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

28 September 2006

Lecture 9: Exploring BioConductor

- How do we load CEL files into an AffyBatch? how can we merge batches? how can we partition batches?
- How do we check that it worked?
- How do we supply the associated phenoData?
- Given an AffyBatch, how do we look at it? boxplot, hist, ma-plots, ratio plots, PLM
- Given an AffyBatch, how do we fit it? expresso, justRMA
- Given an eset, what can we say about its contents?
- How can we get the probe level values for a probeset?

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

- How can we figure out what probeset corresponds to a given gene?
- How can we get the probe sequences for a probeset?

Loading CEL files into an AffyBatch

One theme for today is TMTOWTDI. We're going to try to extend this from Perl over to BioConductor, and to the analysis of Affy data.

To begin with, let's say that we've got a set of CEL files. How do we pull them in?

There are several options. How do I survey them?

Loading CEL files into an AffyBatch

One theme for today is TMTOWTDI. We're going to try to extend this from Perl over to BioConductor, and to the analysis of Affy data.

To begin with, let's say that we've got a set of CEL files. How do we pull them in?

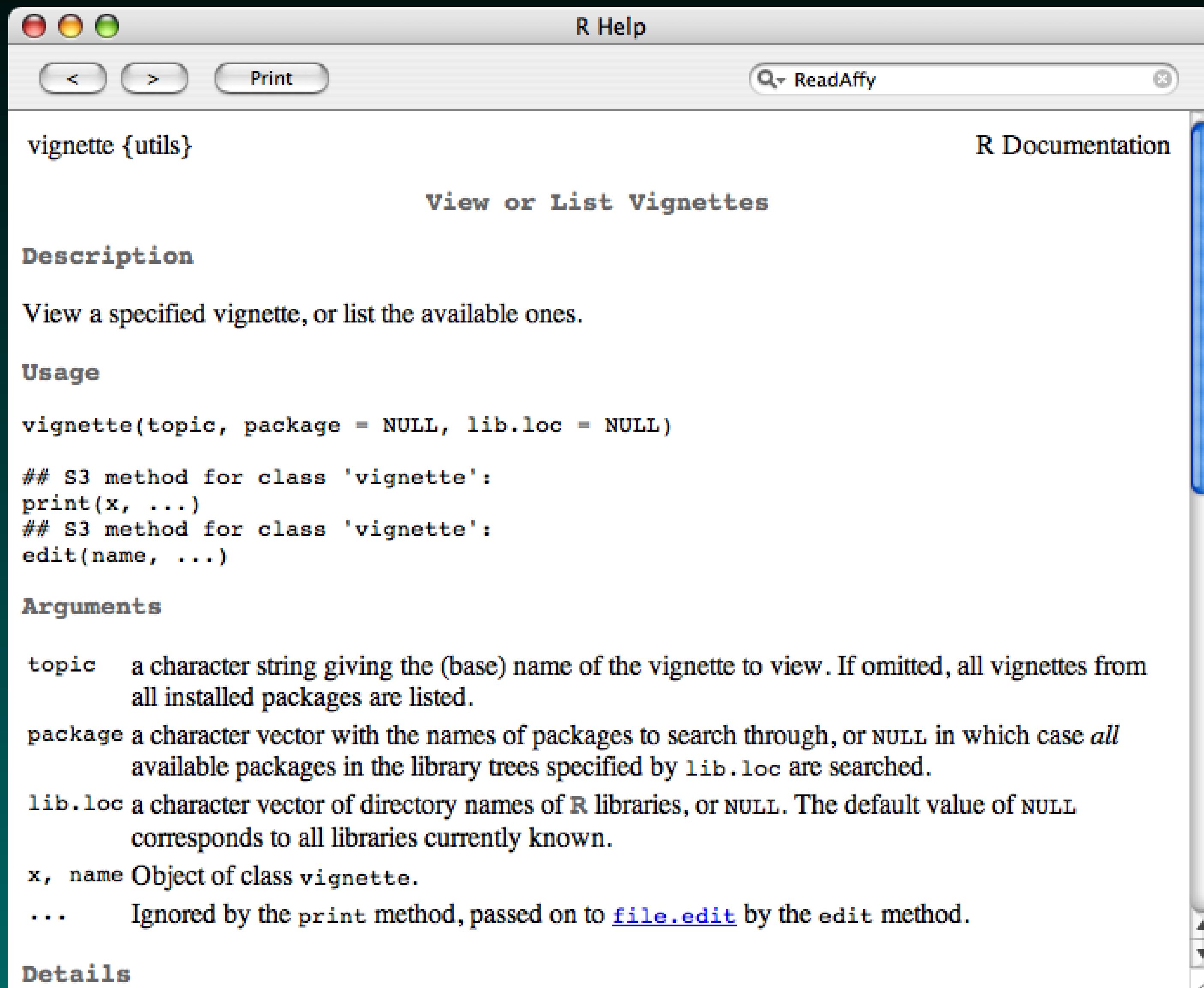
There are several options. How do I survey them?

```
> library("affy");
> vignette("affy");
```

Don't panic, it's not really 271 pages...

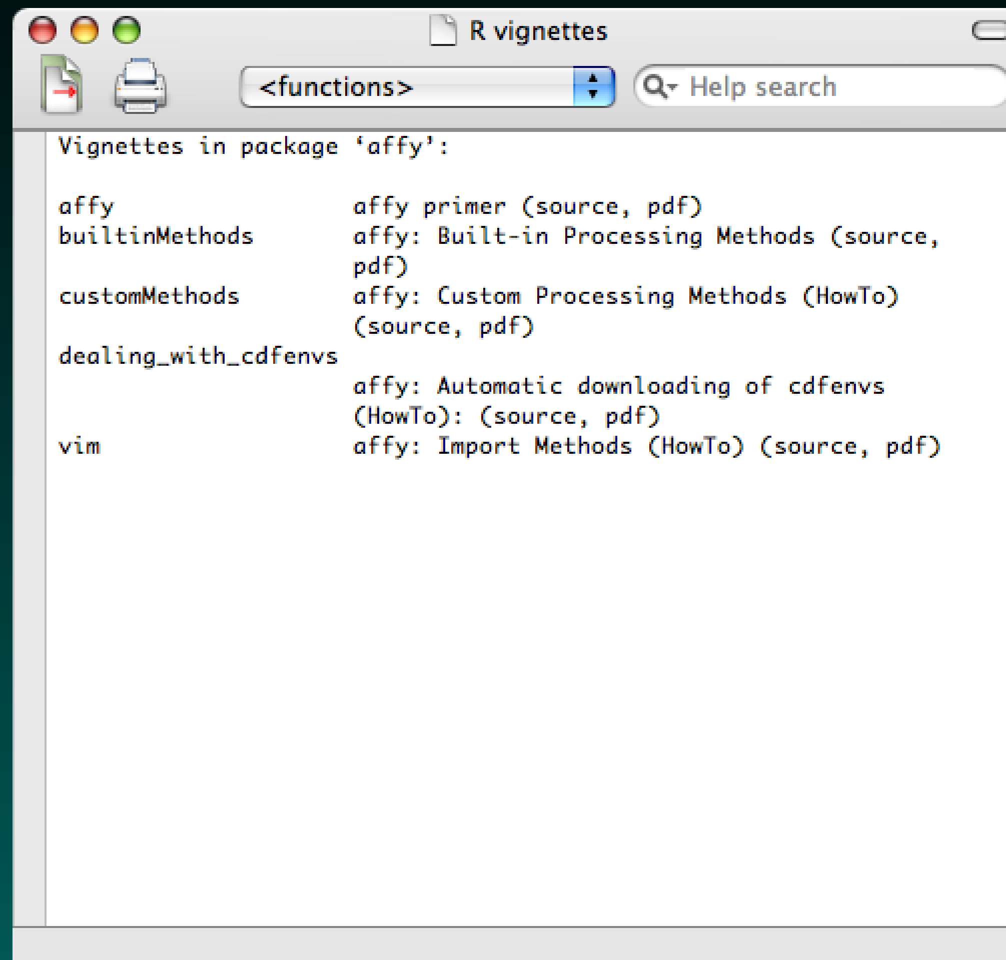
© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Reading the Fine Manual: Vignettes



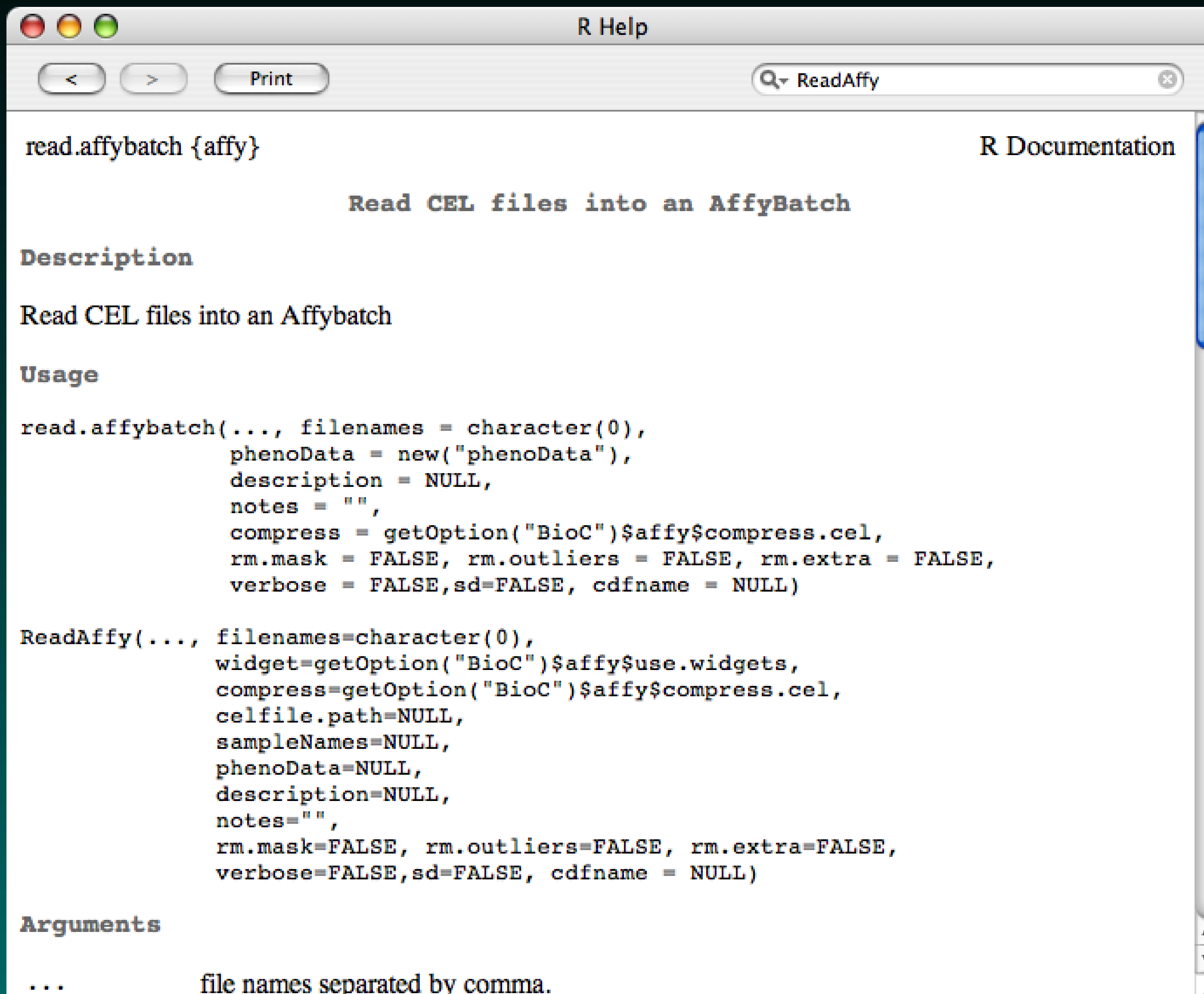
© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Listing Vignettes

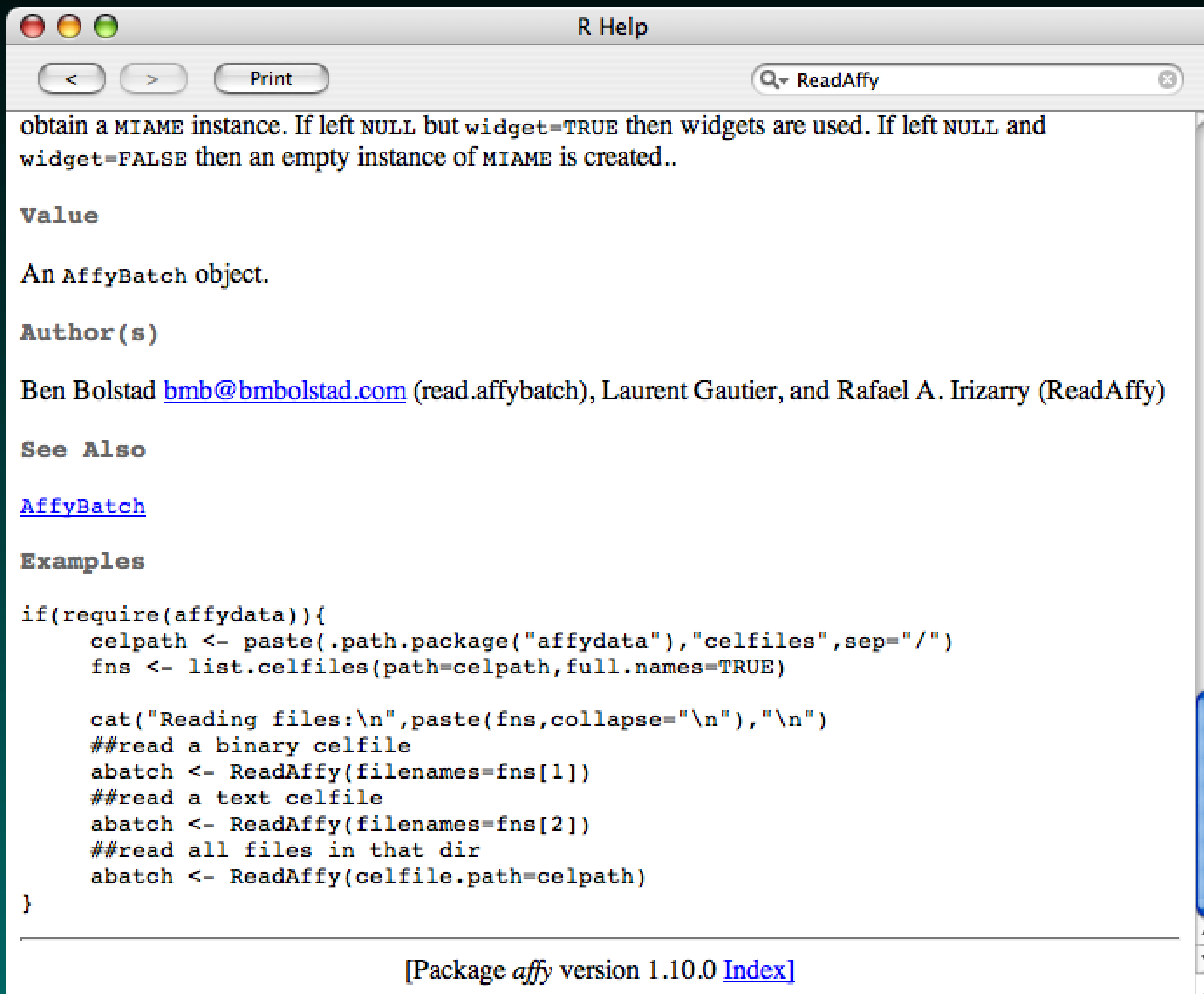



```
> vignette(package = "affy");
```

ReadAffy: Help from Top

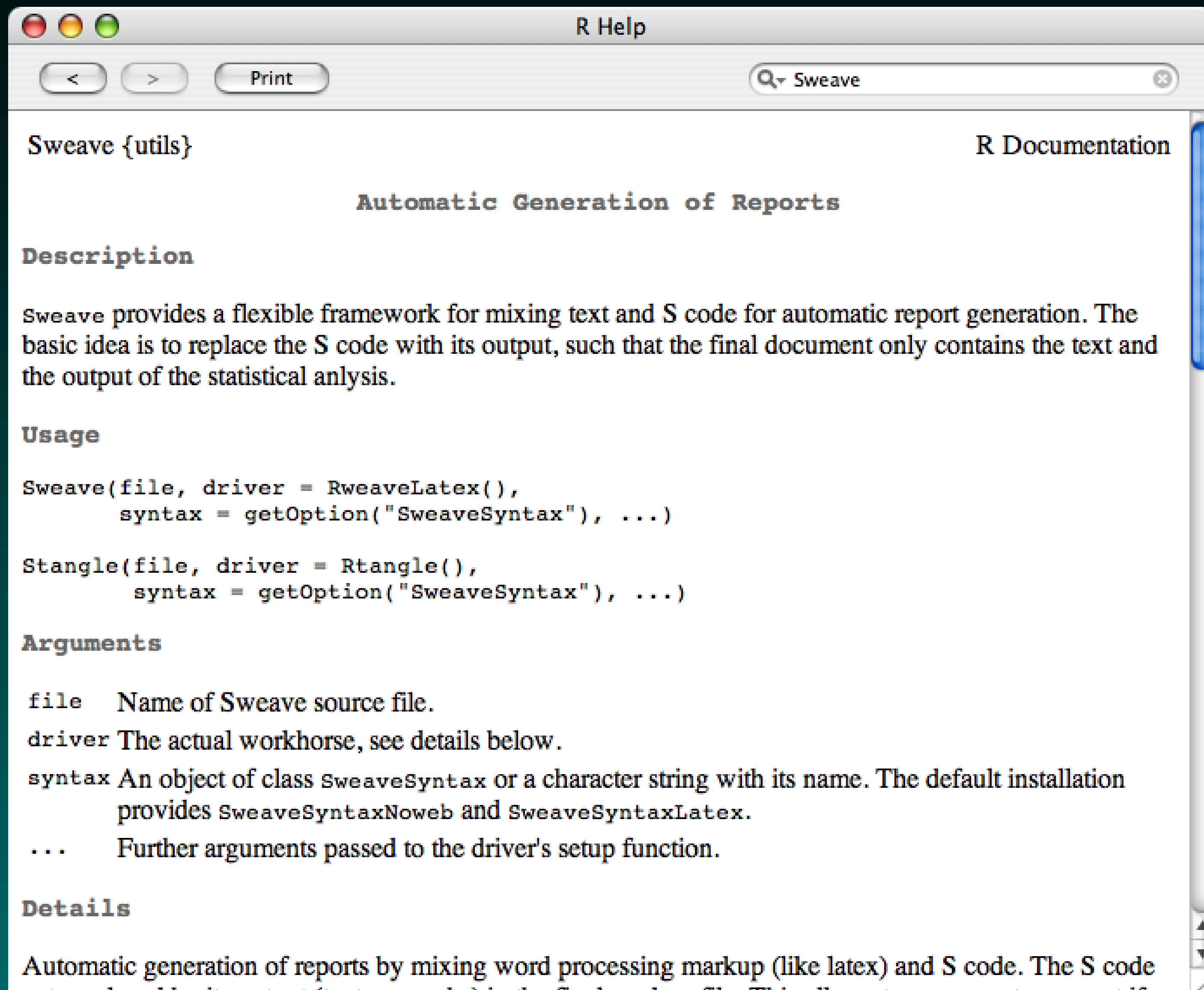


ReadAffy: ... to Bottom



The Affy Index

R documentation: Sweave



Reading a list of files

some_cels.txt:

```
../../DataSets/SinghProstate/N01__normal.CEL
../../DataSets/SinghProstate/N05__normal.CEL
../../DataSets/SinghProstate/N11__normal.CEL
../../DataSets/SinghProstate/N15__normal.CEL
../../DataSets/SinghProstate/N21__normal.CEL
../../DataSets/SinghProstate/N25__normal.CEL
```

```
> celList <- readTable("some_cels.txt");
> ABatch  <- ReadAffy(celList);
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

The Evolution...

The Evolution...

oops...

```
> celList <- readTable("some_cels.txt");
> celList <- as.character(celList$V1);
> ABatch  <- ReadAffy(celList);
```

The Evolution...

oops...

```
> celList <- readTable("some_cels.txt");
> celList <- as.character(celList$V1);
> ABatch  <- ReadAffy(celList);
```

oops...

```
> celList <- readTable("some_cels.txt");
> celList <- as.character(celList$V1);
> ABatch  <- ReadAffy(filenames = celList);
```

Ta Da!

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Checking the Contents

```
> slotNames(ABatch)
 [1] "cdfName"      "nrow"          "ncol"
 [5] "se.exprs"     "description"   "annotation"
 [9] "reporterInfo" "phenoData"

> phenoData(ABatch)
 phenoData object with 1 variables and 6 cases
 varLabels
sample: arbitrary numbering
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Looking at phenoData

```
> slotNames(phenoData(ABatch))
[1] "pData"        "varLabels"    "varMetadata"
> (phenoData(ABatch))@pData
                   sample
N01__normal.CEL         1
N05__normal.CEL         2
N11__normal.CEL         3
N15__normal.CEL         4
N21__normal.CEL         5
N25__normal.CEL         6
> (phenoData(ABatch))@varLabels
$sample
[1] "arbitrary numbering"
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Assigning phenoData

some_pdata.txt:

```
Sample name     Concocted
N01_norm        A
N05_norm        A
N11_norm        A
N15_borm        B
N21_borm        B
N25_borm        B
```

```
> p1 <- read.phenoData("some_pdata.txt");# error
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Assigning phenoData, pt 2

```
> p1 <- read.phenoData("some_pdata.txt",
                           sep="\t")
> p1
 phenoData object with 2 variables
         and 7 cases
 varLabels
V1: read from file
V2: read from file
# Not quite what we want.
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Assigning phenoData, pt 3

```
> p1 <- read.phenoData("some_pdata.txt",
                       sep="\t", header=TRUE)
> p1
 phenoData object with 2 variables
         and 6 cases
 varLabels
Sample.name: read from file
Concocted: read from file
> phenoData(ABatch) <- p1
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Other ways of Reading Data

Are they all in one directory?

What is the list of filenames?

read.affybatch vs ReadAffy

GUI?

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Other ways of Reading Data 1

```
kabagg$ ls ../../DataSets/SinghSmall
N60__normal.CEL N61__normal.CEL N62__normal.CEL

> ABSmall <- ReadAffy(celfile.path=
       "../../DataSets/SinghSmall"); # works
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Other ways of Reading Data 2

```
kabagg$ ls ../../DataSets/SinghSmall2
N60__normal.CEL.gz N61__normal.CEL.gz
N62__normal.CEL.gz
```

```
> ABSmall <- ReadAffy(celfile.path=
       "../../DataSets/SinghSmall2",
       compress=TRUE); # works
```

This takes only about 1/3 the space...

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Other ways of Reading Data 3

```
kabagg$ ls ../../DataSets/SinghSmall3
N60.gz N61.gz N62.gz
```

```
> ABSmall <- ReadAffy(celfile.path=
       "../../DataSets/SinghSmall3",
       compress=TRUE); # fails
```

```
> ABSmall <- ReadAffy(filenames=
       "../../DataSets/SinghSmall3/N60.gz",
       compress=TRUE); # works
```

This still takes only about 1/3 the space...

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Now let's Quantify

```
t0 <- date();
eset0 <- expresso(ABatch,
                  bgcorrect.method="rma",
                  normalize.method="quantiles",
                  pmcorrect.method="pmonly",
                  summary.method="medianpolish");
t1 <- date(); # 151s
eset1 <- justRMA(filenames = celList);
t2 <- date(); #  10s
```

The customized routines are better if they do what you want to do...

(also note that justRMA didn't build an AffyBatch.)

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Just Because I'm Curious

```
> exprs(eset1)[1,]
N01__normal.CEL N05__normal.CEL N11__normal.CEL
       7.789481        7.314639        7.445363
N15__normal.CEL N21__normal.CEL N25__normal.CEL
       7.289881        7.503692        7.412608
```

Can we reconstruct this?

```
> ABatch.BG <- bg.correct.rma(ABatch)
> ABatch.BG.norm <-
    normalize.AffyBatch.quantiles(ABatch.BG)
```

These steps produce AffyBatch objects, with altered exprs.

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

What is the First Gene?

(well, ok, probeset)

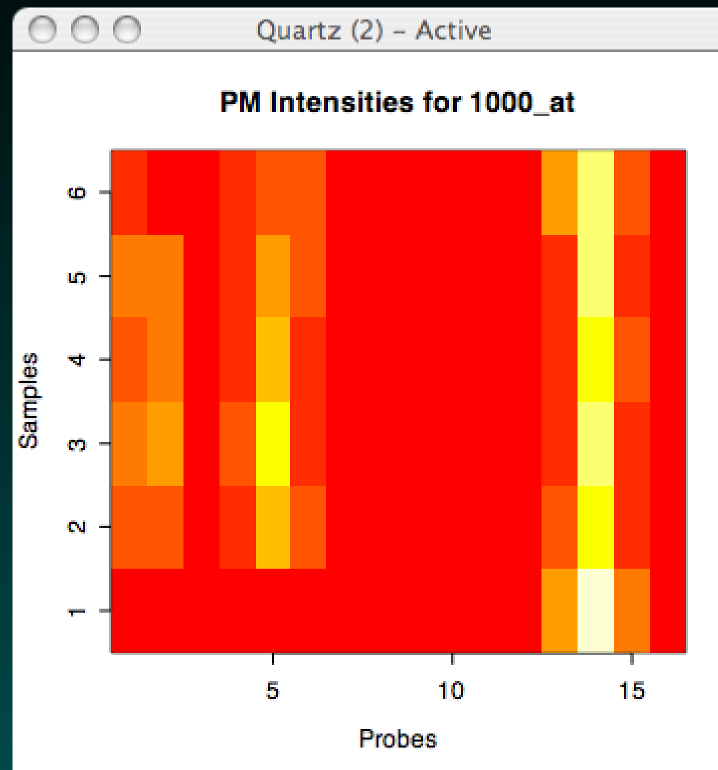
```
> gn1 <- geneNames(ABatch.BG.norm)[1]
> gn1
[1] "1000_at"
```

Ok, now what are the values?

```
pr1 <- pm(ABatch.BG.norm, gn1);
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

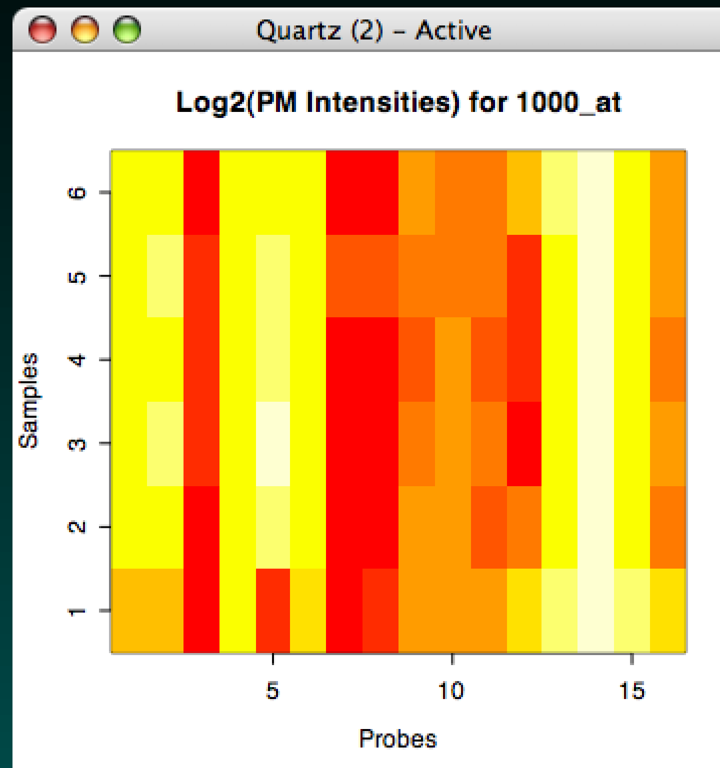
Looking at it, Take 1




```
image(1:nrow(pr1), 1:ncol(pr1), pr1,
      xlab="Probes", ylab="Samples",
      main="PM Intensities for 1000_at")
```

Some parallelism, but we may be missing something...

Looking at it, Take 2




```
image(1:nrow(pr1), 1:ncol(pr1), log2(pr1),
      xlab="Probes", ylab="Samples",
      main="Log2(PM Intensities) for 1000_at")
```

Logs!

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Fitting the Probes

```
> pr1Fit <- medpolish(log2(pr1))
1 : 45.72612
2 : 45.08797
Final: 45.03615
> names(pr1Fit)
[1] "overall"  "row"       "col"       "residuals"
[5] "name"
> pr1Fit$overall + pr1Fit$col
N01__normal.CEL N05__normal.CEL N11__normal.CEL
       7.789481        7.314639        7.445363
N15__normal.CEL N21__normal.CEL N25__normal.CEL
       7.289881        7.503692        7.412608
```

This is what we found before!

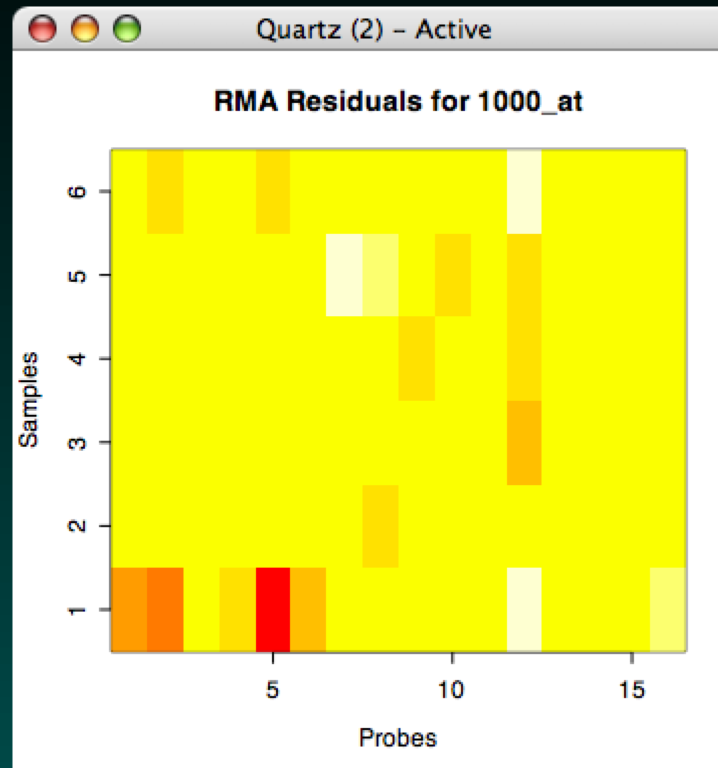
© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

We can Check the Code

```
> medpolish
function (x, eps = 0.01, maxiter = 10,
    trace.iter = TRUE, na.rm = FALSE)
{
    z <- as.matrix(x)
    nr <- nrow(z)
    nc <- ncol(z)
    t <- 0
    r <- numeric(nr)
    c <- numeric(nc)
    oldsum <- 0
    for (iter in 1:maxiter) {
        rdelta <- apply(z, 1, median, na.rm = na.
        z <- z - matrix(rdelta, nr = nr, nc = nc)
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

and Check the Residuals




```
image(1:nrow(pr1), 1:ncol(pr1), pr1Fit$residuals,
      xlab="Probes", ylab="Samples",
      main="RMA Residuals for 1000_at")
```

Trivia Q: Who introduced median polish?

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

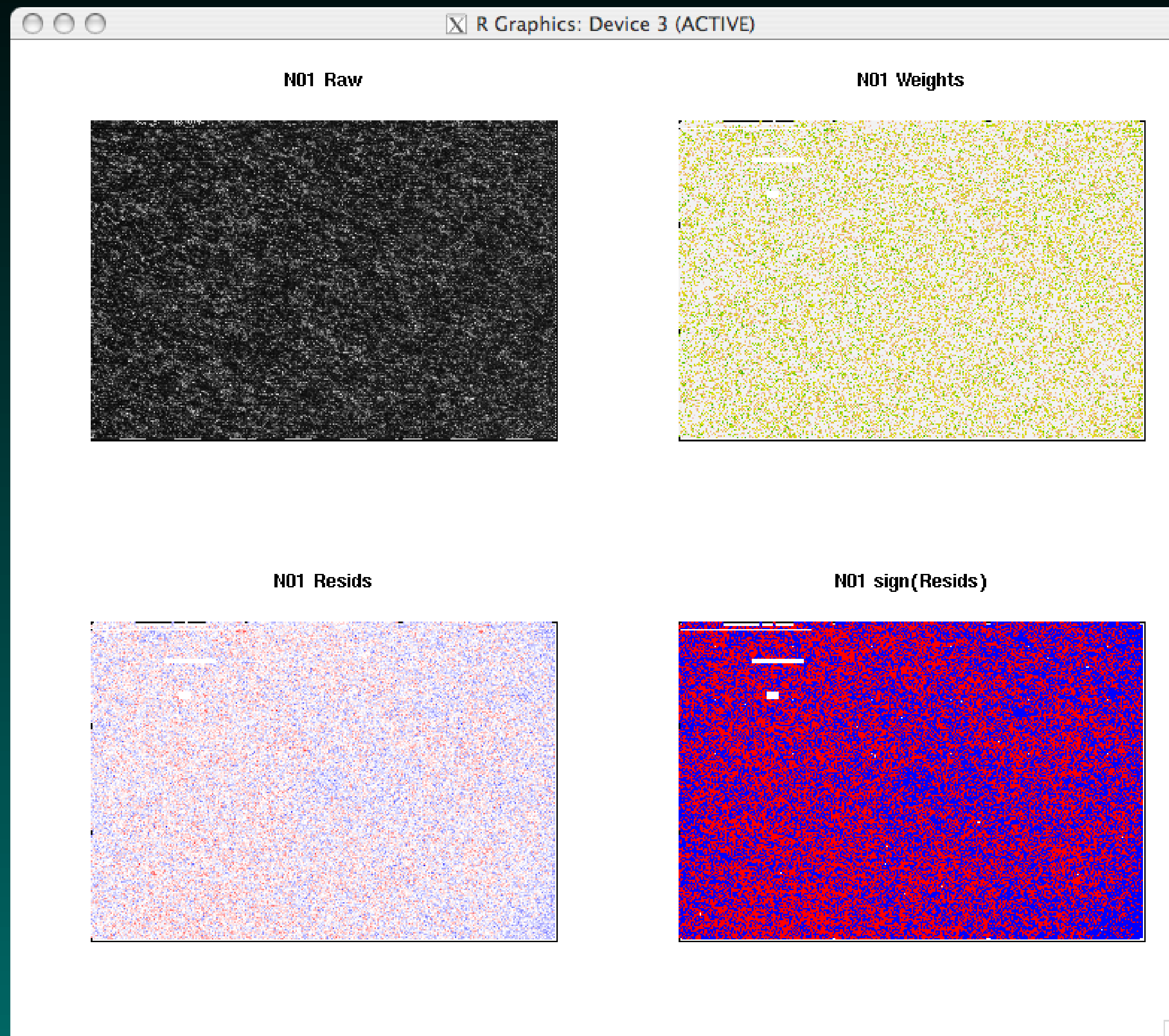
One other Fitting Approach: PLM

PLM = “Probe Level Model”

```
library("affyPLM");
plm1 <- fitPLM(ABatch); # takes a few minutes
par(mfrow=c(2,2));
image(ABatch[, 1],main="N01 Raw");
image(plm1, type="weights", which=1, main="N01 We
image(plm1, type="resids", which=1, main="N01 Res
image(plm1, type="sign.resids", which=1,
      main="N01 sign(Resids)");
```

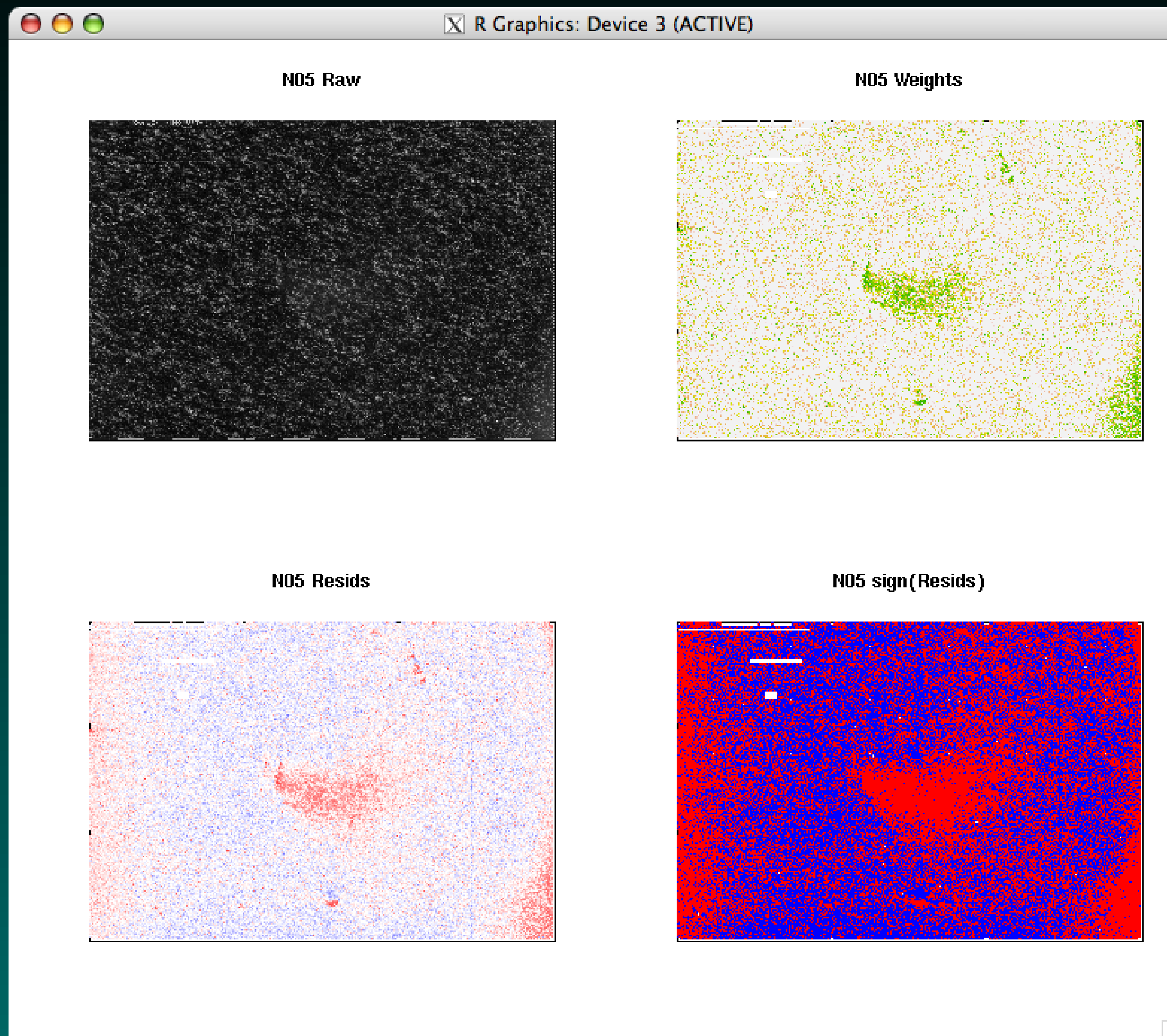
© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Looking at N01



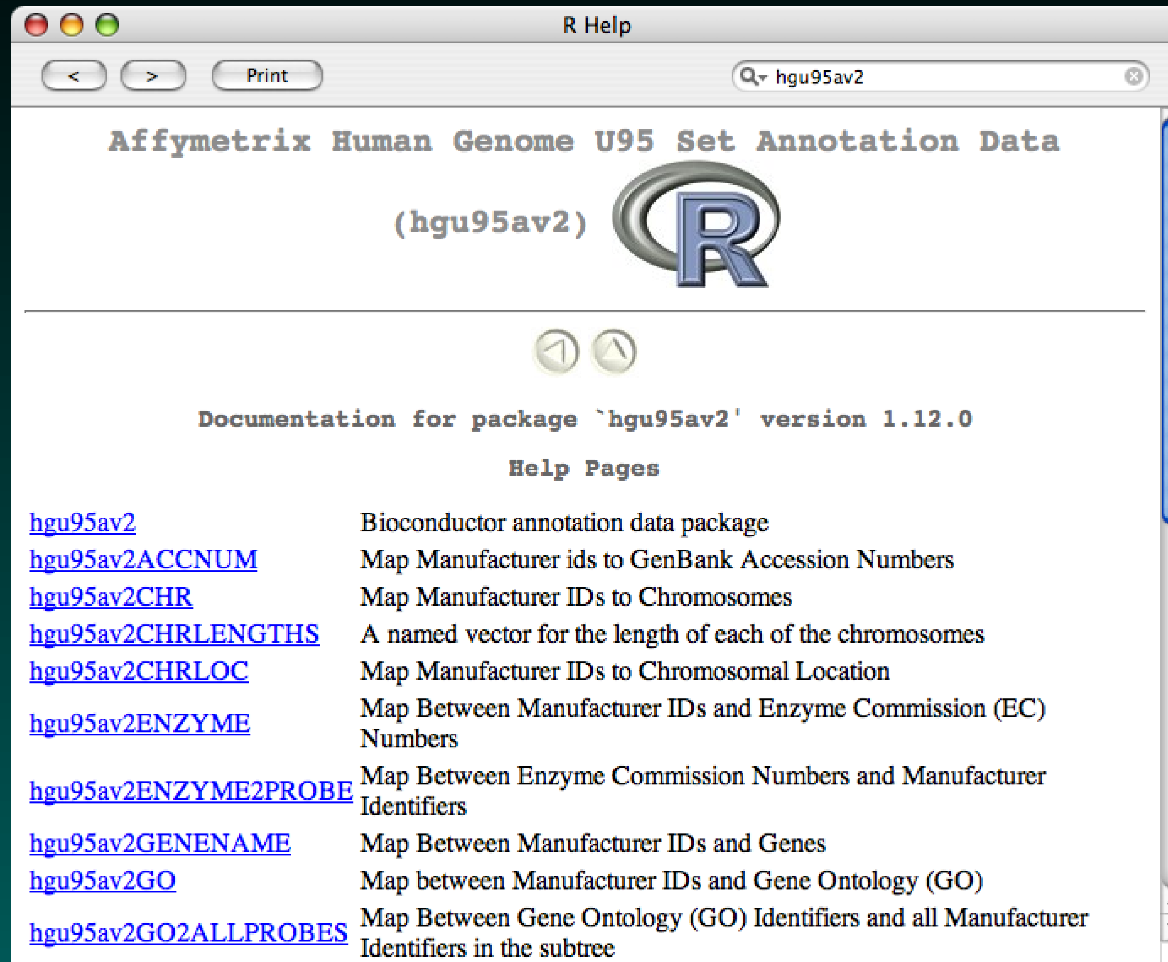
© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Looking at N05



© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

Whence the Gene Name Info?




```
> library("hgu95av2");
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

What Does This Package Contain?

```
> hgu95av2()

Quality control information for  hgu95av2
Date built: Created: Sun Mar 26 00:53:27 2006

Number of probes: 12625
Probe number missmatch: None
Probe missmatch: None
Mappings found for probe based rda files:
 hgu95av2ACCNUM found 12625 of 12625
 hgu95av2CHRLOC found 11716 of 12625
 hgu95av2CHR found 12171 of 12625
 hgu95av2ENZYME found 1922 of 12625
 hgu95av2GENENAME found 11660 of 12625
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

```
 hgu95av2GO found 11101 of 12625
 hgu95av2LOCUSID found 12238 of 12625
 hgu95av2MAP found 12140 of 12625
 hgu95av2OMIM found 9978 of 12625
 hgu95av2PATH found 4252 of 12625
 hgu95av2PMID found 12136 of 12625
 hgu95av2REFSEQ found 12039 of 12625
 hgu95av2SUMFUNC found 0 of 12625
 hgu95av2SYMBOL found 12184 of 12625
 hgu95av2UNIGENE found 12127 of 12625
Mappings found for non-probe based rda files:
   hgu95av2CHRLENGTHS found 25
 hgu95av2ENZYME2PROBE found 660
 hgu95av2GO2ALLPROBES found 6012
 hgu95av2GO2PROBE found 4274
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

```
hgu95av2ORGANISM found 1
hgu95av2PATH2PROBE found 173
hgu95av2PFAM found 10412
hgu95av2PMID2PROBE found 107253
hgu95av2PROSITE found 8193
```

(we can also see this using ls(”package:hgu95av2”).)

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

What Does This Package Contain?

```
> hgu95av2GENENAME
<environment: 0x26519bdc>
```

Almost everything in this package is an “environment”, which is R’s fancy name for a hash table. We can access things by name.

```
> hgu95av2GENENAME$"1000_at"
[1] "mitogen-activated protein kinase 3"
```

We can access a lot of annotation!

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

What was Needed for Quantification?

```
> library("hgu95av2cdf");
> hgu95av2cdf$"1000_at"
          pm     mm
 [1,] 358160 358800
 [2,] 118945 119585
 [3,] 323731 324371
 [4,] 223978 224618
...
[15,] 317054 317694
[16,] 404069 404709
```

These give the indices of the probes within the 409600-long vector of expression intensities.

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

What if We Want to Go in Reverse?

Given a probeset, I can find a gene name. What if I have a gene name, and I want something else?

Can we find “BAD”?

This is a gene symbol, so we probably want to work with the hgu95av2SYMBOL environment.

The key function for extracting items from an environment without the key is “contents”.

```
> tempSYM <- contents(hgu95av2SYMBOL);
> tempSYM[1]
$`986_at`
[1] "CYP19A1"
```

Finding BAD in the Contents

```
> tempSYM[tempSYM == "BAD"]
$`1861_at`
[1] "BAD"

> names(tempSYM[tempSYM == "BAD"])
[1] "1861_at"
```

This gives us the key!

Some of these queries are simplified if we invoke

```
> library("annotate"); # for example,
> getLL("1861_at", "hgu95av2");
[1] 572
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

One More Thing...

sequences?

```
> library("hgu95av2probe");
> data(hgu95av2probe); # a big file

> as.data.frame(hgu95av2probe[1,])
                    sequence   x   y
1 TCTCCTTTGCTGAGGCCTCCAGCTT 399 559

Probe.Set.Name   Probe.Interrogation.Position
1000_at                                1367

Target.Strandedness
Antisense
```

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly

So, What's BAD?

```
> as.data.frame(hgu95av2probe[hgu95av2probe$Probe
                              sequence   x   y Probe.Set
14006 CAACCTCTGGCAGCACAGCGCTAT 403 485          18
14007 AACCTCTGGCAGCACAGCGCTATG 402 485          18
14008 CCTCTGGCAGCACAGCGCTATGGC 207 491          18
14009 TGGGCAGCACAGCGCTATGGCCGCG 436 421         18
14010 GCAGCACAGCGCTATGGCCGCGAGC 285 599         18
...
```

The midpoint of the first probe should be at position 384 within the cDNA sequence.

© Copyright 2004–2006, Kevin R. Coombes and Keith A. Baggerly