GS01 0163
Analysis of Microarray Data

Keith Baggerly and Kevin Coombes
Department of Bioinformatics and Computational Biology
UT M. D. Anderson Cancer Center
kabagg@mdanderson.org
kcoombes@mdanderson.org

2 October 2007

Lecture 10: 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?

- 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.

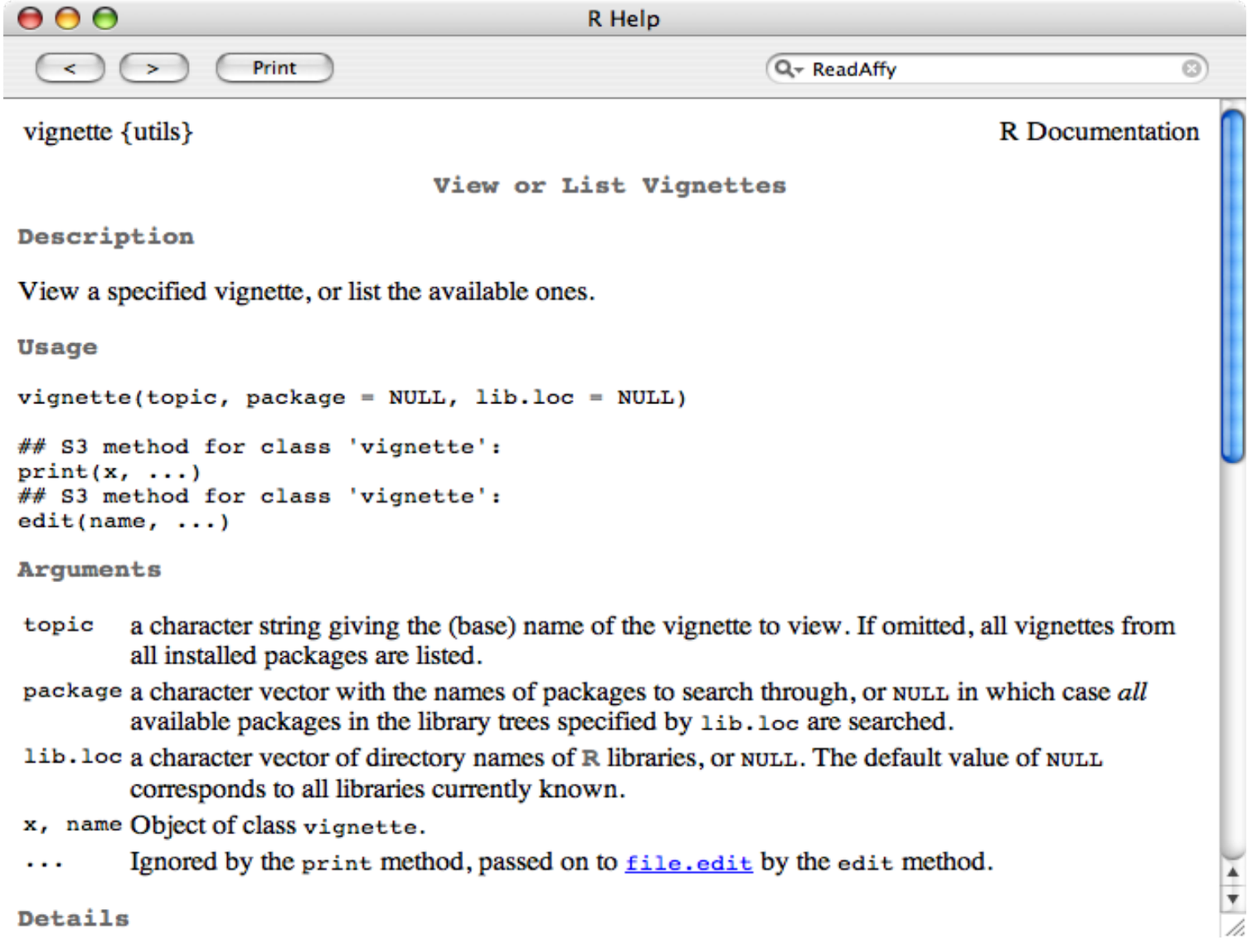
We're going to try to extend this from Perl over to BioConductor, and to the analysis of Affy data.

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?

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

Reading the Fine Manual: Vignettes



R Help

< > Print ReadAffy

vignette {utils} R Documentation

View or List Vignettes

Description

View a specified vignette, or list the available ones.

Usage

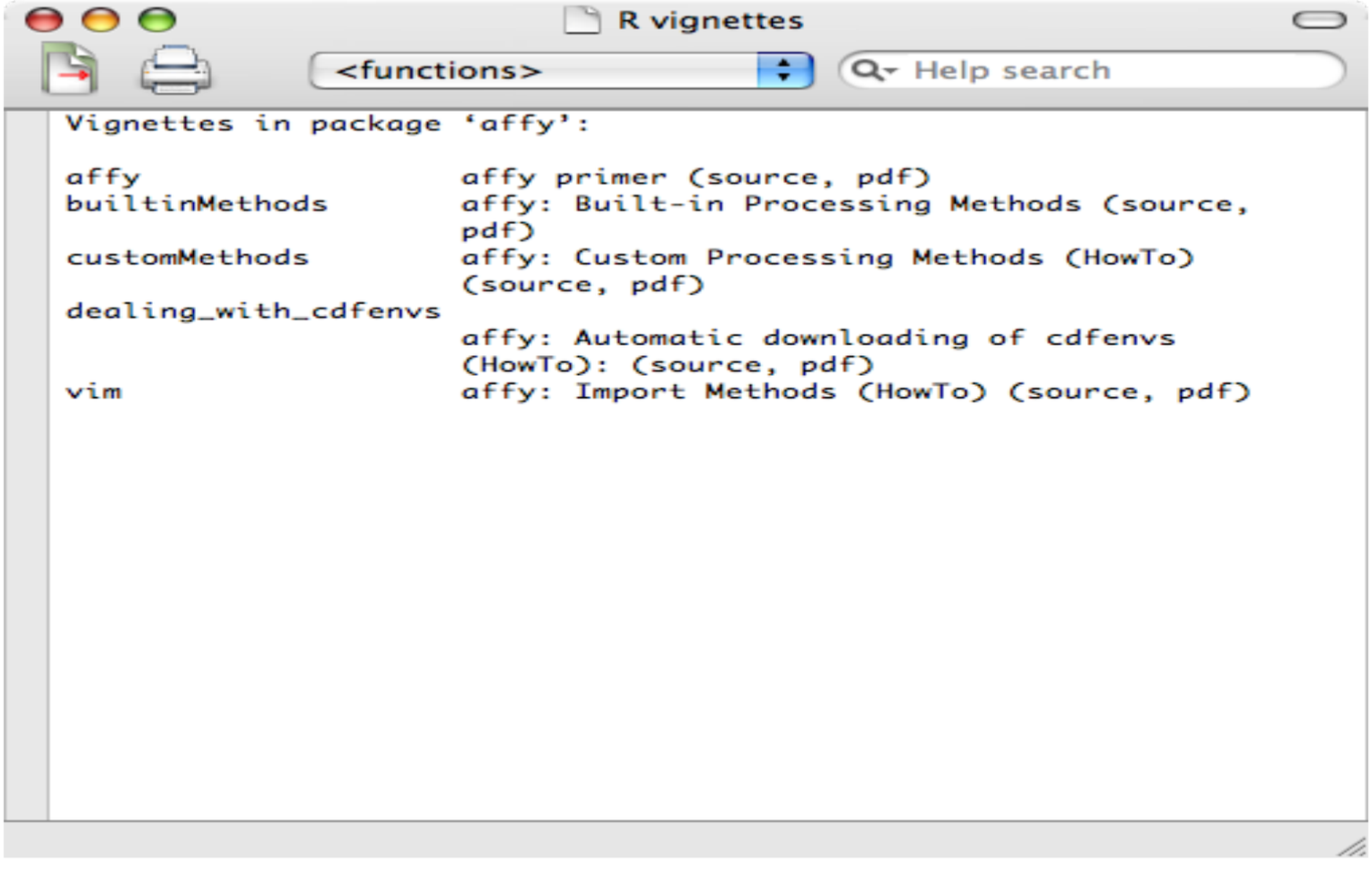
```
vignette(topic, package = NULL, lib.loc = NULL)

## S3 method for class 'vignette':
print(x, ...)
## S3 method for class 'vignette':
edit(name, ...)
```

Arguments

`topic`	a character string giving the (base) name of the vignette to view. If omitted, all vignettes from all installed packages are listed.
`package`	a character vector with the names of packages to search through, or `NULL` in which case *all* available packages in the library trees specified by `lib.loc` are searched.
`lib.loc`	a character vector of directory names of R libraries, or `NULL`. The default value of `NULL` corresponds to all libraries currently known.
`x, name`	Object of class `vignette`.
`...`	Ignored by the `print` method, passed on to `file.edit` by the `edit` method.

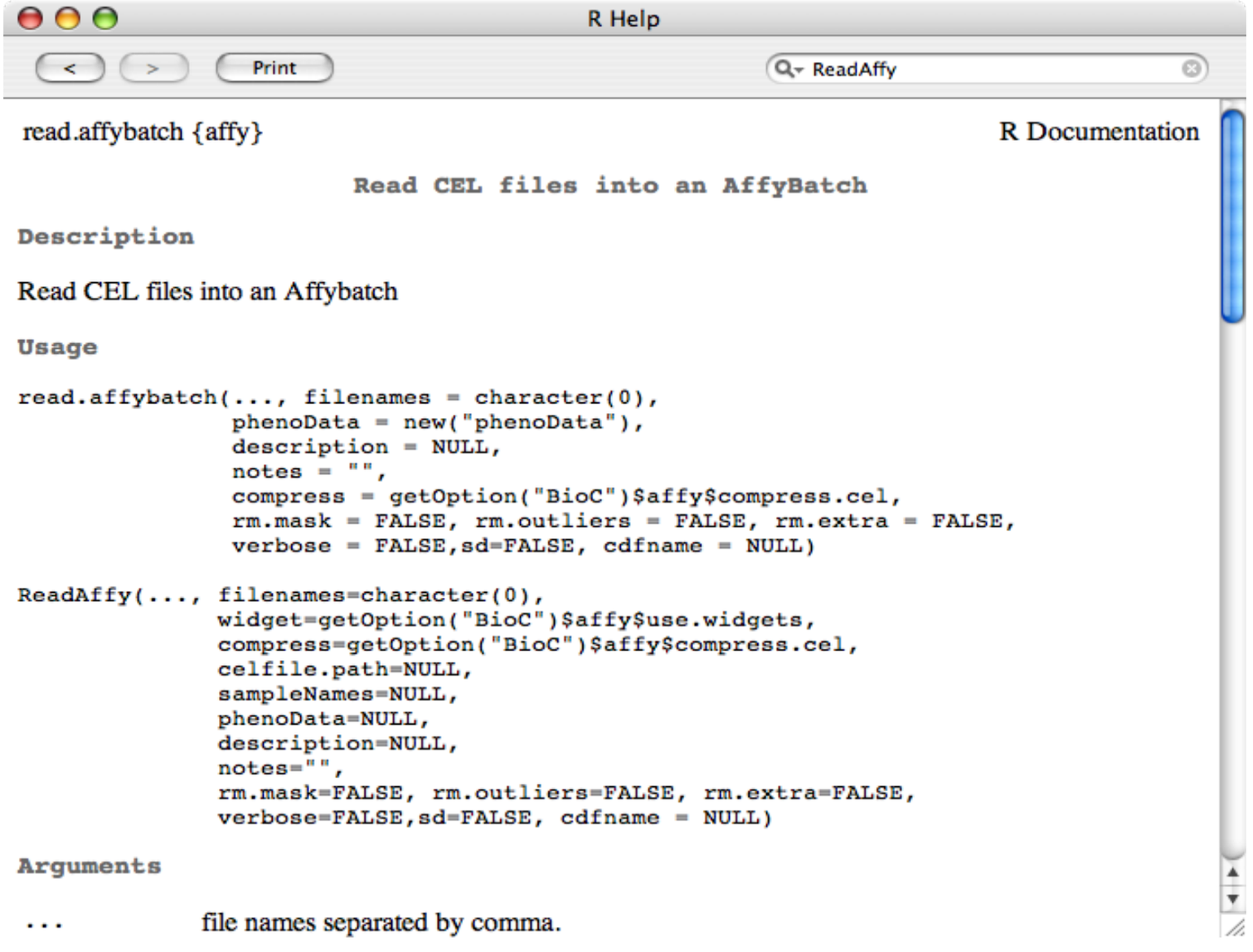
Details

Listing Vignettes

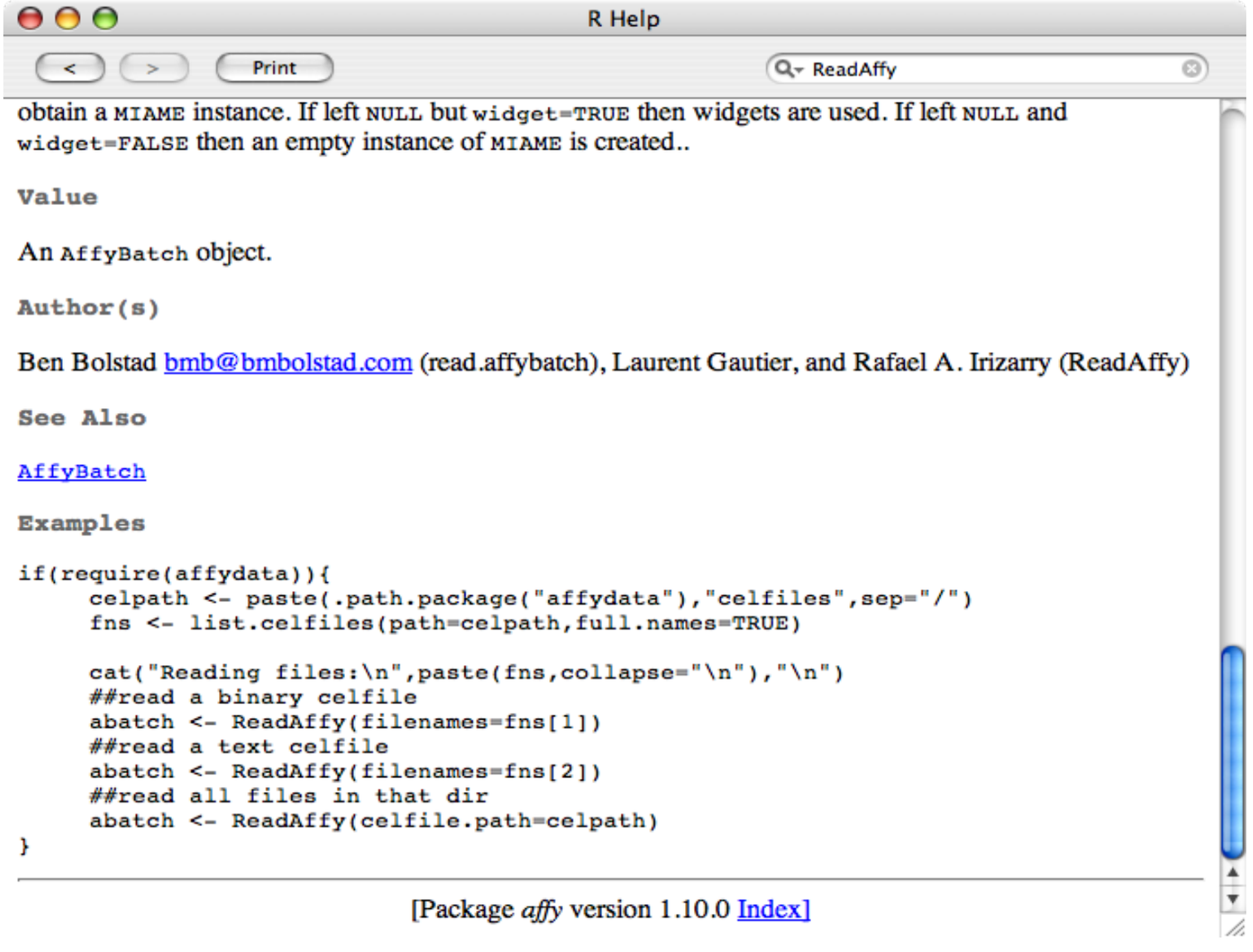



```
> vignette(package = "affy");
```

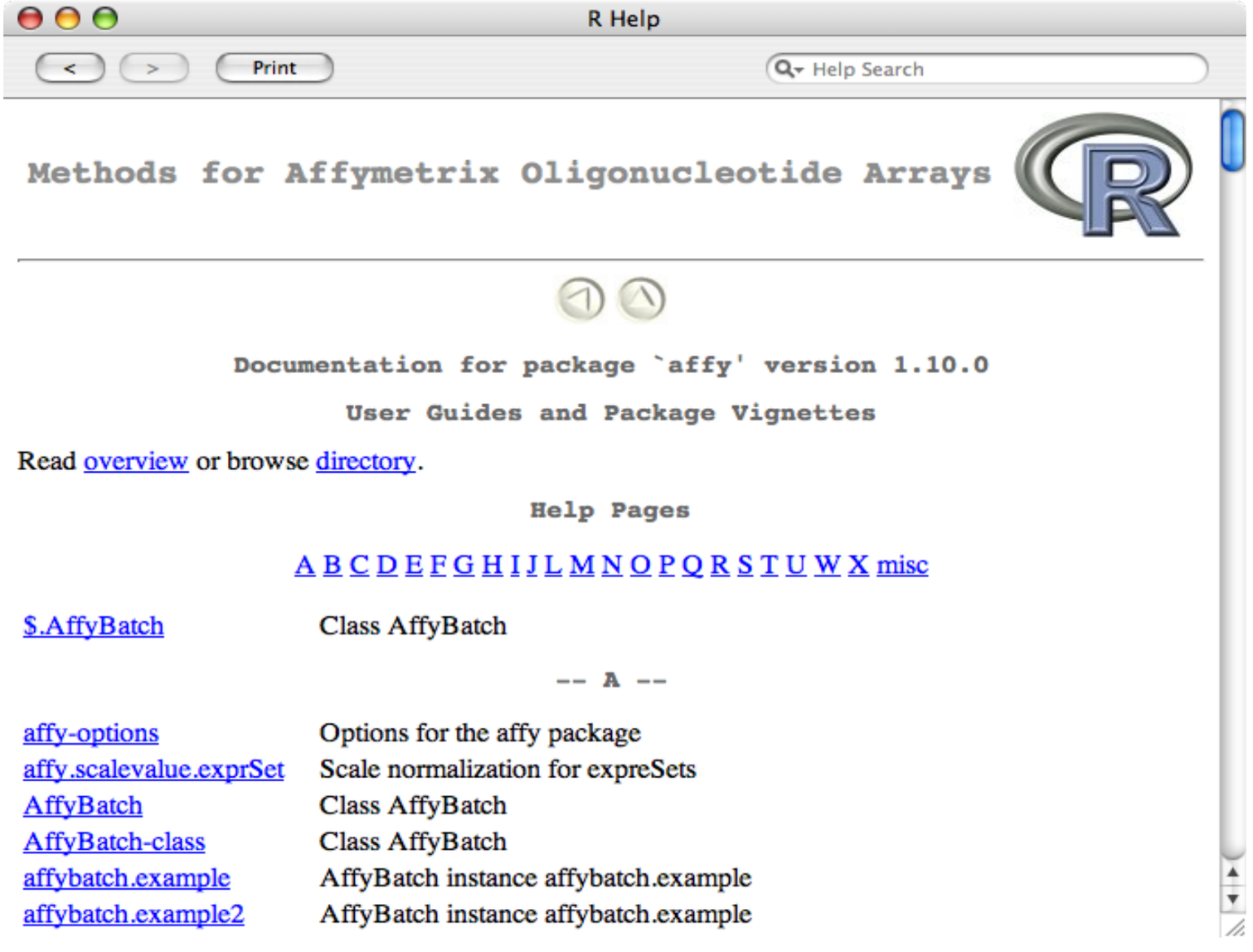
ReadAffy: Help from Top



ReadAffy: ... to Bottom



The Affy Index



Reading a list of files

The file `subcels.txt` contains 20 lines like this:

```
G:/Public/Singh-Prostate-Affymetrix/CelFiles/N01__normal.cel
G:/Public/Singh-Prostate-Affymetrix/CelFiles/N58__normal.cel
G:/Public/Singh-Prostate-Affymetrix/CelFiles/N59__normal.cel
G:/Public/Singh-Prostate-Affymetrix/CelFiles/N61__normal.cel
G:/Public/Singh-Prostate-Affymetrix/CelFiles/N62__normal.cel
...
```

Reading a list of files

```
> basedir <- file.path("G:", "Public", "Singh-Prostate-Affym
> celList <- read.table(file.path(basedir, "subcels.txt"))
> celList[1:6, ]

[1] G:/Public/Singh-Prostate-Affymetrix/CelFiles/N01__normal.
[2] G:/Public/Singh-Prostate-Affymetrix/CelFiles/N58__normal.
[3] G:/Public/Singh-Prostate-Affymetrix/CelFiles/N59__normal.
[4] G:/Public/Singh-Prostate-Affymetrix/CelFiles/N61__normal.
[5] G:/Public/Singh-Prostate-Affymetrix/CelFiles/N62__normal.
[6] G:/Public/Singh-Prostate-Affymetrix/CelFiles/N02__normal.
20 Levels: G:/Public/Singh-Prostate-Affymetrix/CelFiles/N01__
```

Reading the CEL files

```
> ABatch <- ReadAffy(celList)

Error : file names must be specified using a
charactervector, not a 'list'
```

oops...

The Evolution...

```
> celList <- as.character(celList$V1)
```

```
> ABatch <- ReadAffy(celList)
```

```
Error : file names must be specified using a
charactervector, not a 'list'
```

oops...

```
> ABatch <- ReadAffy(filenames = celList)
```

Ta Da!

Checking the Contents

```
> slotNames(ABatch)

[1] "cdfName"           "nrow"
[3] "ncol"              "assayData"
[5] "phenoData"         "featureData"
[7] "experimentData"    "annotation"
[9] ".__classVersion__"

> phenoData(ABatch)

  rowNames: N01__normal.cel, N58__normal.cel, ..., T49__tumor
  varLabels and varMetadata:
    sample: arbitrary numbering
```


Looking at phenoData

```
> class(phenoData(ABatch))

[1] "AnnotatedDataFrame"
attr(,"package")
[1] "Biobase"

> slotNames(phenoData(ABatch))

[1] "varMetadata"        "data"
[3] "dimLabels"          ".__classVersion__"

> pd <- phenoData(ABatch)
```

```
> pd@data
```

	sample
N01__normal.cel	1
N58__normal.cel	2
N59__normal.cel	3
N61__normal.cel	4
N62__normal.cel	5
N02__normal.cel	6
N11__normal.cel	7
N18__normal.cel	8
N21__normal.cel	9
N34__normal.cel	10
T36__tumor.cel	11
T40__tumor.cel	12
T43__tumor.cel	13


```
T58__tumor.cel      14
T59__tumor.cel      15
T06__tumor.cel      16
T20__tumor.cel      17
T24__tumor.cel      18
T26__tumor.cel      19
T49__tumor.cel      20
```

```
> pd@varMetadata
```

```
         labelDescription
sample arbitrary numbering
```

```
> pd@dimLabels
```

```
[1] "rowNames"    "columnNames"
```


Assigning phenoData

subsamples.txt:

```
Array name Sample name Status Batch Cluster
N01__normal N01A Normal B2 A
N58__normal N58A Normal B4 A
N59__normal N59A Normal B3 A
N61__normal N61A Normal B3 A
N62__normal N62A Normal B3 A
N02__normal N02B Normal B2 B
N11__normal N11B Normal B2 B
N18__normal N18B Normal B2 B
...
```


Assigning phenoData

```
> p1 <- read.phenoData(file.path(basedir, "subsamples.txt"))
```

```
Error in scan(file, what, nmax, sep, dec, quote,
skip, nlines, na.strings, : line 2 did not have 7
elements
```

```
In addition: Warning message:
read.phenoData is deprecated, use read.AnnotatedDataFrame
instead
```

Assigning phenoData, pt 2

```
> p1 <- read.AnnotatedDataFrame(file.path(basedir,
+     "subsamples.txt"), sep = "\t")
> p1

  rowNames: 1, 2, ..., 21 (21 total)
  varLabels and varMetadata:
    V1: read from file
    V2: read from file
    ...: ...
    V5: read from file
    (5 total)
```

Not quite what we want.

Assigning phenoData, pt 3

```
> p1 <- read.AnnotatedDataFrame(file.path(basedir,
+     "subsamples.txt"), sep = "\t", header = TRUE)
> p1

  rowNames: 1, 2, ..., 20 (20 total)
  varLabels and varMetadata:
    Array.name: read from file
    Sample.name: read from file
    ...: ...
    Cluster: read from file
    (5 total)

> phenoData(ABatch) <- p1
```


Other ways of Reading Data

Are they all in one directory?

What is the list of filenames?

read.affybatch vs ReadAffy

GUI?

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
```


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...

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...

Quantification

```
> t0 <- date()
> eset0 <- expresso(ABatch, bgcorrect.method = "rma",
+     normalize.method = "quantiles", pmcorrect.method = "pm
+     summary.method = "medianpolish")

background correction: rma
normalization: quantiles
PM/MM correction : pmonly
expression values: medianpolish
background correcting...done.
normalizing...done.
12625 ids to be processed
|                    |
|####################|
```

```
> t1 <- date()
> t0

[1] "Tue Oct 02 12:54:29 2007"

> t1

[1] "Tue Oct 02 13:01:57 2007"
```


Quantification

```
> eset1 <- justRMA(filenames = celList)

Error : the following are not valid files:
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-P
```

```
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr
m:/Private/ArrayCourse/Lectures07/Lecture10/G:/Public/Singh-Pr

> eset1 <- justRMA(filenames = celList, celfile.path = "")

Error : the following are not valid files:
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N01__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N58__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N59__normal.cel
```



```
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N61__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N62__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N02__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N11__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N18__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N21__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/N34__normal.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T36__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T40__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T43__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T58__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T59__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T06__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T20__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T24__tumor.cel
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T26__tumor.cel
```



```
/G:/Public/Singh-Prostate-Affymetrix/CelFiles/T49__tumor.cel

> t2 = date()
> eset1 <- justRMA(filenames = celList, celfile.path = NULL)

Background correcting
Normalizing
Calculating Expression

> t3 = date()
> t2

[1] "Tue Oct 02 13:01:57 2007"

> t3

[1] "Tue Oct 02 13:02:34 2007"
```


The following method will also work:

```
> celpath <- file.path(basedir, "CelFiles")
> cels <- sub(celpath, "", celList)
> eset1 <- justRMA(filenames = cels, celfile.path = celpath)
> t2 <- date()
```

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

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

Just Because I'm Curious

```
> exprs(eset1)[1, 1:3]

N01__normal.cel N58__normal.cel N59__normal.cel
       6.732579        6.893270        7.068655
```

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.

What is the First Gene?

(well, ok, probeset)

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

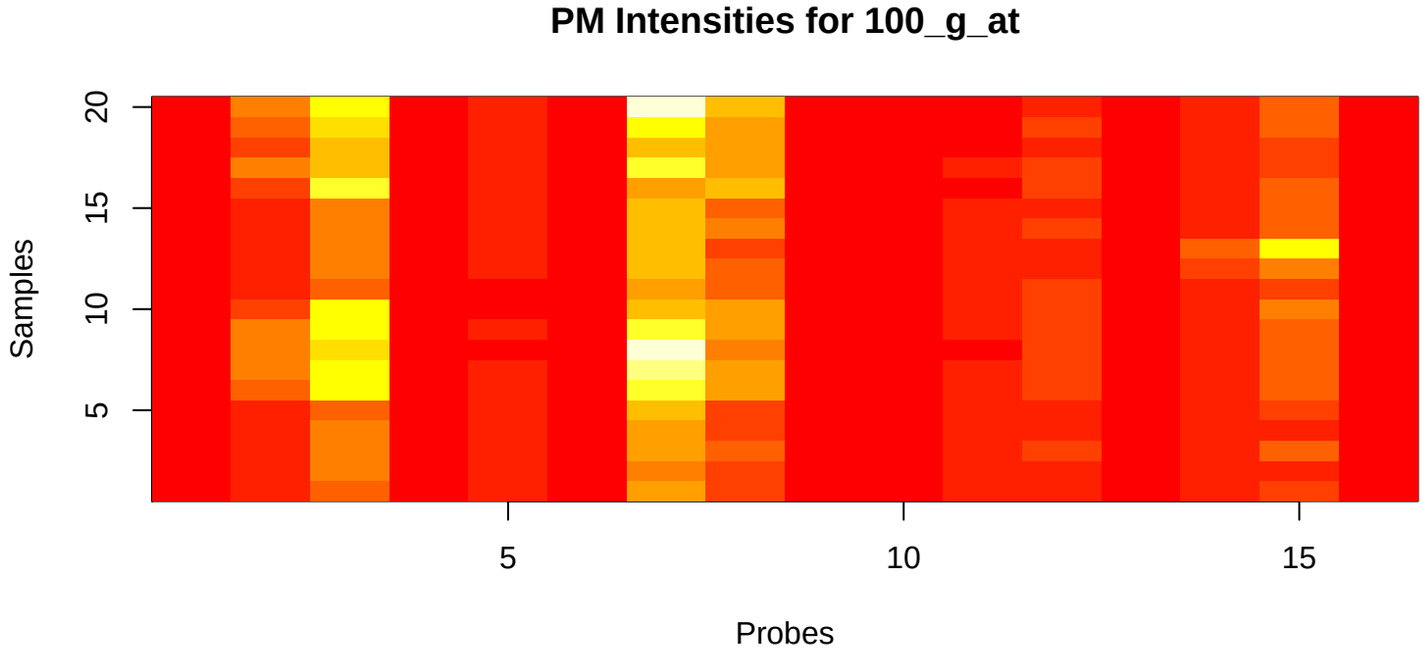
```
[1] "100_g_at"
```

Ok, now what are the values?

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


Looking at it, Take 1

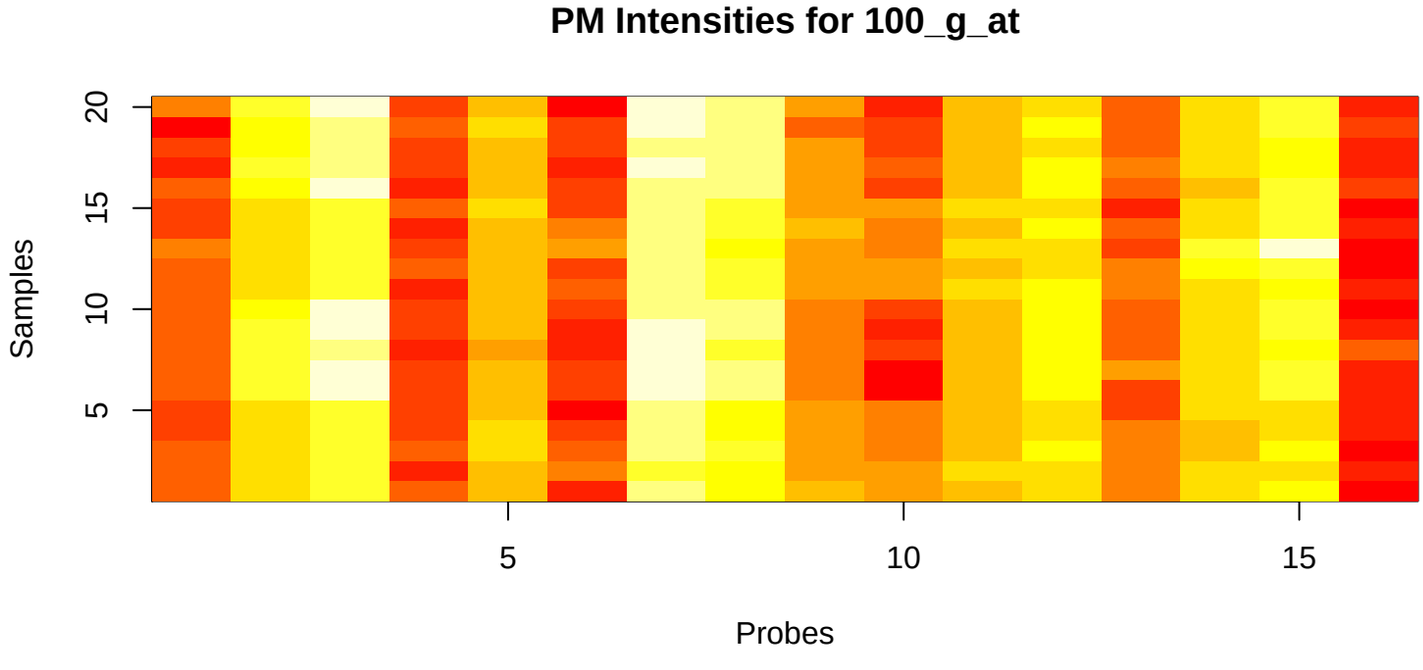
```
> image(1:nrow(pr1), 1:ncol(pr1), pr1, xlab = "Probes",
+     ylab = "Samples", main = paste("PM Intensities for",
+         gn1))
```



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 = paste("PM Intensities for",
+         gn1))
```



Logs!

Fitting the Probes

```
> pr1Fit <- medpolish(log2(pr1))

1 : 121.5456
Final: 120.8225

> names(pr1Fit)

[1] "overall"   "row"       "col"       "residuals"
[5] "name"

> (pr1Fit$overall + pr1Fit$col)[1:3]

       1        2        3
6.732579 6.893270 7.068655
```


This is what we found before!

We can Check the Code

```
> medpolish

function (x, eps = 0.01, maxiter = 10, trace.iter = TRUE, na.
{
    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.rm)
        z <- z - matrix(rdelta, nr = nr, nc = nc)
```



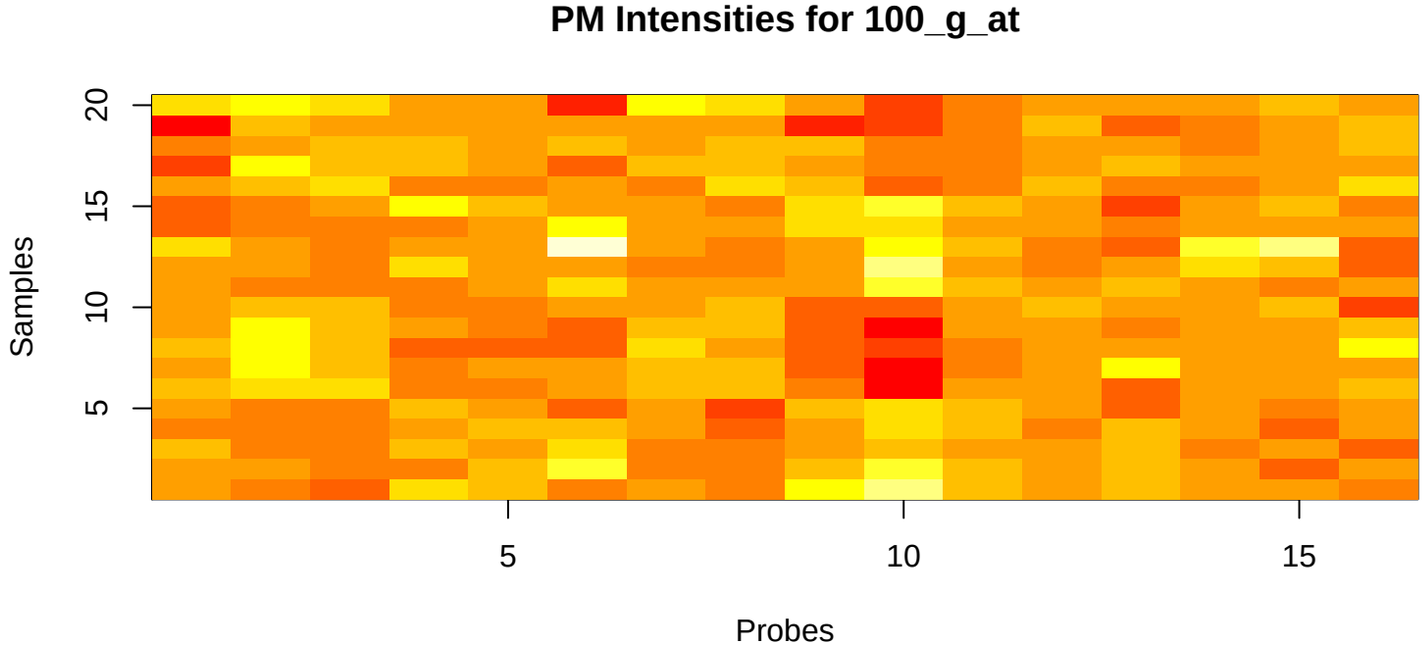
```
        r <- r + rdelta
        delta <- median(c, na.rm = na.rm)
        c <- c - delta
        t <- t + delta
        cdelta <- apply(z, 2, median, na.rm = na.rm)
        z <- z - matrix(cdelta, nr = nr, nc = nc, byrow = TRU
        c <- c + cdelta
        delta <- median(r, na.rm = na.rm)
        r <- r - delta
        t <- t + delta
        newsum <- sum(abs(z), na.rm = na.rm)
        converged <- newsum == 0 || abs(newsum - oldsum) < ep
            newsum
        if (converged)
            break
        oldsum <- newsum
```

```
        if (trace.iter)
            cat(iter, ":", newsum, "\n")
    }
    if (converged) {
        if (trace.iter)
            cat("Final:", newsum, "\n")
    }
    else warning(gettextf("medpolish() did not converge in %d
        maxiter), domain = NA)
    names(r) <- rownames(z)
    names(c) <- colnames(z)
    ans <- list(overall = t, row = r, col = c, residuals = z,
        name = deparse(substitute(x)))
    class(ans) <- "medpolish"
    ans
}
```


<environment: namespace:stats>

and Check the Residuals

```
> image(1:nrow(pr1), 1:ncol(pr1), pr1Fit$residuals,
+     xlab = "Probes", ylab = "Samples", main = paste("PM In
+         gn1))
```

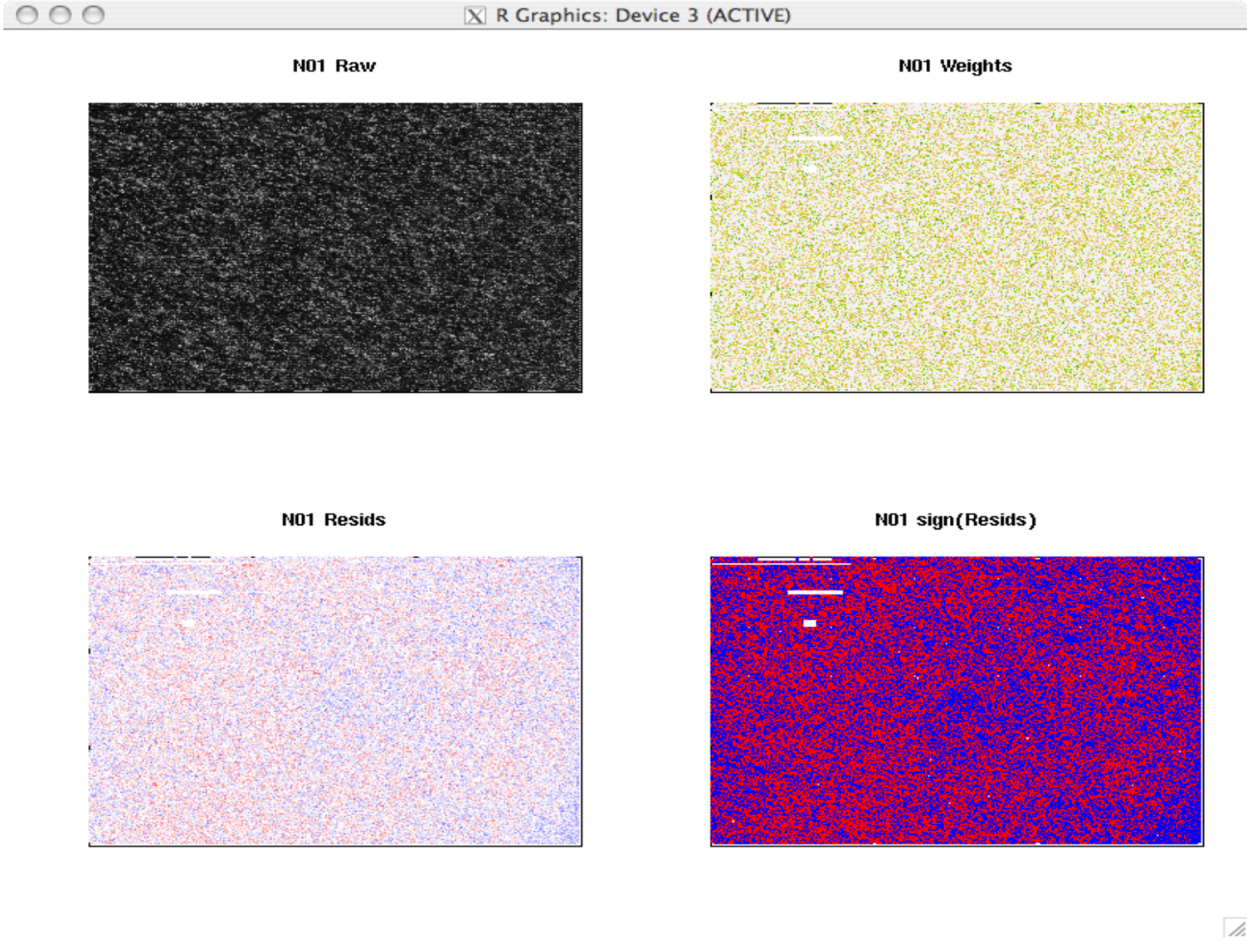


One other Fitting Approach: PLM

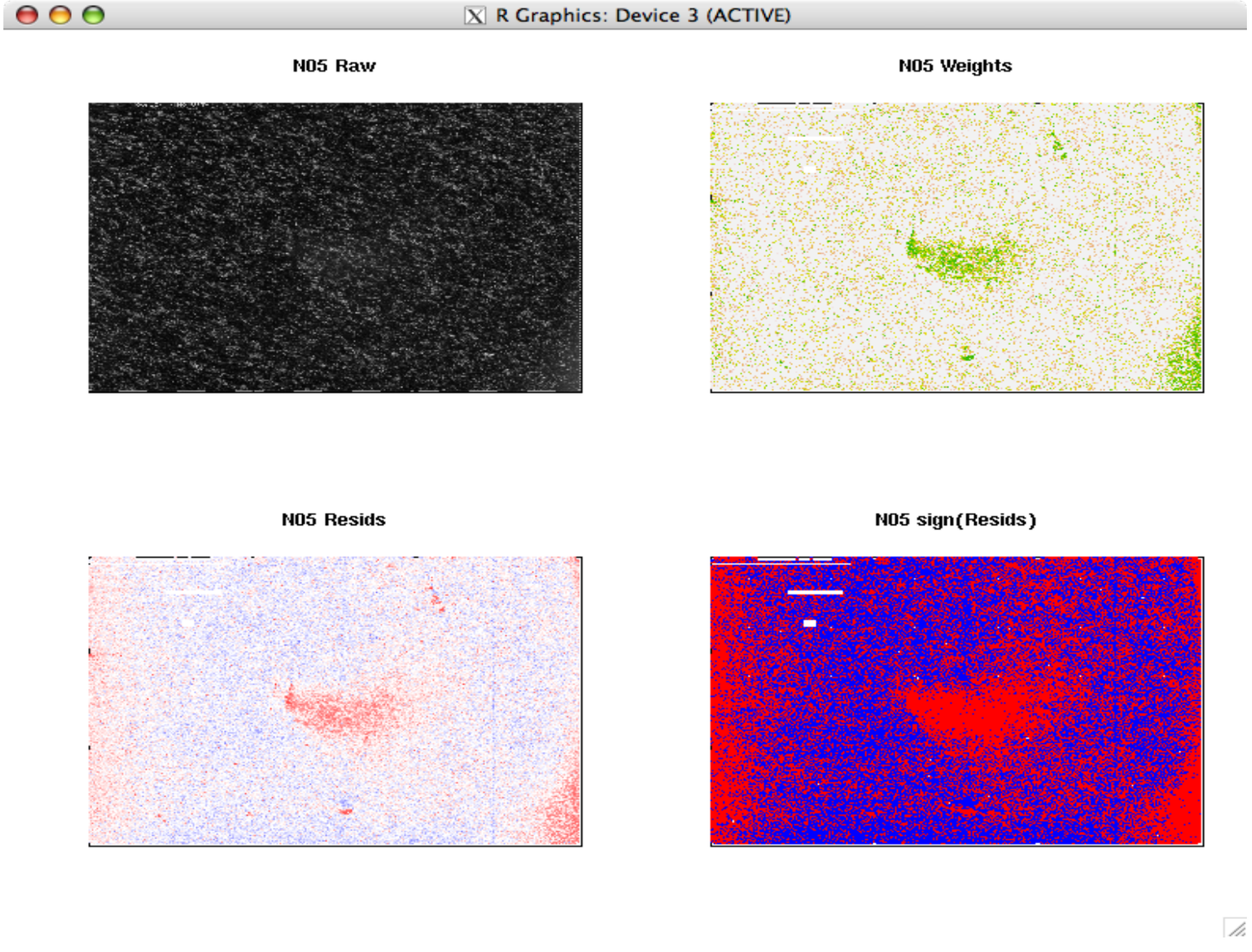
PLM = "Probe Level Model"

```
> library(affyPLM)
> plm1 <- fitPLM(ABatch)

> opar <- par(mfrow = c(2, 2))
> image(ABatch[, 1], main = "NO1 Raw")
> image(plm1, type = "weights", which = 1, main = "NO1 Weigh
> image(plm1, type = "resids", which = 1, main = "NO1 Resids
> image(plm1, type = "sign.resids", which = 1, main = "NO1 s
> par(opar)
```

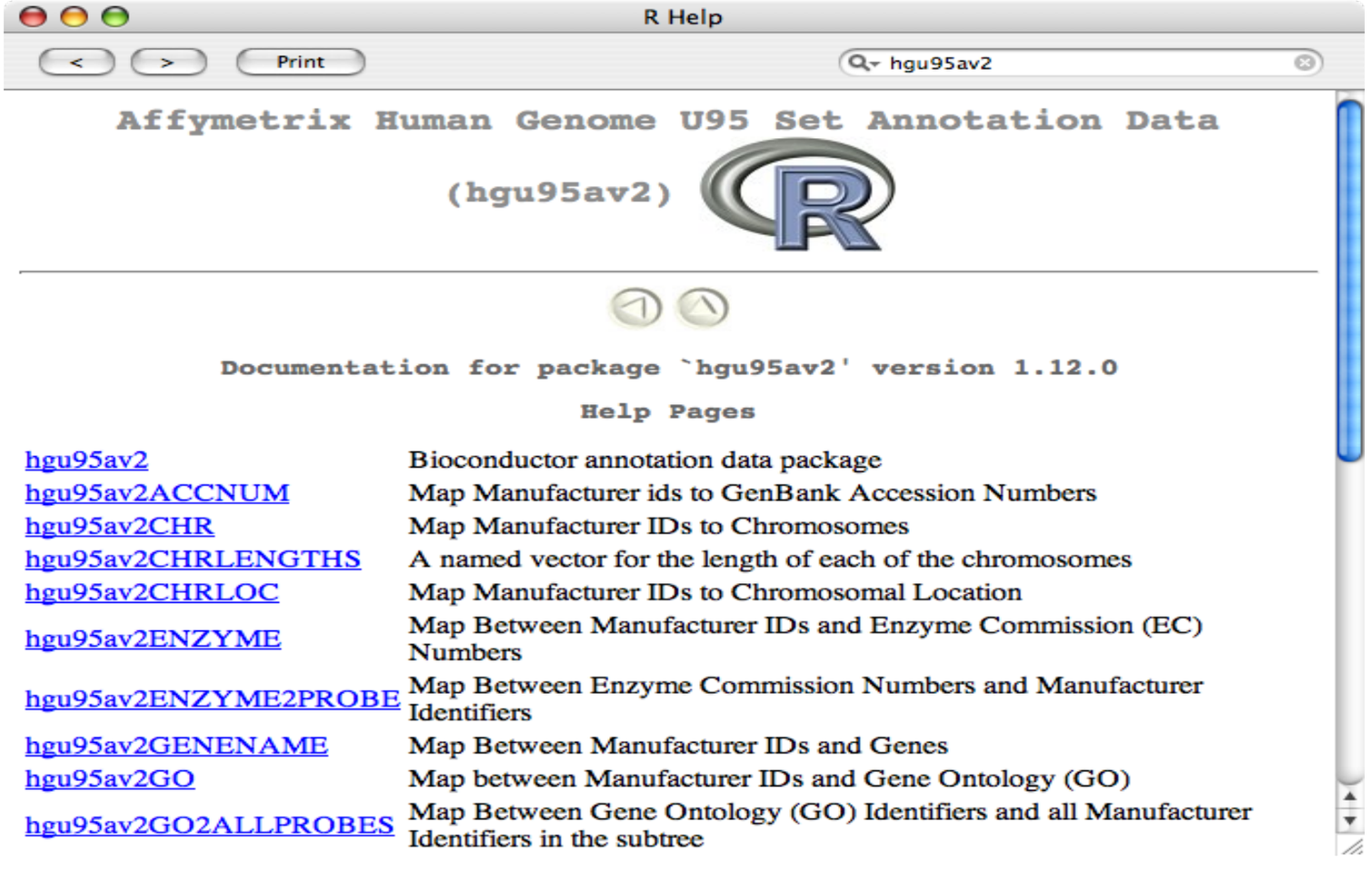

Looking at N01



Looking at N05



Whence the Gene Name Info?




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


What Does This Package Contain?

```
> hgu95av2()

Quality control information for hgu95av2
Date built: Created: Mon Apr 23 12:21:36 2007

Number of probes: 12625
Probe number mismatch: None
Probe missmatch: None
Mappings found for probe based rda files:
        hgu95av2ACCNUM found 12625 of 12625
        hgu95av2CHR found 12149 of 12625
        hgu95av2CHRLOC found 11730 of 12625
        hgu95av2ENZYME found 1861 of 12625
        hgu95av2ENTREZID found 12225 of 12625
```



```
        hgu95av2GENENAME found 12161 of 12625
        hgu95av2GO found 11421 of 12625
        hgu95av2MAP found 12121 of 12625
        hgu95av2OMIM found 10157 of 12625
        hgu95av2PATH found 4322 of 12625
        hgu95av2PFAM found 12046 of 12625
        hgu95av2PMID found 12120 of 12625
        hgu95av2PROSITE found 12046 of 12625
        hgu95av2REFSEQ found 12004 of 12625
        hgu95av2SYMBOL found 12161 of 12625
        hgu95av2UNIGENE found 11973 of 12625
Mappings found for non-probe based rda files:
        hgu95av2CHRLENGTHS found 25
        hgu95av2ENZYME2PROBE found 677
        hgu95av2GO2ALLPROBES found 7501
        hgu95av2GO2PROBE found 5339
```

```
hgu95av2PATH2PROBE found 189
hgu95av2PMID2PROBE found 127350
```

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

What Does This Package Contain?

```
> hgu95av2GENENAME
```

```
<environment: 0x05599244>
```

Almost everything in this package is an "environment", which is the fancy name R uses 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!

What was Needed for Quantification?

```
> library("hgu95av2cdf")
> hgu95av2cdf$"1000_at"
```

```
       pm     mm
 [1,] 358160 358800
 [2,] 118945 119585
 [3,] 323731 324371
 [4,] 223978 224618
 [5,] 313420 314060
 [6,] 349209 349849
 [7,] 199525 200165
 [8,] 213669 214309
 [9,] 236739 237379
[10,] 298099 298739
```

```
[11,] 282744 283384
[12,] 281443 282083
[13,] 349198 349838
[14,] 297953 298593
[15,] 317054 317694
[16,] 404069 404709
```

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

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]
```

```
$`1114_at`
[1] "BMP4"
```

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")
> getLL("1861_at", "hgu95av2")
```



```
1861_at
     572
```

One More Thing...

sequences?

```
> library("hgu95av2probe")
> data(hgu95av2probe)
> as.data.frame(hgu95av2probe[1, ])

                    sequence   x   y Probe.Set.Name
1 TGGCTCCTGCTGAGGTCCCCTTTCC 395 301        1138_at
  Probe.Interrogation.Position Target.Strandedness
1                         2631           Antisense
```


So, What is BAD?

```
> as.data.frame(hgu95av2probe[hgu95av2probe$Probe.Set.Name =
+     "1861_at", ])
```

	sequence	x	y	Probe.Set.Name
14006	CAACCTCTGGGCAGCACAGCGCTAT	403	485	1861_at
14007	AACCTCTGGGCAGCACAGCGCTATG	402	485	1861_at
14008	CCTCTGGGCAGCACAGCGCTATGGC	207	491	1861_at
14009	TGGGCAGCACAGCGCTATGGCCGCG	436	421	1861_at
14010	GCAGCACAGCGCTATGGCCGCGAGC	285	599	1861_at
14011	AGCACAGCGCTATGGCCGCGAGCTC	207	601	1861_at
14012	ACAGCGCTATGGCCGCGAGCTCCGG	632	337	1861_at
14013	CTATGGCCGCGAGCTCCGGAGGATG	196	609	1861_at
14014	TATGGCCGCGAGCTCCGGAGGATGA	624	167	1861_at
14015	GCTCCGGAGGATGAGTGACGAGTTT	440	447	1861_at

```
14016 GATGAGTGACGAGTTTGTGGACTCC 291 467          1861_at
14017 ATGAGTGACGAGTTTGTGGACTCCT 292 467          1861_at
14018 TGACGAGTTTGTGGACTCCTTTAAG 595 539          1861_at
14019 CGAGTTTGTGGACTCCTTTAAGAAG 488 515          1861_at
14020 GAGTTTGTGGACTCCTTTAAGAAGG 478 495          1861_at
14021 TGTGGACTCCTTTAAGAAGGGACTT 502 207          1861_at
      Probe.Interrogation.Position Target.Strandedness
14006                          384         Antisense
14007                          385         Antisense
14008                          387         Antisense
14009                          391         Antisense
14010                          394         Antisense
14011                          396         Antisense
14012                          399         Antisense
14013                          405         Antisense
14014                          406         Antisense
```

14015	417	Antisense
14016	426	Antisense
14017	427	Antisense
14018	432	Antisense
14019	435	Antisense
14020	436	Antisense
14021	441	Antisense