

# Overview: Visualization of Microarray Data

Robert Gentleman

April 22, 2010

## 1 Overview

In this document we present a brief overview of the visualization methods that are available in Bioconductor project. To make use of these tools you will need the packages: *Biobase*, *annotate*, and *genefilter*. These must be installed in your version of R and when you start R you must load them with the `library` command.

A quick word of warning regarding the interpretation of these plots. We can only plot where the gene is supposed to be. If there are translocations or amplifications these will not be detected by microarray analyses.

```
> library(genefilter)
```

## 2 Whole Genome Plots

The functions `cPlot` and `cColor` allow the user to associate microarray expression data with chromosomal location. The plots can include any subset (by default all chromosomes are shown) of chromosomes for the organism being considered.

To make these plots we use the complete reference set of genes for the organism being studied. We must then obtain the chromosomal location (in bases) and orientation (which strand) the gene is on. Chromosomes are represented by straight lines parallel to the  $x$ -axis. Genes are represented by short perpendicular lines. All genes for the experiment (i.e. for an Affymetrix U95A analysis we show all genes on the chips).

The user can then change the color of different sets of the genes according to their needs.

The original setup is done using `cPlot`. The subsequent coloring is done using `cColor`.

We will use the example data in `sample.ExpressionSet` to show how this function might be used.

```
> data(sample.ExpressionSet)
> eset = sample.ExpressionSet
> mytt <- function(y, cov2) {
+   ys <- split(y, cov2)
+   t.test(ys[[1]], ys[[2]])
+ }
> ttout <- esApply(eset, 1, mytt, eset$type)
> s1means <- sapply(ttout, function(x) x$estimate[1])
> s2means <- sapply(ttout, function(x) x$estimate[2])
> deciles <- quantile(c(s1means, s2means), probs = seq(0, 1, 0.1))
> s1class <- cut(s1means, deciles)
> names(s1class) <- names(s1means)
> s2class <- cut(s2means, deciles)
> names(s2class) <- names(s2means)
```

Next we need to set up the graphics output. We do this in a rather complicated way. In the plot below we can compare the mean expression levels for genes in Group 1 with those in Group 2. The Group 1 values are in the left-hand plot and the Group 2 values are in the right-hand plot.

```
cols <- dChip.colors(10)
nf <- layout(matrix(1:3,nr=1), widths=c(5,5,2))
chrObj <- buildChromLocation("hgu95av2")
cPlot(chrObj)
cColor(featureNames(eset), cols[s1class], chrObj)
cPlot(chrObj)
cColor(featureNames(eset), cols[s2class], chrObj)
image(1,1:10,matrix(1:10,nc=10),col=cols, axes=FALSE,
      xlab="", ylab="")
axis(2, at=(1:10), labels=levels(s1class), las=1)
```



