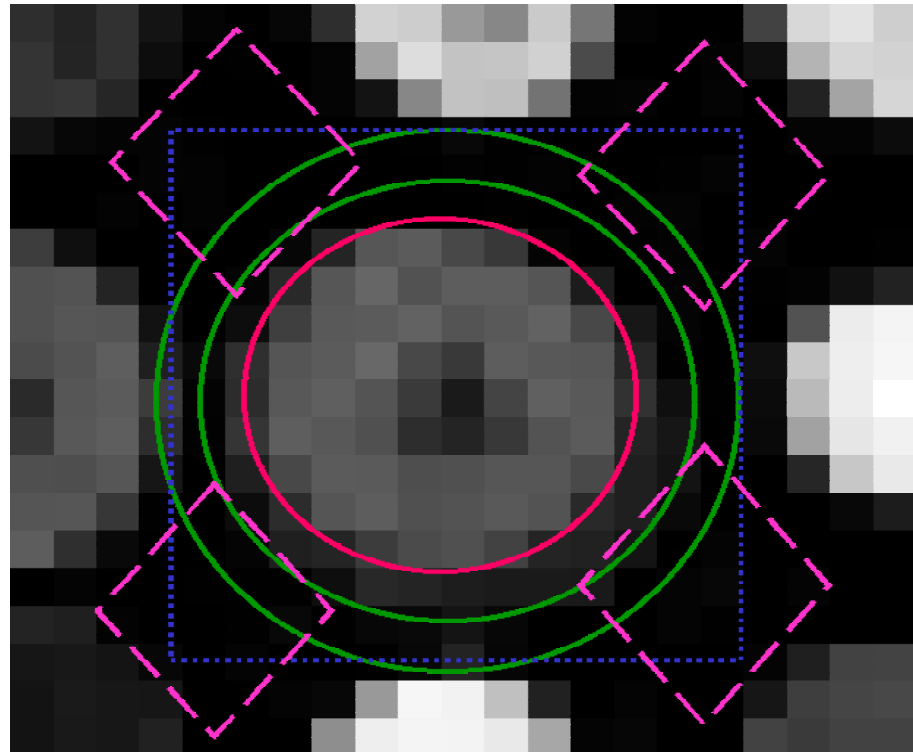
- **Adaptive** segmentation method.
- Requires the input of **seeds**, either individual pixels or groups of pixels, which control the formation of the regions into which the image will be segmented.

Here, based on fitted foreground and background **grids** from the addressing step.

- The decision to add a pixel to a region is based on the absolute gray-level difference of that pixel's intensity and the average of the pixel values in the neighboring region.
- Done on combined red and green images.
- Ref. Adams & Bischof (1994)

Local background

- GenePix
- QuantArray
- ScanAnalyze

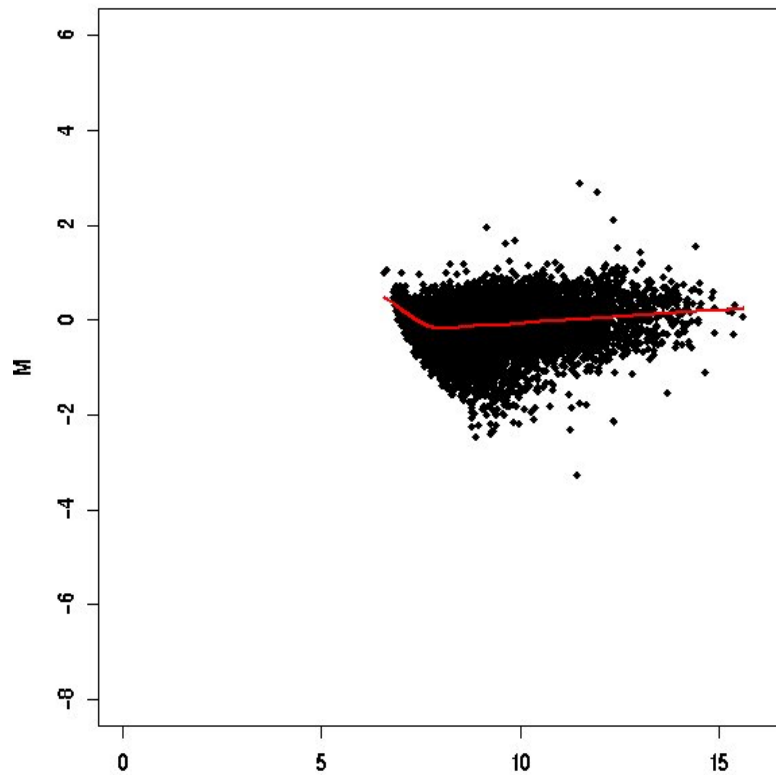


Morphological opening

- The image is probed with a **structuring element**, here, a square with side length about twice the spot-to-spot distance.
- **Erosion** (**Dilation**): the eroded (dilated) value at a pixel x is the **minimum** (**maximum**) value of the image in the window defined by the structuring element when its origin is at x .
- **Morphological opening**: **erosion** followed by **dilation**.
- Done separately for the red and green images.
- Produces an image of the estimated background for the entire slide.

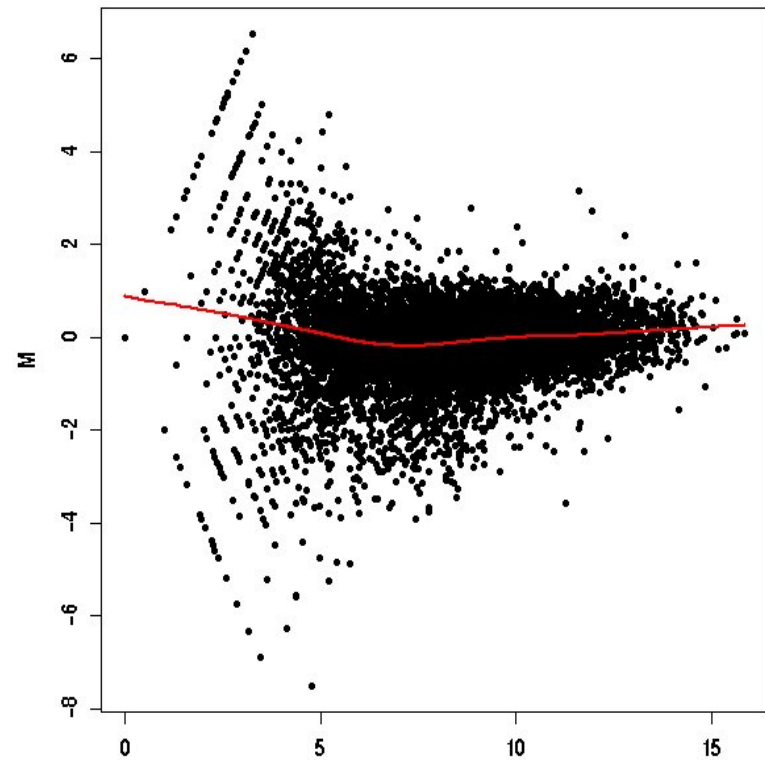
Background matters

Morphological opening



$$M = \log_2^A R - \log_2 G$$

Local background



$$\text{vs. } A = (\log_2^A R + \log_2 G)/2$$

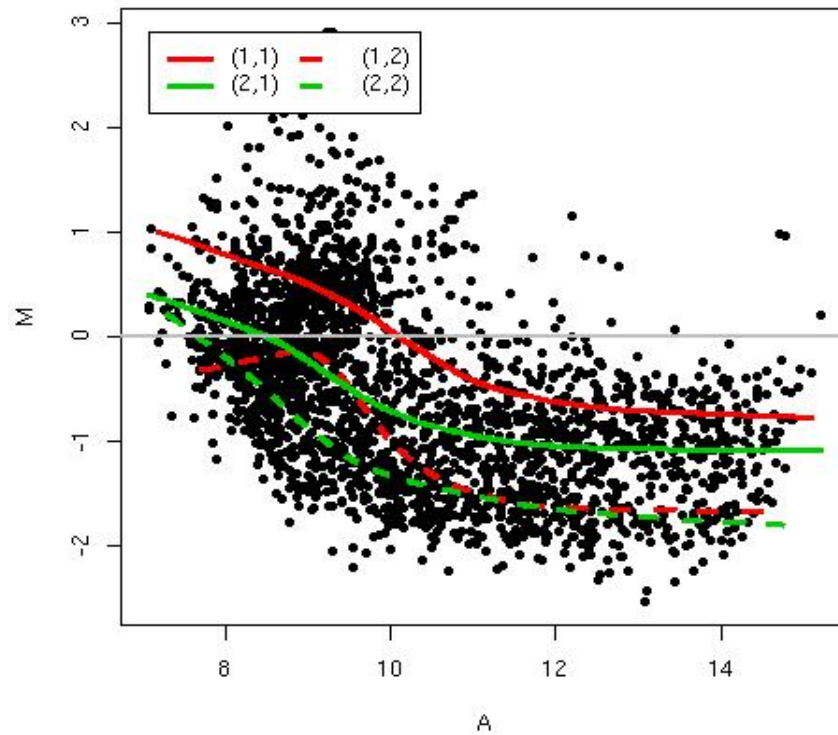
Quality measures

- **Spot quality**
 - **Brightness:** foreground/background ratio;
 - **Uniformity:** variation in pixel intensities and ratios of intensities within a spot;
 - **Morphology:** area, perimeter, circularity.
- **Slide quality**
 - Percentage of spots with no signal;
 - Range of intensities;
 - Distribution of spot signal area, etc.
- How to use quality measures in subsequent analyses?

Spot image analysis software

- Software package **Spot**, built on the **R** language and environment for statistical computing and graphics.
- Batch automatic addressing.
- Segmentation. **Seeded region growing** (Adams & Bischof 1994): **adaptive** segmentation method, no restriction on the size or shape of the spots.
- Information extraction
 - Foreground. Mean of pixel intensities within a spot.
 - Background. **Morphological opening**: non-linear filter which generates an image of the estimated background intensity for the entire slide.
- Spot quality measures.

Normalization



Normalization

- After image processing, we have measures of the red and green fluorescence intensities, **R** and **G**, for each spot on the array.
- **Normalization** is needed to ensure that differences in intensities are indeed due to differential expression, and not some printing, hybridization, or scanning artifact.
- Normalization is necessary before any analysis which involves within or between slides comparisons of intensities, e.g., clustering, testing.

Normalization

- Identify and remove the effects of **systematic variation** in the measured fluorescence intensities, other than differential expression, for example
 - different labeling efficiencies of the dyes;
 - different amounts of Cy3- and Cy5-labeled mRNA;
 - different scanning parameters;
 - print-tip, spatial, time-of-printing, or plate effects, etc.

Normalization

- **Within-slide normalization**: Correct for systematic differences in intensities between co-hybridized samples.
 - **Location** normalization - additive on log-scale.
 - **Scale** normalization - multiplicative on log-scale.
 - **Which spots** to use?
 - **Paired-slides** (dye-swap experiments): self-normalization.
- **Between-slides normalization**: Correct for systematic differences in intensities between samples hybridized to different slides.

Normalization

- **Two-channel normalization:** normalization of log-ratios.
 - For analysis of **relative** expression levels, e.g., gene expressed at a higher level in target sample A than in sample B.
- **Single-channel normalization:** normalization of individual red and green log-intensities.
 - For analysis of **absolute** expression levels, e.g., testing for expression or lack thereof of certain genes in a target sample A.
 - Two-channel within-slide normalization followed by between-slides normalization, cf. normalization methods for Affymetrix data.

Normalization

- The need for normalization can be seen most clearly in **self-self hybridizations**, where the same mRNA sample is labeled with the Cy3 and Cy5 dyes.
- The imbalance in the red and green intensities is usually **not constant** across the spots within and between arrays, and can vary according to overall spot intensity, location, plate origin, etc.
- These factors should be considered in the normalization.

Single-slide data display

- Usually: R vs. G
 $\log_2 R$ vs. $\log_2 G$.

- Preferred

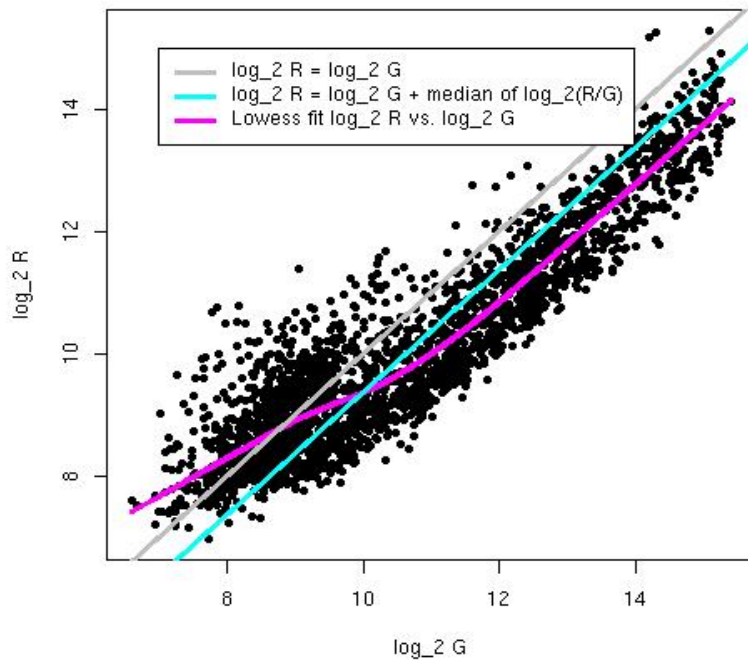
$$M = \log_2 R - \log_2 G$$

vs. $A = (\log_2 R + \log_2 G)/2$.

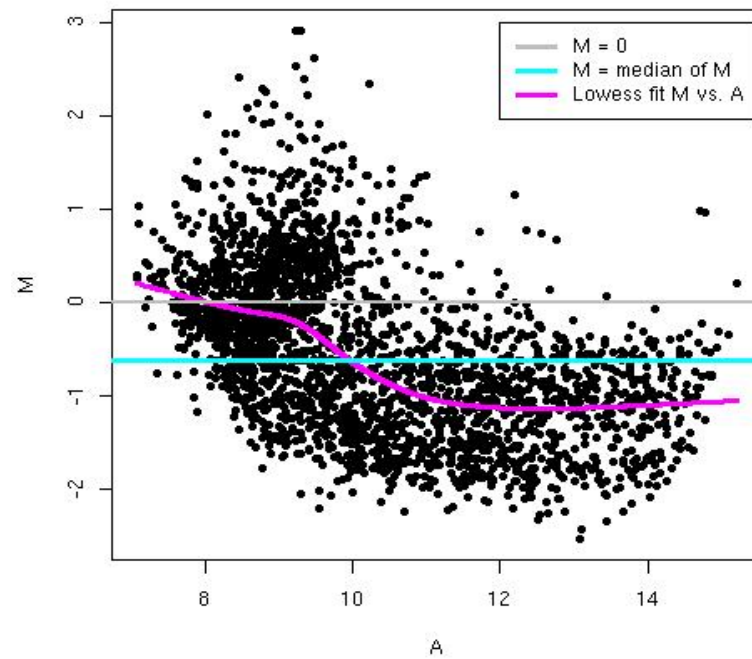
- An MA-plot amounts to a 45° counterclockwise rotation of a $\log_2 R$ vs. $\log_2 G$ plot followed by scaling.

Self-self hybridization

$\log_2 R$ vs. $\log_2 G$

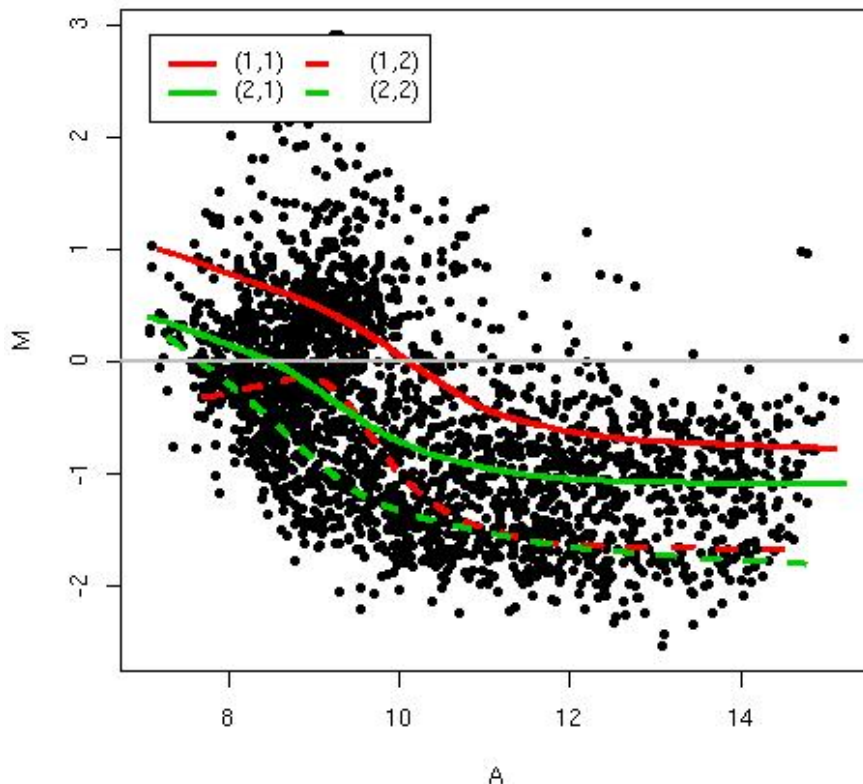


M vs. A



$$M = \log_2 R - \log_2 G, \quad A = (\log_2 R + \log_2 G)/2$$

Self-self hybridization



Robust local regression
within sectors
(print-tip-groups)
of intensity log-ratio M
on average log-intensity
 A .

$$M = \log_2 R - \log_2 G,$$
$$A = (\log_2 R + \log_2 G)/2$$

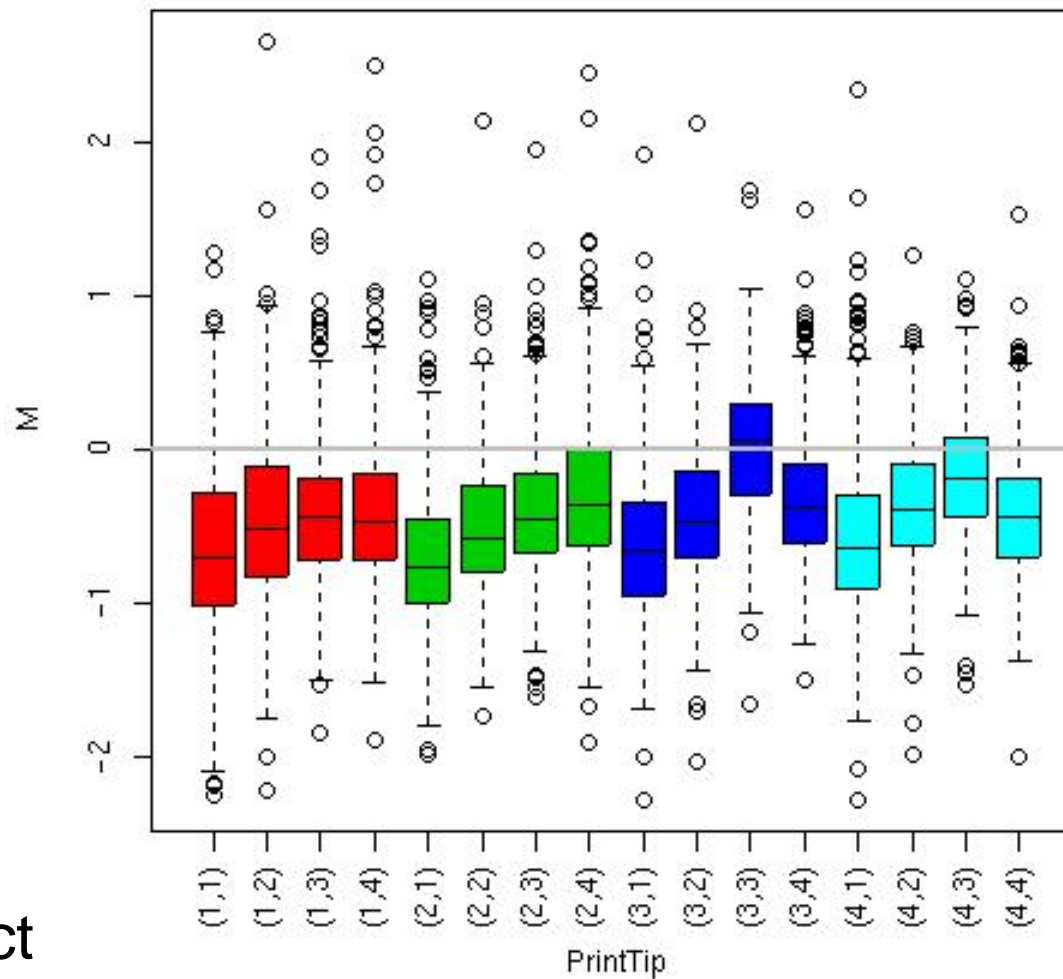
Diagnostic plots

- **RGB overlay** of Cy3 and Cy5 images.
- Diagnostics plots of spot statistics, e.g. red and green log-intensities, intensity log-ratios M , average log-intensities A , spot area
 - **Boxplots, dotplots**;
 - **2D spatial images**;
 - **Scatter-plots**, e.g., **MA-plots**;
 - **Histograms/density plots**.
- **Stratify** plots according to layout parameters, e.g., print-tip-group, plate, time-of-printing.

Boxplots by print-tip-group

Swirl 93 array: pre-normalization log-ratio M

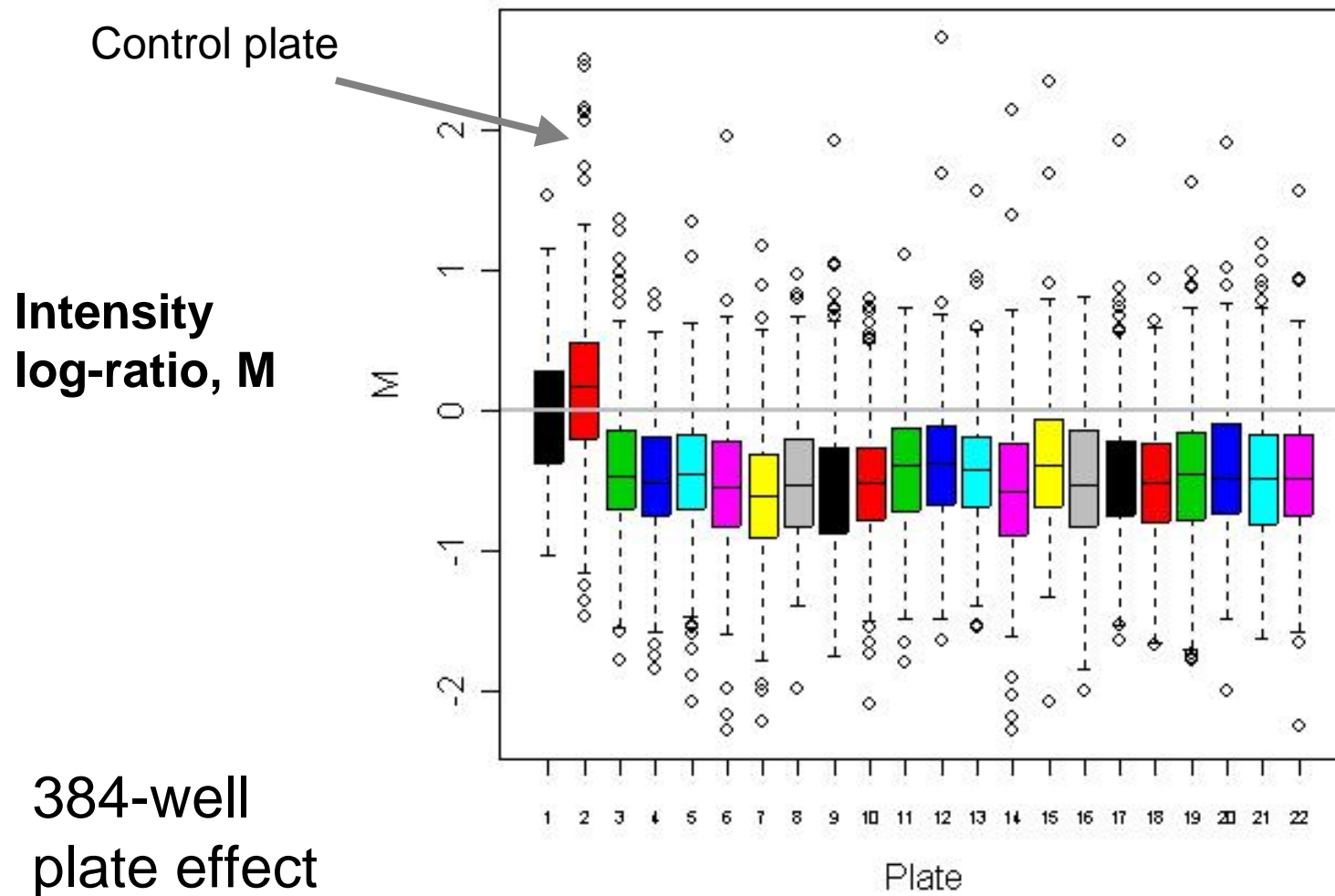
Intensity
log-ratio, M



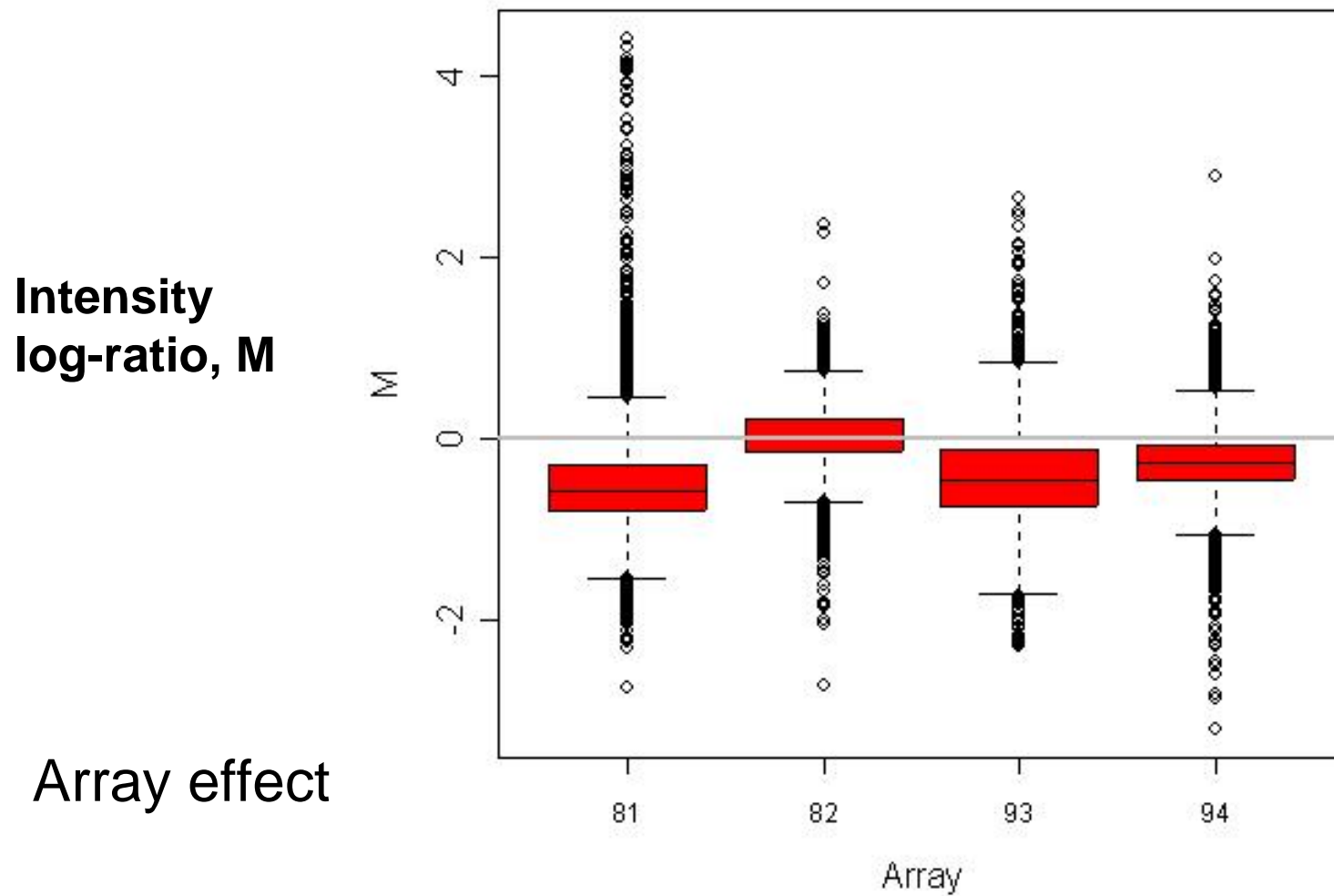
Print-tip effect

Boxplots by plate

Swirl 93 array: pre-normalization log-ratio M



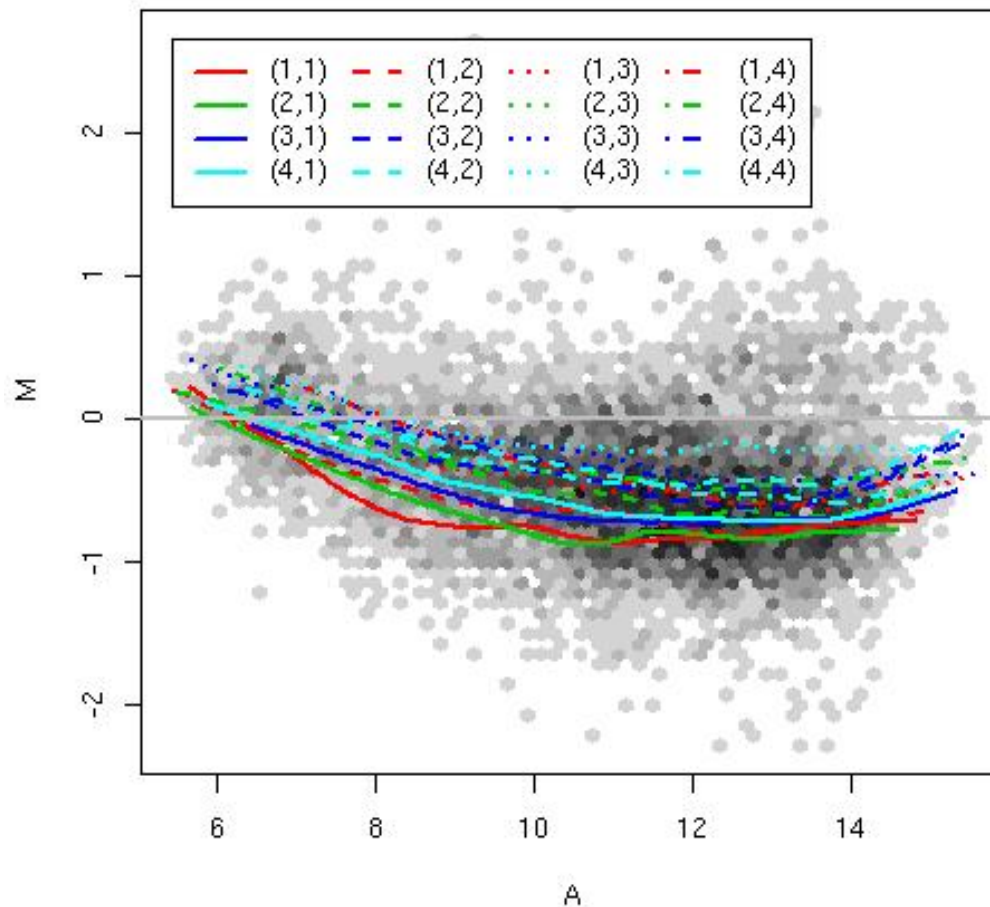
Boxplots by array



MA-plot by print-tip-group

Swirl 93 array: pre-normalization log ratio M

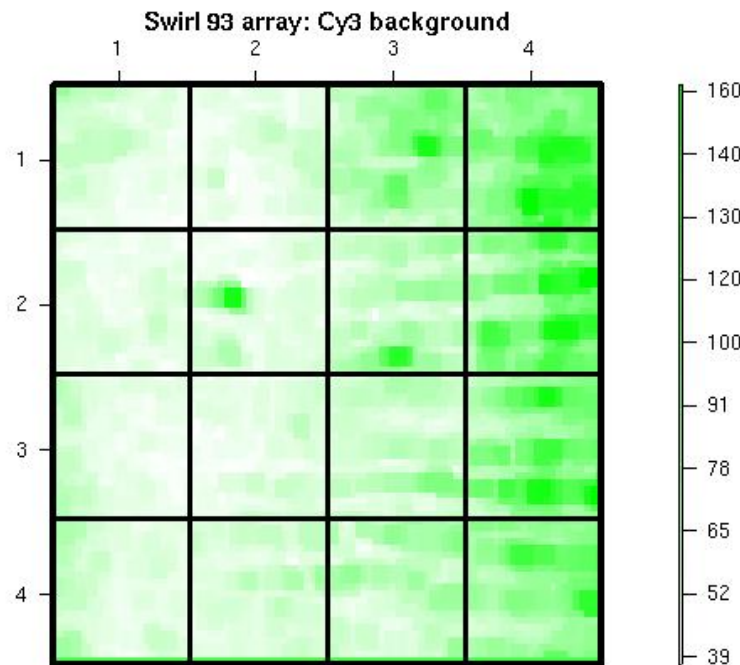
Intensity
log-ratio, M



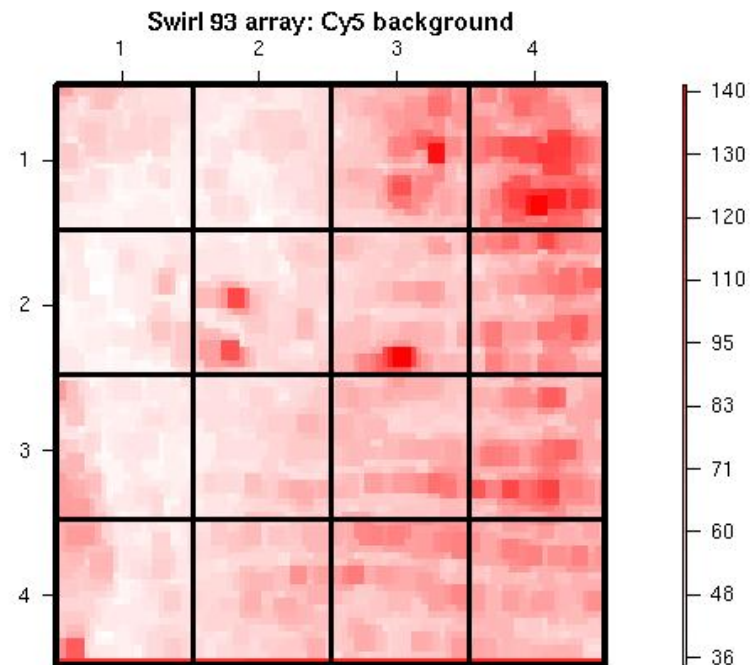
$$M = \log_2 R - \log_2 G,$$
$$A = (\log_2 R + \log_2 G)/2$$

Average
log-intensity, A

2D spatial images

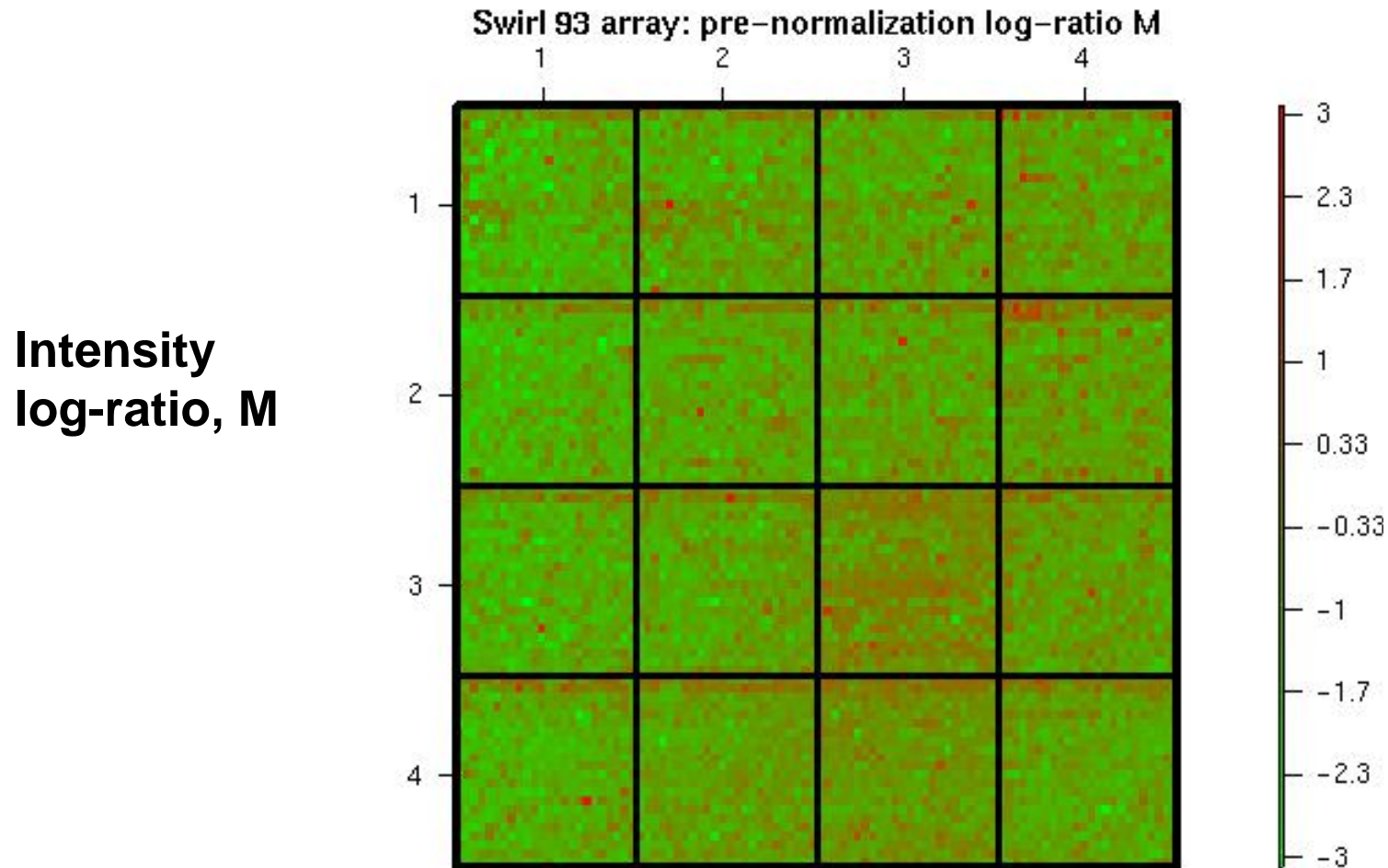


Cy3 background intensity



Cy5 background intensity

2D spatial images



Location normalization

$$\log_2 R/G \leftarrow \log_2 R/G - L(\text{intensity, sector, ...})$$

- **Constant normalization.** Normalization function L is **constant** across the spots.
E.g. mean or median of the log-ratios M , or $\text{sum } R/\text{sum } G$.
- **Adaptive normalization.** Normalization function L depends on a number of **explanatory variables**, such as spot intensity A , sector, plate origin.

Location normalization

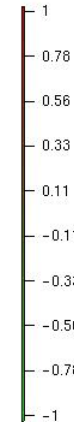
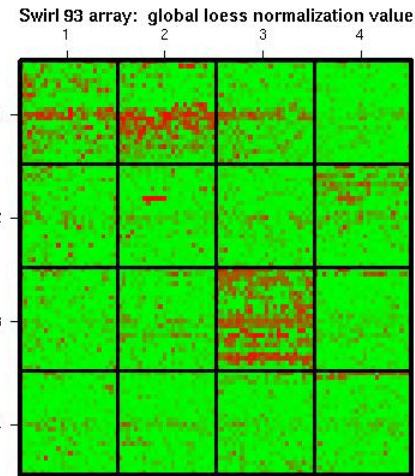
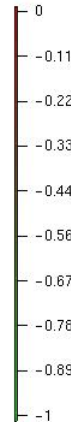
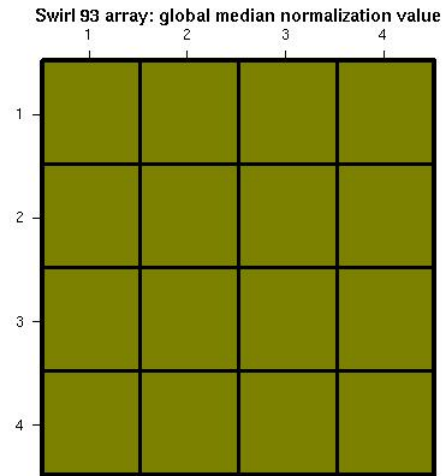
- The normalization function can be obtained by **robust locally weighted regression** of the log-ratios M on explanatory variables.
E.g. regression of M on A within sector.
- E.g. lowess (LOcally WEighted Scatterplot Smoothing) or loess (Cleveland, 1979; Cleveland & Devlin, 1988).

Location normalization

- **Intensity-dependent normalization.**
Regression of M on A (*global loess*).
- **Intensity and sector-dependent normalization.**
Same as above, for each sector separately (*within-print-tip-group loess*).
- **2D spatial normalization.**
Regression of M on 2D-coordinates.
- Other variables: time-of-printing, plate, etc.
- **Composite normalization.** Weighted average of several normalization functions.

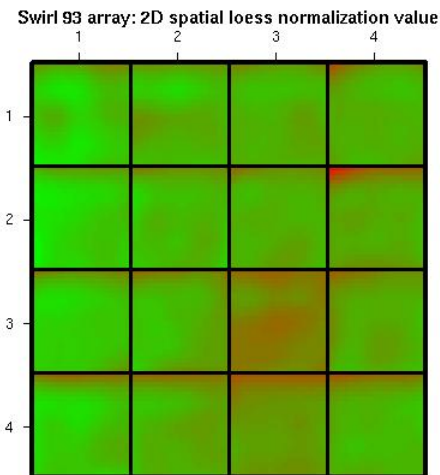
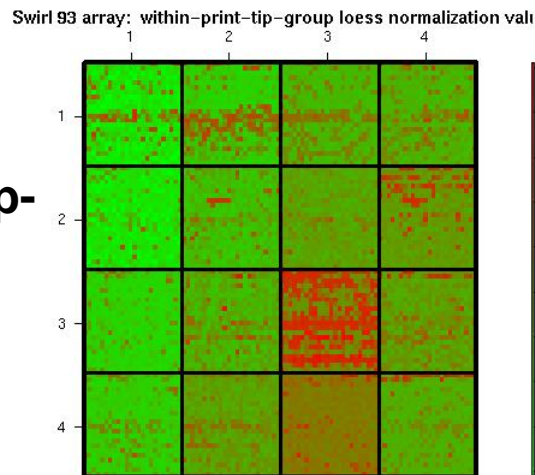
2D images of L values

Global median normalization



Global loess normalization

Within-print-tip-group loess normalization

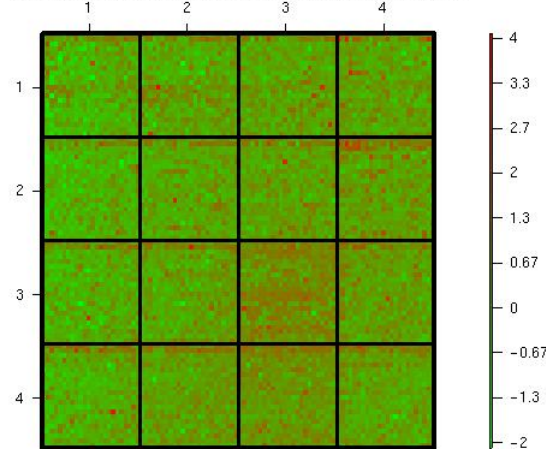


2D spatial normalization

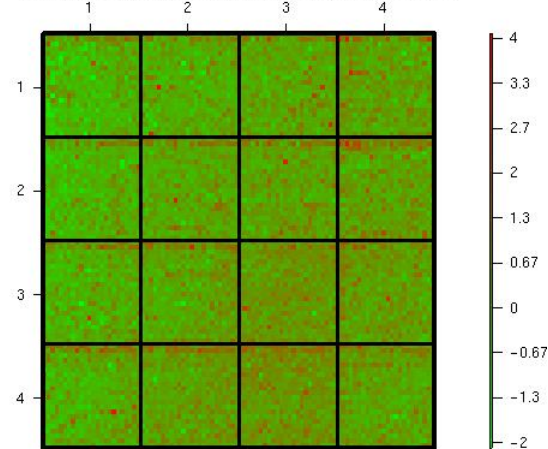
2D images of normalized M-L

Global median normalization

Swirl 93 array: global median normalization log-ratio M



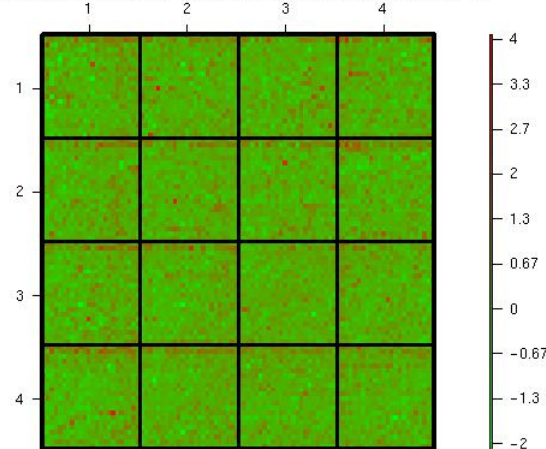
Swirl 93 array: global loess normalization log-ratio M



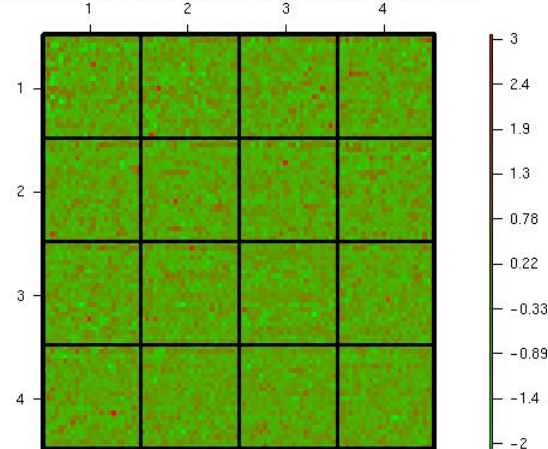
Global loess normalization

Within-print-tip-group loess normalization

Swirl 93 array: within-print-tip-group loess normalization log-ratio M



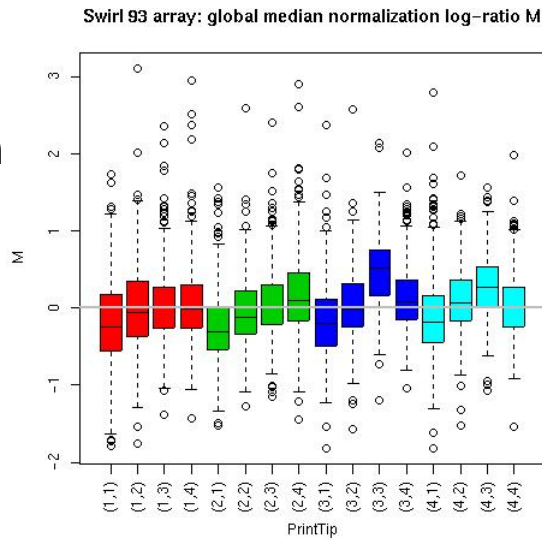
Swirl 93 array: 2D spatial loess normalization log-ratio M



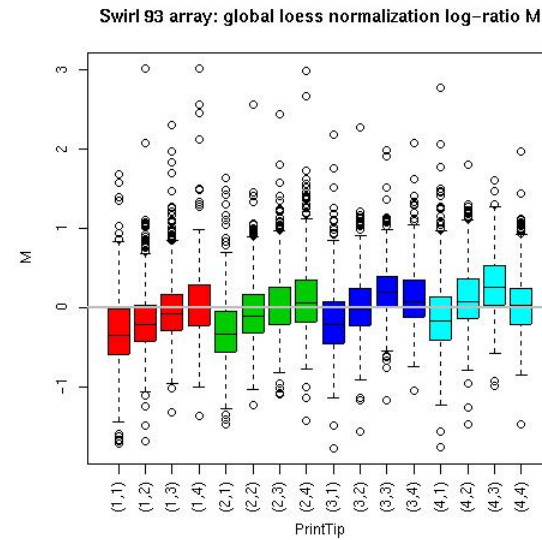
2D spatial normalization

Boxplots of normalized M-L

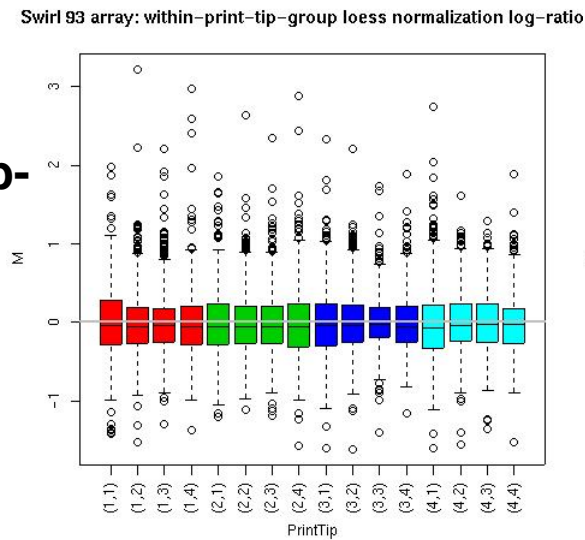
Global median normalization



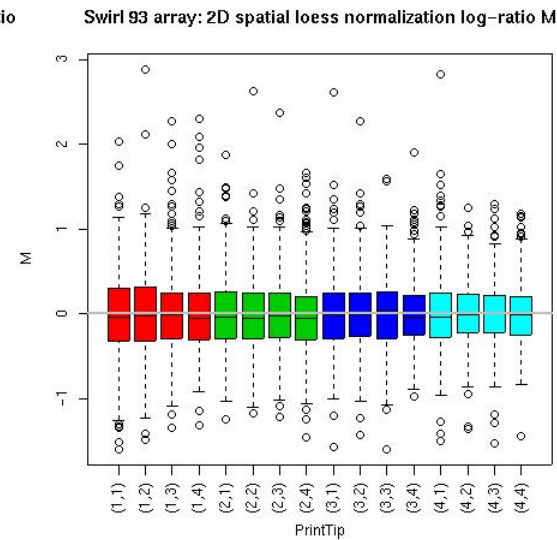
Global loess normalization



Within-print-tip-group loess normalization

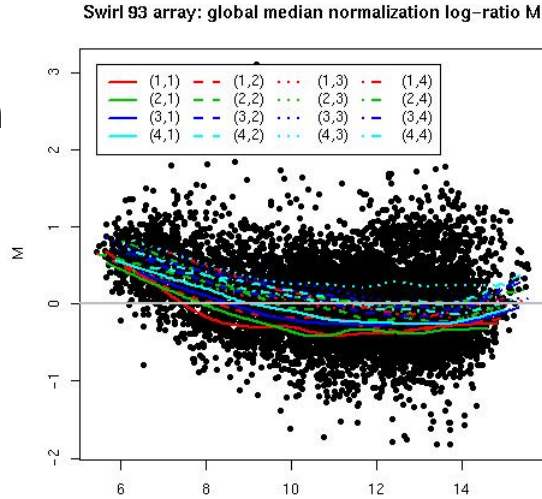


2D spatial normalization

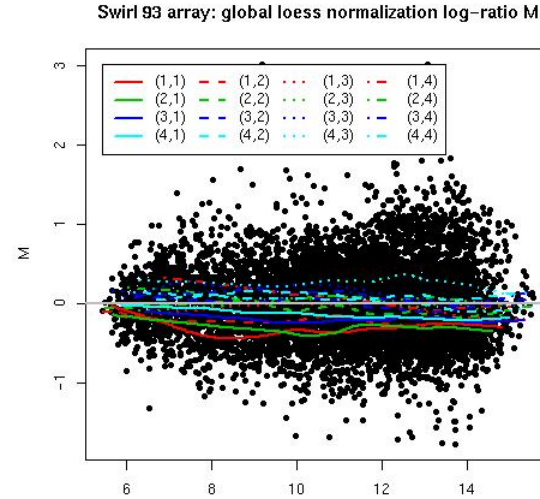


MA-plots of normalized M-L

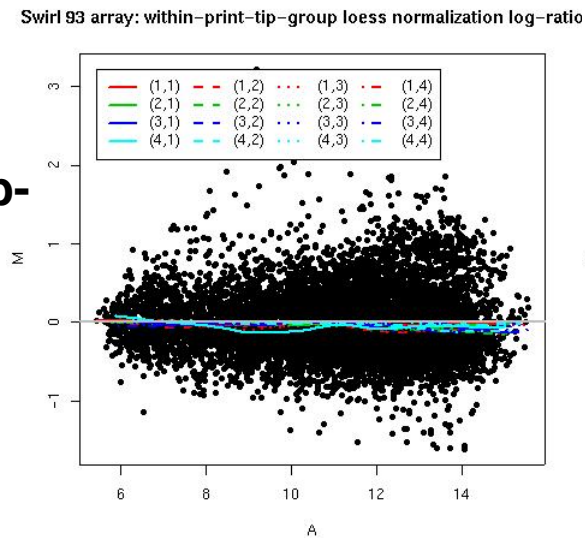
Global median normalization



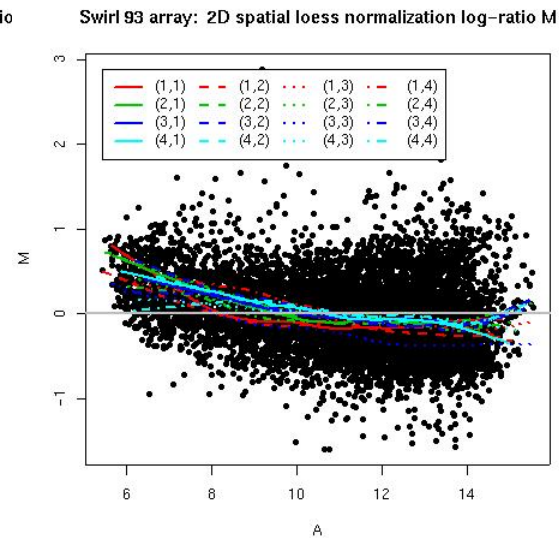
Global loess normalization



Within-print-tip-group loess normalization



2D spatial normalization



Scale normalization

- The log-ratios M from different sectors, plates, or arrays may exhibit different spreads and some **scale** adjustment may be necessary.

$$\log_2 R/G \leftarrow (\log_2 R/G - L)/S$$

- Can use a robust estimate of scale such as the **median absolute deviation (MAD)**
 $MAD = \text{median} | M - \text{median}(M) |.$

Scale normalization

- For print-tip-group scale normalization, assume all print-tip-groups have the same spread in M .
- Denote **true** and **observed** log-ratio by μ_{ij} and M_{ij} , resp., where $M_{ij} = a_i \mu_{ij}$, and i indexes print-tip-groups and j spots. Robust estimate of a_i is

$$\hat{a}_i = \frac{MAD_i}{\sqrt[I]{\prod_{i=1}^I MAD_i}}$$

where MAD_i is MAD of M_{ij} in print-tip-group i .

- Similarly for between-slides scale normalization.

Which genes to use?

- **All spots on the array:**
 - Problem when many genes are differentially expressed.
- **Housekeeping genes:** Genes that are thought to be constantly expressed across a wide range of biological samples (e.g. tubulin, GAPDH).
Problems:
 - sample specific biases (genes are actually regulated),
 - do not cover intensity range.
- **Spiked controls:** e.g. plant genes.

Which genes to use?

- **Genomic DNA titration series:**
 - fine in yeast,
 - but weak signal for higher organisms with high intron/exon ratio (e.g., mouse, human).
- **Rank invariant set** (Schadt et al., 1999; Tseng et al., 2001): genes with same rank in both channels. Problems: set can be small.

Microarray sample pool

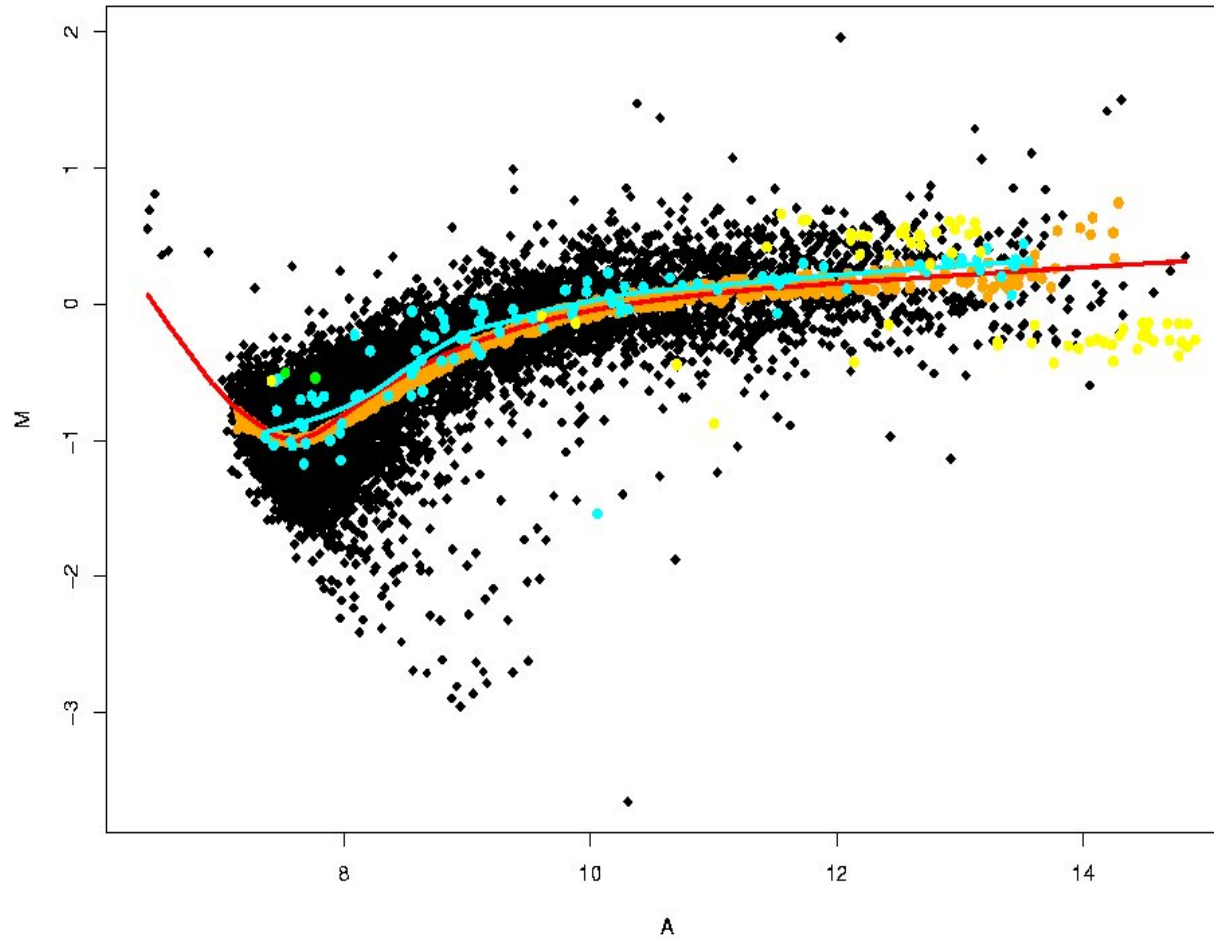
- Collaboration with Ngai Lab, UC Berkeley, Ref. Yang et al. (2002).
- **Microarray Sample Pool, MSP**: Control sample for normalization, in particular, when it is not safe to assume most genes are equally expressed in both channels.
- MSP: **pooled** all 18,816 ESTs from RIKEN release 1 cDNA mouse library.
- Six-step **dilution series** of the MSP.
- MSP samples were spotted in middle of first and last row of each sector.

Microarray sample pool

MSP control spots

- provide potential probes for every target sequence;
- are constantly expressed across a wide range of biological samples;
- cover the intensity range;
- are similar to genomic DNA, but without intron sequences → better signal than genomic DNA in organisms with high intron/exon ratio;
- can be used in composite normalization.

Microarray sample pool



MSP

Rank invariant

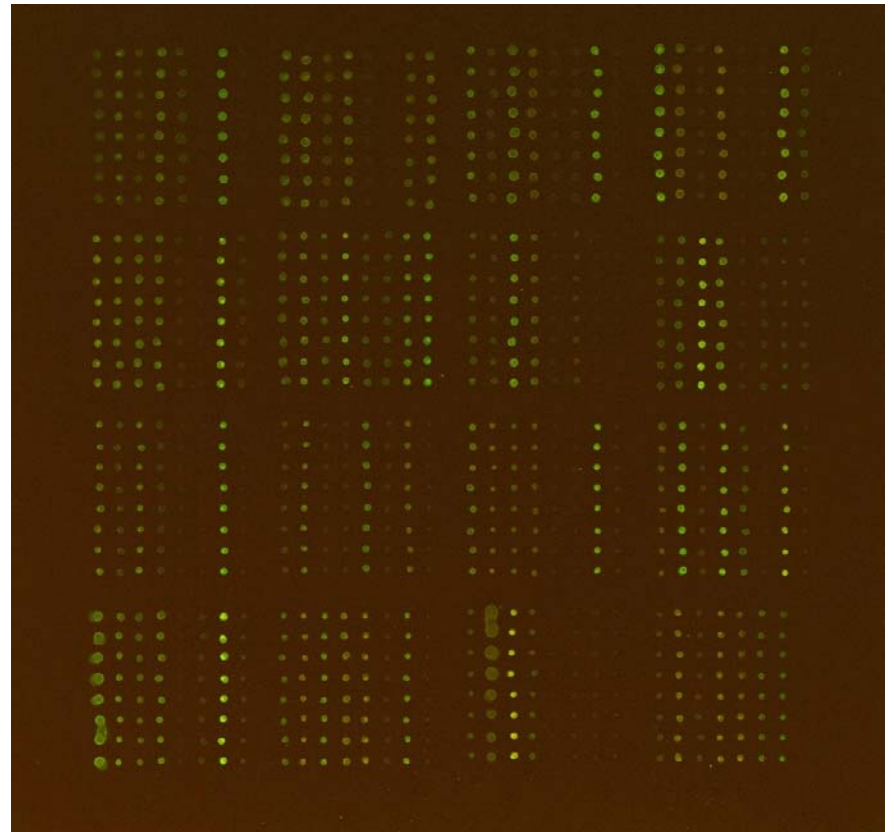
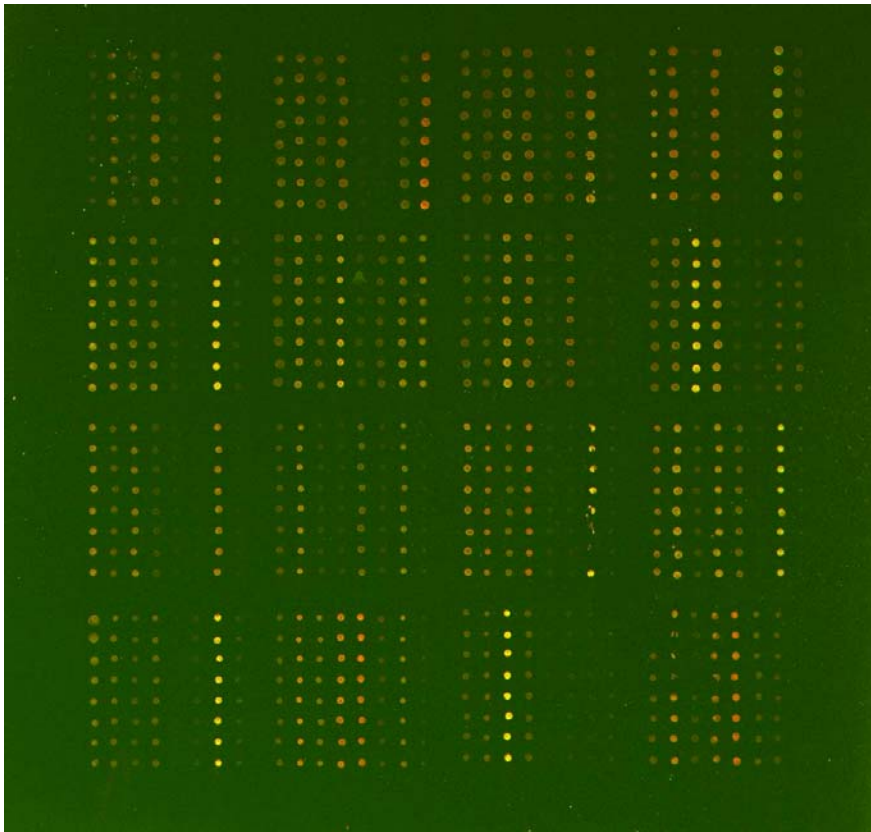
Housekeeping

Tubulin, GAPDH

Dye-swap experiment

- Probes
 - 50 distinct clones thought to be differentially expressed in apo AI knock-out mice compared to inbred C57Bl/6 control mice (largest absolute t-statistics in a previous experiment).
 - 72 other clones.
- Spot each clone 8 times .
- Two hybridizations with dye-swap:
 - Slide 1: trt → red, ctl → green.
 - Slide 2: trt → green, ctl → red.

Dye-swap experiment



Self-normalization

- Slide 1, $M = \log_2 (R/G) - L$
- Slide 2, $M' = \log_2 (R'/G') - L'$

Combine by **subtracting** the normalized log-ratios:

$$M - M'$$

$$= [(\log_2 (R/G) - L) - (\log_2 (R'/G') - L')] / 2$$

$$\approx [\log_2 (R/G) + \log_2 (G'/R')] / 2$$

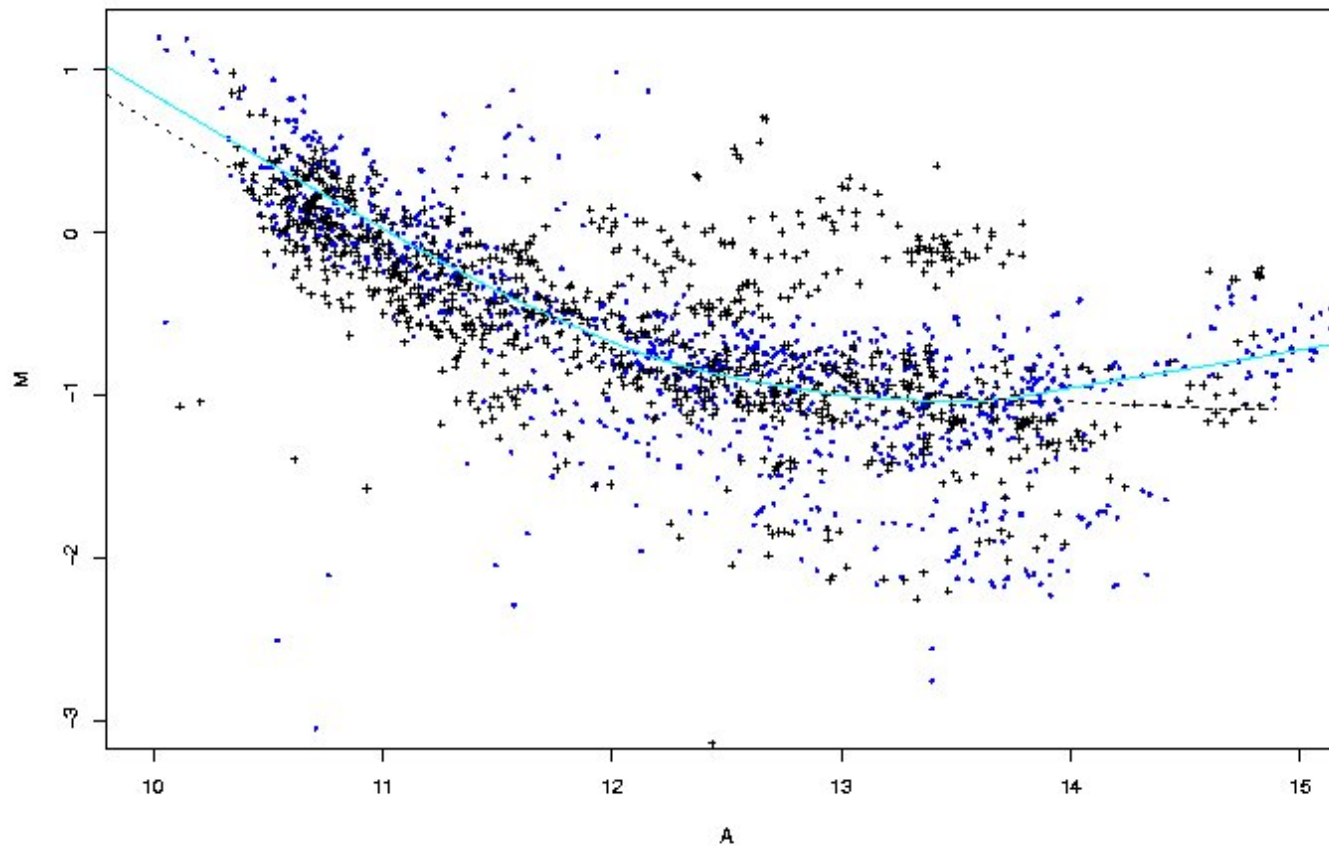
$$\approx [\log_2 (RG'/GR')] / 2$$

provided $L = L'$.

Assumption: the normalization functions are the same for the two slides.

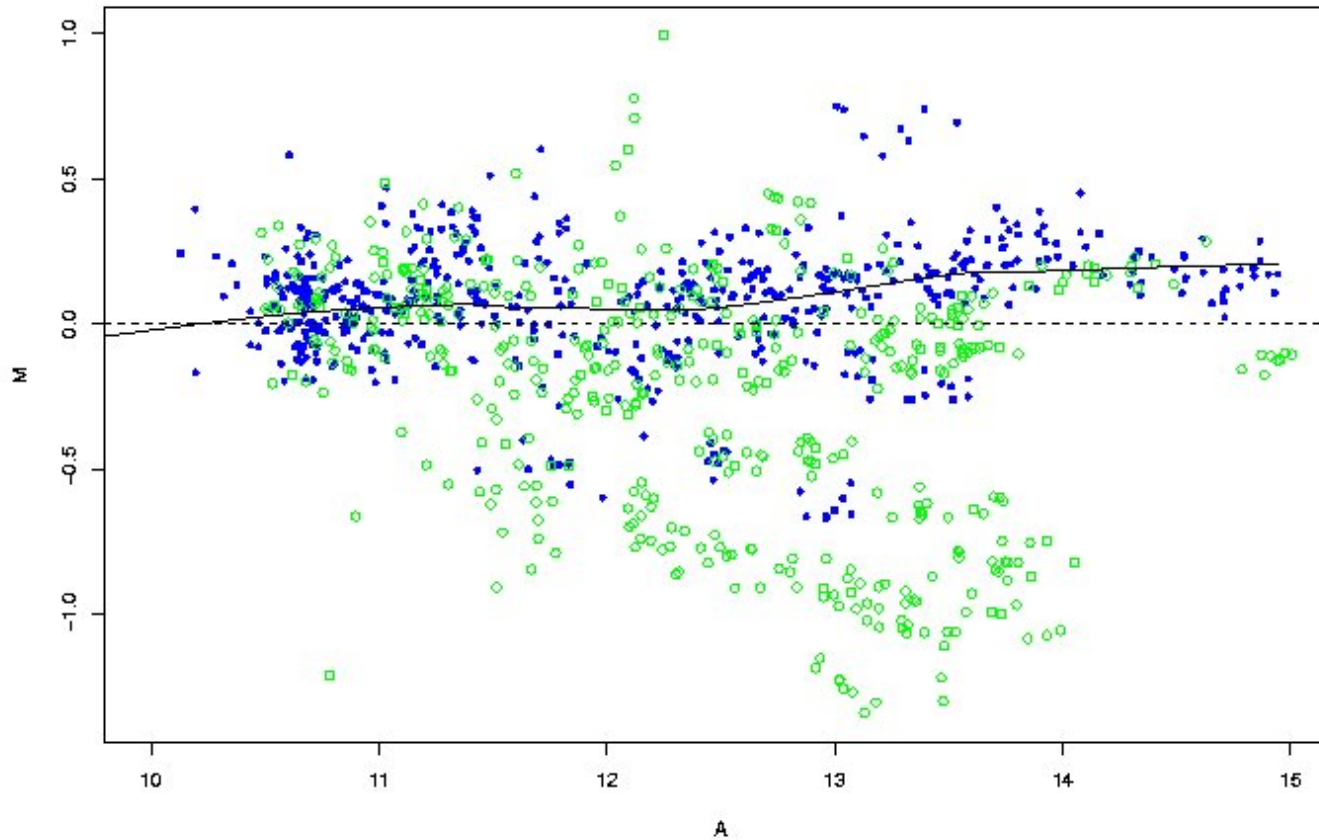
Checking the assumption

MA-plot for slides 1 and 2



Result of self-normalization

$(M - M')/2$ vs. $(A + A')/2$



Pre-processing software

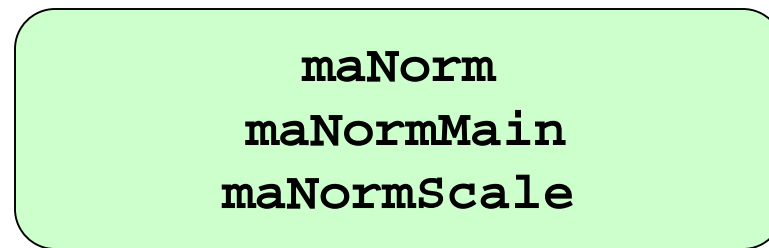
- **affy**: Affymetrix oligonucleotide chips.
- **marray**, **limma**: Spotted DNA microarrays.
- **vsn**: Variance stabilization for both types of arrays.
 - Reading in intensity data, diagnostic plots, normalization, computation of expression measures.
 - The packages start with very different data types, but produce similar objects of class **exprSet**.
 - One can then use other Bioconductor packages, e.g., **genefilter**, **geneplotter**.

marray packages

Image
quantitation
data,
e.g. .gpr, .Spot, .gal



Class `marrayRaw`



Class `marrayNorm`



`as(swirl.norm, "exprSet")`

Class `exprSet`

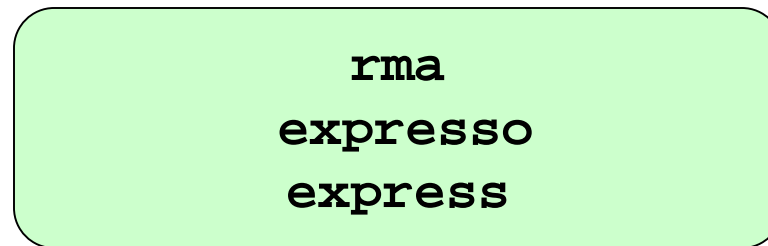
Save data to file using `write.exprs` or continue analysis using other Bioconductor packages

affy package

CEL and CDF
files



Class `AffyBatch`



Class `exprSet`

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

marray: Pre-processing spotted DNA microarray data

- **marrayClasses**:
 - class definitions for spotted DNA microarray data (cf. MIAME);
 - basic methods for manipulating microarray objects: printing, plotting, subsetting, class conversions, etc.
- **marrayInput**:
 - reading in intensity data and textual data describing probes and targets;
 - automatic generation of microarray data objects;
 - widgets for point & click interface.
- **marrayPlots**: diagnostic plots.
- **marrayNorm**: robust adaptive location and scale normalization procedures.
- **marrayTools**: miscellaneous tools for functional genomics cores facilities at UCB and UCSF.

marrayLayout class

Array layout parameters

<code>maNspots</code>	Total number of spots	
<code>maNgr</code>	<code>maNgc</code>	Dimensions of grid matrix
<code>maNsr</code>	<code>maNsc</code>	Dimensions of spot matrices
<code>maSub</code>	Current subset of spots	
<code>maPlate</code>	Plate IDs for each spot	
<code>maControls</code>	Control status labels for each spot	
<code>maNotes</code>	Any notes	

marrayRaw class

Pre-normalization intensity data for a batch of arrays

maRf

maGf

Matrix of red and green foreground intensities

maRb

maGb

Matrix of red and green background intensities

maW

Matrix of spot quality weights

maLayout

Array layout parameters - `marrayLayout`

maGnames

Description of spotted probe sequences
- `marrayInfo`

maTargets

Description of target samples - `marrayInfo`

maNotes

Any notes

marrayNorm class

Post-normalization intensity data for a batch of arrays

maA		Matrix of average log intensities, A
maM		Matrix of normalized intensity log ratios, M
maMloc	maMscale	Matrix of location and scale normalization values
maW		Matrix of spot quality weights
maLayout		Array layout parameters - <code>marrayLayout</code>
maGnames		Description of spotted probe sequences - <code>marrayInfo</code>
maTargets		Description of target samples - <code>marrayInfo</code>
maNormCall		Function call
maNotes		Any notes

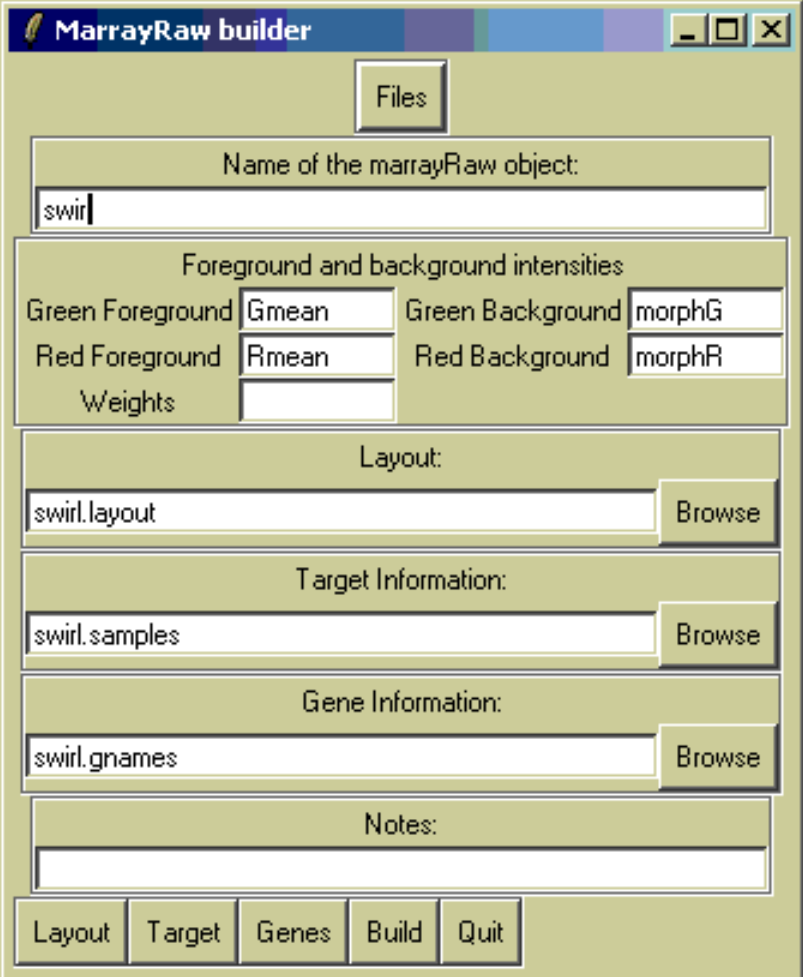
`marrayInput` package

- `marrayInput` provides functions for reading microarray data into R and creating microarray objects of class `marrayLayout`, `marrayInfo`, and `marrayRaw`.
- Input
 - Image quantitation data, i.e., output files from image analysis software.
E.g. `.gpr` for `GenePix`, `.spot` for `Spot`.
 - Textual description of probe sequences and target samples.
E.g. `gal` files, `god` lists.

marrayInput package

- Widgets for graphical user interface

`widget.marrayLayout`,
`widget.marrayInfo`,
`widget.marrayRaw`.



The screenshot shows a window titled "MarrayRaw builder" with a standard Windows-style title bar. The window contains several sections for configuring a marrayRaw object:

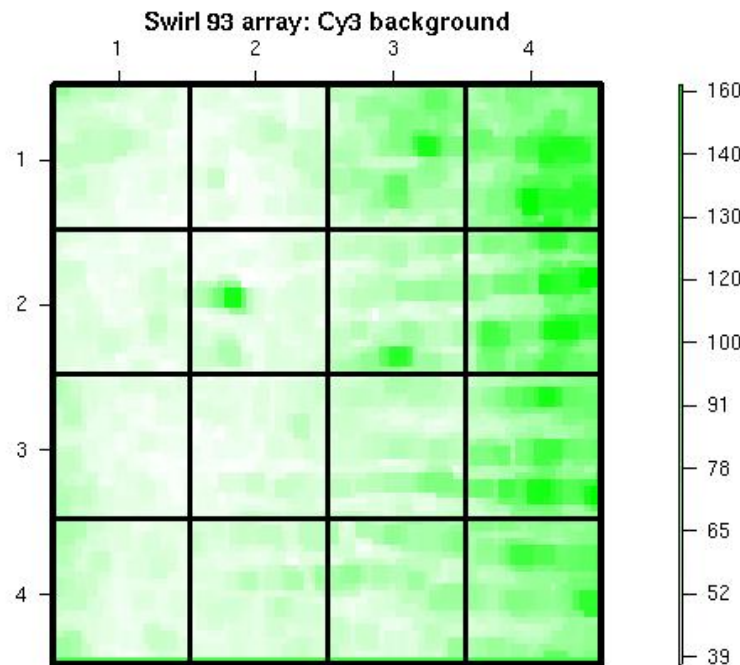
- Files:** A button labeled "Files" is located at the top right of the main area.
- Name of the marrayRaw object:** A text input field containing the text "swirl".
- Foreground and background intensities:** A section with four input fields: "Green Foreground" (Gmean), "Green Background" (morphG), "Red Foreground" (Rmean), and "Red Background" (morphR). Below these is a "Weights" label and an empty input field.
- Layout:** A text input field containing "swirl.layout" and a "Browse" button to its right.
- Target Information:** A text input field containing "swirl.samples" and a "Browse" button to its right.
- Gene Information:** A text input field containing "swirl.gnames" and a "Browse" button to its right.
- Notes:** A large empty text area for entering notes.
- Buttons:** A row of five buttons at the bottom: "Layout", "Target", "Genes", "Build", and "Quit".

marrayPlots package

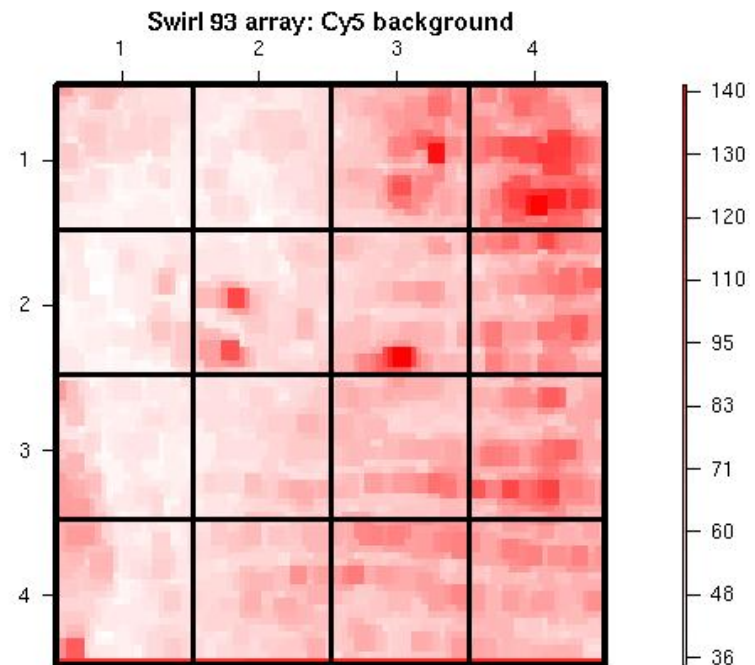
- See `demo(marrayPlots)`.
- **Diagnostic plots** of spot statistics.
E.g. red and green log intensities, intensity log ratios M , average log intensities A , spot area.
 - `maImage`: 2D spatial color images.
 - `maBoxplot`: boxplots.
 - `maPlot`: scatter-plots with fitted curves and text highlighted.
- **Stratify** plots according to layout parameters such as `print-tip-group`, `plate`.
E.g. MA-plots with loess fits by `print-tip-group`.

2D spatial images

maImage



Cy3 background intensity



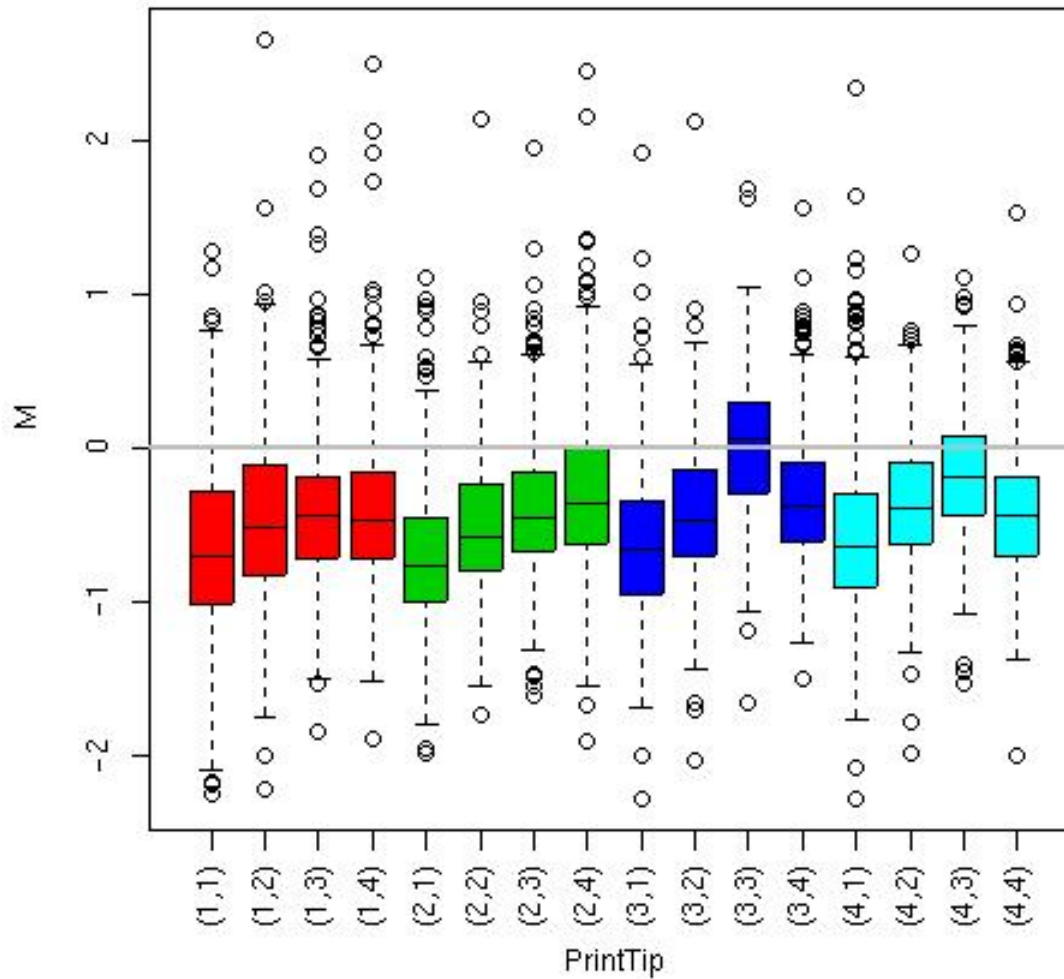
Cy5 background intensity

Boxplots by print-tip-group

maBoxplot

Swirl 93 array: pre-normalization log-ratio M

Intensity
log ratio, M

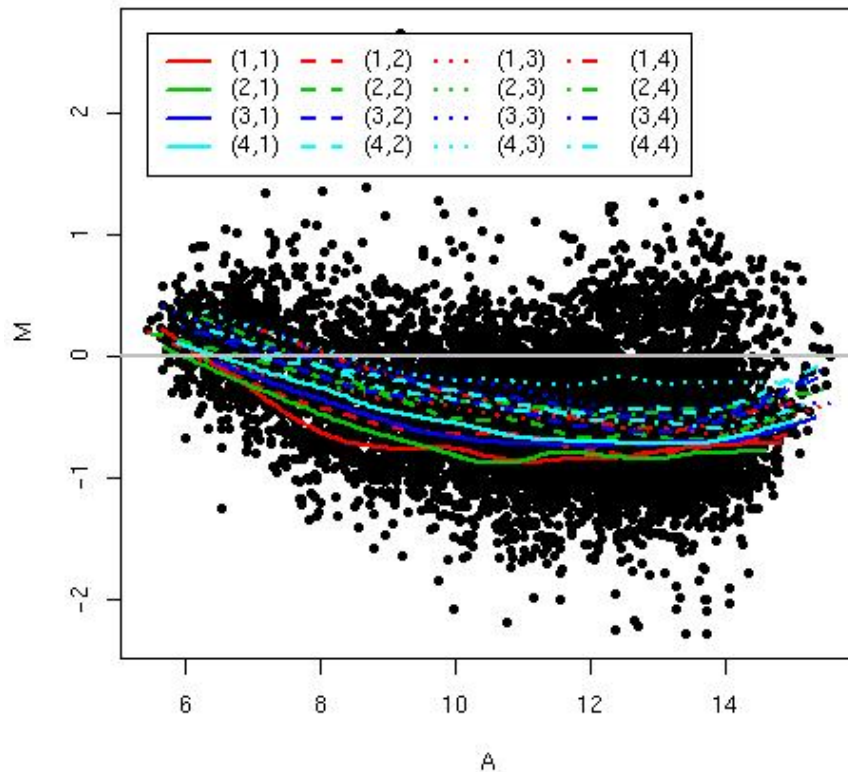


MA-plot by print-tip-group

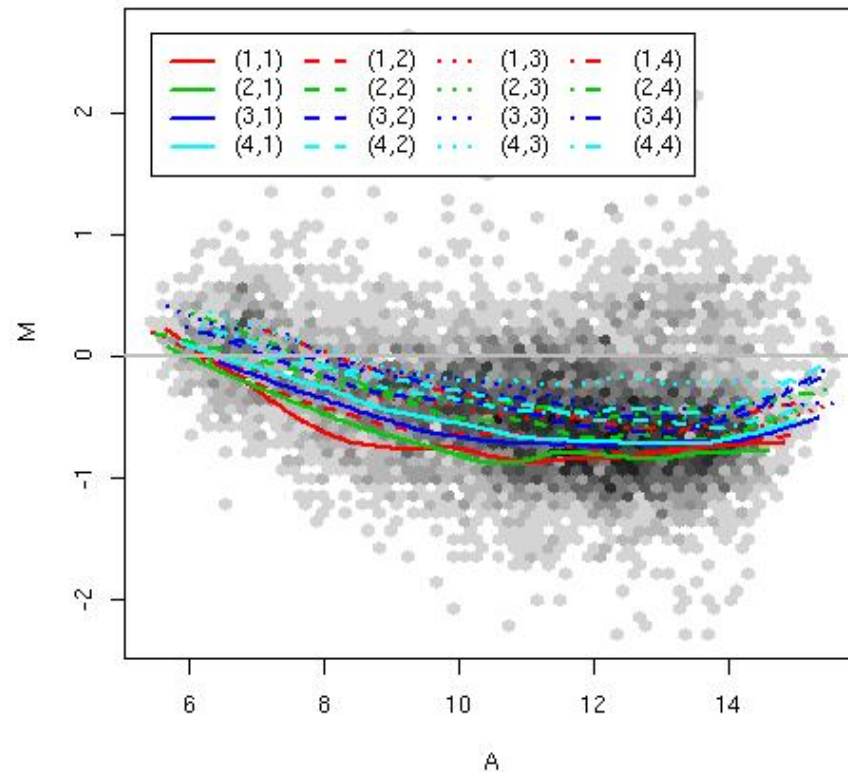
maPlot

$M = \log_2 R - \log_2 G$ vs. $A = (\log_2 R + \log_2 G)/2$

Swirl 93 array: pre-normalization log-ratio M



Swirl 93 array: pre-normalization log ratio M



hexbin

marrayNorm package

- **maNormMain**: main normalization function, allows **robust adaptive location and scale normalization** for a batch of arrays
 - intensity or A-dependent location normalization (**maNormLoess**);
 - 2D spatial location normalization (**maNorm2D**);
 - median location normalization (**maNormMed**);
 - scale normalization using MAD (**maNormMAD**);
 - composite normalization;
 - your own normalization function.
- **maNorm**: simple wrapper function.
maNormScale: simple wrapper function for scale normalization.

marrayTools package

- The **marrayTools** package provides additional functions for handling two-color spotted microarray data (see devel. version).
- The **spotTools** and **gpTools** functions start from Spot and GenePix image analysis output files, respectively, and automatically
 - read in these data into R,
 - perform standard normalization (within print-tip-group loess),
 - create a directory with a standard set of diagnostic plots (jpeg format), excel files of quality measures, and tab delimited files of normalized log ratios M and average log intensities A .

swirl dataset

- Microarrays:
 - 8,448 probes (768 controls);
 - 4 x 4 grid matrix;
 - 22 x 24 spot matrices.
- 4 hybridizations: swirl mutant and wild type mRNA.
- Data stored in object of class `marrayRaw`: `data(swirl)`.

```
> maInfo(maTargets(swirl))[ ,3:4]
experiment Cy3 experiment Cy5
1          swirl          wild type
2      wild type          swirl
3          swirl          wild type
4      wild type          swirl
```

Differential Gene Expression

Combining data across arrays

Data on G genes for n arrays

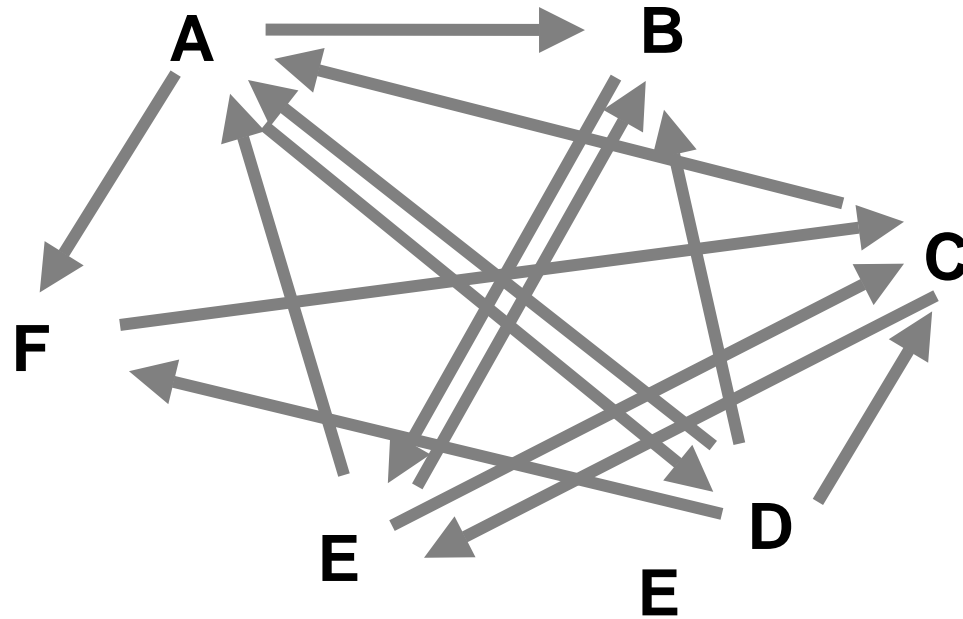
→ $G \times n$ genes-by-arrays data matrix

		Arrays					...
		Array1	Array2	Array3	Array4	Array5	
Genes	Gene1	0.46	0.30	0.80	1.51	0.90	...
	Gene2	-0.10	0.49	0.24	0.06	0.46	...
	Gene3	0.15	0.74	0.04	0.10	0.20	...
	Gene4	-0.45	-1.03	-0.79	-0.56	-0.32	...
	Gene5	-0.06	1.06	1.35	1.09	-1.09	...

$M = \log_2(\text{Red intensity} / \text{Green intensity})$
expression measure, e.g. RMA.

Combining data across arrays

... but the columns have **structure**,
determined by the **experimental design**.



Combining data across arrays

- *cDNA array factorial experiment.* Each column corresponds to a pair of mRNA samples with different drug x dose x time combinations.
- *Clinical trial.* Each column corresponds to a patient, with associated clinical outcome, such as survival and response to treatment.
- **Linear models** and extensions thereof can be used to effectively combine data across arrays for complex experimental designs.

Gene filtering

- A very common task in microarray data analysis is **gene-by-gene selection**.
- Filter genes based on
 - data quality criteria, e.g. absolute intensity or variance;
 - subject matter knowledge;
 - their ability to differentiate cases from controls;
 - their spatial or temporal expression pattern.
- Depending on the experimental design, some highly specialized filters may be required and applied sequentially.

Gene filtering

- *Clinical trial.* Filter genes based on association with survival, e.g. using a Cox model.
- *Factorial experiment.* Filter genes based on interaction between two treatments, e.g. using 2-way ANOVA.
- *Time-course experiment.* Filter genes based on periodicity of expression pattern, e.g. using Fourier transform.

genefilter package

- The **genefilter** package provides tools to sequentially apply filters to the rows (genes) of a matrix or of an instance of the **exprSet** class.
- There are two main functions, **filterfun** and **genefilter**, for assembling and applying the filters, respectively.
- Any number of functions for specific filtering tasks can be defined and supplied to **filterfun**.
E.g. Cox model p-values, coefficient of variation.

genefilter: separation of tasks

1. Select/define functions for specific filtering tasks.
2. Assemble the filters using the **filterfun** function.
3. Apply the filters using the **genefilter** function → a logical vector, **TRUE** indicates genes that are retained.
4. Apply that vector to the **exprSet** to obtain a microarray object for the subset of interesting genes.

genefilter: supplied filters

Filters supplied in the package

- **kOverA** – select genes for which k samples have expression measures larger than A.
- **gapFilter** – select genes with a large IQR or gap (jump) in expression measures across samples.
- **ttest** – select genes according to t-test nominal p-values.
- **Anova** – select genes according to ANOVA nominal p-values.
- **coxfilter** – select genes according to Cox model nominal p-values.

genefilter: writing filters

- It is very simple to write your own filters.
- You can use the supplied filtering functions as templates.
- The basic idea is to rely on **lexical scope** to provide values (bindings) for the variables that are needed to do the filtering.

genefilter: How to?

1. First, build the filters

```
f1 <- anyNA  
f2 <- kOverA(5, 100)
```

2. Next, assemble them in a filtering function

```
ff <- filterfun(f1, f2)
```

3. Finally, apply the filter

```
wh <- genefilter(marrayDat, ff)
```

4. Use `wh` to obtain the relevant subset of the data

```
mySub <- marrayDat[wh, ]
```

Differential gene expression

- Identify genes whose expression levels are **associated** with a response or covariate of interest
 - clinical outcome such as survival, response to treatment, tumor class;
 - covariate such as treatment, dose, time.
- **Estimation**: estimate effects of interest and **variability** of these estimates.
E.g. slope, interaction, or difference in means in a linear model.
- **Testing**: assess the statistical **significance** of the observed associations.

limma: Linear models for microarray data

- Fitting of gene-wise linear models to estimate log-ratios between two or more target samples simultaneously: `lm.series`, `rlm.series`, `glm.series` (handles replicate spots).
- `ebayes`: moderated t-statistics and log-odds of differential expression by empirical Bayes shrinkage of the standard errors towards a common value.

Multiple hypothesis testing

- Large **multiplicity problem**: thousands of hypotheses are tested simultaneously!
 - Increased chance of **false positives**.
 - E.g. chance of at least one p-value $< \alpha$ for G independent tests is $1 - (1 - \alpha)^G$ and converges to one as G increases.
For G=1,000 and $\alpha = 0.01$, this chance is 0.9999568!
 - Individual p-values of 0.01 no longer correspond to significant findings.
- Need to **adjust for multiple testing** when assessing the statistical significance of the observed associations.

Multiple Hypothesis Testing

- Define an appropriate **Type I error** or **false positive rate**.
- Develop multiple testing procedures that
 - provide **strong control** of this error rate,
 - are **powerful** (few false negatives),
 - take into account the **joint distribution** of the test statistics.
- Report **adjusted p-values** for each gene which reflect the **overall** Type I error rate for the experiment.
- **Resampling** methods are useful tools to deal with the unknown joint distribution of the test statistics.

multtest package

- Multiple testing procedures for controlling
 - **Family-Wise Error Rate - FWER**: Bonferroni, Holm (1979), Hochberg (1986), Westfall & Young (1993) maxT and minP;
 - **False Discovery Rate - FDR**: Benjamini & Hochberg (1995), Benjamini & Yekutieli (2001).
- Tests based on t- or F-statistics for one- and two-factor designs.
- **Permutation procedures** for estimating the null distribution (used to calculate adjusted p-values).
- Similar **bootstrap procedures** coming soon!
- Fast permutation algorithm for minP adjusted p-values.
- Documentation: tutorial on multiple testing.

Clustering and Classification

Clustering vs. classification

- **Cluster analysis** (a.k.a. **unsupervised learning**)
 - the classes are unknown a priori;
 - the goal is to discover these classes from the data.
- **Classification** (a.k.a. class prediction, **supervised learning**)
 - the classes are predefined;
 - the goal is to understand the basis for the classification from a set of labeled objects and build a predictor for future unlabeled observations.

Distances

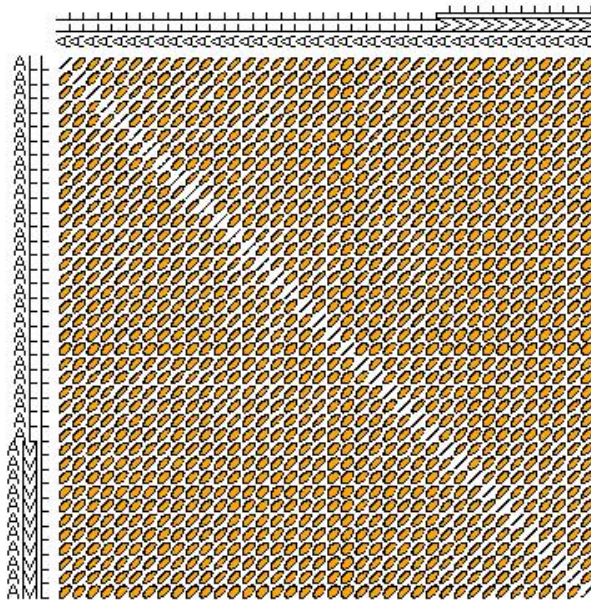
- Microarray data analysis often involves
 - clustering genes or samples;
 - classifying genes or samples.
- Both types of analyses are based on a measure of distance (or similarity) between genes or samples.
- R has a number of functions for computing and plotting distance and similarity matrices.

Distances

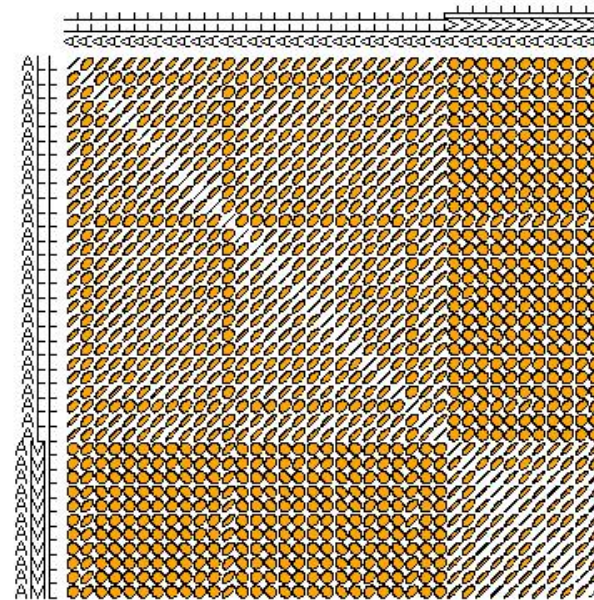
- Distance functions
 - `dist (mva)`: Euclidean, Manhattan, Canberra, binary;
 - `daisy (cluster)`.
- Correlation functions
 - `cor, cov.wt`.
- Plotting functions
 - `image`;
 - `plotcorr (ellipse)`;
 - `plot.cor, plot.mat (sma)`.

Correlation matrices

Correlation matrix for ALL AML data
G=3,051 genes



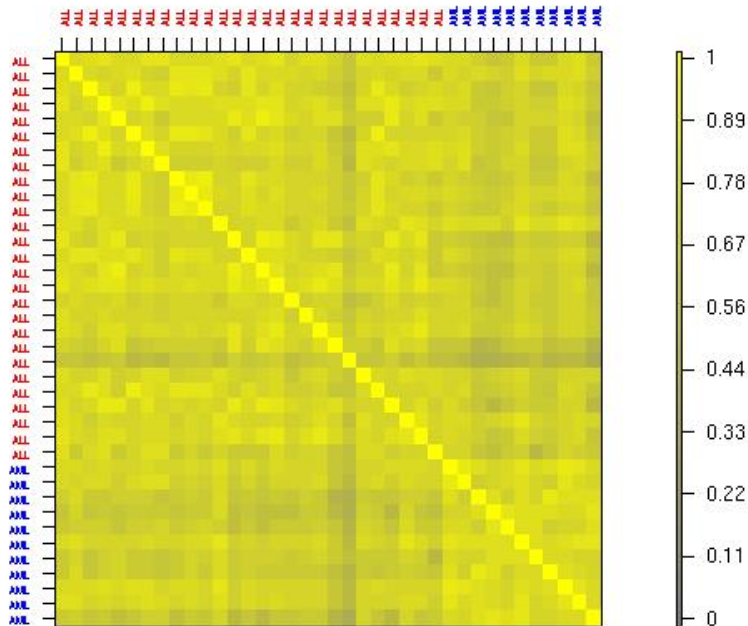
Correlation matrix for ALL AML data
G=39 genes with maxT adjusted p-value < 0.01



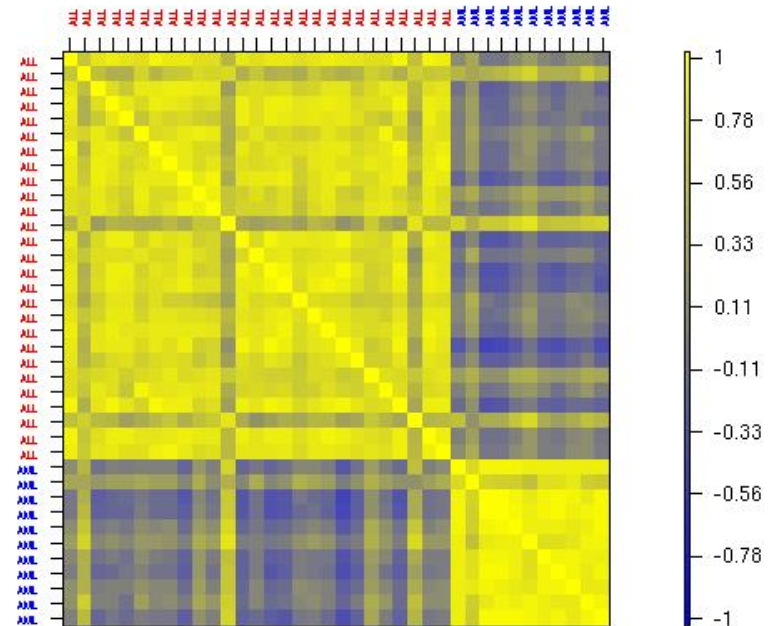
`plotcorr` function from **ellipse** package

Correlation matrices

Correlation matrix for ALL AML data
G=3,051 genes



Correlation matrix for ALL AML data
G=39 genes with maxT adjusted p-value < 0.01



`plot.cor` function from **sma** package

Multidimensional scaling

- Given any $n \times n$ dissimilarity matrix D , **multidimensional scaling** (MDS) is concerned with identifying n points in Euclidean space with a **similar** distance structure D' .
- The purpose is to provide a lower dimensional representation of the distances which conveys information on the relationships between the n objects, such as the existence of clusters or one-dimensional structure in the data (e.g., seriation).

MDS

- There are different approaches for reducing dimensionality, depending on how we define **similarity** between the old and new dissimilarity matrices for the n objects, i.e., depending on the objective or **stress function S** that we seek to minimize.

- **Least-squares scaling** $S(D, D') = \left(\sum (d_{ij} - d'_{ij})^2 \right)^{1/2}$

- **Sammon mapping** $S(D, D') = \sum (d_{ij} - d'_{ij})^2 / d_{ij}$

places more emphasis on smaller dissimilarities (and hence should be preferred for clustering methods).

- **Shepard-Kruskal non-metric scaling** is based on ranks, i.e., the order of the distances is more important than their actual values.

MDS and PCA

- When the distance matrix D is the Euclidean distance matrix between the rows of an $n \times m$ matrix X , there is a duality between **principal component analysis (PCA)** and MDS.
- The k -dimensional **classical solution** to the MDS problem is given by the centered scores of the n objects on the first k principal components.
- The classical solution of MDS in k -dimensional space minimizes the sum of squared differences between the entries of the new and old dissimilarity matrices, i.e., is optimal for least-squares scaling.

MDS

- As with PCA, the quality of the representation will depend on the **magnitude of the first k eigenvalues**.
- The data analyst should choose a value for k that is small enough for ease representation but also corresponds to a substantial “proportion of the distance matrix explained”.

MDS

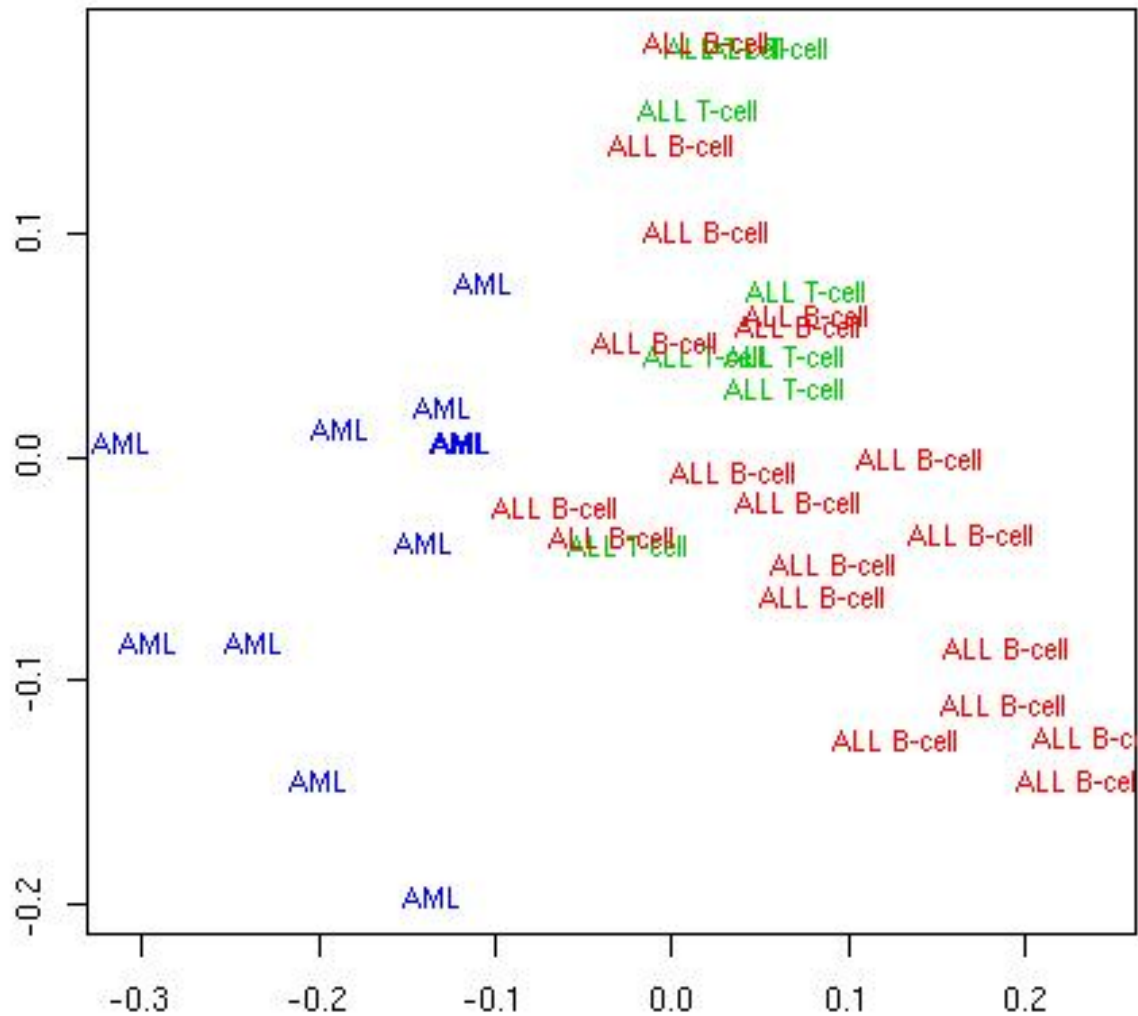
- **N.B.** The MDS solution reflects not only the choice of a distance function, but also the **features selected**.
- If features (genes) were selected to separate the data into two groups (e.g., on the basis of two-sample t-statistics), it should come as no surprise that an MDS plot has two groups. In this instance MDS is not a confirmatory approach.

R MDS software

- `cmdscale`: Classical solution to MDS, in package `mva`.
- `sammon`: Sammon mapping, in package `MASS`.
- `isoMDS`: Kruskal's non-metric MDS, in package `MASS`.

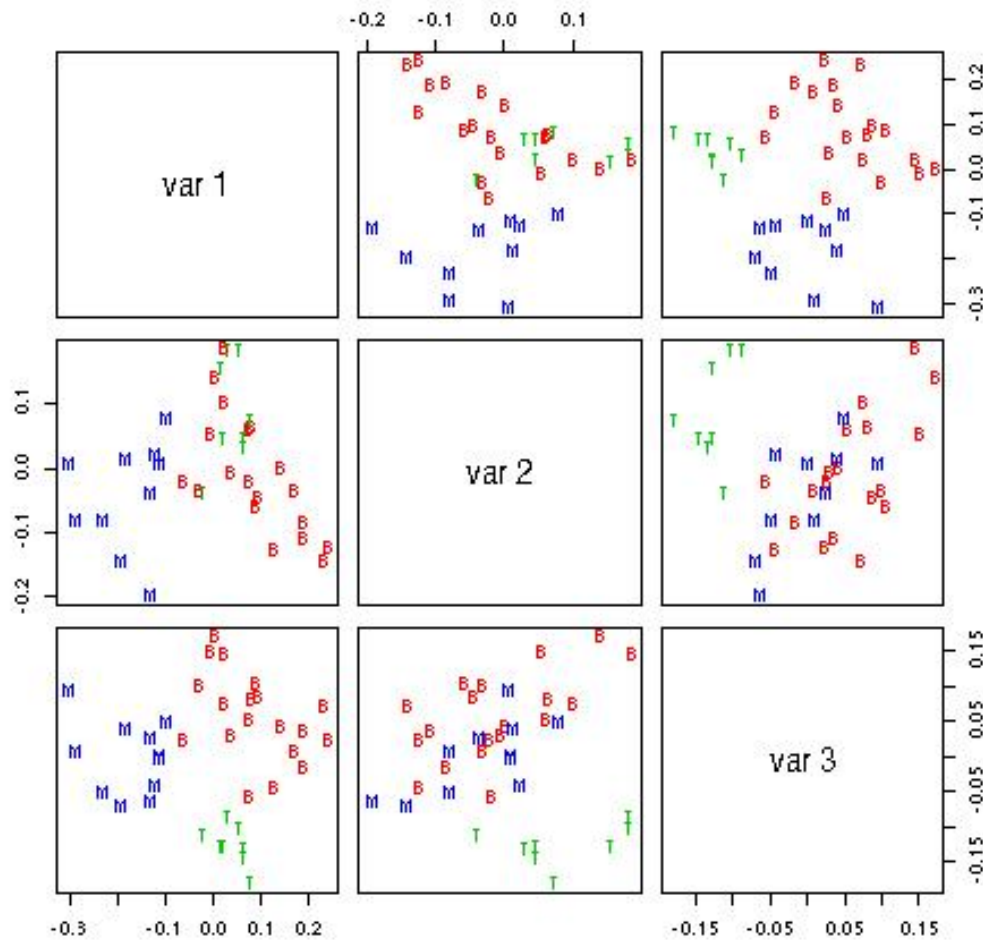
Classical MDS

MDS for ALL AML data, correlation matrix, G=3,051 genes, k=2



Classical MDS

MDS for ALL AML data, correlation matrix, G=3,051 genes, k=3



$$\frac{|\lambda_1| + |\lambda_2|}{\sum |\lambda_i|} = 43\%$$

$$\frac{|\lambda_1| + |\lambda_2| + |\lambda_3|}{\sum |\lambda_i|} = 55\%$$

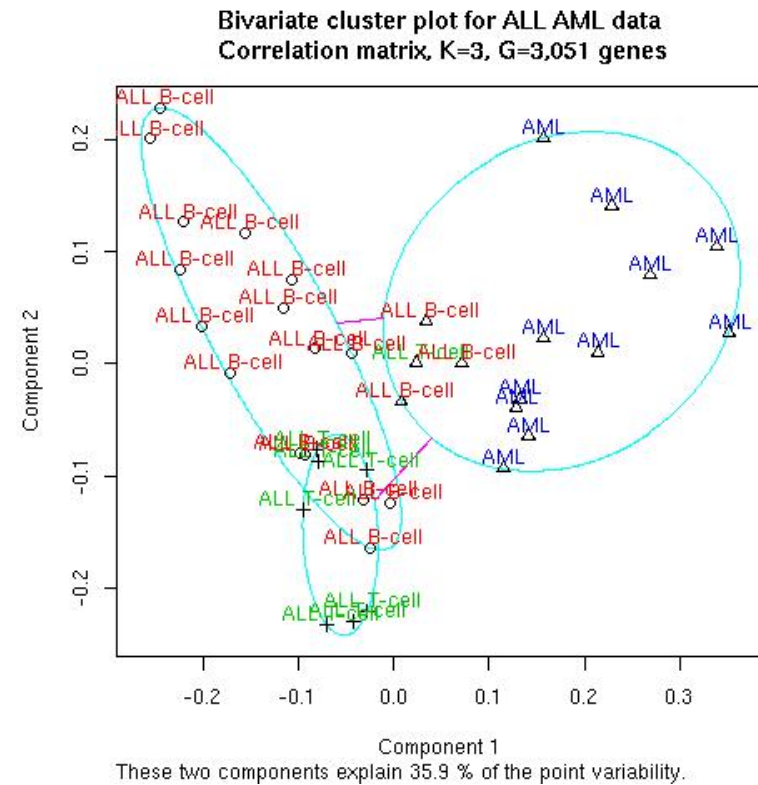
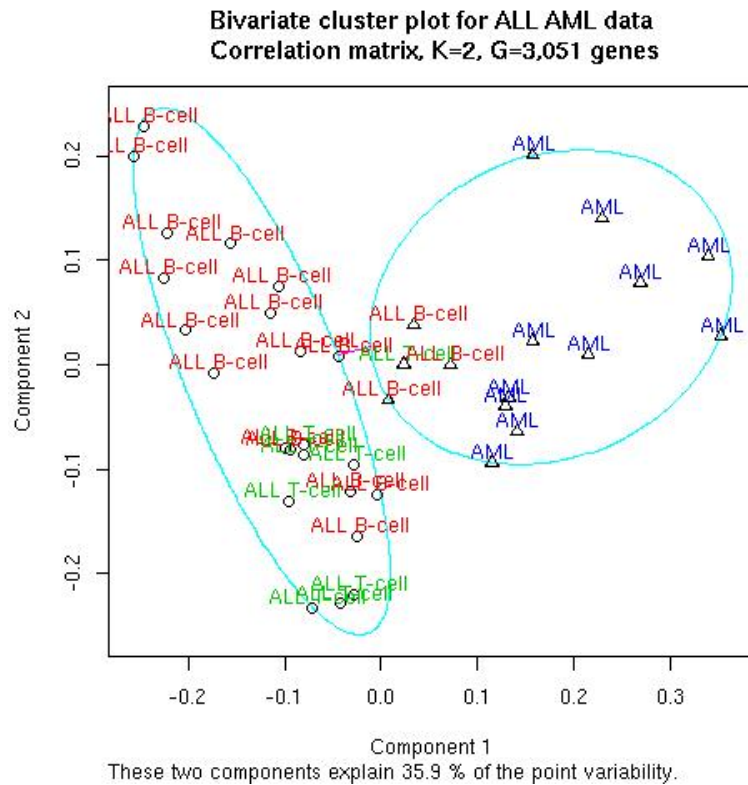
Cluster analysis packages

- **class**: self organizing maps (SOM).
- **cluster**:
 - AGglomerative NESTing (**agnes**),
 - Clustering LARe Applications (**clara**),
 - DIvisive ANALysis (**diana**),
 - Fuzzy Analysis (**fanny**),
 - MONothetic Analysis (**mona**),
 - Partitioning Around Medoids (**pam**),
 - HOPACH (coming soon!).
- **e1071**:
 - fuzzy C-means clustering (**cmeans**),
 - bagged clustering (**bclust**).
- **mva**:
 - hierarchical clustering (**hclust**),
 - k-means (**kmeans**).
- Specialized summary, plot, and print methods for clustering results.

pam

K=2

K=3

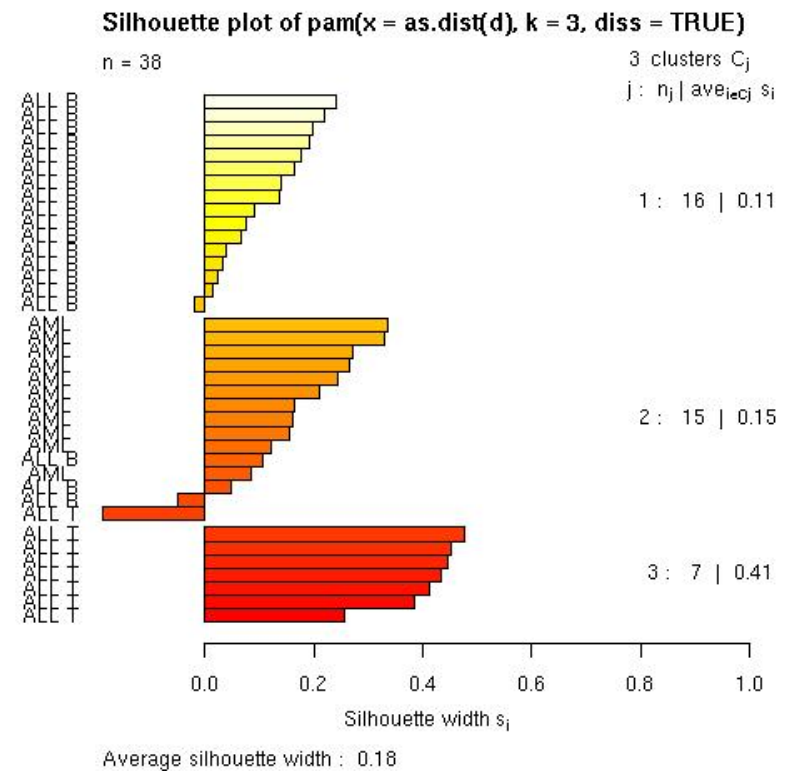
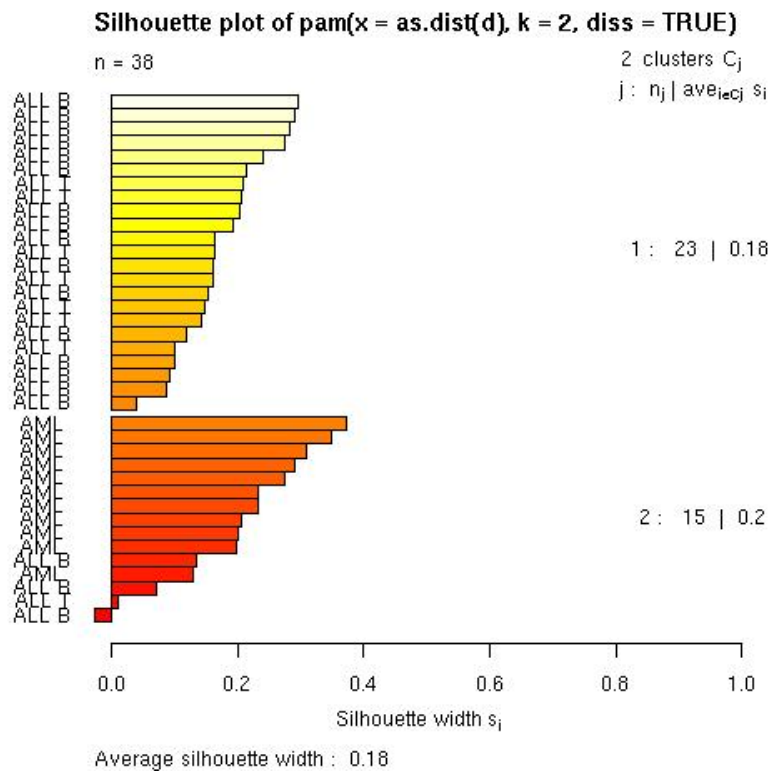


pam and clusplot functions from **cluster** package

pam

K=2

K=3



pam and plot functions from **cluster** package

Dendrogram

- **N.B.** While dendrograms are quite appealing because of their apparent ease of interpretation, **they can be misleading**.
- First, the dendrogram corresponding to a given hierarchical clustering is **not unique**, since for each merge one needs to specify which subtree should go on the left and which on the right --- there are $2^{(n-1)}$ choices.
- The default in the R function `hclust` is to order the subtrees so that the tighter cluster is on the left.

Dendrogram

- Second, they *impose* structure on the data, instead of *revealing* structure in these data.
- Such a representation will be valid only to the extent that the pairwise dissimilarities possess the hierarchical structure imposed by the clustering algorithm.

Dendrogram

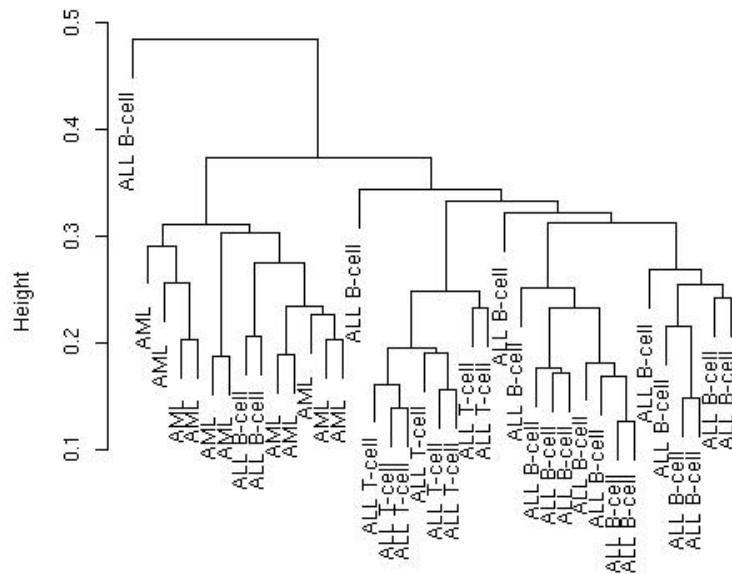
- The **cophenetic correlation coefficient** can be used to measure how well the hierarchical structure from the dendrogram represents the actual distances.
- This measure is defined as the correlation between the $n(n-1)/2$ pairwise dissimilarities between observations and their **cophenetic dissimilarities** from the dendrogram, i.e., the between cluster dissimilarities at which two observations are first joined together in the same cluster.
- Function `cophenetic` in **mva** package.

Dendrogram

Original data,
coph corr = 0.74

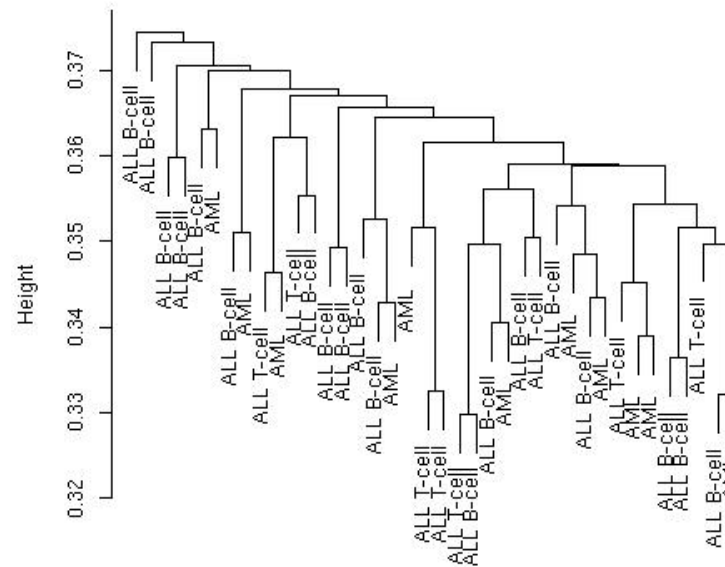
Randomized data
(perm. wi features),
coph corr = 0.57

Hierarchical clustering dendrogram for ALL AML data



as.dist(d)
Average linkage, correlation matrix, G=3,051 genes

Hierarchical clustering dendrogram for randomized ALL AML data



as.dist(d0)
Average linkage, correlation matrix, G=3,051 genes

Classification

- Predict a biological **outcome** on the basis of observable **features**.



- **Outcome:** tumor class, type of bacterial infection, survival, response to treatment.
- **Features:** gene expression measures, covariates such as age, sex.

Classification

- Old and extensive literature on classification, 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.
- Comparison on microarray data:
simple classifiers like k-NN and naïve Bayes perform remarkably well.

Performance assessment

- Classification error rates, or related measures, are usually reported
 - to compare the performance of different classifiers;
 - to support statements such as
“clinical outcome X for cancer Y can be predicted accurately based on gene expression measures”.
- Classification error rates can be estimated by resampling, e.g., bootstrap or cross-validation.

Performance assessment

- It is essential to take into account feature selection and other training decisions in the error rate estimation process.
E.g. number of neighbors in k-NN, kernel in SVMs.
- Otherwise, error estimates can be severely **biased downward**, i.e., overly optimistic.

Important issues

- Standardization;
- Distance function;
- Feature selection;
- Loss function;
- Class priors;
- Binary vs. polychotomous classification.

Classification packages

- **class**:
 - k-nearest neighbor (**knn**),
 - learning vector quantization (**lvq**).
- **e1071**: support vector machines (**svm**).
- **ipred**: bagging, resampling based estimation of prediction error.
- **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.
- **ranForest**, **RanForests**: random forests.
- **rpart**: classification and regression trees.
- **sma**: diagonal linear and quadratic discriminant analysis, naïve Bayes (**stat.diag.da**).

Annotation

Annotation

- One of the largest challenges in analyzing genomic data is associating the experimental data with the available **biological metadata**, e.g., sequence, gene annotation, chromosomal maps, literature.
- Bioconductor provides two main packages for this purpose:
 - **annotate** (end-user);
 - **AnnBuilder** (developer).

WWW resources

- Nucleotide databases: e.g. GenBank.
- Gene databases: e.g. LocusLink, UniGene.
- Protein sequence and structure databases: e.g. SwissProt, Protein DataBank (PDB).
- Literature databases: e.g. PubMed, OMIM.
- Chromosome maps: e.g. NCBI Map Viewer.
- Pathways: e.g. KEGG.
- [Entrez](#) is a search and retrieval system that integrates information from databases at NCBI (National Center for Biotechnology Information).

annotate: matching IDs

Important tasks

- Associate manufacturers or in-house probe identifiers to other available identifiers.

E.g.

Affymetrix IDs → LocusLink LocusID

Affymetrix IDs → GenBank accession number.

- Associate probes with biological data such as chromosomal position, pathways.
- Associate probes with published literature data via PubMed (need PMID).

annotate: matching IDs

Affymetrix identifier HGU95A chips	"41046_s_at"
LocusLink, LocusID	"9203"
GenBank accession #	"X95808"
Gene symbol	"ZNF261"
PubMed, PMID	"10486218" "9205841" "8817323"
Chromosomal location	"X", "Xq13.1"

Annotation data packages

- The Bioconductor project provides [annotation data packages](#), that contain many different mappings to interesting data
 - Mappings between Affy IDs and other probe IDs: [hgu95av2](#) for HGU95Av2 GeneChip series, also, [hgu133a](#), [hu6800](#), [mgu74a](#), [rgu34a](#), [YG](#).
 - Affy CDF data packages.
 - Probe sequence data packages.
- These packages are updated and expanded regularly as new data become available.
- They can be downloaded from the Bioconductor website and also using `installDataPackage`.
- **DPEXplorer**: a widget for interacting with data packages.
- **AnnBuilder**: tools for building annotation data packages.

annotate: matching IDs

- Much of what **annotate** does relies on **matching symbols**.
- This is basically the role of a **hash table** in most programming languages.
- In R, we rely on **environments**.
- The annotation data packages provide R environment objects containing **key** and **value** pairs for the mappings between two sets of probe identifiers.
- Keys can be accessed using the R **ls** function.
- Matching values in different environments can be accessed using the **get** or **multiget** functions.

annotate: matching IDs

```
> library(hgu95a)
> get("41046_s_at", env = hgu95aACCNUM)
[1] "X95808"
> get("41046_s_at", env = hgu95aLOCUSID)
[1] "9203"
> get("41046_s_at", env = hgu95aSYMBOL)
[1] "ZNF261"
> get("41046_s_at", env = hgu95aGENENAME)
[1] "zinc finger protein 261"
> get("41046_s_at", env = hgu95aSUNMFUNC)
[1] "Contains a putative zinc-binding
    motif (MYM)|Proteome"
> get("41046_s_at", env = hgu95aUNIGENE)
[1] "Hs.9568"
```

annotate: matching IDs

```
> get("41046_s_at", env = hgu95aCHR)
[1] "X"
> get("41046_s_at", env = hgu95aCHRLOC)
[1] "66457019@X"
> get("41046_s_at", env = hgu95aCHRORI)
[1] "-@X"
> get("41046_s_at", env = hgu95aMAP)
[1] "Xq13.1"
> get("41046_s_at", env = hgu95aPMID)
[1] "10486218" "9205841" "8817323"
> get("41046_s_at", env = hgu95aGO)
[1] "GO:0003677" "GO:0007275"
```

annotate: matching IDs

- Instead of relying on the general R functions for environments, new user-friendly functions have been written for accessing and working with specific identifiers.
- E.g. `getGO`, `getGODesc`, `getLL`, `getPMID`, `getSYMBOL`.

annotate: matching IDs

```
> getSYMBOL("41046_s_at", data="hgu95a")
41046_s_at
  "ZNF261"
> gg<- getGO("41046_s_at", data="hgu95a")
> getGODesc(gg, "MF")
$"c("GO:0003677", "GO:0007275")"
[1] "DNA binding"
> getLL("41046_s_at", data="hgu95a")
41046_s_at
  9203
> getPMID("41046_s_at", data="hgu95a")
$"41046_s_at"
[1] 10486218  9205841  8817323
```

annotate: querying databases

The **annotate** package provides tools for

- Searching and processing information from various WWW biological databases
 - GenBank,
 - LocusLink,
 - PubMed.
- Regular expression searching of PubMed abstracts.
- Generating nice HTML reports of analyses, with links to biological databases.

annotate: WWW queries

- Functions for querying WWW databases from R rely on the `browseURL` function

```
browseURL("www.r-project.org")
```

Other tools: `HTMLPage` class, `getTDRows`,
`getQueryLink`, `getQuery4UG`, `getQuery4LL`,
`makeAnchor` .

- The `XML` package is used to parse query results.

annotate: querying GenBank

www.ncbi.nlm.nih.gov/Genbank/index.html

- Given a vector of GenBank accession numbers or NCBI UIDs, the **genbank** function
 - opens a browser at the URLs for the corresponding GenBank queries;
 - returns an **XMLdoc** object with the same data.

```
genbank( "X95808" , disp="browser" )
```

<http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?tool=biocductor&cmd=Search&db=Nucleotide&term=X95808>

```
genbank( 1430782 , disp="data" ,  
        type="uid" )
```

annotate: querying LocusLink

www.ncbi.nlm.nih.gov/LocusLink/

- **locuslinkByID**: given one or more LocusIDs, the browser is opened at the URL corresponding to the first gene.

```
locuslinkByID( "9203" )
```

<http://www.ncbi.nlm.nih.gov/LocusLink/LocRpt.cgi?l=9203>

- **locuslinkQuery**: given a search string, the results of the LocusLink query are displayed in the browser.

```
locuslinkQuery( "zinc finger" )
```

<http://www.ncbi.nlm.nih.gov/LocusLink/list.cgi?Q=zinc finger&ORG=Hs&V=0>

- **getQuery4LL**.

annotate: querying PubMed

www.ncbi.nlm.nih.gov

- For any gene there is often a large amount of data available from PubMed.
- The **annotate** package provides the following tools for interacting with PubMed
 - **pubMedAbst**: a class structure for PubMed abstracts in R.
 - **pubmed**: the basic engine for talking to PubMed (`pmidQuery`).

annotate: PubMedAbst class

Class structure for storing and processing
PubMed abstracts in R

- **pmid**
- **authors**
- **abstText**
- **articleTitle**
- **journal**
- **pubDate**
- **abstUrl**

annotate: high-level tools for querying PubMed

- **pm.getabst**: download the specified PubMed abstracts (stored in XML) and create a list of `PubMedAbst` objects.
- **pm.titles**: extract the titles from a list of PubMed abstracts.
- **pm.abstGrep**: regular expression matching on the abstracts.

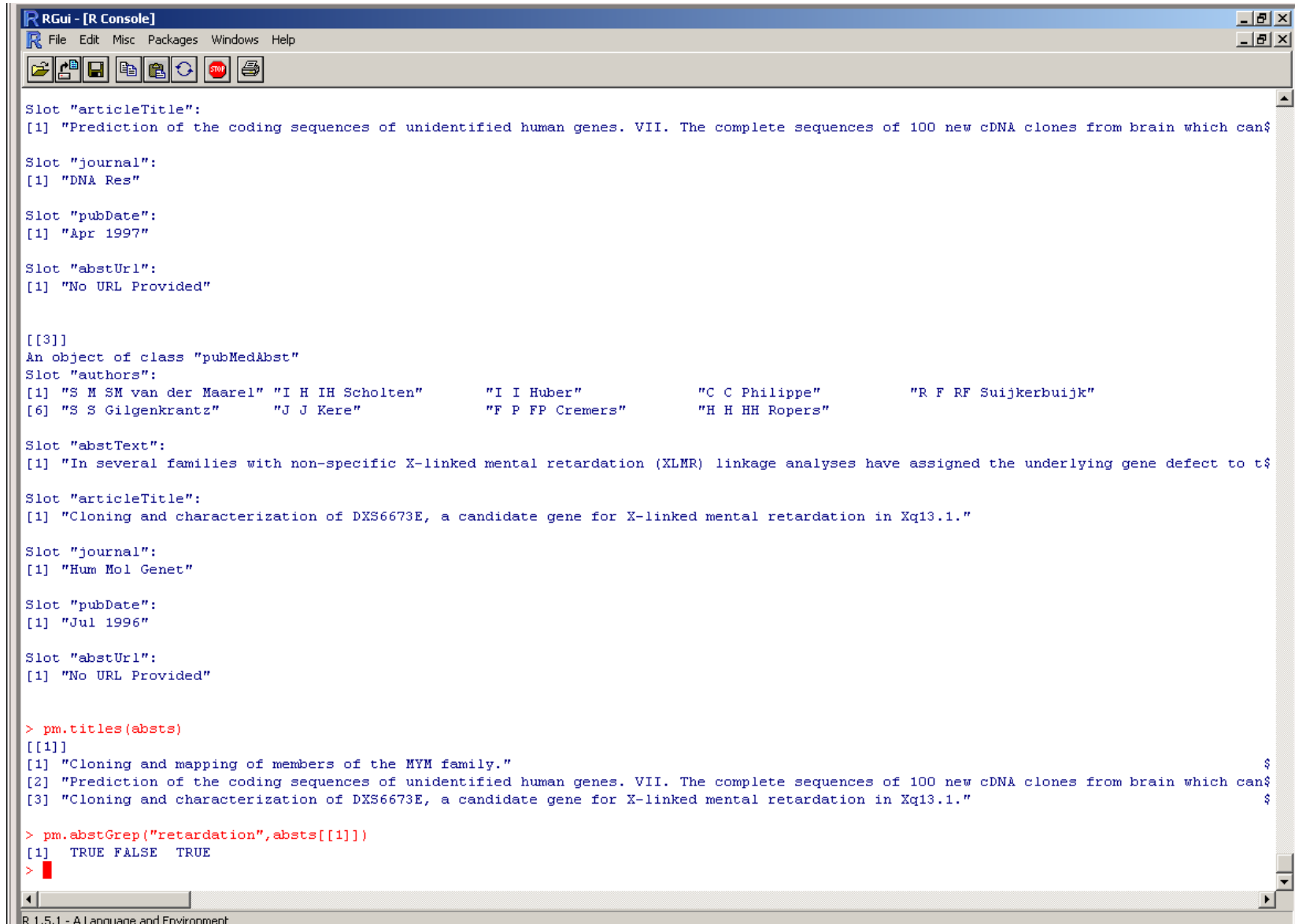
annotate: PubMed example

```
pmid <-get("41046_s_at", env=hgu95aPMID)  
pubmed(pmid, disp="browser")
```

http://www.ncbi.nih.gov/entrez/query.fcgi?tool=bioconductor&cmd=Retrieve&db=PubMed&list_uids=10486218%2c9205841%2c8817323

```
absts <- pm.getabst("41046_s_at",  
  base="hgu95a")  
pm.titles(absts)  
pm.abstGrep("retardation", absts[[1]])
```

annotate: PubMed example



```
RGui - [R Console]
File Edit Misc Packages Windows Help

Slot "articleTitle":
[1] "Prediction of the coding sequences of unidentified human genes. VII. The complete sequences of 100 new cDNA clones from brain which can$

Slot "journal":
[1] "DNA Res"

Slot "pubDate":
[1] "Apr 1997"

Slot "abstUrl":
[1] "No URL Provided"

[[3]]
An object of class "pubMedAbst"
Slot "authors":
[1] "S M SM van der Maarel" "I H IH Scholten" "I I Huber" "C C Philippe" "R F RF Suijkerbuijk"
[6] "S S Gilgenkrantz" "J J Kere" "F P FP Cremers" "H H HH Ropers"

Slot "abstText":
[1] "In several families with non-specific X-linked mental retardation (XLMR) linkage analyses have assigned the underlying gene defect to t$

Slot "articleTitle":
[1] "Cloning and characterization of DXS6673E, a candidate gene for X-linked mental retardation in Xq13.1."

Slot "journal":
[1] "Hum Mol Genet"

Slot "pubDate":
[1] "Jul 1996"

Slot "abstUrl":
[1] "No URL Provided"

> pm.titles(absts)
[[1]]
[1] "Cloning and mapping of members of the MYM family."
[2] "Prediction of the coding sequences of unidentified human genes. VII. The complete sequences of 100 new cDNA clones from brain which can$
[3] "Cloning and characterization of DXS6673E, a candidate gene for X-linked mental retardation in Xq13.1."

> pm.abstGrep("retardation", absts[[1]])
[1] TRUE FALSE TRUE
>
```

R 1.5.1 - A Language and Environment

annotate: PubMed HTML report

- The new function `pmAbst2HTML` takes a list of `pubMedAbst` objects and generates an HTML report with the titles of the abstracts and links to their full page on PubMed.

```
pmAbst2HTML( absts[[1]], filename="pm.html" )
```

BioConductor Abstract List - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Shop Stop

Bookmarks Location: file:///C:/Sandrine/Current/Talks/EMBO03/pm.html What's Related

Google Sandrine Dudoit Welcome to Bioc PH 240D - Sprin Group In Biosta Berkeley Progra Home Page, Stat

BioConductor Abstract List

Article Title	Publication Date
Conditional targeting of the DNA repair enzyme hOGG1 into mitochondria.	Nov 2002
Inter-individual variation, seasonal variation and close correlation of OGG1 and ERCC1 mRNA levels in full blood from healthy volunteers.	Sep 2002
A limited association of OGG1 Ser326Cys polymorphism for adenocarcinoma of the lung.	May 2002
Protection of human lung cells against hyperoxia using the DNA base excision repair genes hOgg1 and Fpg.	Jul 2002
The human OGG1 DNA repair enzyme and its association with orolaryngeal cancer risk.	Jul 2002
Human OGG1 undergoes serine phosphorylation and associates with the nuclear matrix and mitotic chromatin in vivo.	Jun 2002
hOGG1 Ser(326)Cys polymorphism and modification by environmental factors of stomach cancer risk in Chinese.	Jun 2002
Association of the hOGG1 Ser326Cys polymorphism with lung cancer risk.	Apr 2002
Reciprocal "flipping" underlies substrate recognition and catalytic activation by the human 8-oxo-guanine DNA glycosylase.	Mar 2002
Expression of 8-oxoguanine DNA glycosylase is reduced and associated with neurofibrillary tangles in Alzheimer's disease brain.	Jan 2002
Structure and chromosome location of human OGG1.	Month 1999
Expression and differential intracellular localization of two major forms of human 8-oxoguanine DNA glycosylase encoded by alternatively spliced OGG1 mRNAs.	May 1999
Genetic polymorphisms and alternative splicing of the hOGG1 gene, that is involved in the repair of 8-hydroxyguanine in damaged DNA.	Jun 1998
Augmented expression of a human gene for 8-oxoguanine DNA glycosylase (MutM) in B lymphocytes of the dark zone in lymph node germinal centers.	Nov 1997
Opposite base-dependent reactions of a human base excision repair enzyme on DNA containing 7,8-dihydro-8-oxoguanine and abasic sites.	Oct 1997
Molecular cloning and functional expression of a human cDNA encoding the antimutator enzyme 8-hydroxyguanine-DNA glycosylase.	Jul 1997
Cloning and characterization of hOGG1, a human homolog of the OGG1 gene of Saccharomyces cerevisiae.	Jul 1997

Document: Done

pmAbst2html
function from
annotate package

pm.html

annotate: analysis reports

- A simple interface, [ll.htmlpage](#), can be used to generate an HTML report of analysis results.
- The page consists of a table with one row per gene, with links to LocusLink.
- Entries can include various gene identifiers and statistics.

BioConductor Gene Listing

Golub et al. data, genes with permutation maxT adjusted p-value < 0.01

Locus Link Genes

LocusID	Gene name	Chromosome	ALL mean	AML mean	t-statistic	raw p-value	adj p-value
7791	X95735_at	7	-0.295	1.59	-10.6	2e-05	2e-05
1471	M27891_at	20	-0.81	2.08	-9.78	2e-05	2e-05
2184	M55150_at	15	0.488	1.24	-8.03	2e-05	0.00014
4067	M16038_at	8	-0.284	1.1	-7.98	2e-05	0.00016
334	L09209_s_at	11	-0.162	1.36	-7.97	2e-05	2e-04
6929	M31523_at	19	0.855	-0.391	7.55	2e-05	5e-04
5928	X74262_at	1	0.869	-0.565	7.42	2e-05	0.00078
7155	Z15115_at	3	1.94	0.945	7.35	2e-05	0.001
26999	L47738_at	5	0.734	-0.779	7.31	2e-05	0.00114
4602	U22376_cds2_s_at	6	1.86	0.294	7.28	2e-05	0.00116
65108	HG1612-HT1612_at	1	1.91	0.888	7.11	2e-05	0.0017
34	M91432_at	1	0.431	-0.771	7.08	2e-05	0.0018
5925	L41870_at	13	-0.438	-1.3	7.08	2e-05	0.0018
546	U72936_s_at	NA	-0.097	-1.07	7.07	2e-05	0.0018
7430	X51521_at	6	1.92	1.07	7.06	2e-05	0.00186
4056	U50136_mal_at	5	0.71	1.51	-6.97	2e-05	0.00232
54741	Y12670_at	1	-0.167	0.892	-6.96	2e-05	0.00238
7203	X74801_at	1	0.611	-0.183	6.95	2e-05	0.00238
3576	Y00787_s_at	4	-0.371	2.32	-6.87	2e-05	0.00288
6709	J05243_at	9	0.413	-0.982	6.86	2e-05	0.00288
1725	U26266_s_at	19	-0.209	-1.16	6.85	4e-05	0.00294
3205	U82759_at	7	-0.64	0.504	-6.82	2e-05	0.00306
945	M23197_at	19	-0.881	0.354	-6.79	2e-05	0.0033
1509	M63138_at	11	1.21	2.12	-6.77	2e-05	0.00344
6955	M12959_s_at	14	1.13	0.132	6.76	2e-05	0.00352
967	X62654_mal_at	12	0.0513	1.33	-6.76	2e-05	0.00352
5341	X07743_at	2	-0.959	0.535	-6.74	2e-05	0.00378
140465	M31211_s_at	12	0.108	-0.953	6.71	2e-05	0.00404
7336	U62136_at	8	-0.163	-0.92	6.68	2e-05	0.00428
3660	X15949_at	4	-0.541	-1.33	6.61	2e-05	0.00492
9655	U72944_at	14	0.076	0.259	6.61	2e-05	0.00492

l1.htmlpage
function from
annotate
package

[genelist.html](#)

annotate: chromLoc class

Location information for one gene

- **chrom**: chromosome name.
- **position**: starting position of the gene in bp.
- **strand**: chromosome strand +/-.

annotate: chromLocation class

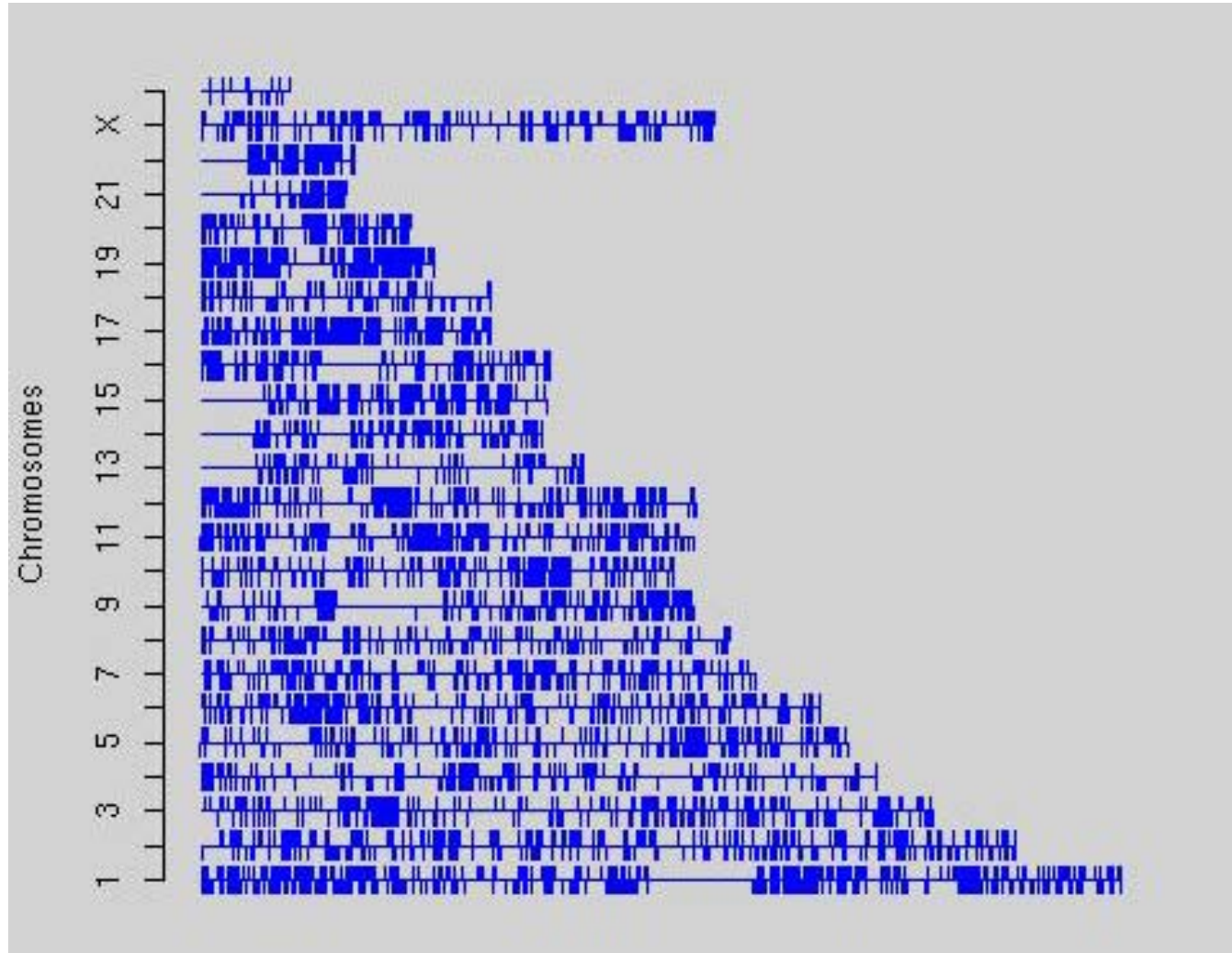
Location information for a set of genes

- **species**: species that the genes correspond to.
- **datSource**: source of the gene location data.
- **nChrom**: number of chromosomes for the species.
- **chromNames**: chromosome names.
- **chromLocs**: starting position of the genes in bp.
- **chromLengths**: length of each chromosome in bp.
- **geneToChrom**: hash table translating gene IDs to location.

Function **buildChromClass**.

Visualization

geneplotter: cPlot



geneplotter: a longChrom

