Using oligonucleotide microarray reporter sequence information for preprocessing and quality assessment

Wolfgang Huber and Robert Gentleman

June 7, 2016

Contents

1 Overview

This document presents some basic and simple tools for dealing with the oligonucleotide microarray reporter sequence information in the Bioconductor *probe* packages. This information is used, for example, in the *gcrma* package.

Probe packages are a convenient way for distributing and storing the probe sequences (and related information) of a given chip.

As an example, the package *hgu95av2probe* provides microarray reporter sequences for Affymetrix' *HgU95a version 2* genechip, and for almost all major Affymetrix genechips, the corresponding packages can be downloaded from the Bioconductor website. If you have the reporter sequence information of a particular chip, you can also create such a package yourself. This is described in in the makeProbePackage vignette of the *AnnotationForge* package.

This document assumes some basic familiarity with R and with the design of the *AffyBatch* class in the *affy* package, Bioconductor's basic container for Affymetrix genechip data.

First, let us load the *Biostrings* package and some other packages we will use.

```
> library(Biostrings)
> library(hgu95av2probe)
> library(hgu95av2cdf)
```

2 Using probe packages

Help for the probe sequence data packages can be accessed through

```
> ? hgu95av2probe
```

One of the issues that you have to deal with is that the *probe* packages do not provide the reporter sequences of all the features present in an *AffyBatch*. Some sequences are missing, some are implied; in particular, the data structure in the *probe* packages does not explicitely contain the sequences of the mismatch probes, since they are implied by the perfect match probes. Also, some other features, typically harbouring control probes or empty, do not have sequences. This is the choice that Affymetrix made when they made files with probe sequences available, and we followed it.

Practically, this means that the vector of probe sequences in a *probe* package does not align 1:1 with the rows of the corresponding *AffyBatch*; you need to keep track of this mapping, and some tools for this are provided and explained below (see Section **??**). It also means that some functions from the *affy* package, such as `pm`, cannot be used when the sequences of the probes corresponding to their result are needed; since `pm` reports the intensities, but not the identity of the probes it has selected, yet the latter would be needed to retrieve their sequences.

2.1 Basic functions

Let us look at some basic operations on the sequences.

2.1.1 Reverse and complementary sequence

DNA sequences can be reversed and/or complemented with the `reverse`, `complement` and `reverseComplement` functions.

```
> ? reverseComplement
```

2.1.2 Matching sets of probes against each other

```
> pm <- DNAStringSet(hgu95av2probe)
> dict <- pm[3801:4000]
> pdict <- PDict(dict)
> m <- vcountPDict(pdict, pm)
> dim(m)
```

```
[1]    200 201800
```

```
> table(rowSums(m))
```

```
  1   2   3
179  15   6
```

```
> which(rowSums(m) == 3)
```

```
[1] 77 79 80 81 82 83
```

```
> ii <- which(m[77, ] != 0)
> pm[ii]
```

```
  A DNAStringSet instance of length 3
    width seq
[1]    25 TCGTCATCAGGTGCATAGCAAGTGA
[2]    25 TCGTCATCAGGTGCATAGCAAGTGA
[3]    25 TCGTCATCAGGTGCATAGCAAGTGA
```

2.1.3 Base content

The base content (number of occurrence of each character) of the sequences can be computed with the function `alphabetFrequency`.

```
> bcpm <- alphabetFrequency(pm, baseOnly=TRUE)
> head(bcpm)
```

```
     A  C G  T other
[1,] 1 10 6  8     0
[2,] 5  5 5 10     0
[3,] 6  4 3 12     0
[4,] 4  7 8  6     0
[5,] 4  5 8  8     0
[6,] 2  7 6 10     0
```

```
> alphabetFrequency(pm, baseOnly=TRUE, collapse=TRUE)
```

```
      A       C       G       T   other
1250858 1235532 1186629 1371981       0
```

2.2 Relating to the features of an *AffyBatch*

```
> nc = hgu95av2dim$NCOL
```

```
[1] 640
```

```
> nr = hgu95av2dim$NROW
```

```
[1] 640
```

Each column of an *AffyBatch* corresponds to an array, each row to a certain probe on the arrays. The probes are stored in a way that is related to their geometrical position on the array. For example, the *hgu95av2* array is geometrically arranged into 640 columns and 640 rows. We call the column and row indices the x- and y-coordinates, respectively. This results in $640 \times 640 = 409600$ probes of the *AffyBatch*; we also call them indices. To convert between x- and y-coordinates and indices, you can use the functions `xy2indices` and `indices2xy` from the *affy* package.

The sequence data in the *probe* packages is addressed by their x and y–coordinates. Let us construct a vector `abseq` that aligns with the indices of an *hgu95av2 AffyBatch* and fill in the sequences:

```
> library(affy)
> abseq = rep(as.character(NA), nc*nr)
> ipm = with(hgu95av2probe, xy2indices(x, y, nc=nc))
> any(duplicated(ipm))  # just a sanity check
```

```
[1] FALSE
```

```
> abseq[ipm] = hgu95av2probe$sequence
> table(is.na(abseq))
```

```
 FALSE   TRUE
201800 207800
```

The mismatch sequences are not explicitely stored in the probe packages. They are implied by the match sequences, by flipping the middle base. This can be done with the `pm2mm` function defined below. For Affymetrix GeneChips the length of the probe sequences is 25, and since we start counting at 1, the middle position is 13.

```
> mm <- pm
> subseq(mm, start=13, width=1) <- complement(subseq(mm, start=13, width=1))
> cat(as.character(pm[[1]]), as.character(mm[[1]]), sep="\n")
```

```
TGGCTCCTGCTGAGGTCCCCTTTCC
TGGCTCCTGCTGTGGTCCCCTTTCC
```

We compute `imm`, the indices of the mismatch probes, by noting that each mismatch has the same x-coordinate as its associated perfect match, while its y-coordinate is increased by 1.

```
> imm = with(hgu95av2probe, xy2indices(x, y+1, nc=nc))
> intersect(ipm, imm)  # just a sanity check
```

```
numeric(0)
```

```
> abseq[imm] = as.character(mm)
> table(is.na(abseq))
```

```
 FALSE   TRUE
403600   6000
```

See Figures **??**–**??** for some applications of the probe sequence information to preprocessing and data quality related plots.

3 Some sequence related "preprocessing and quality" plots

The function `alphabetFrequency` counts the number of occurrences of each of the four bases A, C, G, T in each probe sequence.

```
> freqs <- alphabetFrequency(DNAStringSet(abseq[!is.na(abseq)]), baseOnly=TRUE)
> bc <- matrix(nrow=length(abseq), ncol=5)
> colnames(bc) <- colnames(freqs)
> bc[!is.na(abseq), ] <- freqs
> head(na.omit(bc))
```

```
     A C G  T other
[1,] 6 9 8  2     0
[2,] 5 9 8  3     0
[3,] 6 9 7  3     0
[4,] 6 8 6  5     0
[5,] 4 8 7  6     0
[6,] 5 5 4 11     0
```

Let us define an ordered factor variable for GC content:

```
> GC = ordered(bc[,"G"] + bc[,"C"])
> colores = rainbow(nlevels(GC))
```

And let us create an *AffyBatch* object.

```
> library(affydata)
```

```
     Package
[1,] "affydata"
     LibPath
[1,] "/Library/Frameworks/R.framework/Versions/3.3/Resources/library"
     Item       Title
[1,] "Dilution" "AffyBatch instance Dilution"
```

```
> f <- system.file("extracelfiles", "CL2001032020AA.cel", package="affydata")
> pd <- new("AnnotatedDataFrame", data=data.frame(fromFile=I(f), row.names="f"))
> abatch <- read.affybatch(filenames=f, compress=TRUE, phenoData=pd)
```

Figure **??** shows a barplot of the frequencies of counts in `GC`:

```
> barplot(table(GC), col=colores, xlab="GC", ylab="number")
```

Figure **??**:

```
> boxplot(log2(exprs(abatch)[,1]) ~ GC, outline=FALSE,
+   col=colores, , xlab="GC", ylab=expression(log[2]~intensity))
```

Figure **??**:

```
> png("matchprobes-p2p.png", width=900, height=480)
> plot(exprs(abatch)[ipm,1], exprs(abatch)[imm,1], asp=1, pch=".", log="xy",
+      xlab="PM", ylab="MM", col=colores[GC[ipm]])
> abline(a=0, b=1, col="#404040", lty=3)
> dev.off()
```

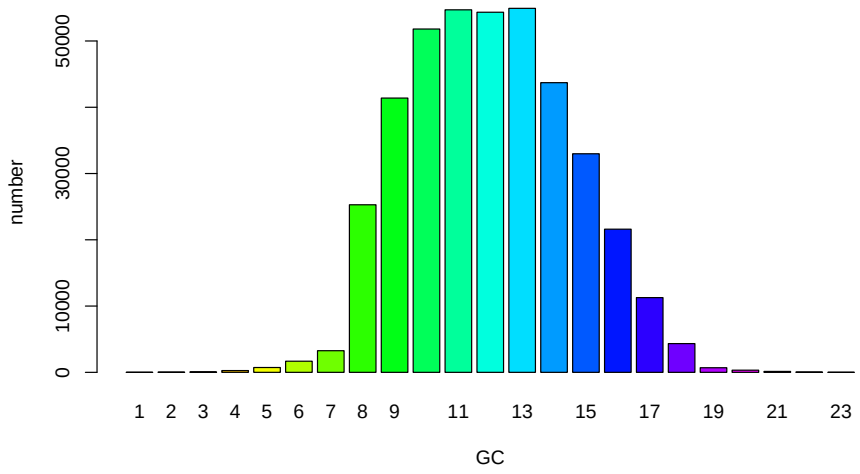

Figure 1: Distribution of probe GC content. The height of each bar corresponds to the number of probes with the corresponing GC content.

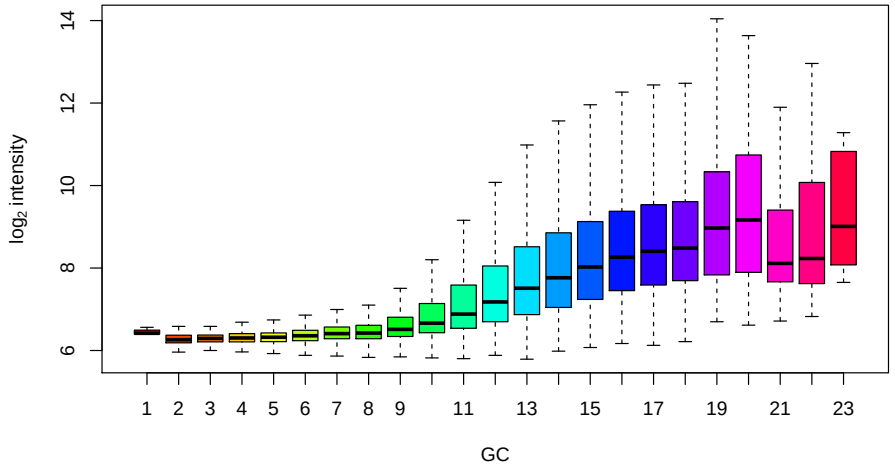

Figure 2: Boxplots of $\log_2$ intensity stratified by probe GC content.

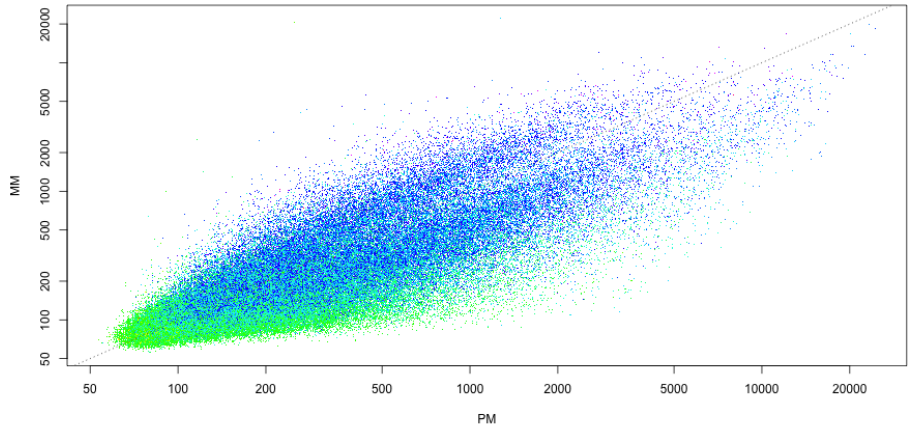

Figure 3: Scatterplot of PM vs MM intensities, colored by probe GC content.