

# ideogram package

Karl Dykema

June 20, 2005

Bioinformatics Special Program, Van Andel Research Institute

## 1 Standard ideogram plots

`ideogram` displays cytogenetic banding information in the plot margin and calls a secondary plotting function to display associated data at the same relative scale. Cytogenetic data for human, mouse, and rat genomes are currently included. We have written this due to a large volume of traditional cytogenetic data currently available.

The data is arranged by associating feature identifiers to genomic location using a `chromLoc` annotation object build using the `buildChromLocation` from the annotation package. As such is a vector of data is to be plotted, the `'names'` attribute of the vector MUST to contain the gene identifiers. Likewise if a matrix of data is to be plotted, the `'rownames'` attribute of the matrix MUST to the gene identifiers.

To date, `'plot'` can be called for vector data, while `'maplot'` and `'image'` can be called for matrix data. Most additional plotting arguments can be passed down via `....`. However, the `ideogram` function plots the axis independently. Therefore arguments like `'cex.axis'` are not passed via the `'plot'` function. The `'cex.axis'`, `'col.axis'`, and `'font.axis'` are intercepted from `...` and redirected to the `'axis'` call. Other parameters that effect the axis should be set via `'par'`.

Below some example data is set up to plot an ideogram.

```
> if (require(hu6800) && require(golubEsets)) {  
+   library(golubEsets)  
+   library(annotate)
```

```

+   library(ideogram)
+   data(golubTrain)
+   hu.chr <- buildChromLocation("hu6800")
+   ex <- golubTrain@exprs[, 1]
+   colors <- rep("black", times = length(ex))
+   colors[ex > 10000] <- "red"
+   pts <- rep(1, times = length(ex))
+   pts[ex > 10000] <- 2
+ }

```

Loading required package: hu6800

Loading required package: golubEsets

Loading required package: Biobase

Welcome to Bioconductor

Vignettes contain introductory material. To view,  
 simply type: `openVignette()`  
 For details on reading vignettes, see  
 the `openVignette` help page.

## 2 Interactive ideogram plots

Many investigators have their favorite region of interest, and would find it useful to dynamically interact with their ideogram plots. This has been added through the `ideograb` function. This has made an easier way for researchers to identify specific genes within cytogenetic regions of interest.

`ideograb` has been written to take an ideogram call as one of its arguments. Two points are first clicked on to determine the coordinates of a rectangular region. All names of genes within that region are then returned in a vector.

## 3 Using ideogram with other packages

Many useful packages can be used in conjunction with `ideogram`. In this section, we will use the *limma* package to determine differentially expressed genes and then plot them with `mideogram`. Please see the *limma* tutorial for more information. Pre-made data files have been included with this package.

The data was used with permission and published in *Haematologica*. 2004 Sep;89(9):1072-81.

The `topTable` function returns a data frame with information about differentially expressed genes. The pre-made data file used in this vignette contains the result of `topTable("t")` and a `chromLocation` object that contains chip-specific annotation information. In this case, we need to create a vector of values with the names attribute set to the corresponding gene name. A colors vector is created with "gray" as the base color, and then altered to accentuate the values with an absolute value of greater than 1.

```
> data(limmaExample)
> data <- t$M
> names(data) <- rownames(t)
> colors <- rep("gray", length(data))
> colors[which(data > 1)] <- "red"
> colors[which(data < (-1))] <- "blue"
> names(colors) <- names(data)
```

```
> ideogram(ex, hu.chr, chr = "1", col = colors, pch = pts, font.axis = 2,  
+         cex.axis = 1)
```

1

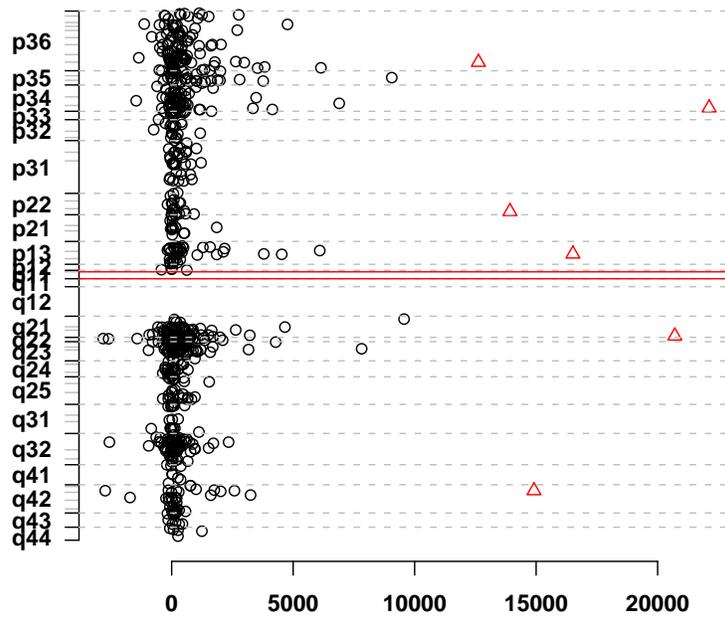


Figure 1: An ideogram

```
> mideogram(data, vai.chr, xlim = c(-5, 5), col = colors, pch = 20)
```

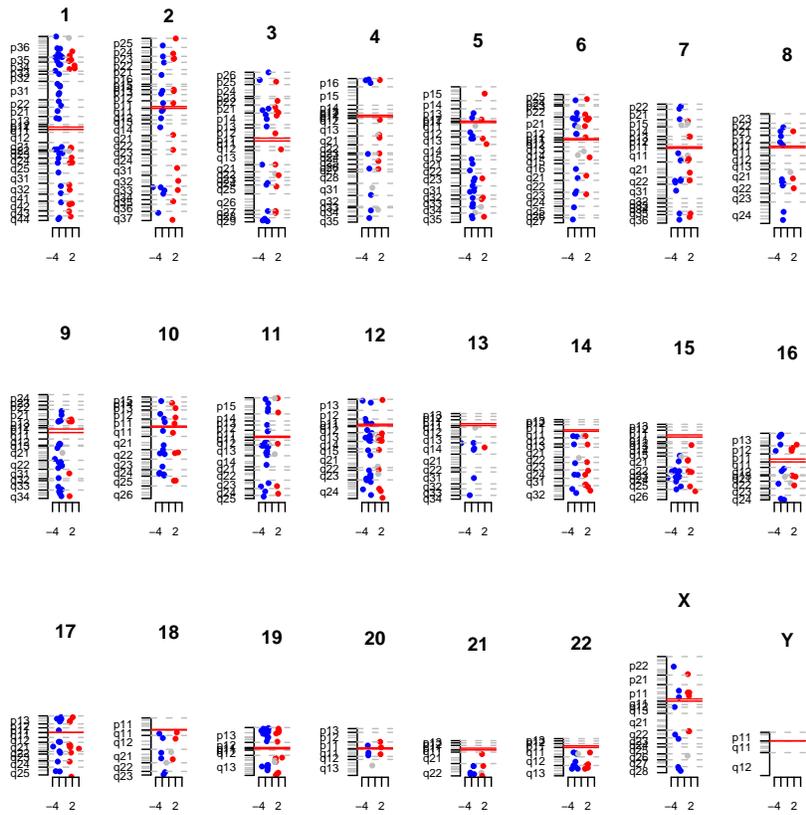


Figure 2: An mideogram