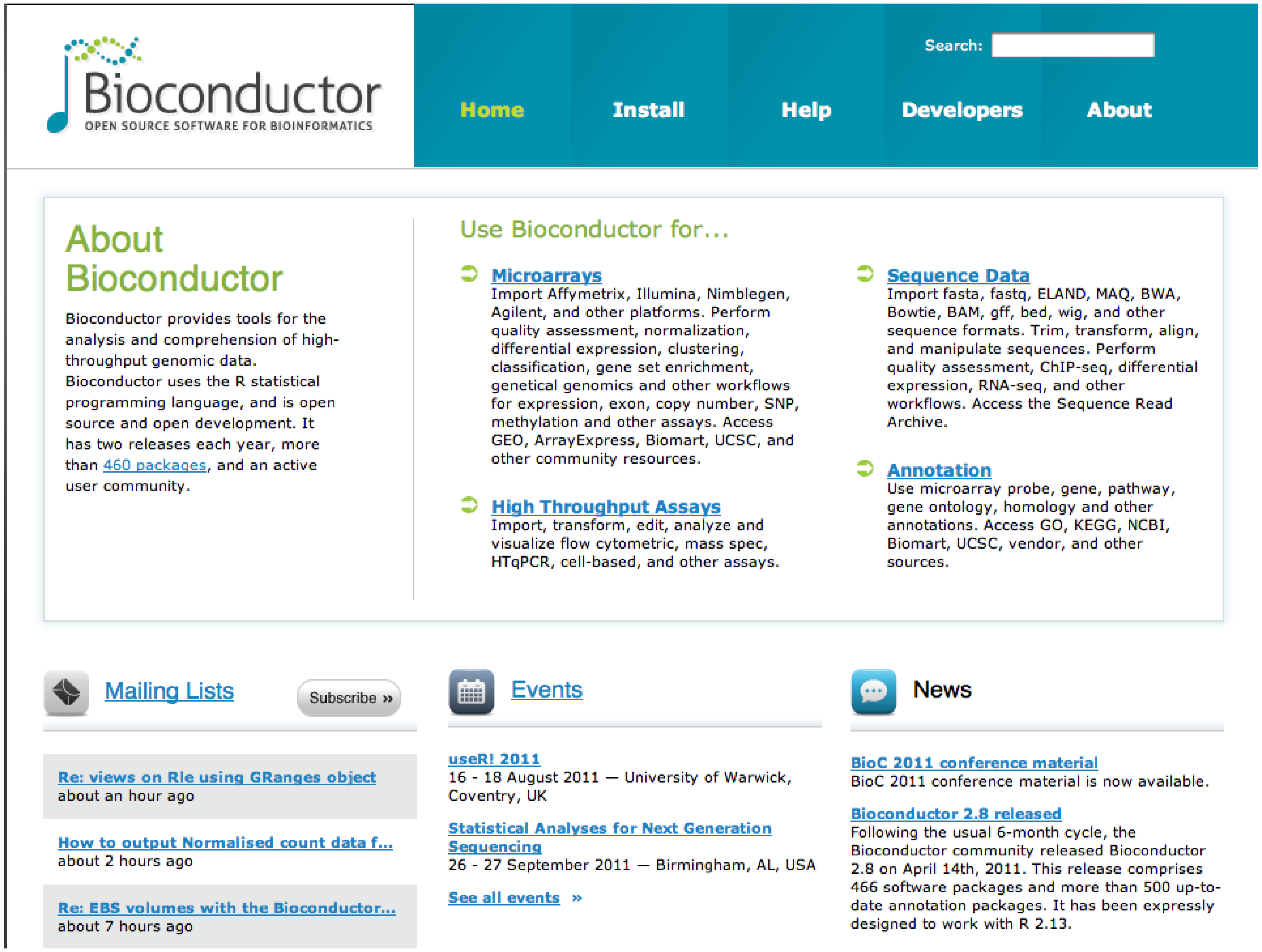
8. Bioconductor Intro

Ken Rice
Thomas Lumley

Universities of Washington and Auckland

Seattle, July 2012

What is Bioconductor?



About Bioconductor

Bioconductor provides tools for the analysis and comprehension of high-throughput genomic data. Bioconductor uses the R statistical programming language, and is open source and open development. It has two releases each year, more than 460 packages, and an active user community.

Use Bioconductor for...

Microarrays
Import Affymetrix, Illumina, Nimblegen, Agilent, and other platforms. Perform quality assessment, normalization, differential expression, clustering, classification, gene set enrichment, genetical genomics and other workflows for expression, exon, copy number, SNP, methylation and other assays. Access GEO, ArrayExpress, Biomart, UCSC, and other community resources.

High Throughput Assays
Import, transform, edit, analyze and visualize flow cytometric, mass spec, HTqPCR, cell-based, and other assays.

Sequence Data
Import fasta, fastq, ELAND, MAQ, BWA, Bowtie, BAM, gff, bed, wig, and other sequence formats. Trim, transform, align, and manipulate sequences. Perform quality assessment, ChIP-seq, differential expression, RNA-seq, and other workflows. Access the Sequence Read Archive.

Annotation
Use microarray probe, gene, pathway, gene ontology, homology and other annotations. Access GO, KEGG, NCBI, Biomart, UCSC, vendor, and other sources.

Mailing Lists Subscribe »

Re: views on Rle using GRanges object
about an hour ago

How to output Normalised count data f...
about 2 hours ago

Re: EBS volumes with the Bioconductor...
about 7 hours ago

Events

useR! 2011
16 - 18 August 2011 — University of Warwick, Coventry, UK

Statistical Analyses for Next Generation Sequencing
26 - 27 September 2011 — Birmingham, AL, USA

See all events »

News

BioC 2011 conference material
BioC 2011 conference material is now available.

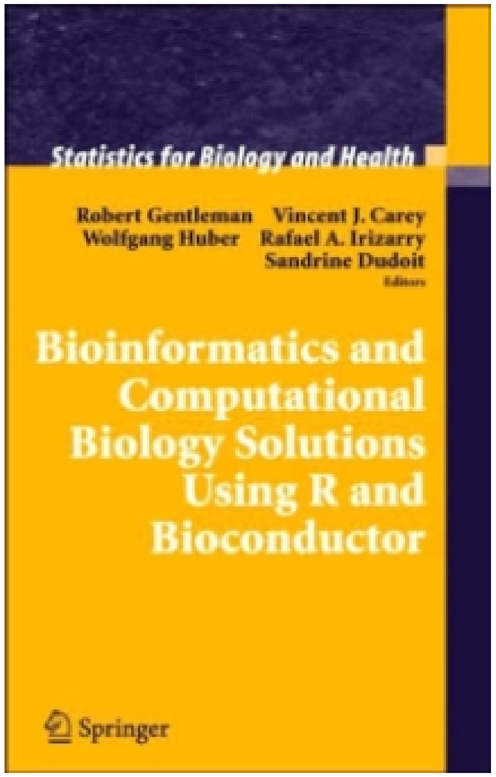
Bioconductor 2.8 released
Following the usual 6-month cycle, the Bioconductor community released Bioconductor 2.8 on April 14th, 2011. This release comprises 466 software packages and more than 500 up-to-date annotation packages. It has been expressly designed to work with R 2.13.

What is Bioconductor?

- `www.bioconductor.org`
- Software project for analysis of genomic data – and related tools, resources/datasets
- **Open source** and **Open development**
- **Free**

You could use commercial software; but experts typically write `R` code first. Also, the help manuals are not a sales pitch and encourage appropriate use.

Bioconductor basics


Statistics for Biology and Health

Robert Gentleman Vincent J. Carey
Wolfgang Huber Rafael A. Irizarry
Sandrine Dudoit
Editors

Bioinformatics and Computational Biology Solutions Using R and Bioconductor

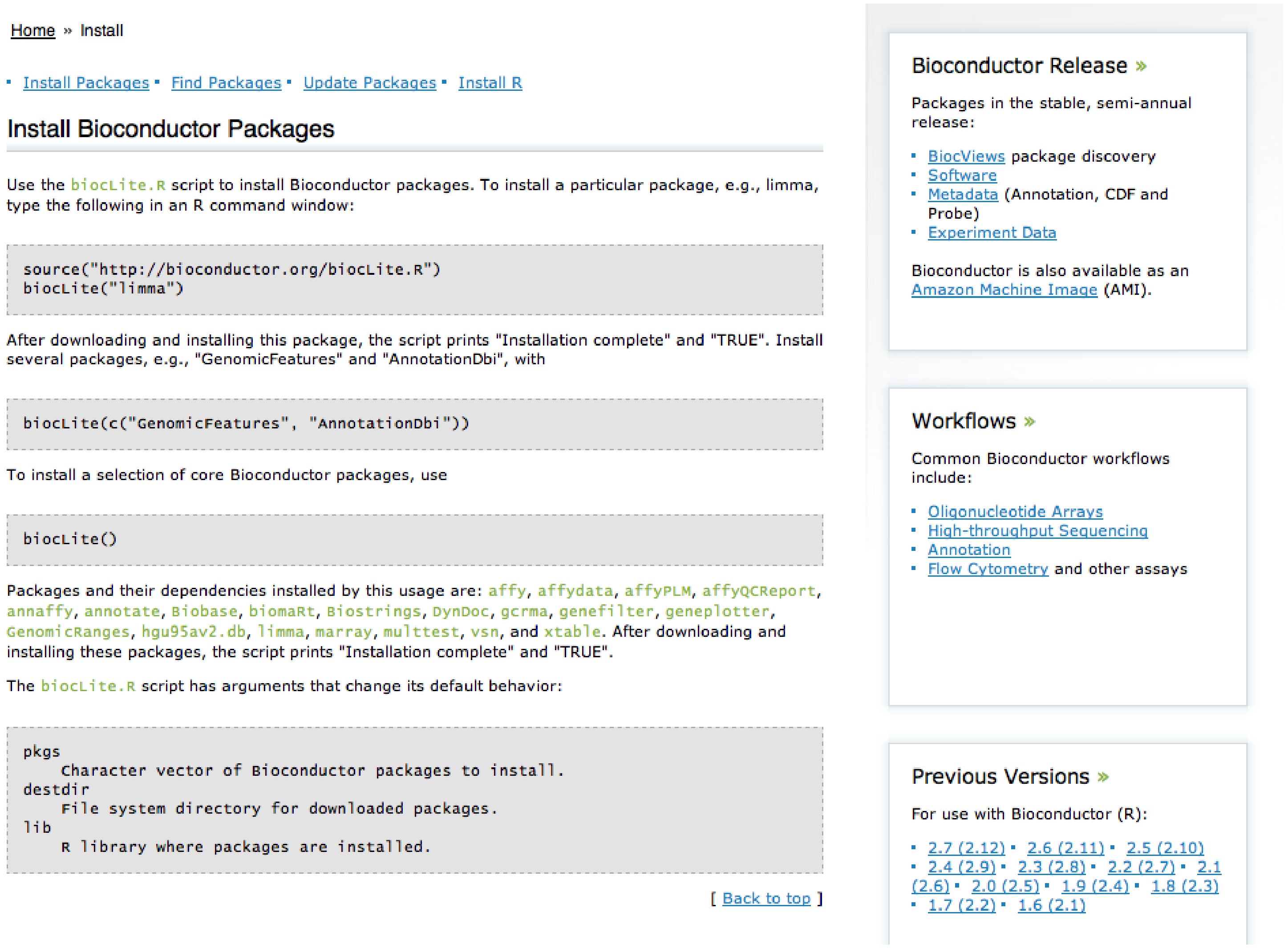
Springer

- Begun in 2001, based at Harvard and now FHCRC (Seattle)
- A large collection of `R` packages (they also convert good software to `R`)
- Far too much for our little course!

We'll give examples of what Bioconductor can do, and how to learn more. Gentleman et al (above) is a helpful reference text

Bioconductor basics

Getting started...



Home » Install

▪ Install Packages ▪ Find Packages ▪ Update Packages ▪ Install R

Install Bioconductor Packages

Use the biocLite.R script to install Bioconductor packages. To install a particular package, e.g., limma, type the following in an R command window:

```
source("http://bioconductor.org/biocLite.R")
biocLite("limma")
```

After downloading and installing this package, the script prints "Installation complete" and "TRUE". Install several packages, e.g., "GenomicFeatures" and "AnnotationDbi", with

```
biocLite(c("GenomicFeatures", "AnnotationDbi"))
```

To install a selection of core Bioconductor packages, use

```
biocLite()
```

Packages and their dependencies installed by this usage are: affy, affydata, affyPLM, affyQCReport, annaffy, annotate, Biobase, biomaRt, Biostrings, DynDoc, gcrma, genefilter, geneplotter, GenomicRanges, hgu95av2.db, limma, marray, multtest, vsn, and xtable. After downloading and installing these packages, the script prints "Installation complete" and "TRUE".

The biocLite.R script has arguments that change its default behavior:

```
pkgs
    Character vector of Bioconductor packages to install.
destdir
    File system directory for downloaded packages.
lib
    R library where packages are installed.
```

[Back to top]

Bioconductor Release »

Packages in the stable, semi-annual release:

- BiocViews package discovery
- Software
- Metadata (Annotation, CDF and Probe)
- Experiment Data

Bioconductor is also available as an Amazon Machine Image (AMI).

Workflows »

Common Bioconductor workflows include:

- Oligonucleotide Arrays
- High-throughput Sequencing
- Annotation
- Flow Cytometry and other assays

Previous Versions »

For use with Bioconductor (R):

▪ 2.7 (2.12) ▪ 2.6 (2.11) ▪ 2.5 (2.10) ▪ 2.4 (2.9) ▪ 2.3 (2.8) ▪ 2.2 (2.7) ▪ 2.1 (2.6) ▪ 2.0 (2.5) ▪ 1.9 (2.4) ▪ 1.8 (2.3) ▪ 1.7 (2.2) ▪ 1.6 (2.1)

Bioconductor basics

```
> source("http://bioconductor.org/biocLite.R")
> biocLite()
```

installs the following general-purpose libraries;

`Biobase, IRanges, AnnotationDbi`

... then you use e.g. `library("Biobase")` as before. (NB older versions used to download much more than this)

`vignette(package="Biobase")` tells you to look at `vignette("esApply")` for a worked example – a very helpful introduction. (Or use e.g. `openVignette()`, which is in the `Biobase` package itself)

Bioconductor basics

To get other packages, use the `source()` command as before, then use e.g.

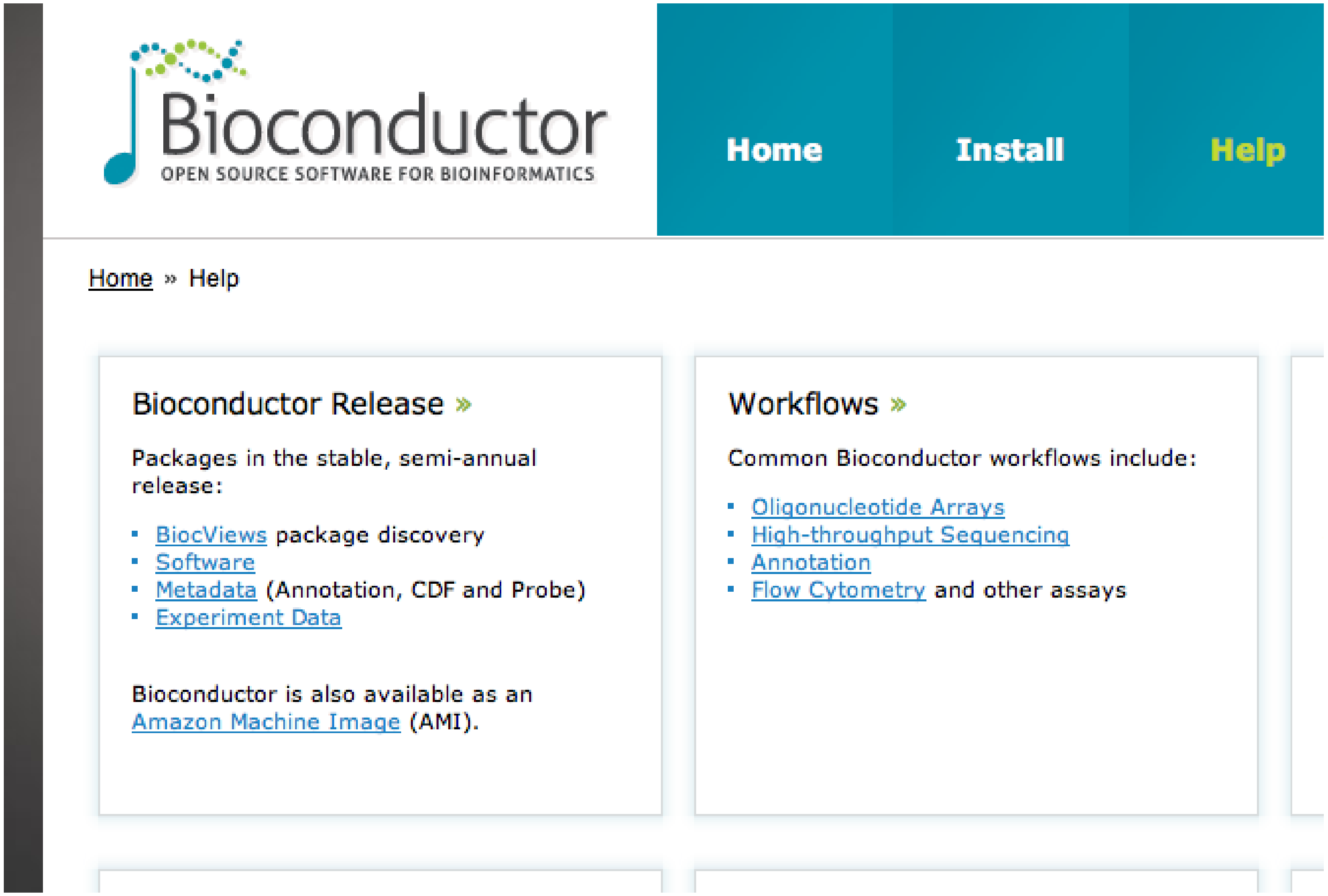
```
biocLite("SNPchip")
biocLite(c("limma", "siggenes"))
```

You do not need to type `biocLite()` again (even in a new R session). This would install the general-purpose packages again – which is harmless, but a waste of time.

Note; if, due to access privileges, you need to write to non-default directories, follow the onscreen commands and then start again. On Windows, 'Run as Administrator' may cut out this step.

What to install?

Back to the front page – click 'Help'



Home » Help

Bioconductor Release »

Packages in the stable, semi-annual release:

- BiocViews package discovery
- Software
- Metadata (Annotation, CDF and Probe)
- Experiment Data

Bioconductor is also available as an Amazon Machine Image (AMI).

Workflows »

Common Bioconductor workflows include:

- Oligonucleotide Arrays
- High-throughput Sequencing
- Annotation
- Flow Cytometry and other assays

What to install?

- **Software** – probably what you want
- **Metadata** – e.g. annotation data, probe sequence data for microarrays of different types
- **Experiment data** – e.g. datasets from `hapmap.org`, some expression datasets

Simple QC graphics

The "splots" package plots values from 96 or 384-well plates, for QC purposes

First, install it

```
biocLite("splots")
```

Then load into R

```
library("splots")
```

There is a single function: `plotScreen()` for displaying the results

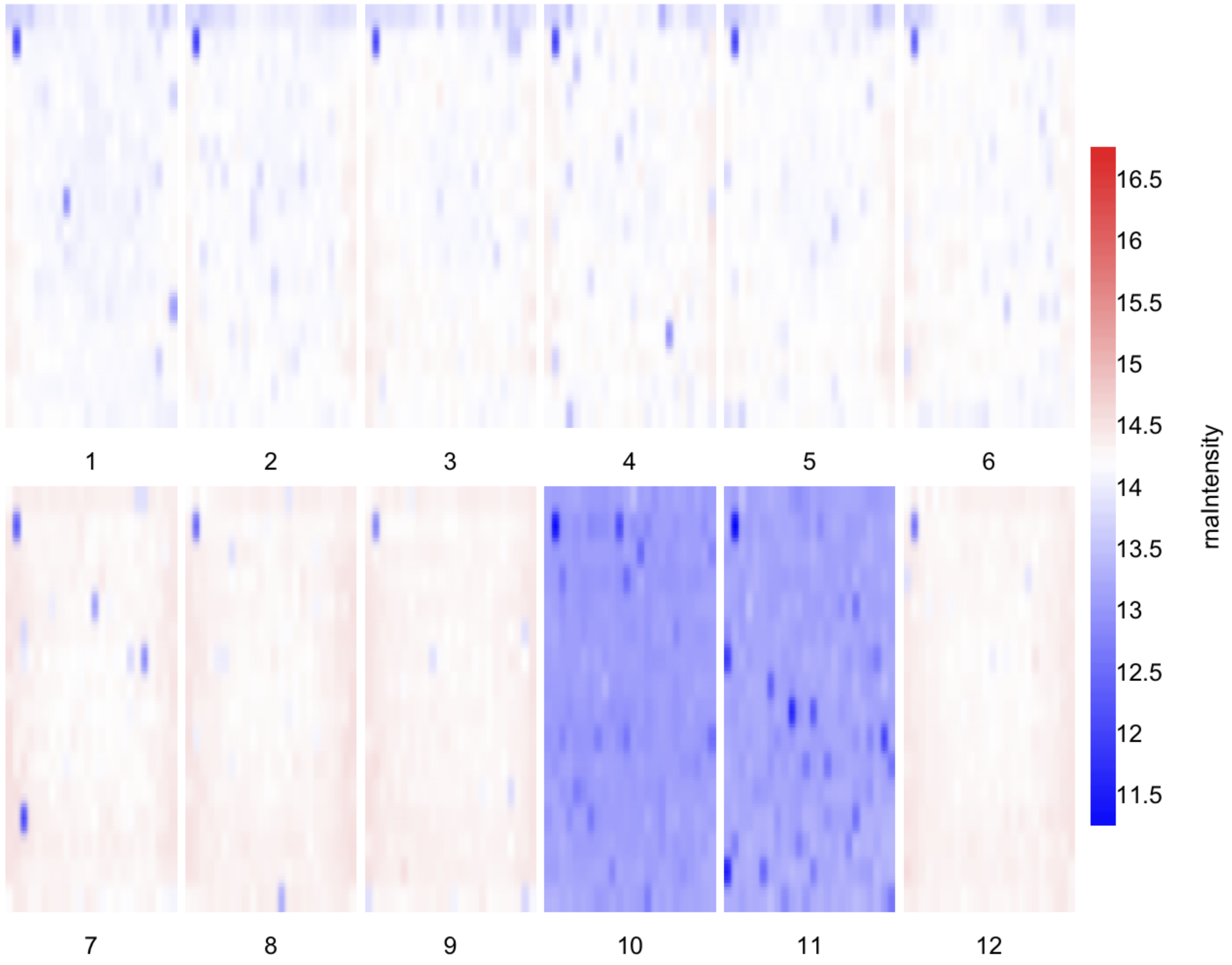
Example

The file "drosophila.rda" contains 12 of 114 plates from a RNAi gene-knockout study in fruit flies. Each spot represents a gene, and the intensity is low if knockout of that gene is lethal (data from the "RNAither" package)

```
> load("drosophila.rda")
> ls()
[1] "rnai"
> str(rnai)
List of 12
 $ : num [1:384] 13.7 13.8 13.8 13.7 14 ...
 $ : num [1:384] 13.8 13.8 13.6 13.8 13.9 ...
> plotScreen(rnai) # see output on next slide
```

The positive controls in the same position each plate are clear, and there are obvious plate effects that you might need to correct by normalization.

Example



1
2
3
4
5
6
7
8
9
10
11
12
16.5
16
15.5
15
14.5
14
13.5
13
12.5
12
11.5
rmaIntensity

GWAS analysis

Genome-Wide Association Studies (GWAS) are currently popular – typically, these genotype e.g. 1M SNPs on several thousand subjects in (large) established studies

- Usually on 1000's of subjects
- 'Simple' t-tests, regressions, for each SNP (like microarrays)
- 1M *anything* takes a long time! (up to 72 hours)
- Just **loading** big datasets is non-trivial – but some tools are available

GWAS analysis

`snpStats` is a Bioconductor package for GWAS analysis – maintained by David Clayton (analysis lead on Wellcome Trust)

```
> source("http://bioconductor.org/biocLite.R") # in a new session
> biocLite("snpStats")
> library("snpStats")
> data(for.exercise)
> ls()
[1] "snp.support"     "snps.10"           "subject.support"
```

A ‘little’ case-control dataset (Chr 10) based on HapMap – three objects; `snp.support`, `subject.support` and `snps.10`

GWAS analysis

```
> summary(snp.support)
   chromosome    position              A1         A2
 Min.    :10   Min.    :   101955   A:14019   C: 2349
 1st Qu.:10   1st Qu.: 28981867   C:12166   G:12254
 Median :10   Median : 67409719   G: 2316   T:13898
 Mean    :10   Mean    : 66874497
 3rd Qu.:10   3rd Qu.:101966491
 Max.    :10   Max.    :135323432
> summary(subject.support)
       cc           stratum
 Min.    :0.0    CEU     :494
 1st Qu.:0.0    JPT+CHB:506
 Median :0.5
 Mean    :0.5
 3rd Qu.:1.0
 Max.    :1.0
```

GWAS analysis

```
> show(snps.10)
A SnpMatrix with  1000 rows and  28501 columns
Row names:  jpt.869 ... ceu.464
Col names:  rs7909677 ... rs12218790
> summary(snps.10)
$rows
   Call.rate      Certain.calls Heterozygosity
 Min.   :0.9879   Min.   :1     Min.   :0.0000
 1st Qu.:0.9896   1st Qu.:1     1st Qu.:0.2993
 Median :0.9900   Median :1     Median :0.3078
 Mean   :0.9900   Mean   :1     Mean   :0.3074
 3rd Qu.:0.9904   3rd Qu.:1     3rd Qu.:0.3159
 Max.   :0.9919   Max.   :1     Max.   :0.3386
```

[continues]

GWAS analysis

```
$cols
     Calls          Call.rate      Certain.calls        RAF               MAF        
 Min.    : 975   Min.    :0.975   Min.    :1      Min.    :0.0000   Min.    :0.0000  
 1st Qu.: 988   1st Qu.:0.988   1st Qu.:1      1st Qu.:0.2302   1st Qu.:0.1258  
 Median : 990   Median :0.990   Median :1      Median :0.5030   Median :0.2315  
 Mean    : 990   Mean    :0.990   Mean    :1      Mean    :0.5001   Mean    :0.2424  
 3rd Qu.: 992   3rd Qu.:0.992   3rd Qu.:1      3rd Qu.:0.7671   3rd Qu.:0.3576  
 Max.    :1000   Max.    :1.000   Max.    :1      Max.    :1.0000   Max.    :0.5000  
      P.AA              P.AB              P.BB             z.HWE         
 Min.    :0.00000   Min.    :0.0000   Min.    :0.00000   Min.    :-21.9725  
 1st Qu.:0.06559   1st Qu.:0.2080   1st Qu.:0.06465   1st Qu.: -2.8499  
 Median :0.26876   Median :0.3198   Median :0.27492   Median : -1.1910  
 Mean    :0.34617   Mean    :0.3074   Mean    :0.34647   Mean    : -1.8610  
 3rd Qu.:0.60588   3rd Qu.:0.4219   3rd Qu.:0.60362   3rd Qu.: -0.1014  
 Max.    :1.00000   Max.    :0.5504   Max.    :1.00000   Max.    :  3.7085  
                                                         NA's    :4        
```

- 28501 SNPs, all with Allele 1, Allele 2
- 1000 subjects, 500 controls (`cc=0`) and 500 cases (`cc=1`)
- **Far too much** data for a regular `summary()` of `snps.10` – even in this small example

GWAS analysis

We'll use just the column summaries, and a (mildly) 'clean' subset;

```
> snpsum <- col.summary(snps.10)
> use <- with(snpsum, MAF > 0.01 & z.HWE^2 < 200)

> table(use)
use
FALSE  TRUE
  317 28184
```

GWAS analysis

Now do single-SNP tests for each SNP, and extract the p-value for each SNP, along with its location;

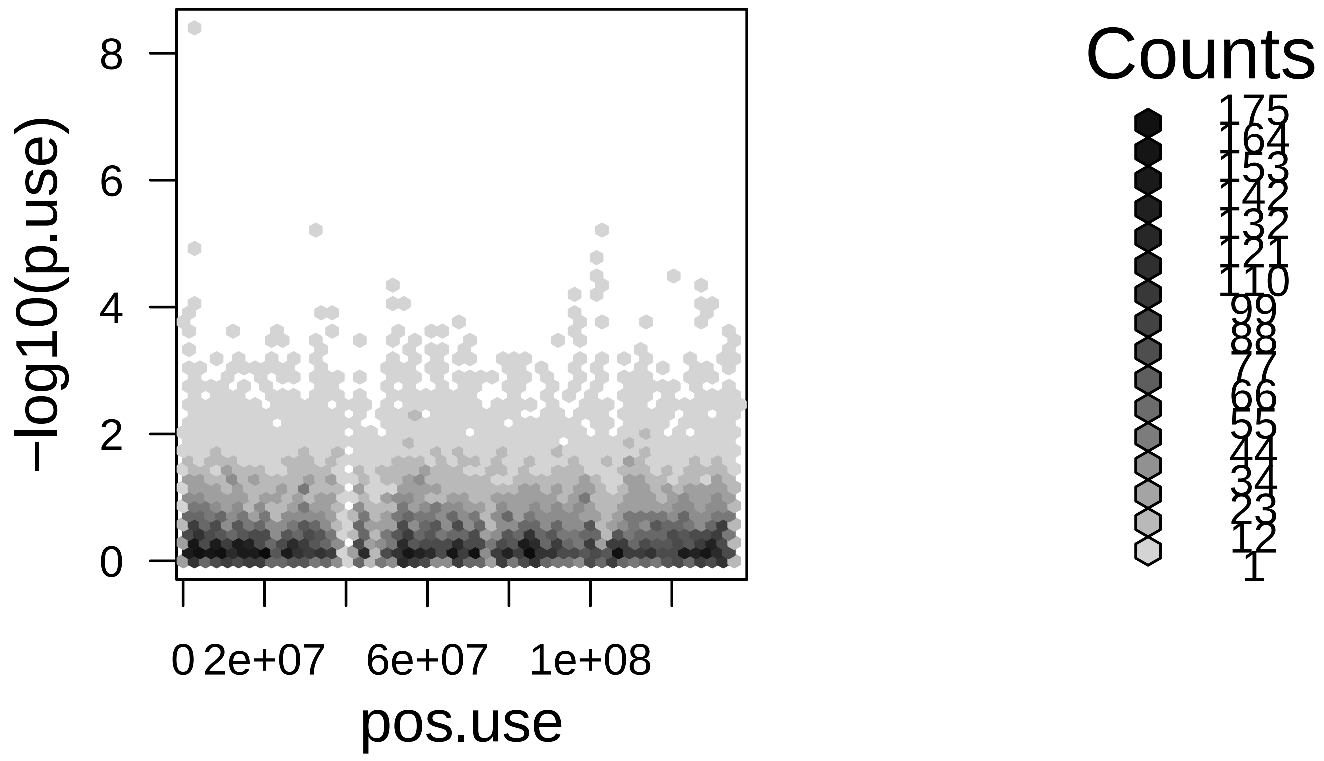
```
tests <- single.snp.tests(cc, data = subject.support,
+ snp.data = snps.10)
```

```
pos.use <- snp.support$position[use]
p.use   <- p.value(tests, df=1)[use]
```

We'd usually give a table of 'top hits,' but...

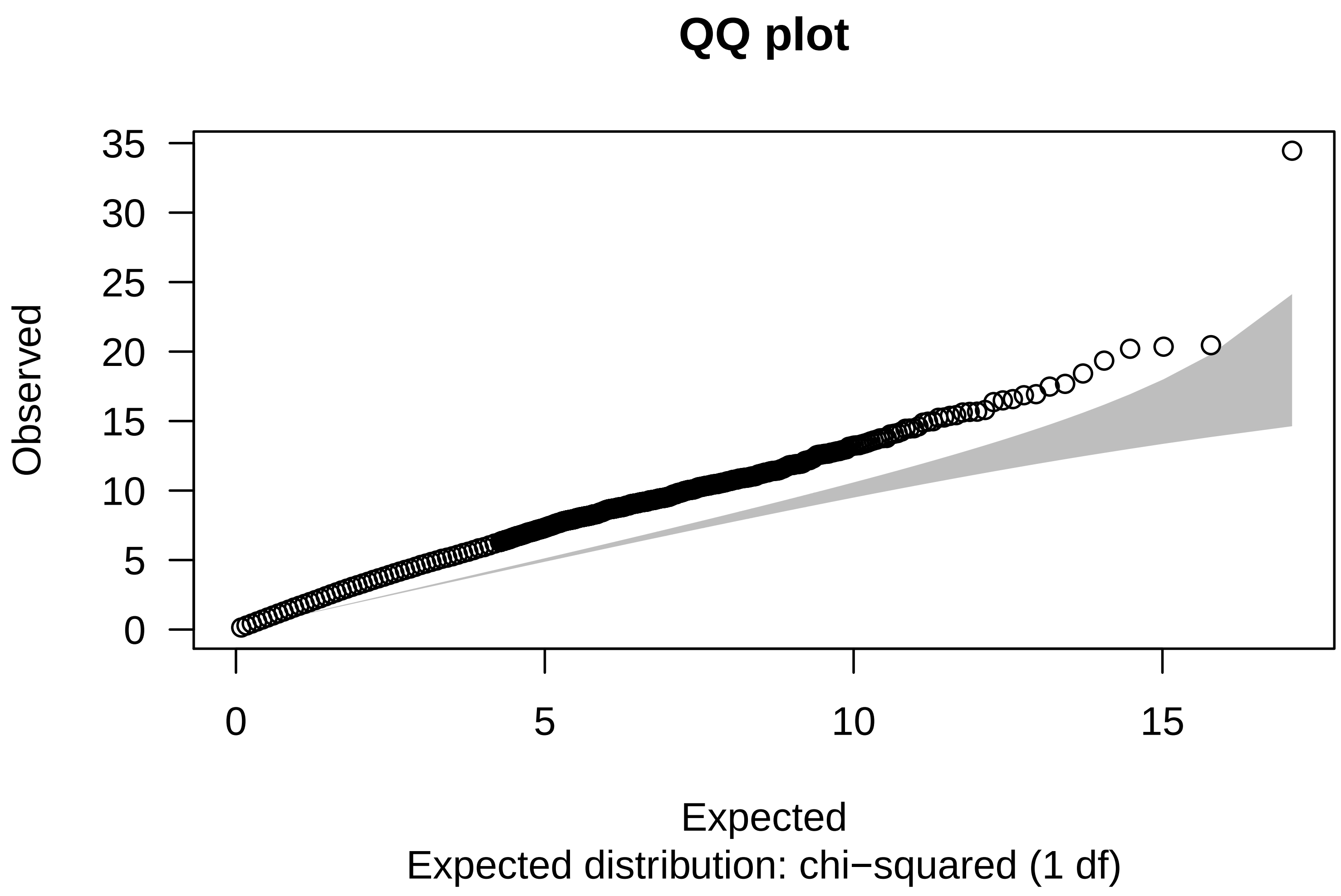
GWAS analysis

```
plot(hexbin(pos.use, -log10(p.use), xbin = 50))
```



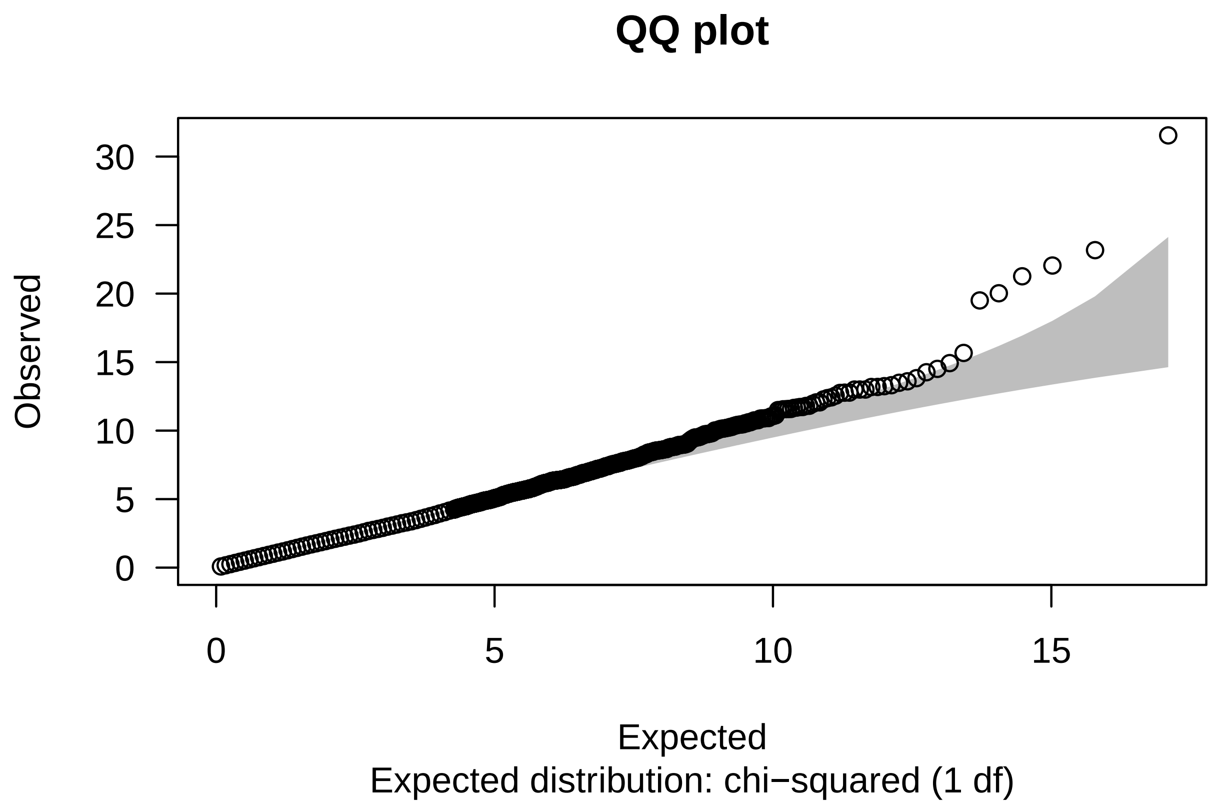
GWAS analysis

```
qq.chisq(chi.squared(tests, df=1)[use], df=1)
```



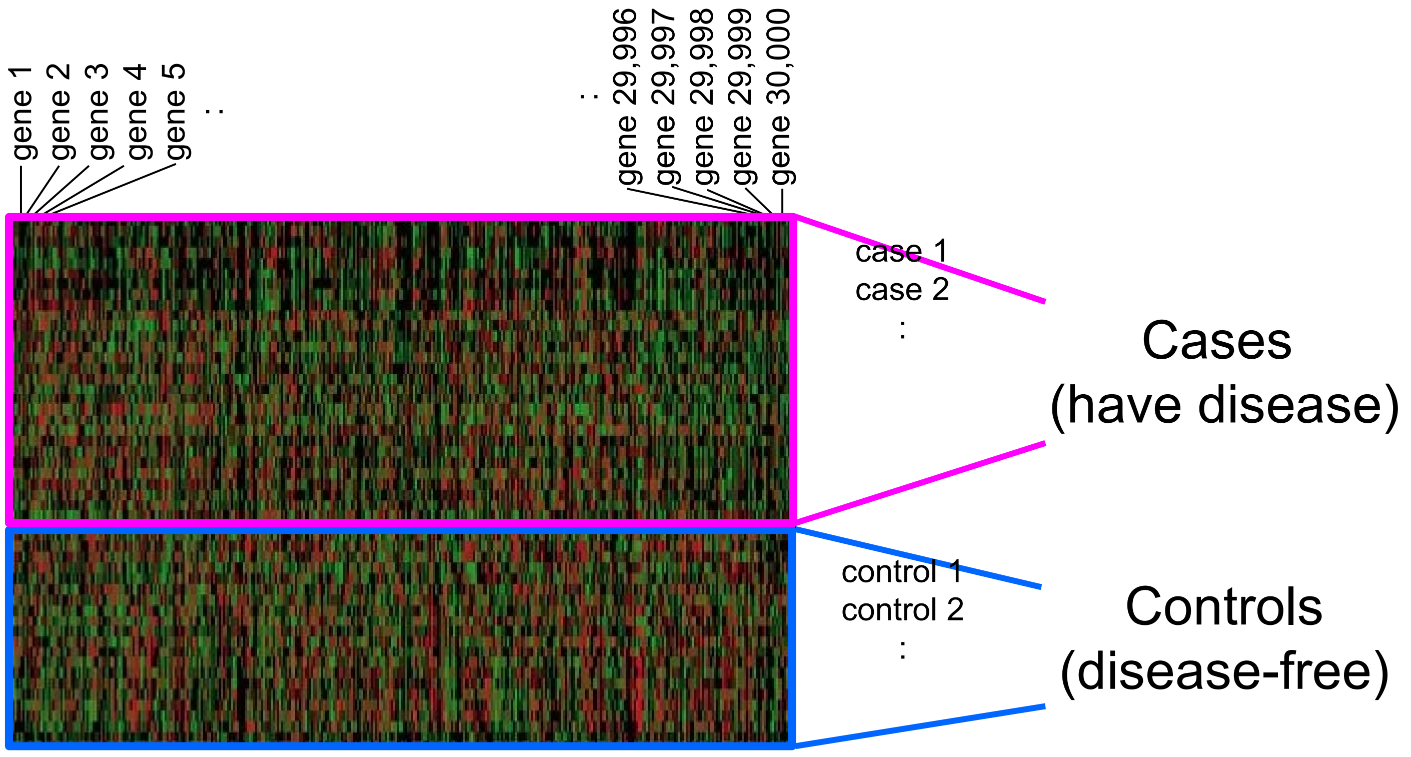
GWAS analysis

```
tests2 <- single.snp.tests(cc,stratum=stratum,data=subject.support,
+ snp.data = snps.10)
qq.chisq(chi.squared(tests2, 1)[use], 1)
```



Significance Analysis of Microarrays

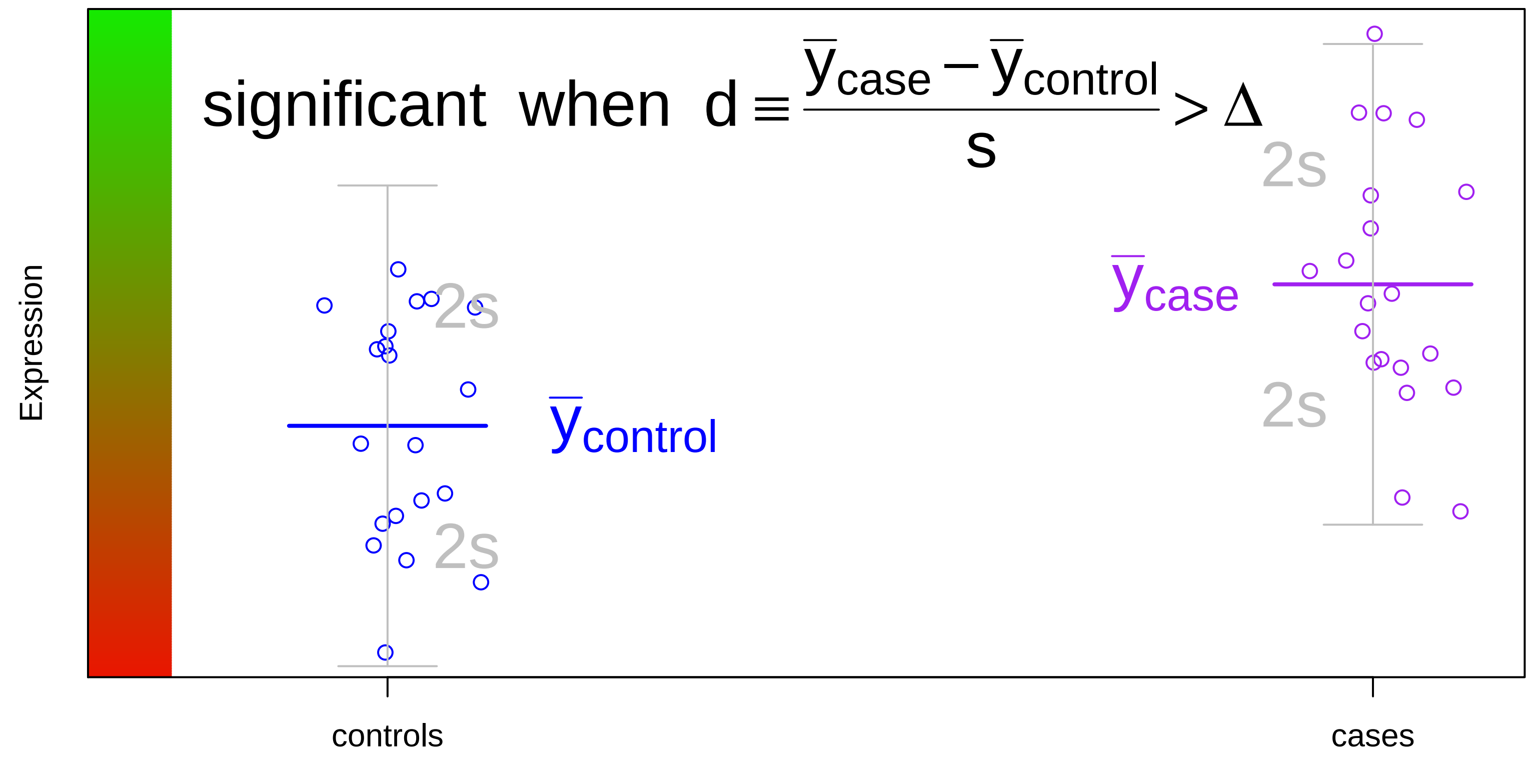
Significance Analysis of Microarrays (SAM) is a popular method (Tusher *et al* 2001) which identifies differentially expressed genes



i.e. large red/green difference between cases and controls

Significance Analysis of Microarrays

Why so popular? Here's the traditional method;



Do this $\times$30,000 genes; d in each is **quite unstable**. Small values of s give large d, which may give **false positive** results

Significance Analysis of Microarrays

SAM has a quick fix for this problem;

Traditional	SAM
$d_i = \dfrac{\bar{y}_{i,\text{case}} - \bar{y}_{i,\text{control}}}{s_i}$	$d_i = \dfrac{\bar{y}_{i,\text{case}} - \bar{y}_{i,\text{control}}}{s_i + s_0}$

For each gene (each i), SAM's s_0 **borrows strength** from the other genes.

SAM (and `siggenes`) then does some clever permutation testing to produce False Discovery Rates

Significance Analysis of Microarrays

Golub et al (1999) give differential expression for 3,051 genes, in 27 ‘controls’ (ALL) and 11 ‘cases’ (AML)

```
> library("multtest")
> data(golub)
> table(golub.cl)
 0  1
27 11
```

Now let’s do the SAM analysis; we give a **random seed** for the permutations – and tell `R` how many to do;

```
> library("siggenes")
> sam.out <- sam(golub, golub.cl, B=100, rand = 123)
```

... takes only a few seconds. Use `B=1000` or more if you can

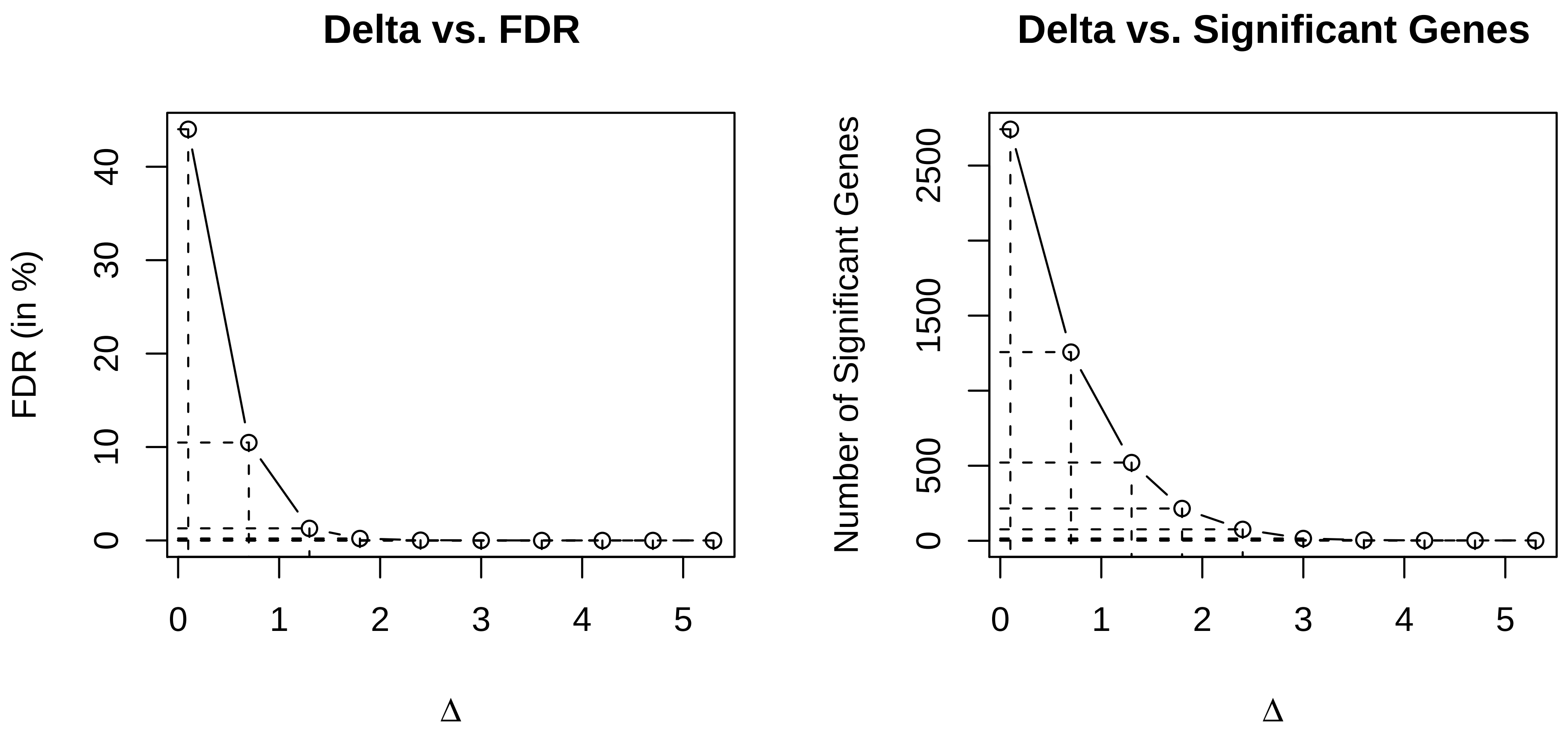
Significance Analysis of Microarrays

```
> summary(sam.out)
SAM Analysis for the Two-Class Unpaired Case Assuming Unequal Variances
  s0 = 0.0584  (The 0 % quantile of the s values.)
  Number of permutations: 100
 MEAN number of falsely called variables is computed.
   Delta  p0   False Called      FDR cutlow cutup   j2   j1
1    0.1 0.5 2424.77   2739  0.44276 -0.177 0.228 1434 1747
2    0.7 0.5  262.21   1248  0.10508 -1.264 1.438  737 2541
3    1.3 0.5   12.11    507  0.01195 -2.299 2.488  311 2856
4    1.8 0.5    0.74    210  0.00176 -3.154 3.311  134 2976
5    2.4 0.5    0.01     76 6.58e-05 -4.157 4.259   44 3020
6    3.0 0.5       0     15        0 -5.577 5.139    4 3041
7    3.6 0.5       0      5        0   -Inf 5.971    0 3047
8    4.1 0.5       0      2        0   -Inf 7.965    0 3050
9    4.7 0.5       0      2        0   -Inf 7.965    0 3050
10   5.3 0.5       0      0        0   -Inf   Inf    0 3052
```

p0 is the prior probability of differential expression. (Also note that the FDR values are rounded)

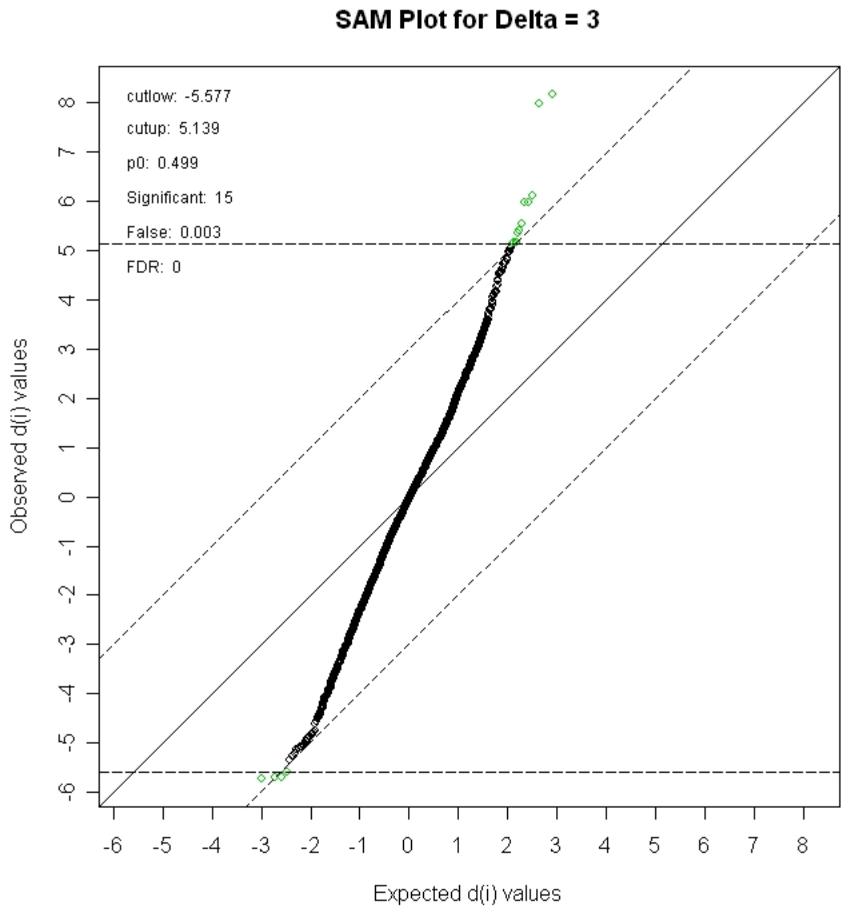
Significance Analysis of Microarrays

```
> plot(sam.out)
```



Significance Analysis of Microarrays

```
> plot(sam.out, 3) #specifies Delta; also sam.plot2()
```



Microarray analysis with `limma`

The `limma` package can do **several** analyses for microarrays. It reads in **raw data**, in standard formats

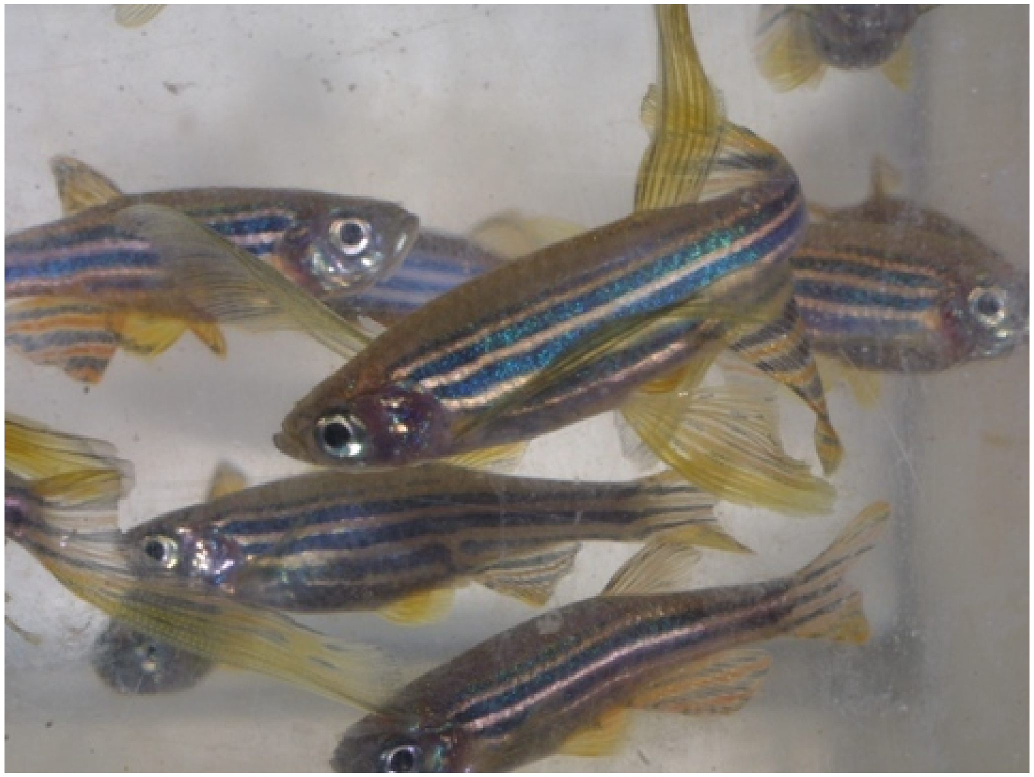
```
> library("limma")
> my.files <- dir(pattern=".spot")
> my.files
[1] "swirl.1.spot" "swirl.2.spot" "swirl.3.spot" "swirl.4.spot"
> RG <- read.maimages(my.files, source="spot")
Read swirl.1.spot
Read swirl.2.spot
Read swirl.3.spot
Read swirl.4.spot

RG$genes   <- readGAL("fish.gal") # gene names
RG$printer <- getLayout(RG$genes) # 4x4x22x24 print layout
```

Microarray analysis with `limma`

What is `swirl`? A mutation affecting **zebrafish**

Microarray analysis with `limma`

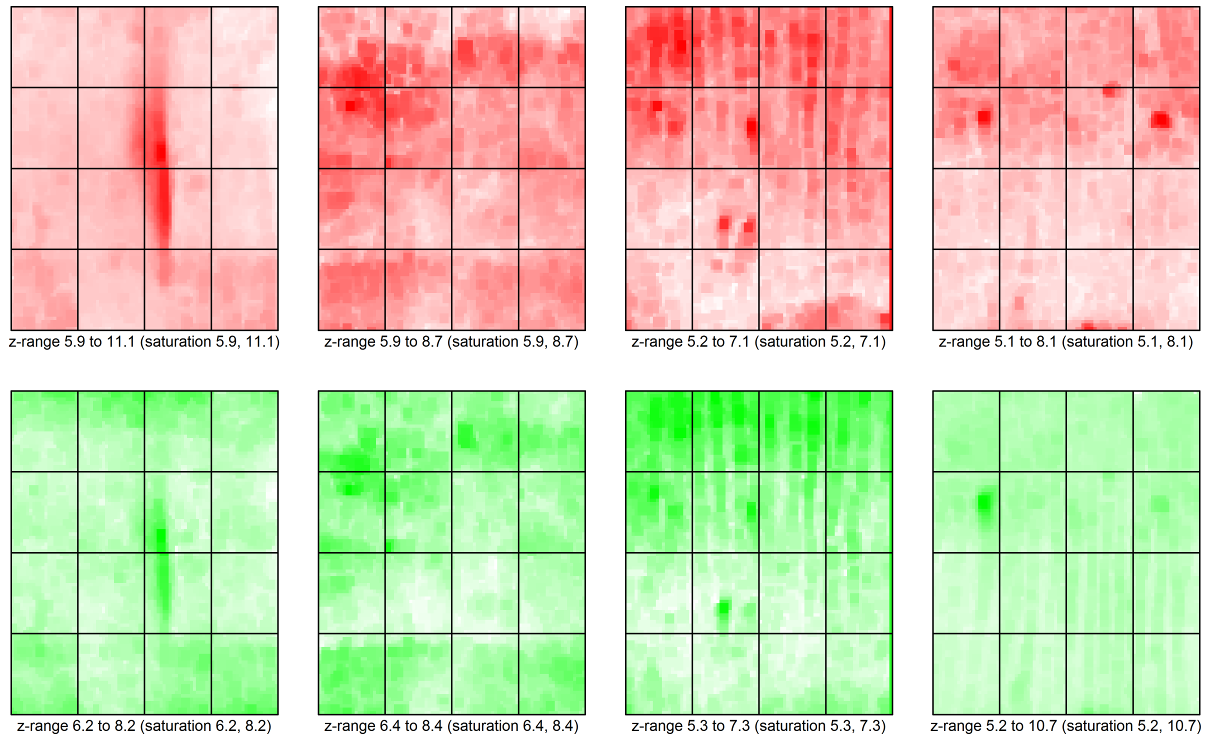
What is `swirl`? A mutation affecting **zebrafish**



We have 2 mutants, and 2 wild-type fish

Microarray analysis with `limma`

Here are the red/green intensities from each microarray;

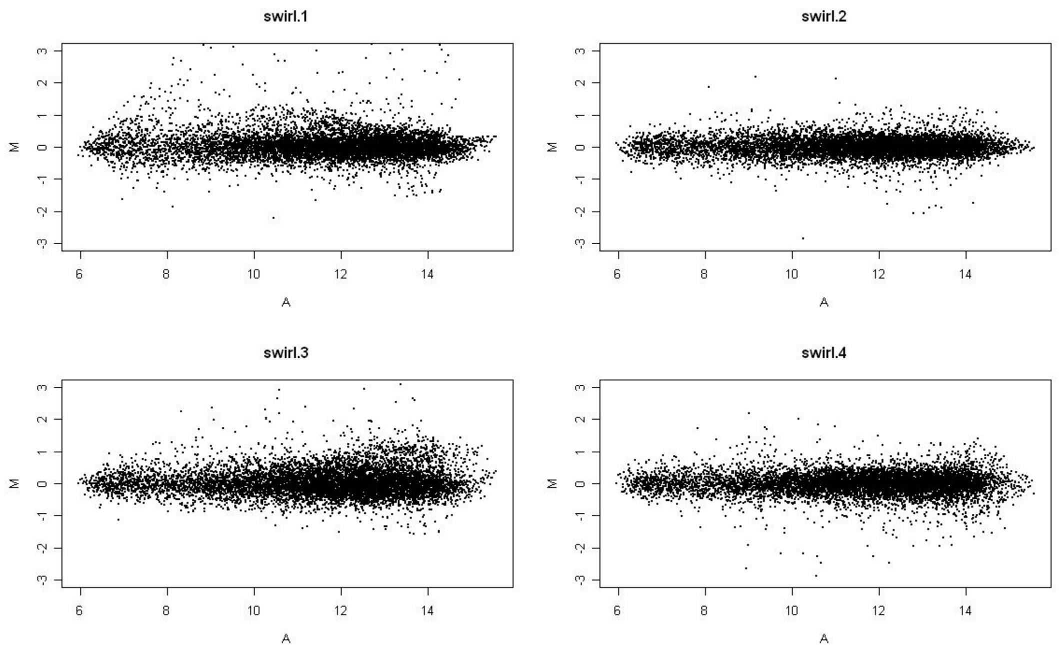


– obtained using `imageplot()`. Need to **normalize** each array (or get a bigger sample!)

Microarray analysis with `limma`

`limma` has ‘default’ normalization techniques;

```
> MA1 <- normalizeWithinArrays(RG)
> MA2 <- normalizeBetweenArrays(MA1)
> for(i in 1:4){plotMA(MA2,i)}
```



‘M’ measures differences in log RG intensity; ‘A’ measures average intensity. Can you guess where the ‘signals’ are?

Microarray analysis with `limma`

`limma` fits 'plain' models to each gene, and also 'robustifies' them with an Empirical Bayes approach (much the same as SAM)

```
> fit1 <- lmFit(MA2, design=c(-1,1,-1,1))
> options(digits=3); toptable(fit1, n=30, adjust="fdr")
         M     t  P.Value adj.P.Val    B
2961 -2.66 -20.8 1.44e-07   0.00121 7.55
3723 -2.19 -17.6 4.59e-07   0.00194 6.75
1611 -2.19 -16.1 8.44e-07   0.00238 6.29
7649 -1.60 -14.2 2.02e-06   0.00326 5.58
515   1.26  13.7 2.55e-06   0.00326 5.39

> fit2 <- eBayes(fit1)
> options(digits=3); topTable(fit2, n=30, adjust="fdr")
     Block Row Column      ID   Name     M     A     t  P.Value adj.P.Val    B
2961     6  14      9 fb85d05 18-F10 -2.66 10.33 -20.8 1.44e-07   0.00121 7.55
3723     8   2      3 control   Dlx3 -2.19 13.24 -17.6 4.59e-07   0.00194 6.75
1611     4   2      3 control   Dlx3 -2.19 13.45 -16.1 8.44e-07   0.00238 6.29
7649    15  11     17 fb58g10 11-L19 -1.60 13.49 -14.2 2.02e-06   0.00326 5.58
515      1  22     11 fc22a09 27-E17  1.26 13.19  13.7 2.55e-06   0.00326 5.39
```