GS01 0163
Analysis of Microarray Data

Keith Baggerly and Kevin Coombes
Section of Bioinformatics
Department of Biostatistics and Applied Mathematics
UT M. D. Anderson Cancer Center

kabagg@mdanderson.org

kcoombes@mdanderson.org

16 October 2007

Lecture 14: Reporting Differential Expression

- Differential Expression in BioC

- Annotation and Reporting of Data

Putting Some Pieces Together

Let's examine some aspects of differential expression in R, using some of the datasets from BioConductor.

```
> library(affy)
> library(ALL);
> data("ALL");
```

This is an ExpressionSet derived from 128 U95Av2 arrays, quantified using RMA. The phenoData has 21 variables, including "mol.biol". This specifies cytogenetic abnormalities, such as "BCR/ABL" or "NEG".

Skimming the Data

```
> class(ALL)
[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"
> slotNames(ALL)
[1] "assayData"  "phenoData"  "featureData"
[4] "experimentData"  "annotation"
[6] ".__classVersion__"
> phenoData(ALL)
  sampleNames: 01005, 01010, ..., LAL4 (128 total)
  varLabels and varMetadata:
    cod:  Patient ID
    diagnosis:  Date of diagnosis
    ...: ...
```

```
date last seen:   date patient was last seen
(21 total)
```

Learning about the Experiment

```
> experimentData(ALL)
Experiment data
  Experimenter name: Chiaretti et al.
  Laboratory: Department of Medical Oncology, Dan
  Contact information:
  Title: Gene expression profile of adult T-cell
  URL:
  PMIDs: 14684422 16243790

  Abstract: A 187 word abstract is available. Use
> abstract(ALL)
[1] "Gene expression profiles were examined in 33
```

Picking Something to Focus On

```
> varLabels(phenoData(ALL))
 [1] "cod"  "diagnosis"  "sex"   "age"  "BT"
 [6] "remission"   "CR"  "date.cr"   "t(4;11)"
[10] "t(9;22)"  "cyto.normal"  "citog"  "mol.biol
[14] "fusion protein"  "mdr"  "kinet"  "ccr"
[18] "relapse"  "transplant"  "f.u"  "date last se
> table(phenoData(ALL)$mol.biol)
```

```
ALL1/AF4  BCR/ABL E2A/PBX1       NEG   NUP-98  p15,
      10       37        5        74        1
```

Subsetting the Group

```
> mySubset <- ALL$mol.biol %in%
      c("BCR/ABL", "NEG");
> ALLs <- ALL[, mySubset];
```

There are 37 samples with the BCR/ABL fusion, and 74 samples that are negative for this.

Let's contrast these 2 groups.

Looking for Differences

```
> library("genefilter");
> g <- ALLs$mol.biol; # choose a factor
> ALLs.t <- rowttests(ALLs, g);
```

The rowttests function is written in C and is pretty fast. For each row, it returns
"statistic" "dm" "df" "p.value"
(dm is the difference in means.) We tend to use MultiTtest from the ClassComparison package available on our website, but that's only because we wrote it.

Unfortunately...

Looking for Differences

```
> ALLs.t <- rowttests(ALLs, g);
  Error in rowttests(ALLs, g) : Number of
  groups must be <= 2 for 'rowttests'.
> levels(g)
[1] "ALL1/AF4" "BCR/ABL"  "E2A/PBX1"
[4] "NEG"      "NUP-98"   "p15/p16"
```

Subsetted factors remember where they came from...

Looking for Differences

```
> ALLs.t <- rowttests(ALLs, g);
  Error in rowttests(ALLs, g) : Number of
  groups must be <= 2 for 'rowttests'.
> levels(g)
[1] "ALL1/AF4" "BCR/ABL"  "E2A/PBX1"
[4] "NEG"      "NUP-98"   "p15/p16"
```

Subsetted factors remember where they came from...

```
> ALLs$mol.biol <- factor(ALLs$mol.biol);
> g <- ALLs$mol.biol;
> ALLs.t <- rowttests(ALLs, g); # works
```

Are There Differences?
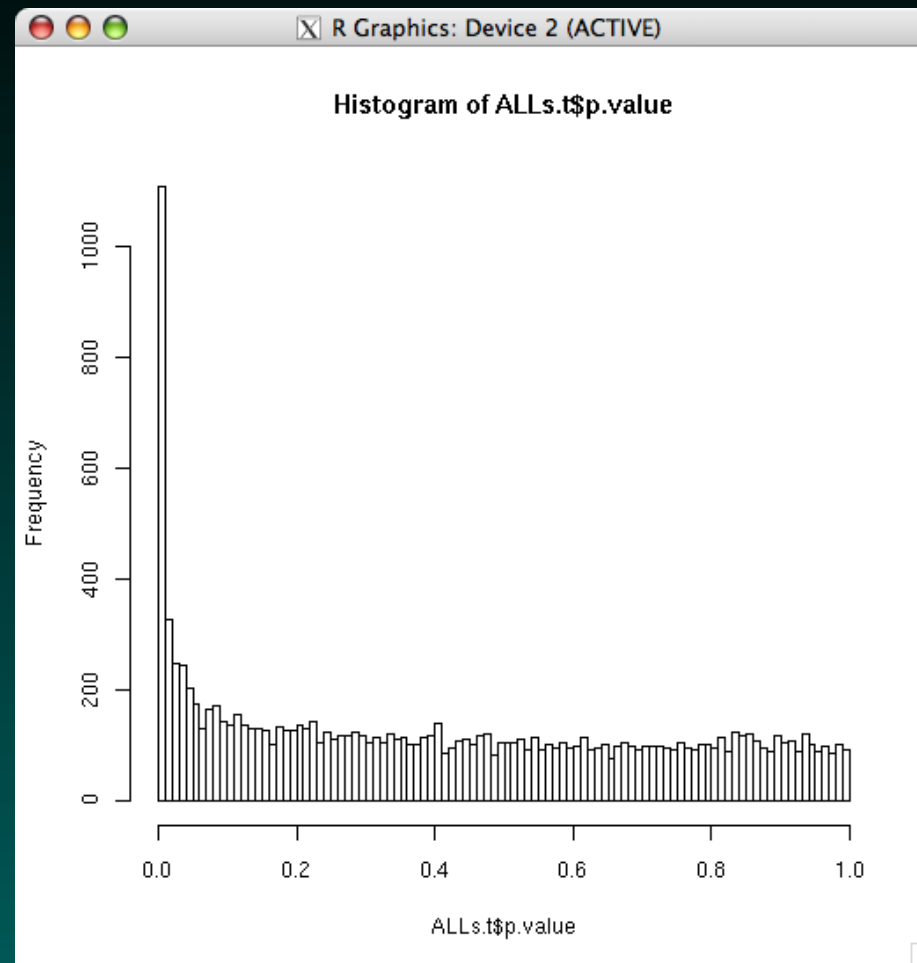

```
> hist(ALLs.t$p.value, breaks=100);
```

Ok, Can We See Them?

```
> heatmap(exprs(ALLs)); # BAD.
```

Why?

Ok, Can We See Them?

```
> heatmap(exprs(ALLs)); # BAD.
```

Why?

We're considering too many genes at present. (Quick quiz: how many?) Clustering will hang your computer.

We need to filter our list down.

Some Filtering

```
> meanThresh <- 100;
> filt1 <- rowMeans(exprs(ALLs)[, g ==
+          levels(g)[1]]) > meanThresh;
> filt2 <- rowMeans(exprs(ALLs)[, g ==
+          levels(g)[2]]) > meanThresh;
> selProbes <- (filt1 | filt2);
> ALLfilt <- ALLs[selProbes, ];
> dim(exprs(ALLfilt));
dim(exprs(ALLfilt));
[1]    0 111
```

Some Filtering

```
> meanThresh <- 100;
> filt1 <- rowMeans(exprs(ALLs)[, g ==
+        levels(g)[1]]) > meanThresh;
> filt2 <- rowMeans(exprs(ALLs)[, g ==
+        levels(g)[2]]) > meanThresh;
> selProbes <- (filt1 | filt2);
> ALLfilt <- ALLs[selProbes, ];
> dim(exprs(ALLfilt));
dim(exprs(ALLfilt));
[1]    0 111


> rowMeans(exprs(ALLs))[1:3]
  1000_at    1001_at 1002_f_at
```

```
7.565085  5.019850  3.884797
```

Some Filtering (Take Logs!)

```
> meanThresh <- log2(100);
> filt1 <- rowMeans(exprs(ALLs)[, g ==
+           levels(g)[1]]) > meanThresh;
> filt2 <- rowMeans(exprs(ALLs)[, g ==
+           levels(g)[2]]) > meanThresh;
> selProbes <- (filt1 | filt2);
> ALLfilt <- ALLs[selProbes, ];
> dim(exprs(ALLfilt)); # 3660 by 111, a bit big

> meanThresh <- log2(200);
...
> dim(exprs(ALLfilt)); # 1771 by 111, better
```
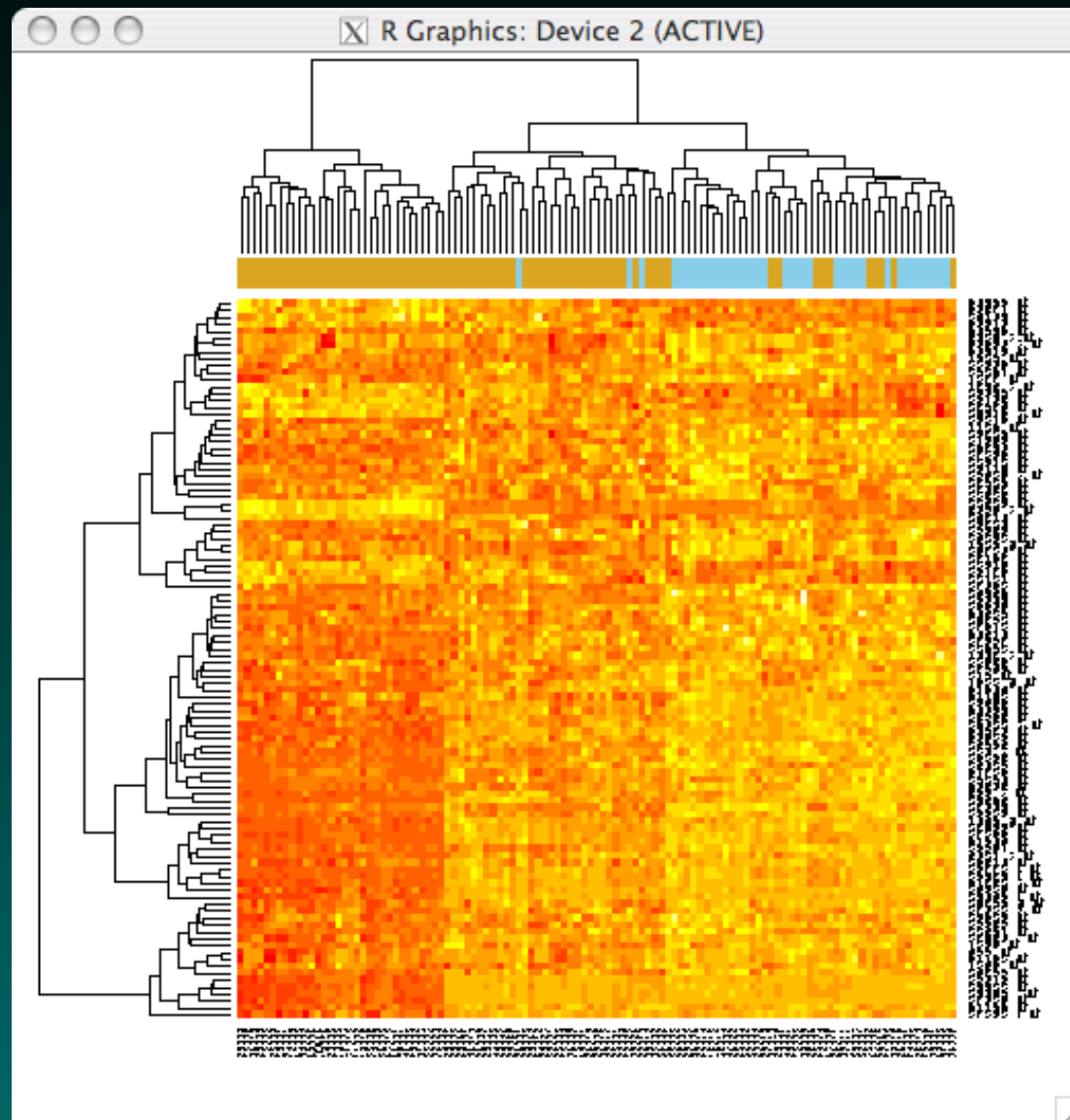
Focus on the Interesting Ones

```
> filt3 <- ALLs.t$p.value < 0.0001;
> selProbes <- (filt1 | filt2) & filt3;
> ALLfilt <- ALLs[selProbes, ];
> dim(exprs(ALLfilt)); # 104 by 111, ok
```

Try picturing this...
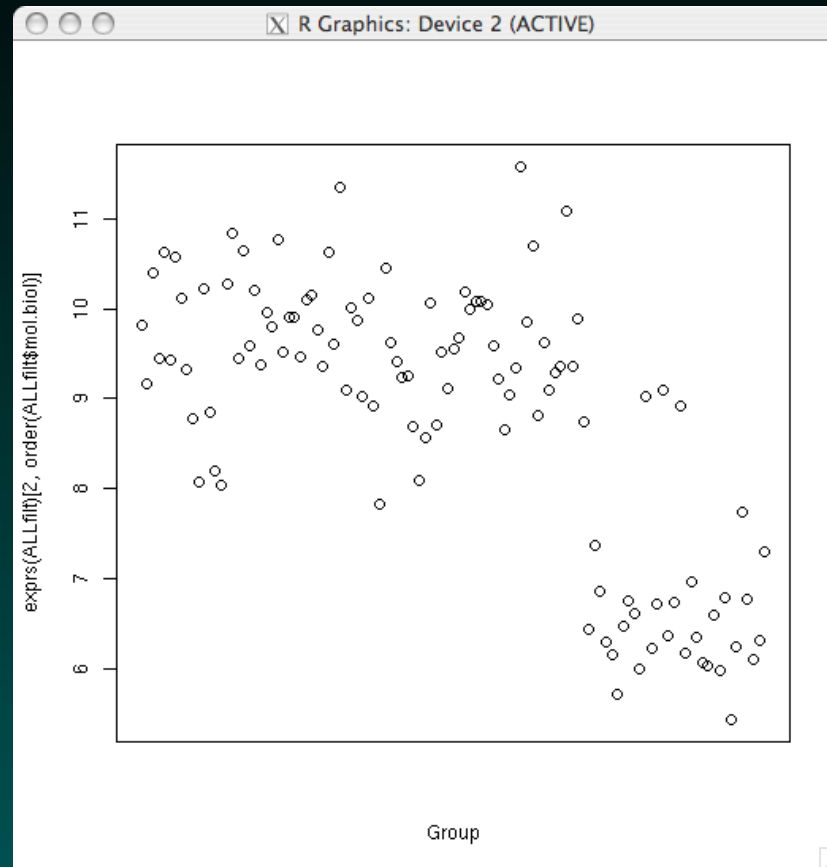
```
> spcol <- ifelse(ALLfilt$mol.biol == "NEG",
+                  "goldenrod", "skyblue")
> heatmap(exprs(ALLfilt), ColSideColors=spcol);
```
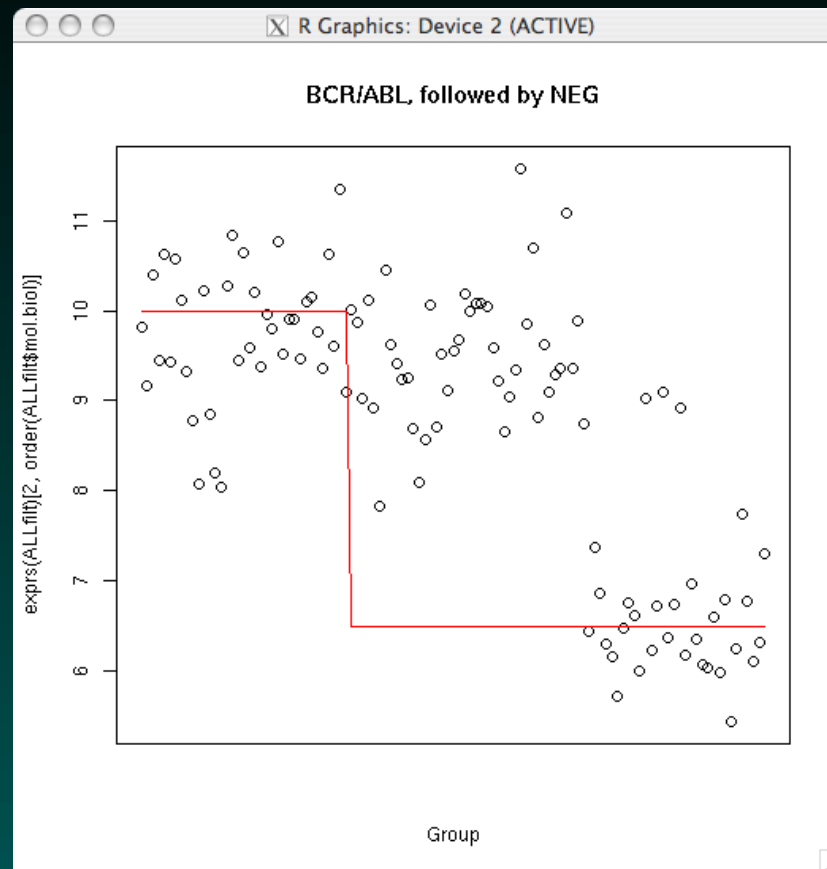
Huzzah! (Right?)

That Was Odd...


```
> plot(exprs(ALLfilt)[2,order(ALLfilt$mol.biol)],
    xaxt='n', xlab='Group'); # row 1 was boring.
```

That Was Odd... Right?


```
> lines(10 - 3.5*(ALLfilt$mol.biol[order(
      ALLfilt$mol.biol)] == 'NEG'), col='red');
> title(main = 'BCR/ABL, followed by NEG');
```

What's Going On?

```
> names(pData(ALLfilt))
 [1] "cod"        "diagnosis" "sex"       "age"
 [5] "BT"         "remission" "CR"        "date.cr"
 [9] "t(4;11)"   "t(9;22)"    "cyto.norm" "citog"
[13] "mol.biol"  "fus prot"   "mdr"        "kinet"
[17] "ccr"        "relapse"    "transplant" "f.u"
[21] "date last seen"
```

Are there other variables that may dominate the one I chose?

What Cells?

```
> ALLfilt$BT
  [1]  B2 B2 B4 B2 B1 B1 B1 B2 B2 B3
 [11]  B3 B2 B3 B  B2 B3 B2 B3 B2 B2
 [21]  B2 B1 B2 B2 B2 B  B  B2 B2 B2
 [31]  B2 B2 B2 B2 B2 B4 B2 B2 B2 B4
 [41]  B2 B2 B3 B3 B3 B3 B4 B3 B3 B1
 [51]  B1 B3 B3 B3 B3 B3 B3 B3 B3 B3
 [61]  B1 B2 B2 B1 B3 B4 B4 B2 B2 B3
 [71]  B4 B4 B4 B2 B2 B2 B1 B2 B  T
 [81]  T2 T2 T3 T2 T  T4 T2 T3 T3 T
 [91]  T2 T3 T2 T2 T2 T1 T4 T  T2 T3
[101]  T2 T2 T2 T2 T3 T3 T3 T2 T3 T2
[111]  T
Levels:  B B1 B2 B3 B4 T T1 T2 T3 T4
```

Another View

```
> table(pData(ALLfilt)$BT, pData(ALLfilt)$mol.biol
      BCR/ABL NEG
  B         2    2
  B1        1    8
  B2       19   16
  B3        8   14
  B4        7    2
  T         0    5
  T1        0    1
  T2        0   15
  T3        0    9
  T4        0    2
```

Matching Patterns

We want entries that begin with B. This is a "regular expression", and one of the tools for extracting these is "grep".

```
> BT <- as.character(ALLfilt$BT);
> grep("B", BT); # returns 1..79
> grep("^B", BT); # same
> grep("^T", BT); # 80..111
> grep("B*", BT); # 1..111 everything!
> grep("B.*",BT); # 1..79
> grep("B$", BT); # 14,26,27,79
> grep("^B$",BT); # same
> grep("^b", BT); # null
> grep("^b", BT, ignore.case=TRUE);
```

Once More Unto the Breach!

```
> plot(exprs(ALLfilt)[2,], xlab='Sample');
> y1 <- rep(0,111);
> y1[grep("^T",BT)] = 1;
> lines(10 - 3.5*y1, col='red')
> title(main="B Cells, then T Cells");
```

Finally!
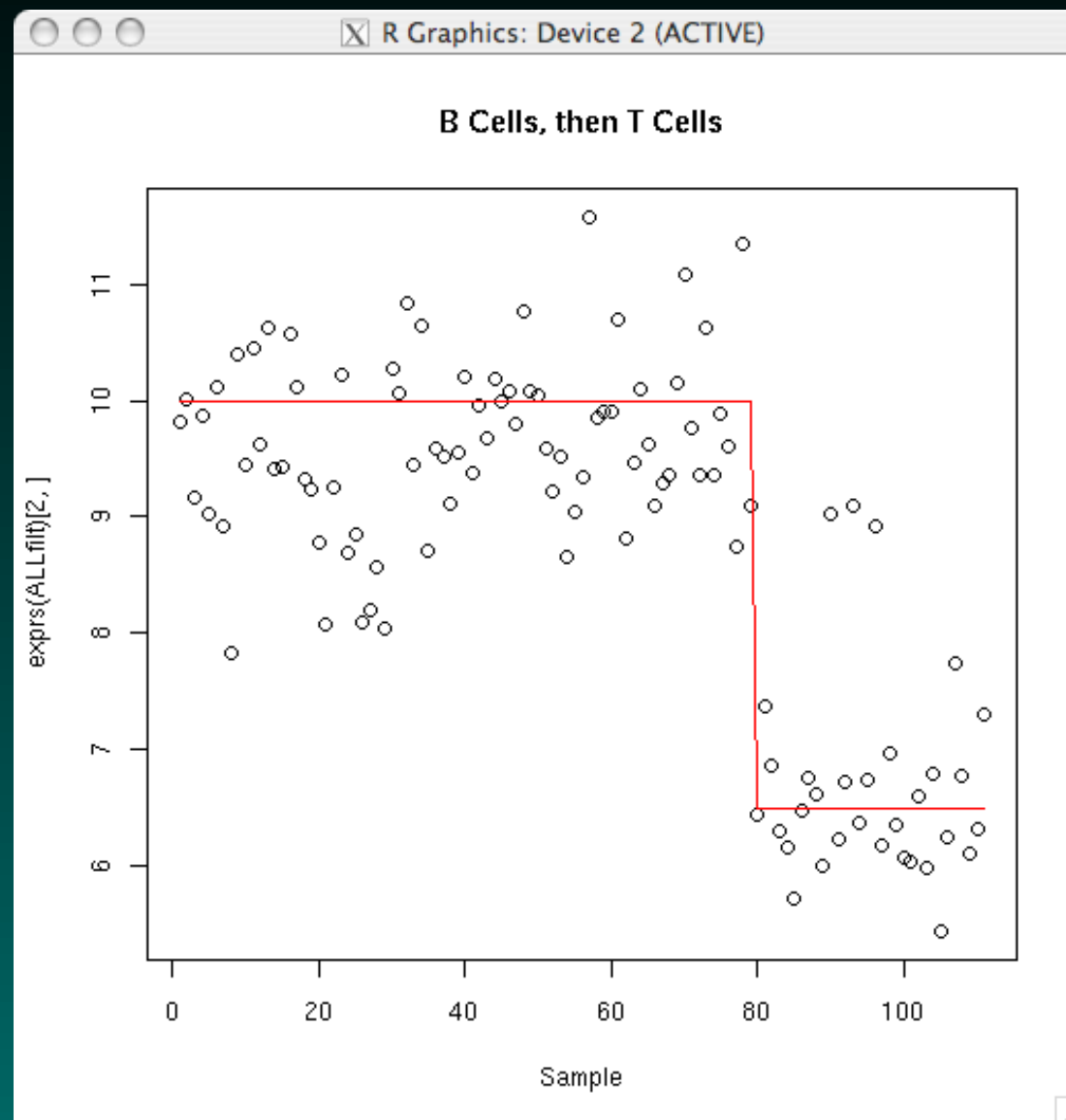
Analysis Redux 1

```
> mySubset1 <- grep("^B", ALL$BT);
> ALLs1 <- ALL[,mySubset1];
> dim(exprs(ALLs1))
[1] 12625    95
> mySubset2 <- ALLs1$mol.biol %in% c("BCR/ABL", "
> ALLs2 <- ALLs1[,mySubset2];
> dim(exprs(ALLs2))
[1] 12625    79
> ALLs2$mol.biol <- factor(ALLs2$mol.biol);
> g <- ALLs2$mol.biol;
> ALLs2.t <- rowttests(exprs(ALLs2), g);
```
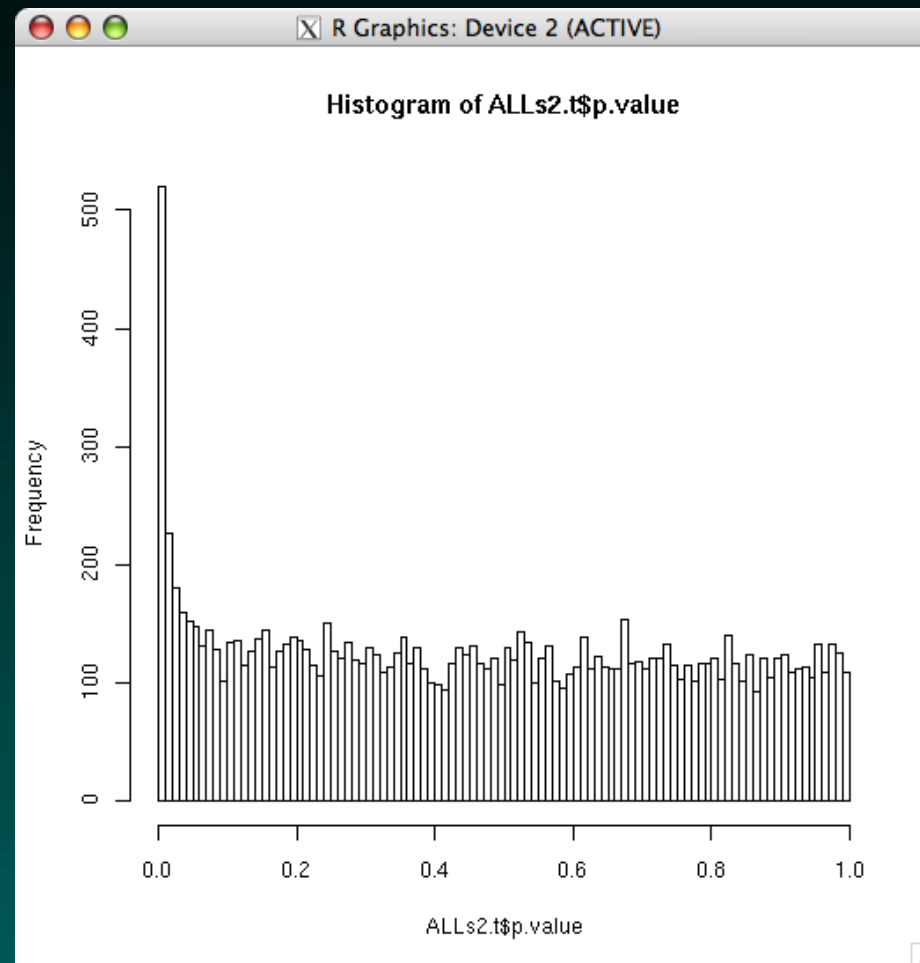
Analysis Redux 2


```
> hist(ALLs2.t$p.value, breaks=100);
```
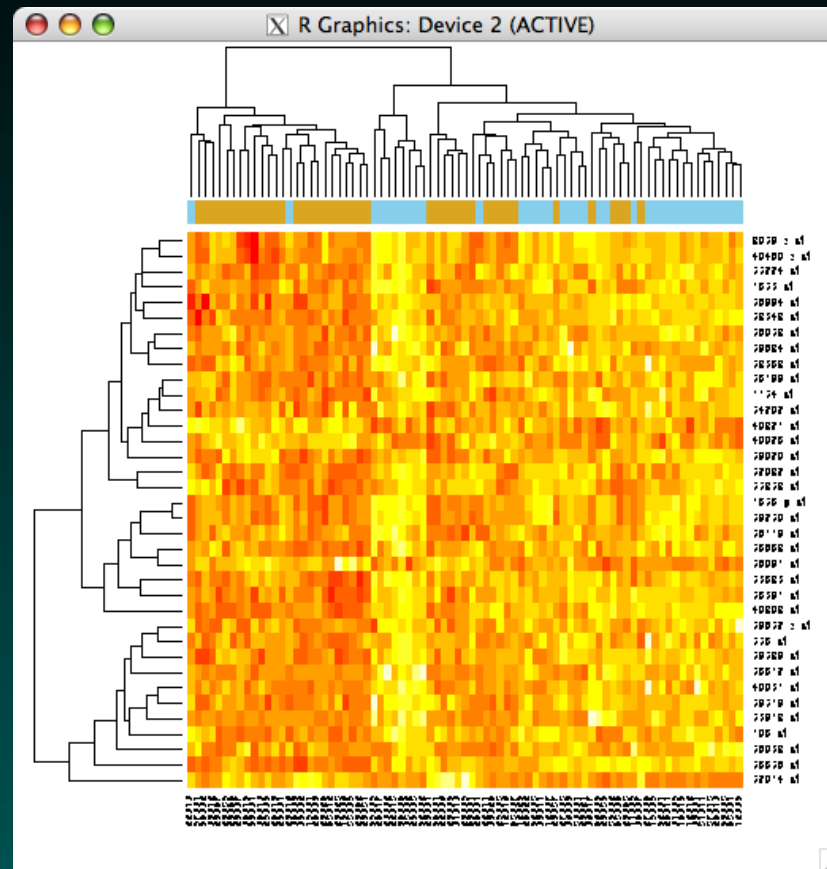
Analysis Redux 3

```
> meanThresh <- log2(100);
> filt1 <- rowMeans(exprs(ALLs2)[, g ==
        levels(g)[1]]) > meanThresh;
> filt2 <- rowMeans(exprs(ALLs2)[, g ==
        levels(g)[2]]) > meanThresh;
> filt3 <- ALLs2.t$p.value < 0.0001;
> selProbes <- (filt1 | filt2) & filt3;
> ALLs2Filt <- ALLs2[selProbes,];
> dim(exprs(ALLs2Filt))
[1]   36 79
```

A Better Figure


```
> spcol <- ifelse(ALLs2Filt$mol.biol ==
       "NEG", "goldenrod", "skyblue");
> heatmap(exprs(ALLs2Filt), ColSideColors=spcol);
```

So, What About These Genes?

Through all this processing, the gene identities have been preserved, so we can access them easily.

```
> featureNames(ALLs2Filt)[1:3]
[1] "106_at"  "1134_at" "1635_at"


> ALLs2Filt.t <- rowttests(exprs(ALLs2Filt), g);
> plot(ALLs2Filt.t$statistic)
> index <- order(abs(ALLs2Filt.t$statistic),
                 decreasing = TRUE);
> probeids <- featureNames(ALLs2Filt)[index]
> probeids[1:3]
[1] "1636_g_at" "39730_at"  "1635_at"
```

Let's Add to the Report

Grab some of the annotation from the environment

```
> library("annaffy")
Loading required package: GO
Loading required package: KEGG
> library("hgu95av2")
> syms <- unlist(mget(probeids,
          hgu95av2SYMBOL))
> locuslinks <- unlist(mget(
          probeids, hgu95av2ENTREZID))
> library("annotate")
```

Make a Web Page

```
> geneList <- list(probeids);
> repository <- list("affy");
> otherNames <- list(syms, locuslinks);
> head <- c("Probe ID", "Symbol", "LocusLink");
> fileName <- "out1.html";
> htmlpage(genelist = geneList, filename =
      fileName, title="ALL Interesting",
      othernames = otherNames,
      table.head = head,
      repository = repository)
```

Note: the elements in geneList will be live links!

The Output

Differentially Expressed Genes

ALL Interesting

1636_g_at	ABL1	25
39730_at	ABL1	25
1635_at	ABL1	25
40202_at	KLF9	687
37027_at	AHNAK	79026
39837_s_at	ZNF467	168544
40480_s_at	FYN	2534
33774_at	CASP8	841
36591_at	TUBA1	7277
37014_at	MX1	4599

```
<TD> <A HREF="https://www.affymetrix.com/
    LinkServlet?&probeset=1636_g_at">1636_g_at
    </A></TD> <TD>ABL1</TD><TD>25</TD>
```

More Details

The Affy links take you to NetAffx; you must log in there to see the details.

For htmlpage, values to be linked must come first.

Calling this function without a list of values to be linked (ie, all columns are "othernames") will break.

More than one set of links can be included in a single table; every such set requires us to specify a repository.

The documentation for htmlpage doesn't directly state what the set of repositories is...

Checking Code

```
> htmlpage
...
    for (i in seq(along = repository)) {
        rows <- paste(rows,
            getTDRows(genelist[[i]],
                        repository[[i]]))
...
```

checking the help for "getTDRows":

repository: A character string for the name of a public repository. Valid values include "ll", "ug", "gb", "sp", "omim", "affy", "en", and "fb".

What are These?

- LL: LocusLink

- UG: UniGene

- GB: GenBank

- SP: SwissProt

- OMIM: Online Mendelian Inheritance in Man

- AFFY: NetAffx

- EN: EntrezGene (replaces LocusLink)

- FB: FlyBase

Getting More Sophisticated

The annaffy package contains quite a few more accessors for different types of databases and information. These begin with "aaf" (Annotation for AFfy), and return lists of information.

```
library(annaffy)
help(package=annaffy)
```

The list (all prefixed by aaf): ChromLoc, Chromosome, Cytoband, Description, Function, GenBank, GO, LocusLink, Pathway, Probe, PubMed, Symbol, UniGene.

Mapping

Most queries share a common syntax:

```
> ALLbands <- aafCytoband(probeids, "hgu95av2");
> ALLbandLinks <- getURL(ALLbands);
> ALLbandLinks[[1]]
[1] "http://www.ncbi.nlm.nih.gov/mapview/
   map_search.cgi?direct=on&query=U07563%5BACCN%5D
> ALLbands[[1]]
An object of class "aafCytoband"
Slot "band":
[1] "9q34.1"
Slot "genbank":
[1] "U07563"
```

Assembling a Table, Take 2

```
> aaf.handler()
 [1] "Probe"     "Symbol"      "Description"
 [4] "Function"  "Chromosome"  "Chromosome Location
 [7] "GenBank"   "LocusLink"   "Cytoband"
[10] "UniGene"   "PubMed"      "Gene Ontology"
[13] "Pathway"
> ALLTable <- aafTableAnn(probeids, "hgu95av2");
```

the argument "colnames" can be used to produce subsets of this output, or to rearrange the order.

Assembling a Table, Take 2

```
> ALLCols <- aaf.handler()[c(1:3, 9, 5:6)];
> ALLTable <- aafTableAnn(probeids,
      "hgu95av2", colnames=ALLCols);
```

These approaches produce tables which are lists of lists:

```
> ALLTable[[1]][[1]]
[1] "1636_g_at"
attr(,"class")
[1] "aafProbe"
> ALLTable[[2]][[1]]
[1] "ABL1"
attr(,"class")
[1] "aafSymbol"
```

Making a Page, Take 2

```
> saveHTML(ALLTable, "out2.html")
> saveHTML(ALLTable2, "out3.html")
```

Why do it twice?

The Sparse Page

Bioconductor Affymetrix Probe Listing

Probe	Symbol	Description	Function	Chromosome	Chromosome Location	GenBank	LocusLink	Cytoband
1636_g_at	ABL1	v-abl Abelson murine leukemia viral oncogene homolog 1		9	130740384, 130618821	U07563	25	9q34.1

The Dense Page

Bioconductor Affymetrix Probe Listing

Probe	Symbol	Description	Cytoband	Chromosome	Chromosome Location
1636_g_at	ABL1	v-abl Abelson murine leukemia viral oncogene homolog 1	9q34.1	9	130740384, 130618821
39730_at	ABL1	v-abl Abelson murine leukemia viral oncogene homolog 1	9q34.1	9	130740384, 130618821
1635_at	ABL1	v-abl Abelson murine leukemia viral oncogene homolog 1	9q34.1	9	130740384, 130618821
40202_at	KLF9	Kruppel-like factor 9	9q13	9	-70229068
37027_at	AHNAK	AHNAK nucleoprotein (desmoyokin)	11q12.2	11	-62039950, -61957591
39837_s_at	ZNF467	zinc finger protein 467	7q36.1	7	-148899099
40480_s_at	FYN	FYN oncogene related to SRC, FGR, YES	6q21	6	-112089179, -112089186
33774_at	CASP8	caspase 8, apoptosis-related cysteine peptidase	2q33-q34	2	201923686, 201948284, 201950747, 201923693
36591_at	TUBA1	tubulin, alpha 1 (testis specific)	2q35	2	-219940505
37014_at	MX1	myxovirus (influenza virus) resistance 1, interferon-inducible protein p78 (mouse)	21q22.3	21	41720023
39329_at	ACTN1	actinin, alpha 1	14q24.1-q24.2|14q24|14q22-q24	14	-68410792
32542_at	FHL1	four and a half LIM domains 1	Xq26	X	134955199
40051_at	TRAM2	translocation associated membrane protein 2	6p21.1-p12	6	-52470160
		FYN oncogene related to SRC, FGR			-112089179