Lab 2: Introduction to Bioconductor Annotation Packages

Sandrine Dudoit, Robert Gentleman, and Katherine S. Pollard

August 13, 2003

One of the largest challenges in analyzing genomic data is associating the experimental data with the available biological *metadata*, e.g., sequence, gene annotation, chromosomal maps, literature. Bioconductor provides two main packages for this purpose: `annotate` (end-user) and `AnnBuilder` (developer). These packages can be used for querying databases such as GenBank, GO, LocusLink, and PubMed, from R, and for processing the query results in R.

In this lab, we focus on annotation resources for Affymetrix chips. Since Affymetrix chips have a standard layout, Bioconductor can provide annotation data packages for the main types of Affymetrix chips, e.g., hu6800, hgu33, hgu95, mgu74, and rgu34 series. These data packages are built using the `AnnBuilder` package. By contrast, there is no standard array design for two-color spotted microarray experiments; each lab tends to have its own custom design. In this case, specific annotation data packages will have to be created for each facility using `AnnBuilder`. Once annotation data packages are constructed to provide mappings between different sets of gene identifiers, the tools in `annotate` can be used in a similar manner for both platforms.

1 Mappings between different gene identifiers

A main task in annotation is to associate manufacturer (e.g., for Affymetrix chips) or in-house (e.g., for custom cDNA spotted arrays) probe identifiers to other available identifiers (e.g., PMID, GenBank accession number).

In this lab, we will use the Bioconductor annotation data packages to map between Affymetrix identifiers and identifiers for biological metadata available on the WWW. Because these data sources are very large, are constantly evolving, and are similar across species, we have adopted the strategy of distributing data in regularly updated R packages. Each mapping is contained in an *environment*. For our purposes, an environment is simply a *hash table*. It provides a very fast way of looking up *values* for specific text

keys. This implementation is not the best and we hope to develop real hash tables for R but that will take some time.

We will again use the Golub et al. (1999) dataset as a case study and will explore annotation resources for the Affymetrix HU6800 chip used in this study. The Bioconductor annotation data package for the HU6800 chip is `hu6800` and can be downloaded from the "Data packages" section of the Bioconductor website. To load the packages needed for this lab

```
> library(Biobase)
> library(annotate)
> library(tkWidgets)
> library(geneplotter)
> library(golubEsets)
> library(hu6800)
> library(GO)
```

Alternately, since all these packages are required by the `UCSC03` course package, simply use

```
>library(UCSC03)
```

After loading the `hu6800` package, we have access to a number of different datasets (environments). These provide the many different mappings from Affymetrix identifiers to other probe identifiers, such as chromosomal location (in the `hu6800CHR` environment) and PMID (in the `hu6800PMID` environment). Quality control data, counts, etc., for the mappings are available by calling the function `hu6800()` and by examining the help page, `?hu6800`.

The main R functions we will use for dealing with environments are `ls` (to list the names of objects in a specified environment) and `get` (to search for an R object with a given name and return its value, if found, in the specified environment). We obtain the Affymetrix identifiers using `ls` and note that there are 7,129 identifiers for the HU6800 chip (this matches the Golub `exprSets`).

```
> affyID <- ls(env = hu6800CHR)
> length(affyID)
```

```
[1] 7129
```

We now arbitrarily select the probe set with Affymetrix identifier `"U18237_at"` and obtain a number of other IDs corresponding to this probe set. Note that there may be several PMIDs, i.e., PubMed abstracts, associated with a given gene.

```
> mygene <- affyID[4001]
> mygene
```

```
[1] "U18237_at"

> get(mygene, env = hu6800ACCNUM)

[1] "U18237"

> get(mygene, env = hu6800LOCUSID)

[1] 23456

> get(mygene, env = hu6800SYMBOL)

[1] "ABCB10"

> get(mygene, env = hu6800GENENAME)

[1] "ATP-binding"

> get(mygene, env = hu6800SUMFUNC)

[1] NA

> get(mygene, env = hu6800UNIGENE)

[1] "Hs.1710"

> get(mygene, env = hu6800CHR)

[1] "1"

> get(mygene, env = hu6800CHRLOC)

         1
-226093836

> get(mygene, env = hu6800MAP)

[1] "1q42"

> get(mygene, env = hu6800PMID)

[1] "10922475" "10748049" "7766993"

> get(mygene, env = hu6800GO)
```

```
         IEA           NAS           NAS            ND           NAS            ND
"GO:0000166" "GO:0005524" "GO:0004009" "GO:0005554" "GO:0006810" "GO:0000004"
         NAS           IEA            ND           IEA
"GO:0005739" "GO:0016021" "GO:0008372" "GO:0019866"
```

In some cases, we need to obtain data on several genes at once. We wrote a special function for this purpose: `multiget`.

```
> fivegenes <- affyID[6:10]
> fivegenes

[1] "AB000409_at"   "AB000410_s_at" "AB000449_at"   "AB000450_at"
[5] "AB000460_at"

> multiget(fivegenes, env = hu6800PMID)

$"AB000409_at"
[1] "12477932" "10859165" "9155018"

$"AB000410_s_at"
 [1] "12807753" "12717837" "12644468" "12592398" "12578369" "12244119"
 [7] "12189194" "12164330" "12119232" "12117782" "12034821" "11992556"
[13] "11927502" "11902834" "11837743" "11827746" "10449904" "10233168"
[19] "9681819"  "9348312"  "9321410"  "9223306"  "9223305"  "9207108"
[25] "9197244"  "9190902"  "9187114"

$"AB000449_at"
[1] "9344656"

$"AB000450_at"
[1] "12477932" "9344656"

$"AB000460_at"
[1] "9734812"
```

Instead of relying on the general R functions for environments (e.g., `get`), the development version of the `annotate` package also provides new and more user-friendly functions for accessing specific identifiers.

```
> getSYMBOL(fivegenes, data = "hu6800")

  AB000409_at AB000410_s_at   AB000449_at   AB000450_at   AB000460_at
      "MKNK1"        "OGG1"        "VRK1"        "VRK2"      "C4orf8"

> getLL(fivegenes, data = "hu6800")
```

```
  AB000409_at AB000410_s_at   AB000449_at   AB000450_at   AB000460_at
         8569          4968          7443          7444          8603
```

```
> getPMID(fivegenes, data = "hu6800")
```

```
$"AB000409_at"
[1] "12477932" "10859165" "9155018"

$"AB000410_s_at"
 [1] "12807753" "12717837" "12644468" "12592398" "12578369" "12244119"
 [7] "12189194" "12164330" "12119232" "12117782" "12034821" "11992556"
[13] "11927502" "11902834" "11837743" "11827746" "10449904" "10233168"
[19] "9681819"  "9348312"  "9321410"  "9223306"  "9223305"  "9207108"
[25] "9197244"  "9190902"  "9187114"

$"AB000449_at"
[1] "9344656"

$"AB000450_at"
[1] "12477932" "9344656"

$"AB000460_at"
[1] "9734812"
```

```
> gg <- getGO(fivegenes, data = "hu6800")
> getGOdesc(gg[[2]], "MF")
```

```
$"GO:0008534"

"purine-specific oxidized base lesion DNA N-glycosylase activity"

$"GO:0004519"

"endonuclease activity"

$"GO:0000703"

"pyrimidine-specific oxidized base lesion DNA N-glycosylase activity"

$"GO:0016798"

"hydrolase activity, acting on glycosyl bonds"
```

```
$"GO:0016829"

"lyase activity"
```

A variety of information about the probes can now easily be obtained by, for example, querying PubMed, GenBank, and LocusLink, as described below. In addition, one can also use R to compute on these data and perform a number of quality control and exploratory analyses.

```
> whChrom <- multiget(affyID, env = hu6800CHR)
> table(unlist(whChrom))

  1  10  11  12  13  14  15  16  17  18  19   2  20  21  22   3   4   5   6   7
675 251 416 417 107 235 182 268 442  99 624 442 150 100 170 367 276 293 424 325
  8   9   X   Y
238 252 325  26

> vv <- sapply(whChrom, length)
> table(vv)

vv
   1    2
7119   10

> whChrom[vv == 2]

$"D49410_at"
[1] "X" "Y"

$"HG2868-HT3012_s_at"
[1] "X" "Y"

$"HG3936-HT4206_at"
[1] "X" "Y"

$"J03592_at"
[1] "X" "Y"

$"L39064_rna1_at"
[1] "X" "Y"

$"M16279_at"
[1] "X" "Y"
```

```
$"U11090_at"
[1] "X" "Y"

$"U13706_at"
[1] "X" "Y"

$"U82668_rna1_at"
[1] "X" "Y"

$"X17648_at"
[1] "X" "Y"
```

This yields the distribution of probe sets by chromosome. Note that there are 10 genes that have been assigned two chromosome locations. Based on OMIM the these genes are localized to the so called *pseudoautosomal region* where the X and Y cromosomes are similar and there is actual recombination going on between them. For now we can just put them all on the X chromosome (they are all X,Y).

```
> vv2 <- sapply(whChrom, function(x) x[1])
> vv2 <- factor(vv2)
> length(vv2)

[1] 7129

> table(unlist(vv2))

  1  10  11  12  13  14  15  16  17  18  19   2  20  21  22   3   4   5   6   7
675 251 416 417 107 235 182 268 442  99 624 442 150 100 170 367 276 293 424 325
  8   9   X   Y
238 252 325  16
```

2 Querying PubMed

The National Library of Medicine (NLM) provides a great deal of information on biological data. We have only just started to explore these resources. We have already developed tools for interacting with PubMed, but need to develop further functionality on MeSH and other available resources. A mapping has been done between LocusLink and PubMed. This associates specific articles with specific genes. The articles can be queried interactively using tools available in R and other systems. Full text abstracts are readily available; in some cases full text articles are also available (we don't have the tools to make use of the later). We next demonstrate how to interact with PubMed using functions in `annotate`.

Bioconductor functions for querying PubMed and other WWW databases from R rely on the `browseURL` function and the R `XML` package to parse query results. For more details, please consult articles in R News (Gentleman & Gentry (2002), *R News* 2(2); Temple Lang (2001), *R News* 1(1)).

```
> browseURL("www.r-project.org")
```

A `pubMedAbst` class structure was defined for handling PubMed abstracts in R. The slots are

```
> slotNames("pubMedAbst")
```

```
[1] "pmid"         "authors"      "abstText"     "articleTitle" "journal"
[6] "pubDate"      "abstUrl"
```

The basic engine for talking to PubMed is the function `pubmed`. Given a vector of PMIDs, the function either has a browser display a URL showing the results of the PubMed query for those identifiers (`disp="browser"`) or creates an `XMLdoc` object with the same data (`disp="data"`).

```
> pmids <- getPMID(mygene, data = "hu6800")
> pubmed(pmids, disp = "browser")
> absts1 <- pubmed(pmids, disp = "data")
```

The function `pm.getabst` provides a simpler way to download the specified PubMed abstracts (stored in XML) and create a list of `pubMedAbst` objects. The following commands can be used to store the PubMed abstracts for 5 genes in a list of objects of class `pubMedAbst`, to compute the number of abstracts retrieved for each gene, and to print the abstract(s) for the first gene. We can see, for example, that one of the genes has 21 abstracts associated with it.

```
> absts2 <- pm.getabst(fivegenes, "hu6800")
```

```
Loading required package: XML
```

```
> lapply(absts2, length)
```

```
$"AB000409_at"
[1] 3

$"AB000410_s_at"
[1] 27

$"AB000449_at"
[1] 1
```

```
$"AB000450_at"
[1] 2

$"AB000460_at"
[1] 1
```

> absts2[[1]]

```
[[1]]
An object of class pubMedAbs
Slot "pmid":
[1] "12477932"

Slot "authors":
 [1] "RL Strausberg"  "EA Feingold"    "LH Grouse"      "JG Derge"
 [5] "RD Klausner"    "FS Collins"     "L Wagner"       "CM Shenmen"
 [9] "GD Schuler"     "SF Altschul"    "B Zeeberg"      "KH Buetow"
[13] "CF Schaefer"    "NK Bhat"        "RF Hopkins"     "H Jordan"
[17] "T Moore"        "SI Max"         "J Wang"         "F Hsieh"
[21] "L Diatchenko"   "K Marusina"     "AA Farmer"      "GM Rubin"
[25] "L Hong"         "M Stapleton"    "MB Soares"      "MF Bonaldo"
[29] "TL Casavant"    "TE Scheetz"     "MJ Brownstein"  "TB Usdin"
[33] "S Toshiyuki"    "P Carninci"     "C Prange"       "SS Raha"
[37] "NA Loquellano"  "GJ Peters"      "RD Abramson"    "SJ Mullahy"
[41] "SA Bosak"       "PJ McEwan"      "KJ McKernan"    "JA Malek"
[45] "PH Gunaratne"   "S Richards"     "KC Worley"      "S Hale"
[49] "AM Garcia"      "LJ Gay"         "SW Hulyk"       "DK Villalon"
[53] "DM Muzny"       "EJ Sodergren"   "X Lu"           "RA Gibbs"
[57] "J Fahey"        "E Helton"       "M Ketteman"     "A Madan"
[61] "S Rodrigues"    "A Sanchez"      "M Whiting"      "A Madan"
[65] "AC Young"       "Y Shevchenko"   "GG Bouffard"    "RW Blakesley"
[69] "JW Touchman"    "ED Green"       "MC Dickson"     "AC Rodriguez"
[73] "J Grimwood"     "J Schmutz"      "RM Myers"       "YS Butterfield"
[77] "MI Krzywinski"  "U Skalska"      "DE Smailus"     "A Schnerch"
[81] "JE Schein"      "SJ Jones"       "MA Marra"       "M LastName"

Slot "abstText":
        The National Institutes of Health Mammalian Gene Collection (MGC) Prog...

Slot "articleTitle":
        Generation and initial analysis of more than 15,000 full-length human ...
```

```
Slot "journal":
[1] "Proc Natl Acad Sci U S A"

Slot "pubDate":
[1] "Dec 2002"

Slot "abstUrl":
[1] "No URL Provided"


[[2]]
An object of class pubMedAbs
Slot "pmid":
[1] "10859165"

Slot "authors":
[1] "R Cuesta"     "G Laroia"     "RJ Schneider"

Slot "abstText":
        Inhibition of protein synthesis during heat shock limits accumulation ...

Slot "articleTitle":
        Chaperone hsp27 inhibits translation during heat shock by binding eIF4...

Slot "journal":
[1] "Genes Dev"

Slot "pubDate":
[1] "Jun 2000"

Slot "abstUrl":
[1] "No URL Provided"


[[3]]
An object of class pubMedAbs
Slot "pmid":
[1] "9155018"

Slot "authors":
[1] "R Fukunaga" "T Hunter"
```

```
Slot "abstText":
        We have developed a novel expression screening method for identifying ...

Slot "articleTitle":
        MNK1, a new MAP kinase-activated protein kinase, isolated by a novel e...

Slot "journal":
[1] "EMBO J"

Slot "pubDate":
[1] "Apr 1997"

Slot "abstUrl":
[1] "No URL Provided"
```

The functions `pm.titles` and `pm.abstGrep` can then be used to extract the titles from a set of PubMed abstracts and for regular expression matching on these abstracts, respectively. For the genes with Affy IDs `fivegenes`, the following commands extract the abstract titles and search the abstracts for the word "protein", with lower case "p" or upper case "P".

```
> pm.titles(absts2)

[[1]]
[1] "Generation and initial analysis of more than 15,000 full-length human and mouse cD
[2] "Chaperone hsp27 inhibits translation during heat shock by binding eIF4G and facili
[3] "MNK1, a new MAP kinase-activated protein kinase, isolated by a novel expression sc

[[2]]
 [1] "Suppressive activities of OGG1 and MYH proteins against G:C to T:A mutations caus
 [2] "hOGG1 Ser326Cys polymorphism modifies the significance of the environmental risk
 [3] "Mammalian 8-oxoguanine DNA glycosylase 1 incises 8-oxoadenine opposite cytosine i
 [4] "Product-assisted catalysis in base-excision DNA repair."
 [5] "Structural and biochemical exploration of a critical amino acid in human 8-oxogua
 [6] "Conditional targeting of the DNA repair enzyme hOGG1 into mitochondria."
 [7] "Inter-individual variation, seasonal variation and close correlation of OGG1 and
 [8] "A limited association of OGG1 Ser326Cys polymorphism for adenocarcinoma of the lu
 [9] "Protection of human lung cells against hyperoxia using the DNA base excision repa
[10] "The human OGG1 DNA repair enzyme and its association with orolaryngeal cancer ris
[11] "Human OGG1 undergoes serine phosphorylation and associates with the nuclear matri
[12] "hOGG1 Ser(326)Cys polymorphism and modification by environmental factors of stoma
[13] "Association of the hOGG1 Ser326Cys polymorphism with lung cancer risk."
[14] "Reciprocal \"flipping\" underlies substrate recognition and catalytic activation
```

```
[15] "Expression of 8-oxoguanine DNA glycosylase is reduced and associated with neurofi
[16] "Radiation sensitivity depends on OGG1 activity status in human leukemia cell line
[17] "Structure and chromosome location of human OGG1."
[18] "Expression and differential intracellular localization of two major forms of huma
[19] "Genetic polymorphisms and alternative splicing of the hOGG1 gene, that is involve
[20] "Augmented expression of a human gene for 8-oxoguanine DNA glycosylase (MutM) in E
[21] "Opposite base-dependent reactions of a human base excision repair enzyme on DNA c
[22] "Molecular cloning and functional expression of a human cDNA encoding the antimuta
[23] "Cloning and characterization of hOGG1, a human homolog of the OGG1 gene of Saccha
[24] "Cloning and characterization of a mammalian 8-oxoguanine DNA glycosylase."
[25] "A mammalian DNA repair enzyme that excises oxidatively damaged guanines maps to a
[26] "Cloning of a human homolog of the yeast OGG1 gene that is involved in the repair
[27] "Cloning and characterization of mammalian 8-hydroxyguanine-specific DNA glycosyla

[[3]]
[1] "Identification of two novel human putative serine/threonine kinases, VRK1 and VRK2

[[4]]
[1] "Generation and initial analysis of more than 15,000 full-length human and mouse cD
[2] "Identification of two novel human putative serine/threonine kinases, VRK1 and VRK2

[[5]]
[1] "The primary structure and genomic organization of five novel transcripts located c
```

```
> sapply(absts2, function(x) pm.abstGrep("[Pp]rotein", x))

$"AB000409_at"
[1] FALSE  TRUE  TRUE

$"AB000410_s_at"
 [1]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE
[13] FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE
[25]  TRUE  TRUE  TRUE

$"AB000449_at"
[1] TRUE

$"AB000450_at"
[1] FALSE  TRUE

$"AB000460_at"
[1] TRUE
```

The new function `pmAbst2html` from the development version of `annotate` takes a list of `pubMedAbst` objects and generates an HTML report with the titles of the abstracts and links to their full page on PubMed. The report will be stored in the file `pm.html` in the working directory.

```
> pmAbst2html(absts2[[2]],filename="pm.html")
```

3 Querying GenBank

Another source of data is GenBank. We can interact with GenBank in much the same way as with PubMed; the main function for this purpose is `genbank`. The argument `disp="data"` returns an XMLDoc, while `disp="browser"` displays information in the user's browser.

```
> gbacc <- multiget(fivegenes, hu6800ACCNUM)
> genbank(gbacc, disp = "browser")
```

4 Querying LocusLink

Finally, we consider interactions with LocusLink to obtain sequence and other biological information on probe sets of interest.

```
> llid <- getLL(fivegenes, data = "hu6800")
> locuslinkByID(llid)
```

5 HTML reports

The function `ll.htlmpage` generates HTML reports, with one row per gene and a clickable entry which opens the LocusLink webpage for that gene. Additional information can be displayed using the `othernames` argument as shown below. Such HTML reports provide an easy way to communicate results with other researchers working on the same analysis.

```
> tengenes <- affyID[1:10]
> llid <- getLL(tengenes, data = "hu6800")
> symb <- getSYMBOL(tengenes, data = "hu6800")
> map <- multiget(tengenes, hu6800MAP)
> res <- data.frame(tengenes, cbind(unlist(symb), unlist(map)))
> names(res) <- c("Affy ID", "Gene symbol", "Chromosomal location")
> ll.htmlpage(llid, filename = "ll.html", title = "HTML report for 10 genes",
+     othernames = res, table.head = c("LocusID", names(res)),
+     table.center = TRUE)
```

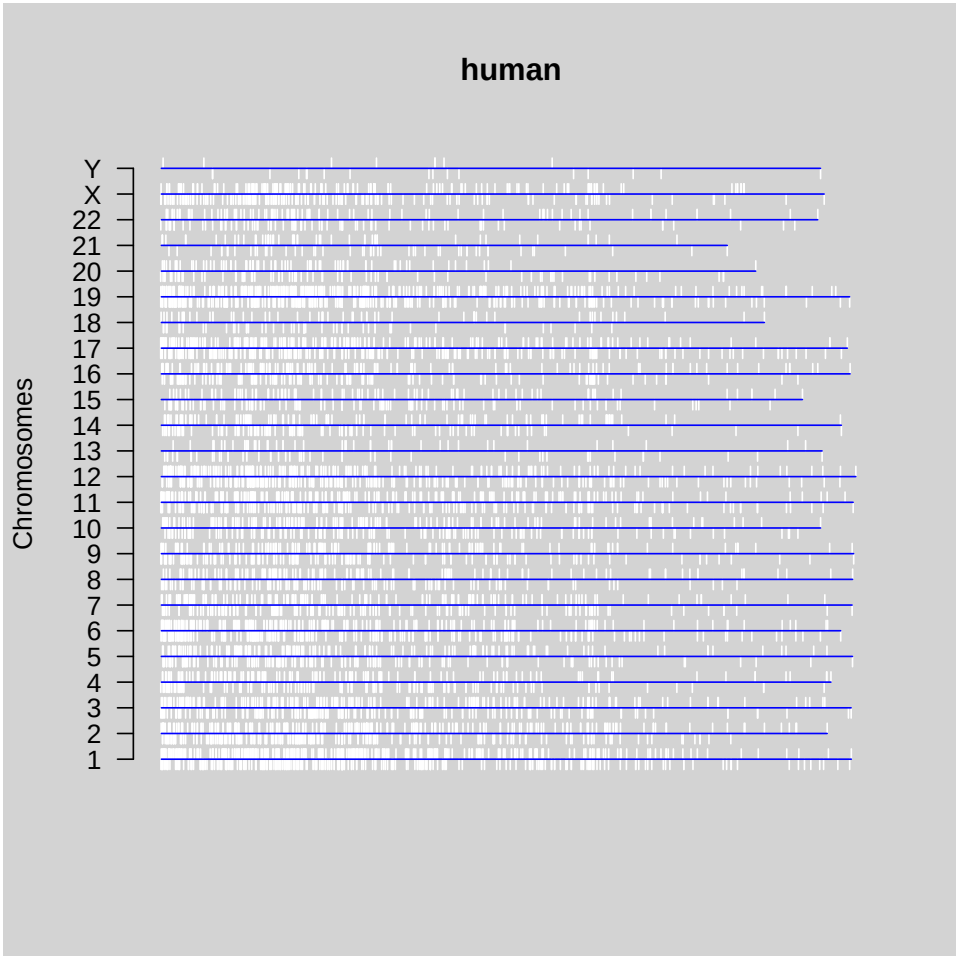
6 Plotting genomic data

The classes `chromLoc` and `chromLocation` are used to keep track of location information for a single gene and a set of genes (entire genome), respectively. We will use the `geneplotter` functions to create an instance of the class `chromLocation` for the Affymetrix HU6800 chip.

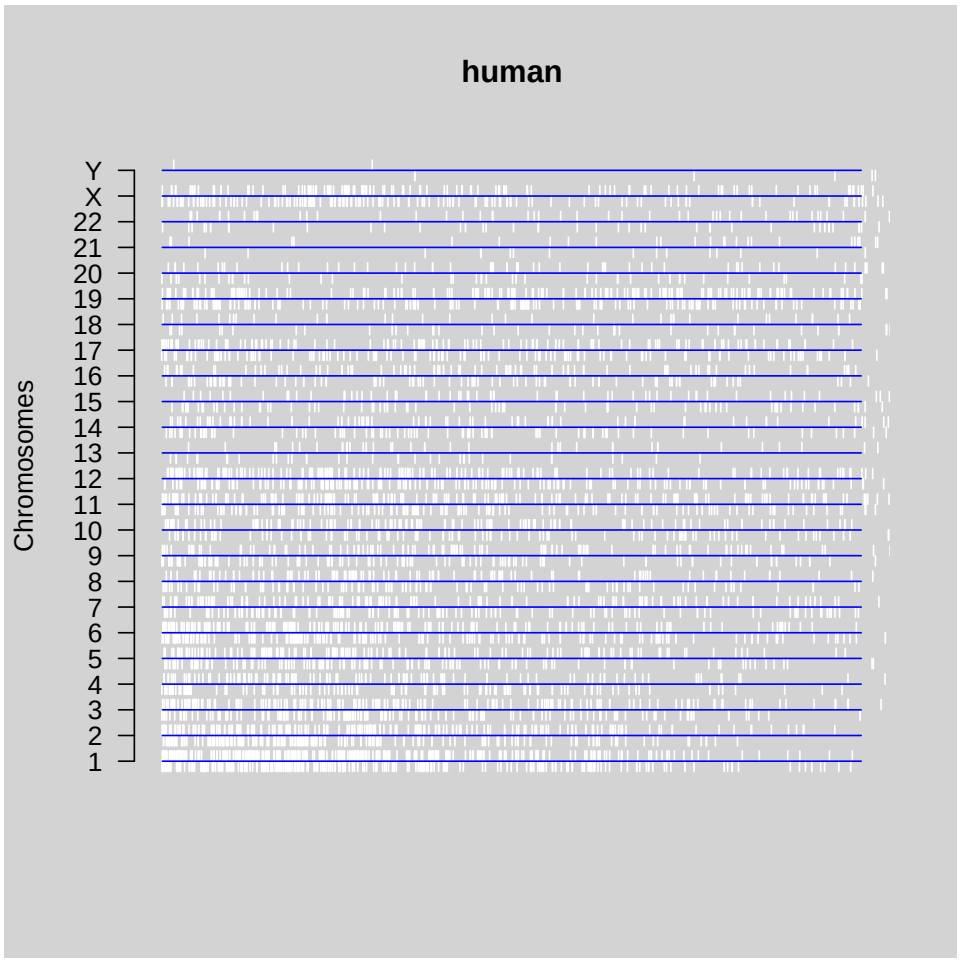
```
> slotNames("chromLoc")

[1] "chrom"    "position" "strand"

> slotNames("chromLocation")

[1] "organism"      "dataSource"    "chromLocs"     "probesToChrom"
[5] "chromInfo"     "geneSymbols"

> strand <- multiget(affyID, env = hu6800CHRLOC)
> splits <- split(strand, vv2)
> length(splits)

[1] 24

> names(splits)

 [1] "1"  "10" "11" "12" "13" "14" "15" "16" "17" "18" "19" "2"  "20" "21" "22"
[16] "3"  "4"  "5"  "6"  "7"  "8"  "9"  "X"  "Y"

> hu6800ChrClass <- buildChromLocation("hu6800")
> organism(hu6800ChrClass)

[1] "human"

> chromNames(hu6800ChrClass)

 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
[16] "16" "17" "18" "19" "20" "21" "22" "X"  "Y"
```

The `geneplotter` package provides two main functions for plotting genomic data: `cPlot` and `alongChrom`. These functions operate on instances of the class `chromLocation`. With `cPlot`, we can render all chromosomes the same length or we can scale them by their relative lengths. Note that the `mva` package has functions for heat maps and dendrograms.

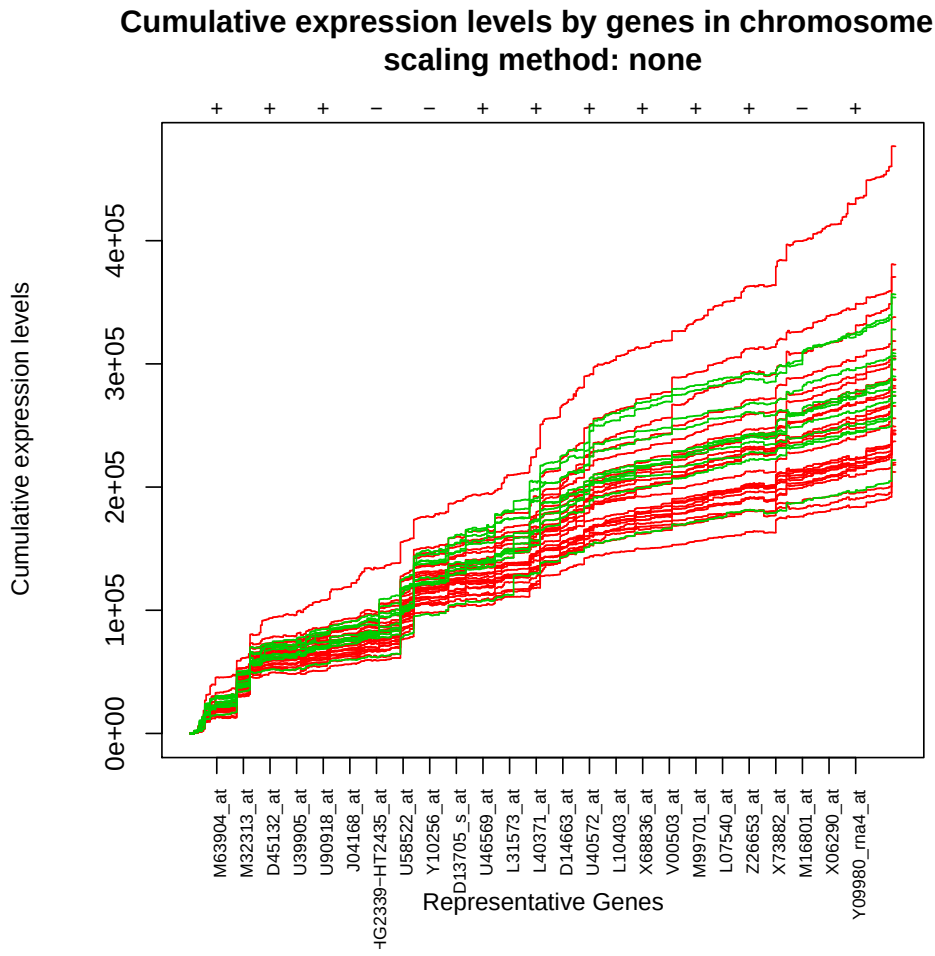
```
> cPlot(hu6800ChrClass, scale = "relative")
```



```
> cPlot(hu6800ChrClass, scale = "max")
```



```
> data(golubTrain)
> cols <- as.numeric(pData(golubTrain)$ALL.AML) + 1
> alongChrom(golubTrain, chrom = "1", specChrom = hu6800ChrClass,
+     col = cols)

<environment: 0xb339394>
```



```
> alongChrom(golubTrain, chrom = "22", specChrom = hu6800ChrClass,
+     plotFormat = "image")
```