GenomeGraphs: Integrated Genomic Visualization

James H. Bullard
bullard@stat.berkeley.edu
Graduate Group in Biostatistics
www.stat.berkeley.edu/~bullard

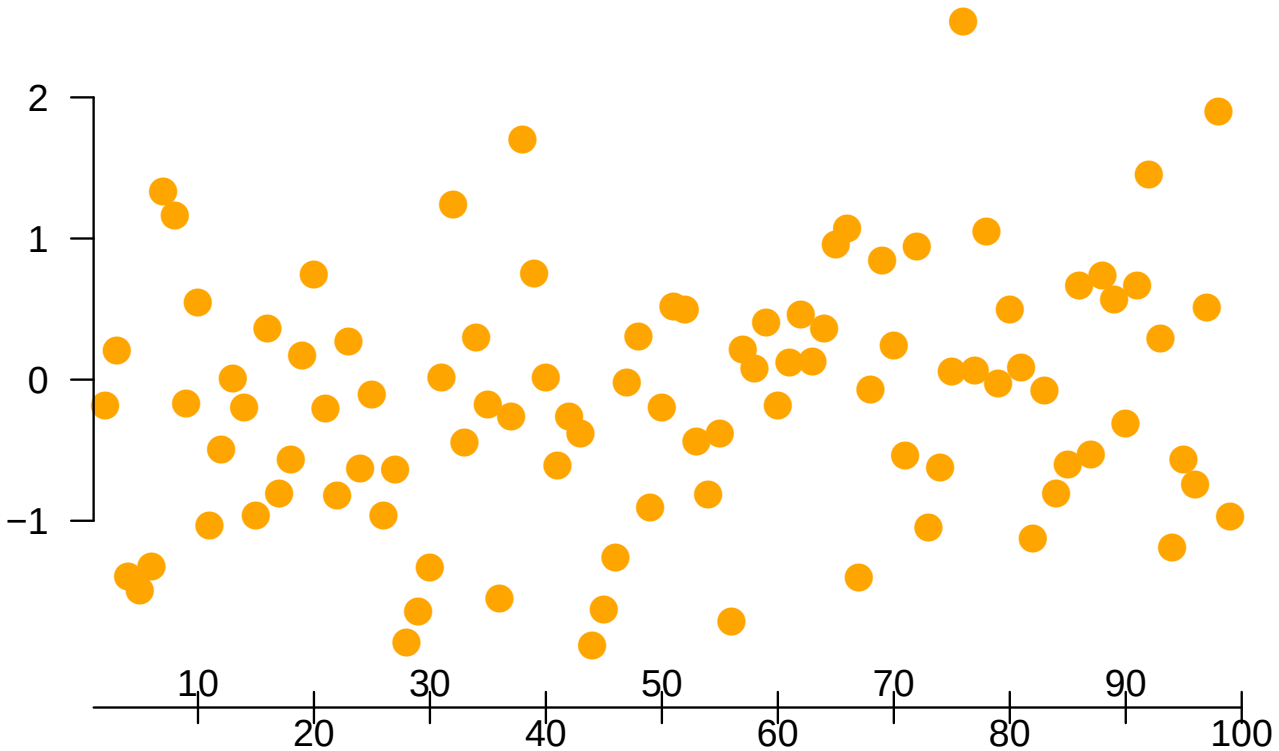
Bioconductor Conference 2009

- Goals:
  - High-throughput data sets
  - Changing annotation
  - Various data sources/formats
- Other Tools:
  - rtracklayer
  - Ensembl Genome Browser
  - NCBI Entrez Map Viewer
  - UCSC's Golden Path Genome Browser
  - Statistical Viewer, interpretation of linkage and association data by providing a plug-in for data uploaded to the Ensembl Genome Browser.
  - X:Map genome annotation database and exonmap for exon arrrays.

```
> require(GenomeGraphs)
> args(gdPlot)

function (gdObjects, minBase = NA, maxBase = NA, overlays =
    labelColor = "black", labelCex = 1, labelRot = 90)
NULL
```

- gdObject
  - Gene, GeneRegion, Transcript, TranscriptRegion, Ideogram, AnnotationTrack
  - Title, Legend, GenomeAxis
  - GenericArray, ExonArray, BaseTrack
- DisplayPars
- Overlay, RectangleOverlay, TextOverlay
- Segmentable

```
> lst <- list(makeBaseTrack(1:100,
+     rnorm(100)), makeGenomeAxis())
> gdPlot(lst)
```

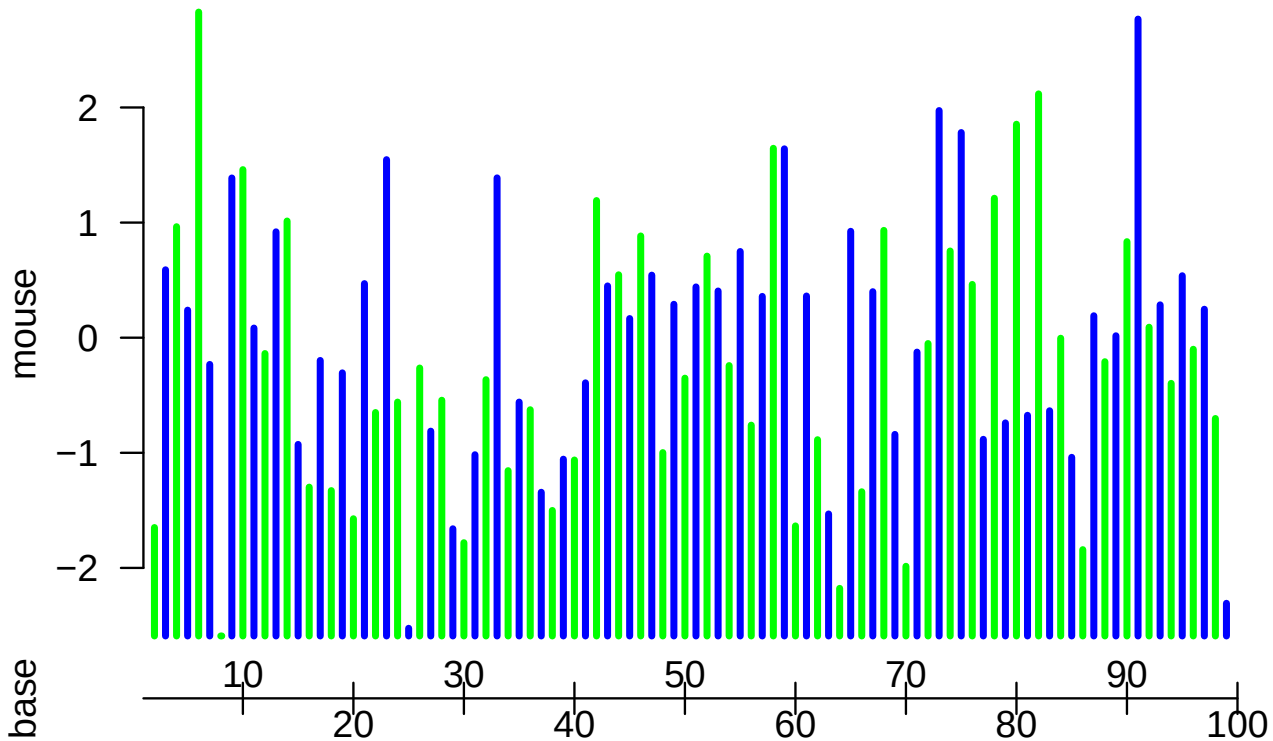


Exercise 0

- Annotate the y-axis (name the tracks).
- Change the color of the plotting symbols.
- Change the plot type (vertical bars).

Exercise 0: Answers

```
> gdPlot(list(mouse = makeBaseTrack(1:100,
+     rnorm(100), dp = DisplayPars(type = "h",
+         lwd = 3, color = c("blue",
+             "green"))), base = makeGenomeAxis()))
```



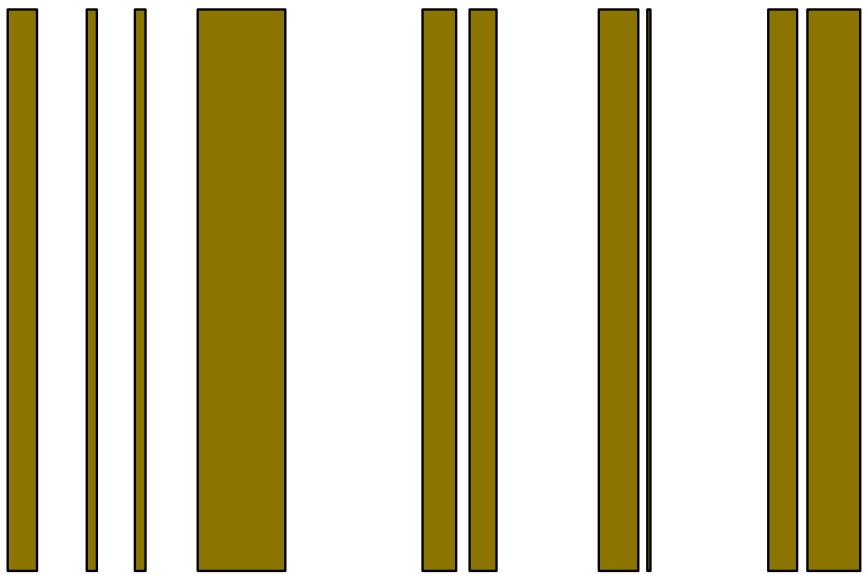
Querying Display Parameters

```
> showDisplayOptions("BaseTrack")

color  =  orange
lty  =  solid
lwd  =  1
size  =  5
type  =  p

> showDisplayOptions("GenericArray")

color  =  darkred
lty  =  solid
lwd  =  1
pch  =  16
pointSize  =  0.2
size  =  5
type  =  point
```

```
> mart <- useMart("ensembl", "scerevisiae_gene_ensembl")
> data("seqDataEx")
> head(seqDataEx$david)
```

```
  chr location strand  expr
1   4  1300003     -1 -0.20
2   4  1300007      1  0.61
3   4  1300011     -1  0.07
4   4  1300015      1  1.25
5   4  1300019     -1 -0.29
6   4  1300023      1  0.61
```

```
> gdPlot(makeGeneRegion(10000, 50000,
+     chr = "IV", strand = "+", biomart = mart),
+     10000, 50000)
```



Exercise 1: Tying Annotation with Data

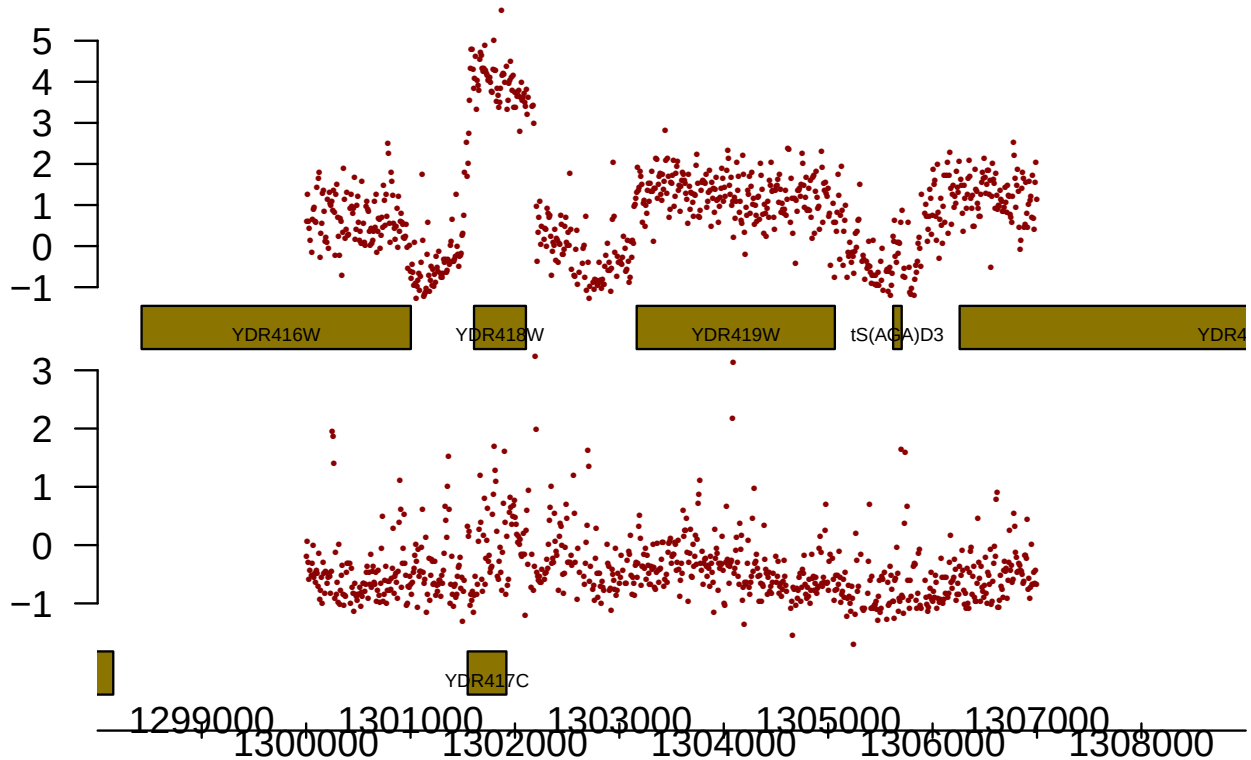
- Add the microarray data of David et al.
- Split things by strand, both annotation and array.
- Add a GenomicAxis.
- Add the gene names.

Exercise 1: Tying Annotation with Data

```
> array <- as.data.frame(seqDataEx$david)
> lst <- lapply(c("+", "-"), function(s) {
+     a <- as.matrix(subset(array,
+         strand == ifelse(s == "+",
+             1, -1)))
+     c(makeGenericArray(a[, "expr",
+         drop = FALSE], a[, "location"]),
+         makeGeneRegion(start = min(array[,
+             "location"]), end = max(array[,
+             "location"]), chr = "IV",
+             strand = s, biomart = mart,
+             dp = DisplayPars(plotId = TRUE,
+                 idRotation = 0,
+                 cex = 0.5, idColor = "black")))
+ })
```

Exercise 1: Tying Annotation with Data

```
> gdPlot(yeastLst <- c(unlist(lst),
+     makeGenomeAxis()))
```

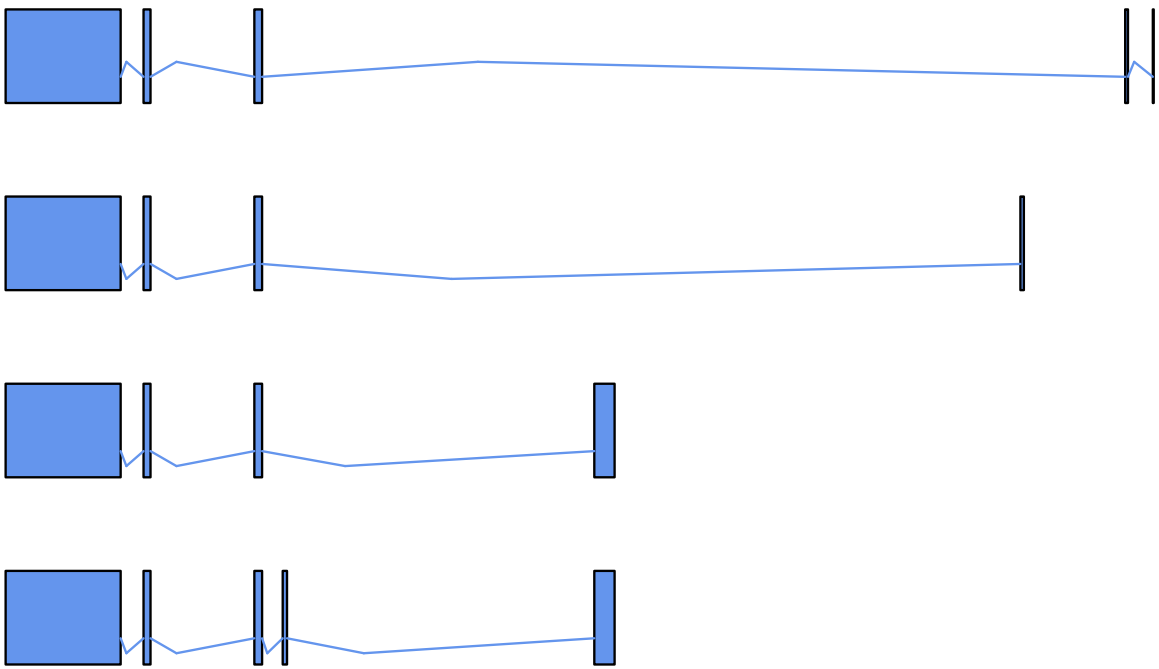


Exercise 2: Human Annotation

- Plot all the transcripts for gene: ENSG00000168309

Exercise 2: Human Annotation

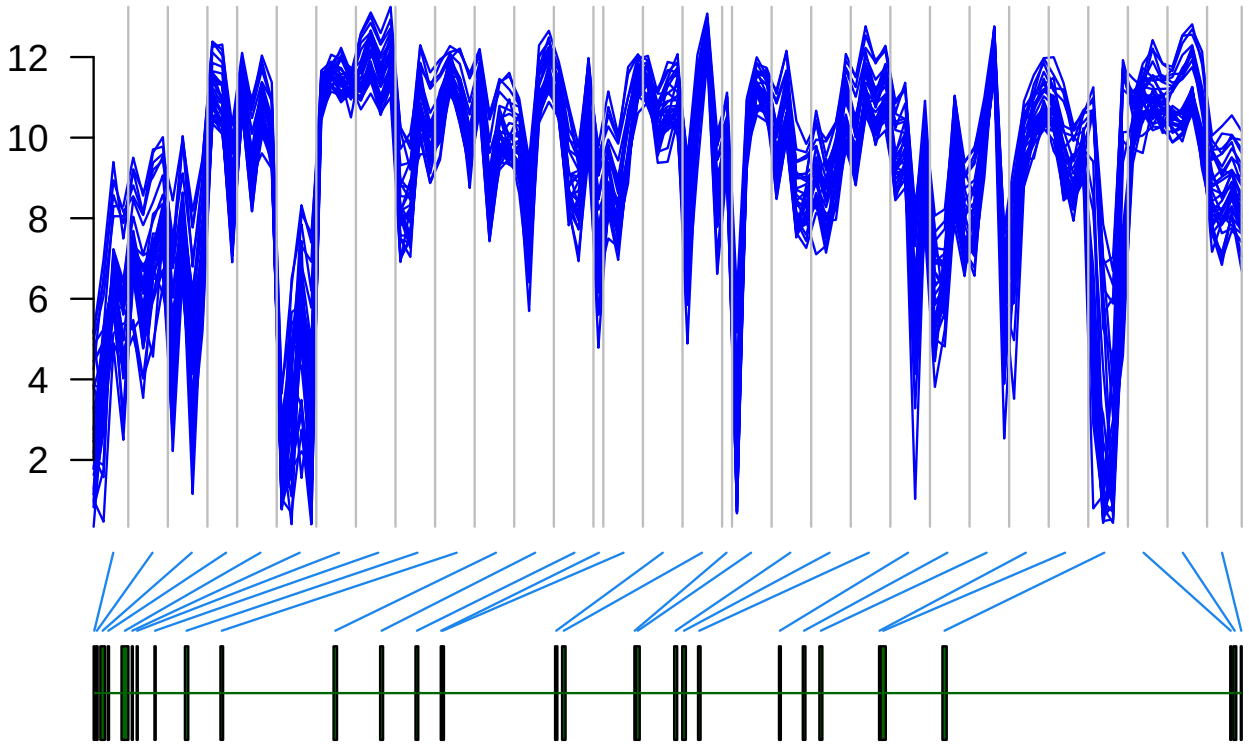
```
> hMart <- useMart("ensembl", "hsapiens_gene_ensembl")
> gdPlot(makeTranscript("ENSG00000168309",
+     biomart = hMart))
```



```
> data("unrData", package = "GenomeGraphs")
> class(unrData)

[1] "matrix"

> head(unrPositions, 2)

  probesetId chromosome     start
1    2429278          1 115061081
2    2429279          1 115061152
       stop
1 115061119
2 115061198
```

```
> exon <- makeExonArray(intensity = unrData,
+     probeStart = unrPositions[,
+         3], probeEnd = unrPositions[,
+         4], probeId = as.character(unrPositions[,
+         1]), nProbes = unrNProbes,
+     dp = DisplayPars(color = "blue",
+         mapColor = "dodgerblue2"),
+     displayProbesets = FALSE)
> geneModel <- makeGeneModel(start = unrPositions[,
+     3], end = unrPositions[, 4])
> gdPlot(list(exon, geneModel))
```

Exon Arrays

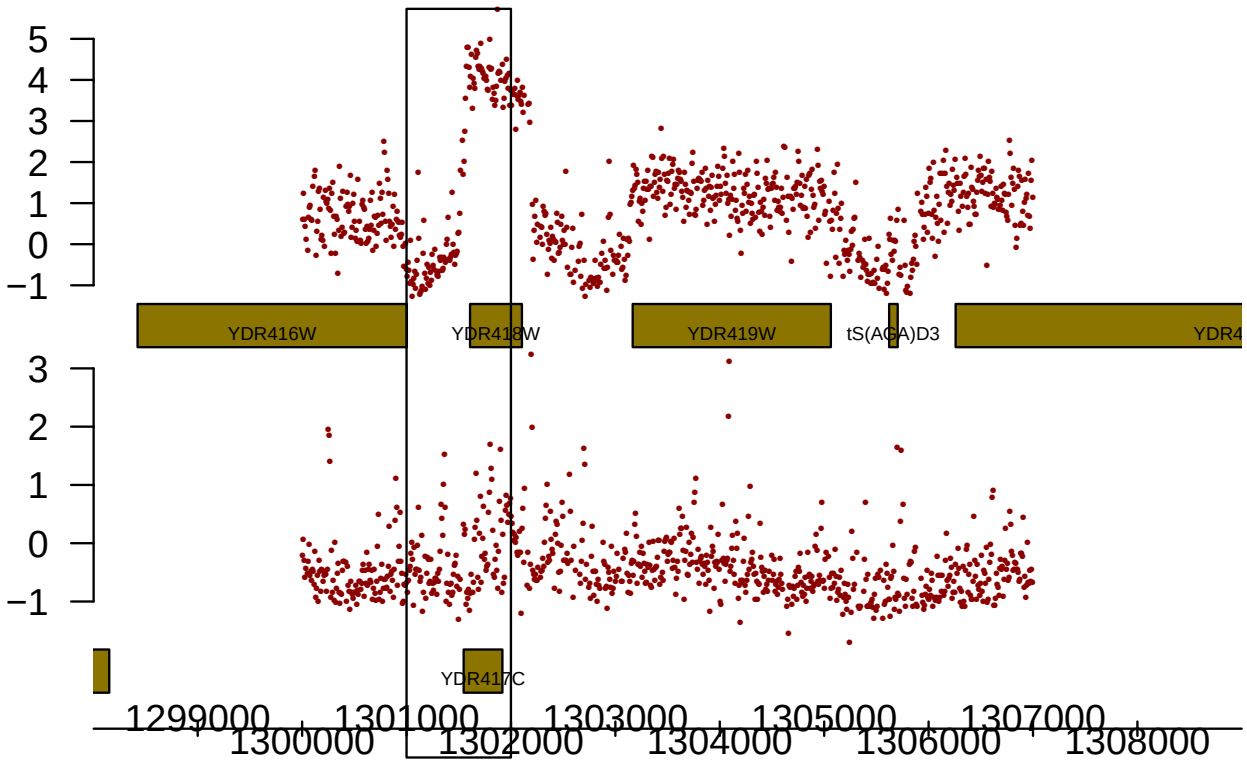


Overlays

Overlays provide a way to plot across tracks, either using genomic coordinates or absolute coordinates. `Overlay` is an abstract class, you cannot instanteate it, only its subclasses: `RectangleOverlay`, `TextOverlay`

```
> ovlay <- makeRectangleOverlay(1301000,
+     1302000)
> gdPlot(yeastLst, overlays = ovlay)
```

Overlays


5
4
3
2
1
0
−1
YDR416W
YDR418W
YDR419W
tS(AGA)D3
YDR4
3
2
1
0
−1
YDR417C
1299000
1300000
1301000
1302000
1303000
1304000
1305000
1306000
1307000
1308000

Exercise 3: Overlays

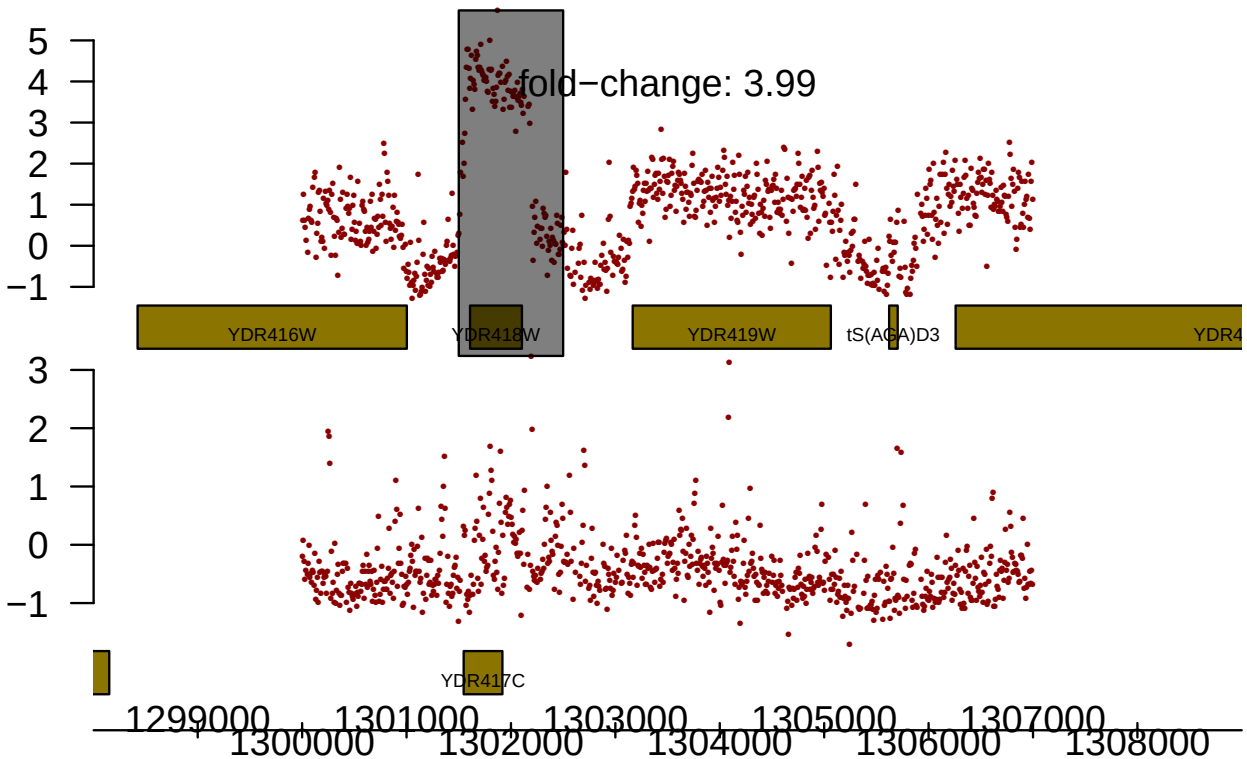
- Fix the overlay to only cover the "+" strand tracks (genes and data).
- Make the overlay semi-transparent.
- Add a text overlay adjacent to the upregulated region with the average expression of the probes in the gene YDR418W.

Exercise 3: Overlays

```
> array <- array[array$strand ==
+     1, ]
> anno <- yeastLst[[2]]@ens
> anno <- anno[anno$ensembl_gene_id ==
+     "YDR418W", c("exon_chrom_start",
+     "exon_chrom_end")]
> foldChange <- paste("fold-change:",
+     round(mean(split(array$expr,
+         findInterval(array$location,
+             as.numeric(anno)))[[2]]),
+         2))
> ovlay <- makeRectangleOverlay(1301500,
+     1302500, region = c(1, 2),
+     dp = DisplayPars(alpha = 0.5))
> tovlay <- makeTextOverlay(foldChange,
```

Exercise 3: Overlays

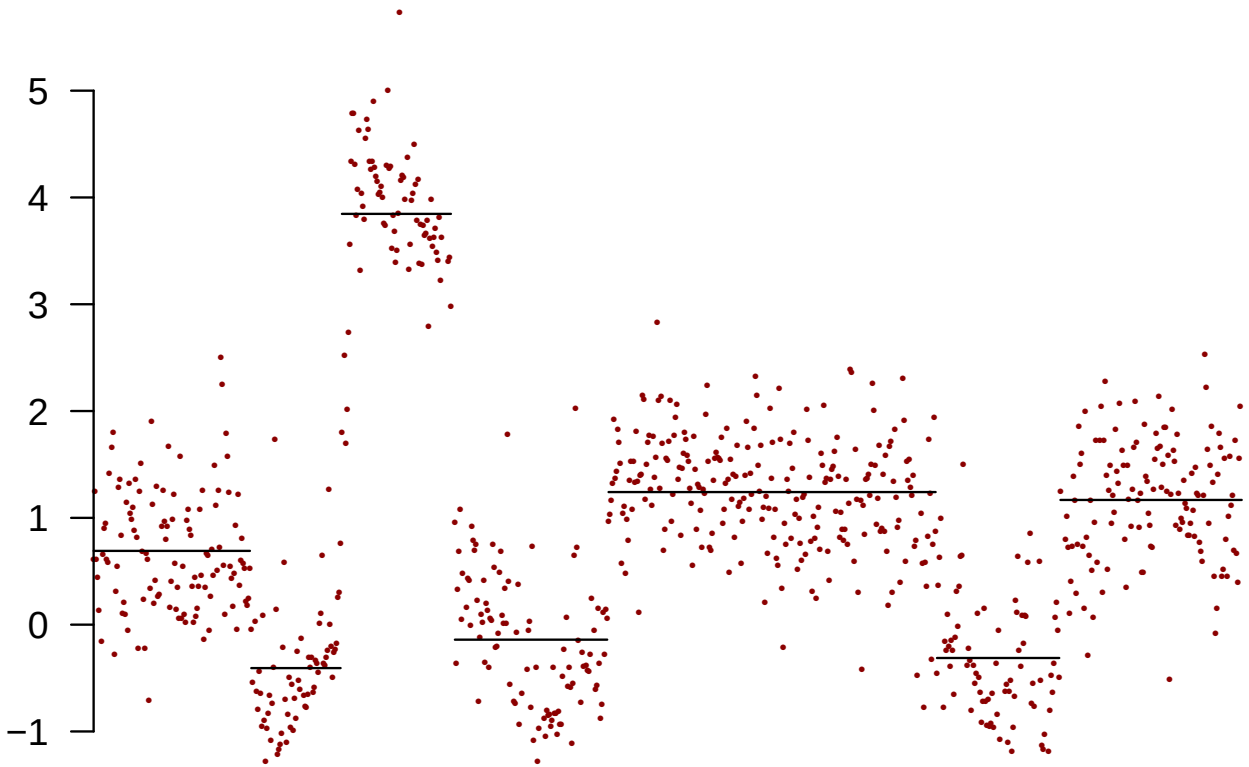
```
+      1303500, 0.75, region = c(1,
+          1), dp = DisplayPars(color = "black"))
> gdPlot(yeastLst, overlays = c(ovlay,
+      tovlay))
```



Data classes implement the interface `Segmentable` which allows one to add a piecewise constant function as an overlay. In the future, this will be expanded to more general functions.

```
> require(tilingArray)
> Y <- yeastLst[[1]]
> s <- segment(Y@intensity, maxk = 500,
+     maxseg = 20)
> bins <- findInterval(1:length(Y@intensity),
+     s@breakpoints[[7]])
> means <- tapply(Y@intensity, bins,
+     mean)
> ranges <- do.call(rbind, tapply(Y@probeStart,
+     bins, range))
> Y@segmentation <- makeSegmentation(ranges[,
```

```
+      1], ranges[, 2], means)
> gdPlot(Y)
```

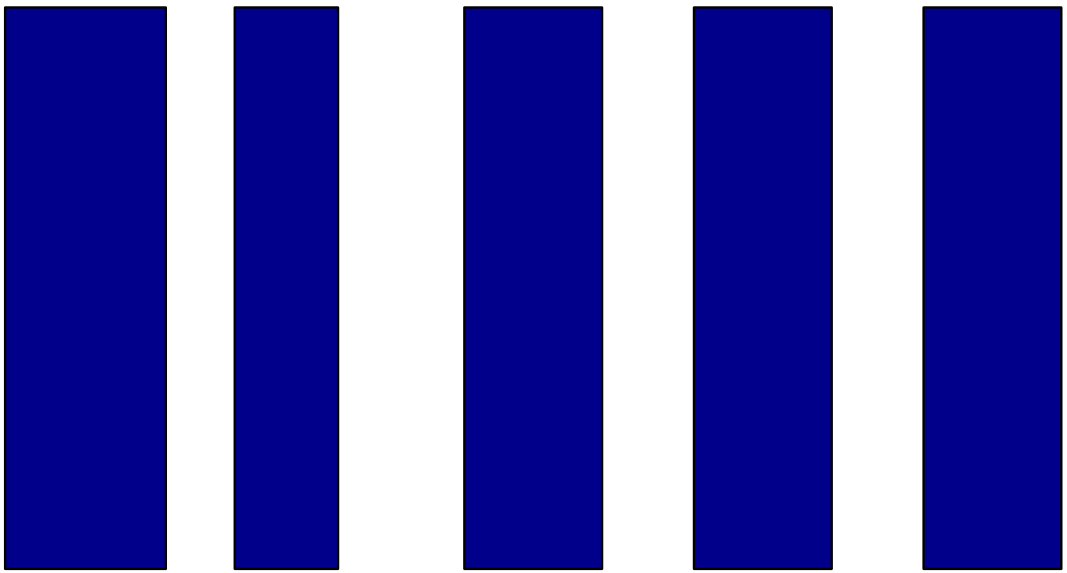


Custom Annotation

We can construct custom annotation by using the `makeAnnotationTrack` function.

```
> S <- seq(1, 100, by = 20)
> E <- S + rpois(length(S), 10)
> anno <- makeAnnotationTrack(start = S,
+     end = E, feature = "gene",
+     dp = DisplayPars(gene = "darkblue"))
> gdPlot(anno, minBase = 0, maxBase = 100)
```

Custom Annotation



Exercise 4: Custom Annotation

Download the annotation file:

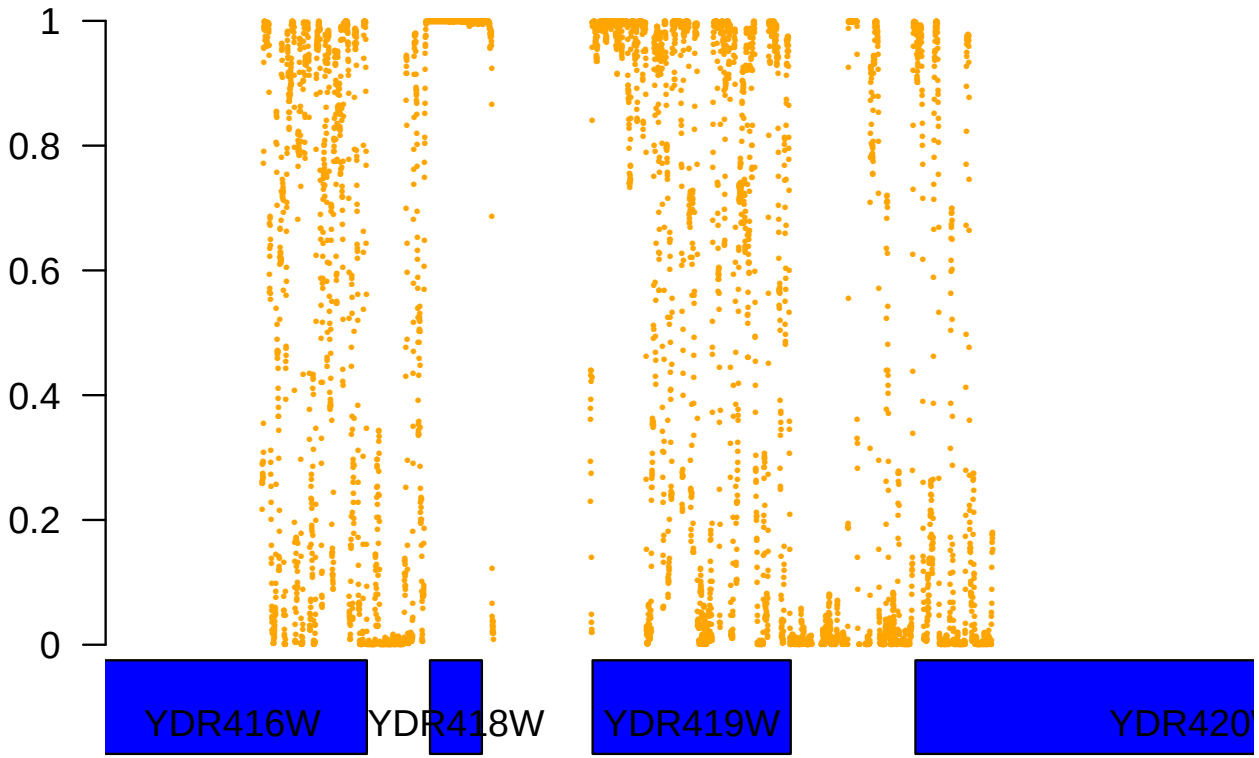
`wiki.biostat.berkeley.edu/~bullard/aleatoria/yeast_anno_4.c`

Construct a annotation track using data from this file and plot along with conservation data from the example dataset `seqDataEx`. Add gene names and determine what the group column does.

Exercise 4: Custom Annotation

```
> anno <- read.csv("http://wiki.biostat.berkeley.edu/~bulla
> data("seqDataEx")
> aTrack <- makeAnnotationTrack(start = anno$start[anno$fea
+     "CDS"], end = anno$end[anno$feature ==
+     "CDS"], feature = "CDS", ID = anno$name[anno$feature
+     "CDS"], dp = DisplayPars(CDS = "blue",
+     plotId = TRUE, idColor = "black",
+     idRotation = 0))
> bTrack <- makeBaseTrack(seqDataEx$conservation[,
+     "location"], seqDataEx$conservation[,
+     "score"], dp = DisplayPars(type = "p",
+     lwd = 0.2))
> gdPlot(list(bTrack, aTrack), minBase = 1300000 -
+     1500, maxBase = 1307000 + 2500)
```

Exercise 4: Custom Annotation



Functionality to Add

- pch option to BaseTrack
- image option for generic array
- Add a suitable minBase and maxBase determiner for Annotation objects