Plotting using Genominator and GenomeGraphs (Beta)

James Bullard Kasper Daniel Hansen

Modified: April 18, 2010, Compiled: October 30, 2018

This vignette is preliminary, and should be viewed as subject to change. A number of the functions are not directly exported by the package – there is a reason for that.

In this vignette we demonstrate how to visualize data using the *GenomeGraphs* package. The main idea is that we want to build a plotting function which we can use to plot regions. The simplest case is the following:

First, we make a database:

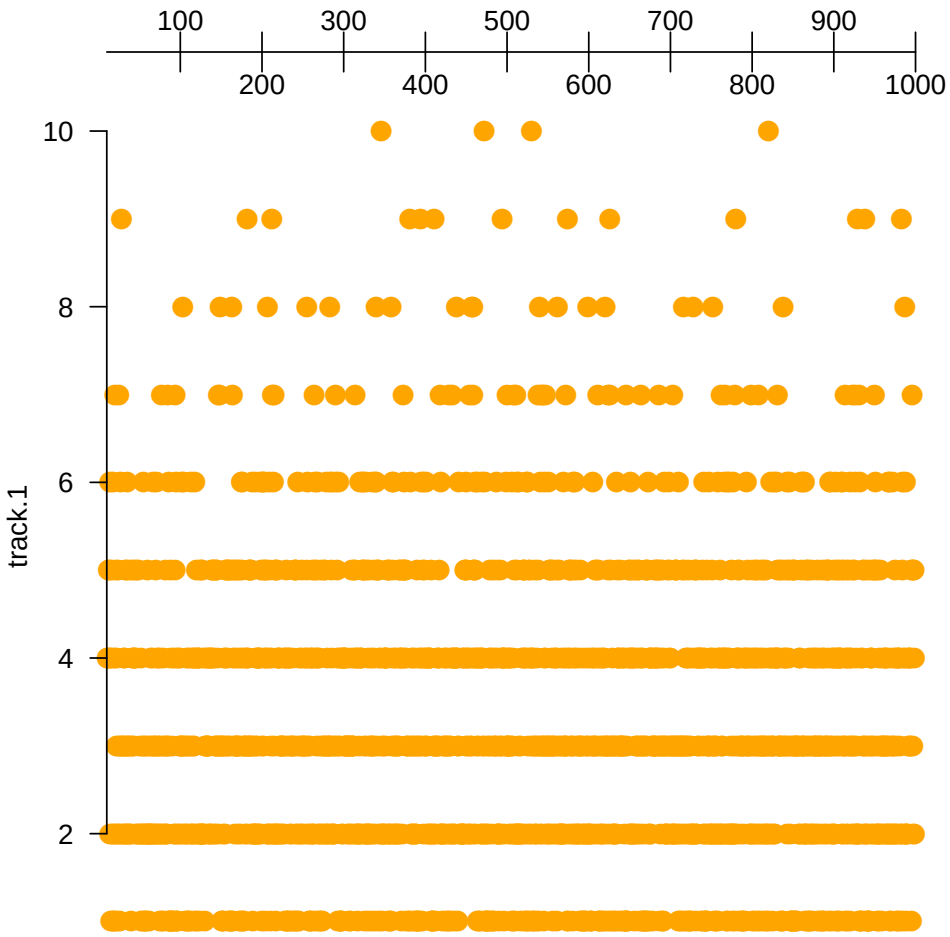
```
> require(Genominator)
> options(verbose = FALSE)
> N <- 100000 # the number of observations.
> K <- 100    # the number of annotation regions, not less than 10
> df <- data.frame(chr = sample(1:16, size = N, replace = TRUE),
+                  location = sample(1:1000, size = N, replace = TRUE),
+                  strand = sample(c(1L,-1L), size = N, replace = TRUE))
> eData <- aggregateExpData(importToExpData(df, dbFilename = "pmy.db", overwrite = TRUE, tablename = "ex
> annoData <- data.frame(chr = sample(1:16, size = K, replace = TRUE),
+                        strand = sample(c(1, -1), size = K, replace = TRUE),
+                        start = (st <- sample(1:1000, size = K, replace = TRUE)),
+                        end = st + rpois(K, 75),
+                        feature = c("gene", "intergenic")[sample(1:2, size = K, replace = TRUE)])
> rownames(annoData) <- paste("elt", 1:K, sep = ".")

> rp <- Genominator:::makeRegionPlotter(list("track.1" = list(expData = eData, what = "counts")))
> args(rp)

function (chr, start, end, overlays = NULL, title = NULL, ...)
NULL
```

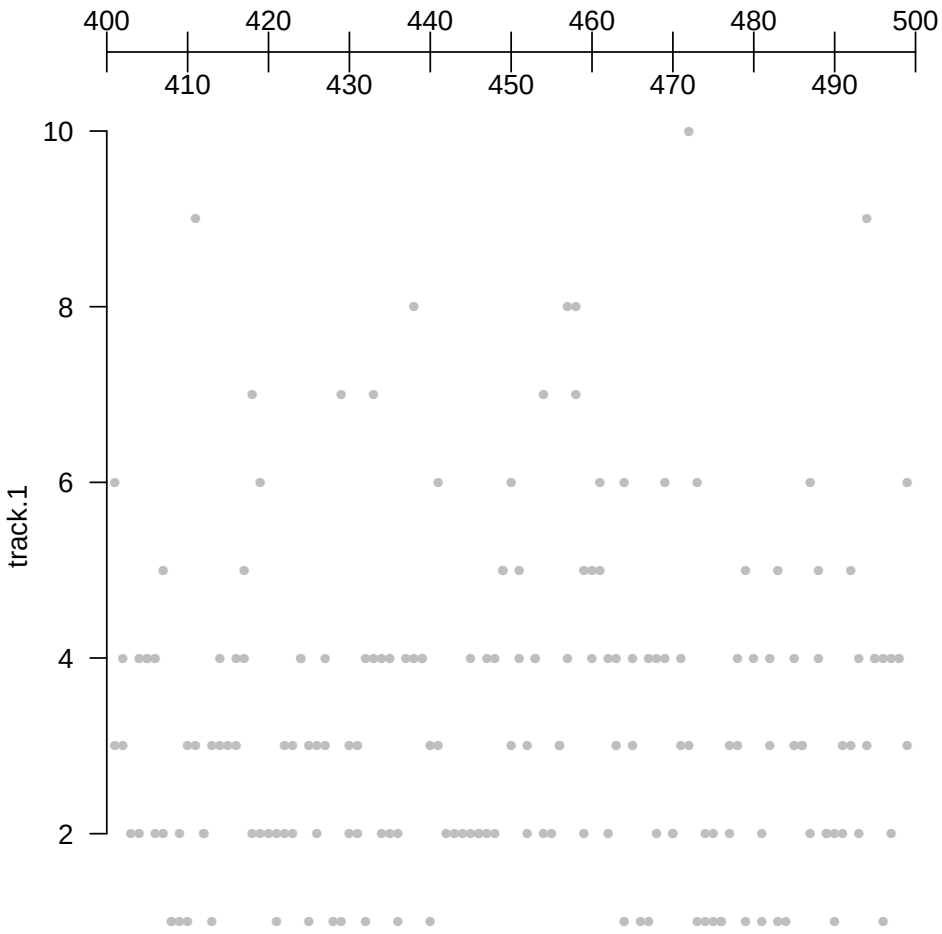
This constructs a function which can be called to view particular pieces of data.

```
> rp(1, 10, 1000)
```



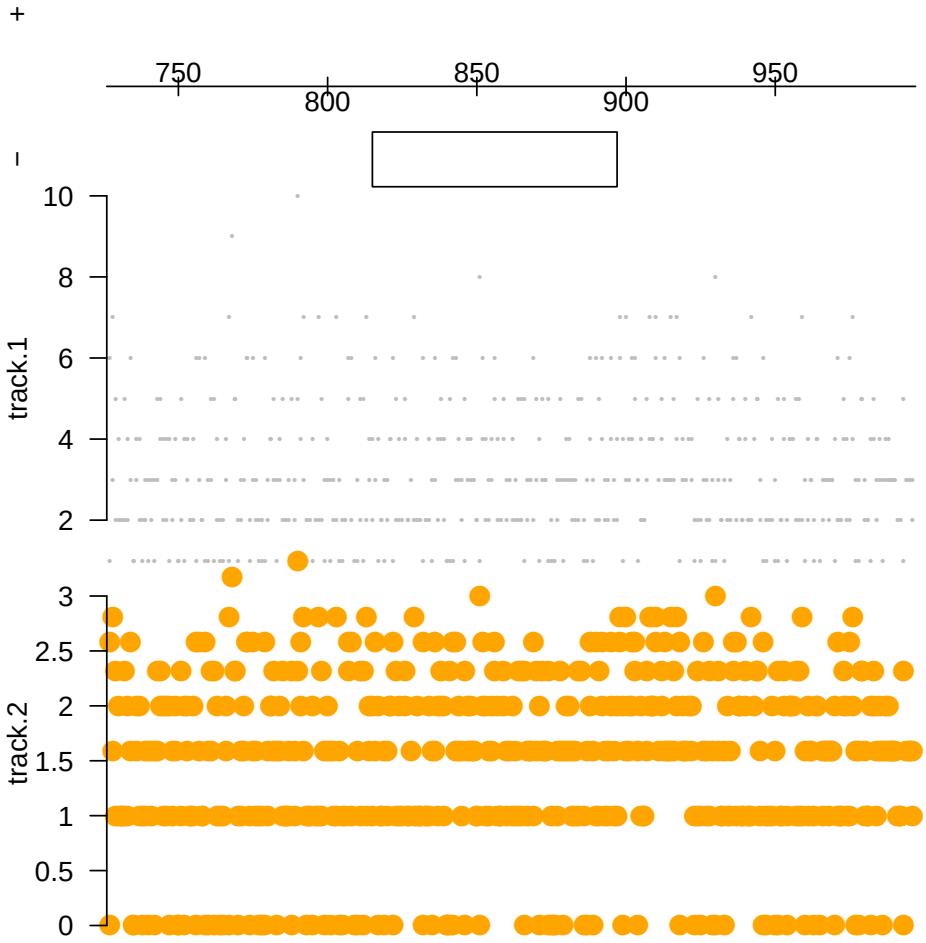
GenomeGraphs provides a wealth of customization options and means of plotting which for the most part are transferable using the list.

```
> rp <- Genominator:::makeRegionPlotter(list("track.1" = list(expData = eData, what = "counts",
+                                                dp = DisplayPars(lwd = .45, color = "grey"))))
> rp(1, 400, 500)
```



Here we can plot our annotation using the annotation factory construct. This is probably a little advanced. An easier thing is to use Ensembl to do the plotting of the annotation. Often, however, you will want to augment the annotation produced by Ensembl.

```
> annoFactory <- Genominator:::makeAnnoFactory(annoData, featureColumnName = "feature",
+                                              groupColumnName = NULL, idColumnName = NULL,
+                                              dp = DisplayPars("gene" = "blue",
+                                              "intergenic" = "green"))
> rp <- Genominator:::makeRegionPlotter(list("track.1" = list(expData = eData, what = "counts",
+                                            dp = DisplayPars(lwd=.2, color = "grey")),
+                                            "track.2" = list(expData = eData, what = "counts",
+                                            fx = log2, DisplayPars(lwd=.3, color = "black"))),
+                                       annoFactory = annoFactory)
> rp(annoData[1,"chr"], annoData[1, "start"] - 100, annoData[1, "end"] + 100)
```



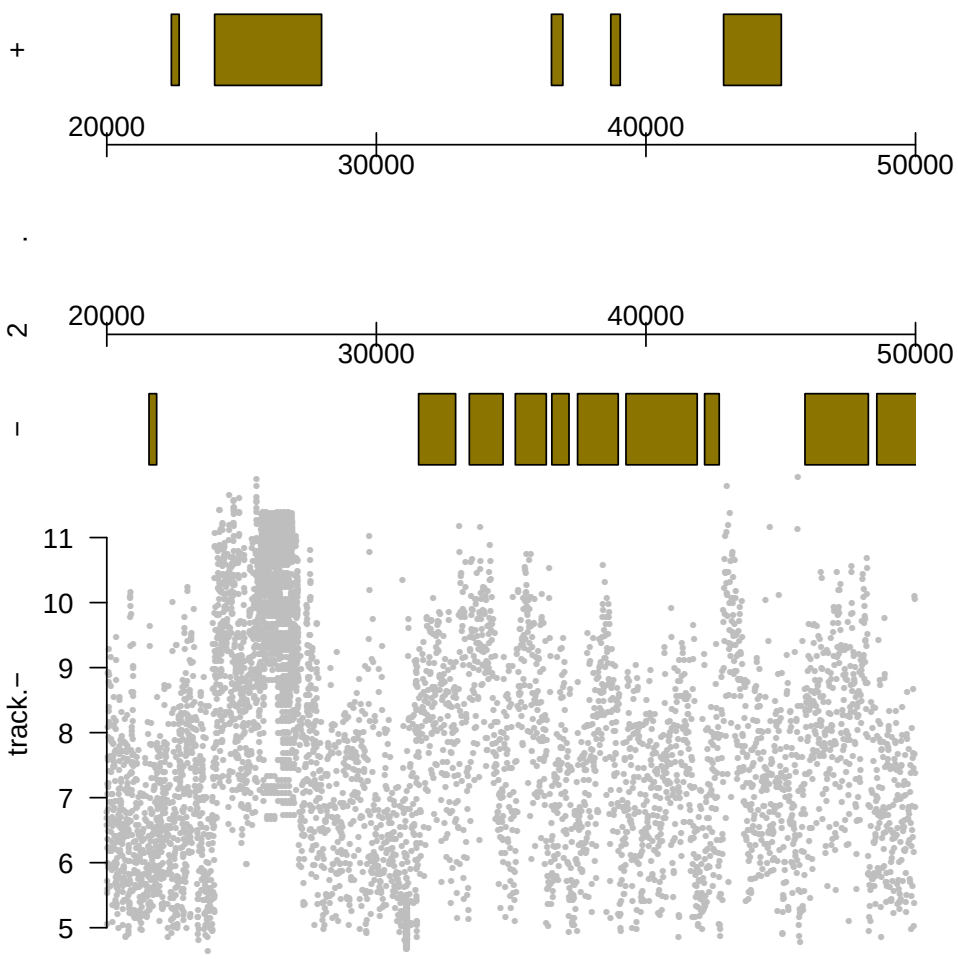
GenomeGraphs also offers a nice way to plot annotation for a given region using data from Ensembl or other sources of annotation - in some cases you have to do a little work because of the way that Biomart indexes the annotation and the way the *Genominator* package works (in this case yeast annotation is stored with Roman numerals denoting the chromosomes).

```
> require("biomaRt")
> mart <- useMart("ensembl", dataset = "scerevisiae_gene_ensembl")
> annoFactory <- Genominator:::makeAnnoFactory(mart, chrFunction = function(chr) as.roman(chr))
> load(system.file("data", "chr1_yeast.rda", package = "Genominator"))
> head(chr1_yeast)
```

```
  chr location strand   mRNA_1   mRNA_2
1   1        1     -1 9.038919 8.614710
2   1        1     -1 9.172428 8.558421
3   1        2     -1 9.422065 9.131857
4   1        2     -1 8.679480 8.442943
5   1        2     -1 8.546894 8.794416
6   1        2     -1 8.784635 8.918863
```

```
> yData <- importToExpData(chr1_yeast, dbFilename = "my.db", tablename = "yeast",
+                          overwrite = TRUE)
> rp <- Genominator:::makeRegionPlotter(list("track.-" = list(expData = yData, what = c("mRNA_1", "mRNA_
+                                            fx = rowMeans, strand = -1,
```

```
+                                                dp = DisplayPars(lwd=.3, color = "grey"))),
+                                          annoFactory = annoFactory)
> rp(1, 20000, 50000)
```



SessionInfo

- R version 3.5.1 Patched (2018-07-12 r74967), `x86_64-apple-darwin15.6.0`
- Locale: `C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Running under: `OS X El Capitan 10.11.6`
- Matrix products: default
- BLAS: `/Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib`
- LAPACK: `/Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib`
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, stats4, utils
- Other packages: BiocGenerics 0.28.0, DBI 1.0.0, GenomeGraphs 1.42.0, Genominator 1.36.0, IRanges 2.16.0, RSQLite 2.1.1, S4Vectors 0.20.0, biomaRt 2.38.0

- Loaded via a namespace (and not attached): AnnotationDbi 1.44.0, Biobase 2.42.0, R6 2.3.0, RCurl 1.95-4.11, Rcpp 0.12.19, XML 3.98-1.16, assertthat 0.2.0, bit 1.1-14, bit64 0.9-7, bitops 1.0-6, blob 1.1.1, compiler 3.5.1, crayon 1.3.4, curl 3.2, digest 0.6.18, hms 0.4.2, httr 1.3.1, magrittr 1.5, memoise 1.1.0, pkgconfig 2.0.2, prettyunits 1.0.2, progress 1.2.0, rlang 0.3.0.1, stringi 1.2.4, stringr 1.3.1, tools 3.5.1