# Работа с геномными интервалами Визуализация

Анастасия Жарикова

azharikova89@gmail.com

14/15 декабря 2020

# Разбиение данных по бинам

```
x = rnorm(1000)

breaks = c(-3,-2,-1,0,1,2,3)

f = cut(x, breaks)

summary(f)
```

```
## (-3,-2] (-2,-1]  (-1,0]   (0,1]   (1,2]   (2,3]    NA's
##      16     128     337     347     143      25       4
```

# Разбиение данных по бинам

фактор

```
x = rnorm(1000)

breaks = c(-3,-2,-1,0,1,2,3)

f = cut(x, breaks)

summary(f)
```

```
## (-3,-2] (-2,-1]  (-1,0]   (0,1]   (1,2]   (2,3]    NA's
##      16     128     337     347     143      25       4
```

# Разбиение данных по бинам

```
f = cut(x, breaks, labels = c('A','B','C','D','E','F'))

summary(f)
```

```
##    A    B    C    D    E    F NA's
##   30  136  345  330  144   14    1
```

# Поиск в данных

```
head(genes)
```

```
##    chr start   end strand                            genetype   gene_name
## 1 chr1 11869 14409      + transcribed_unprocessed_pseudogene     DDX11L1
## 2 chr1 14404 29570      -              unprocessed_pseudogene      WASH7P
## 3 chr1 17369 17436      -                             miRNA   MIR6859-1
## 4 chr1 29554 31109      +                           lncRNA MIR1302-2HG
## 5 chr1 30366 30503      +                             miRNA   MIR1302-2
## 6 chr1 34554 36081      -                           lncRNA     FAM138A
```

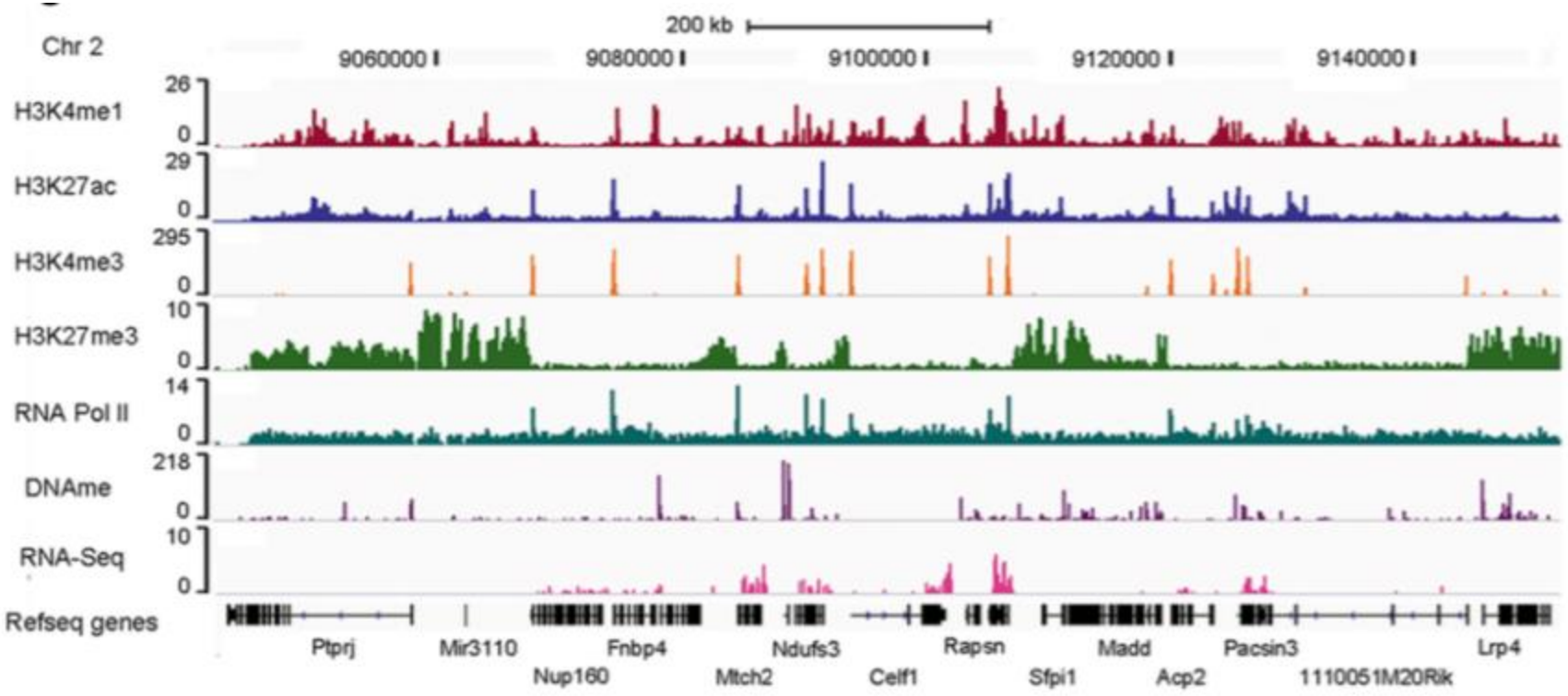
# Поиск в данных

```
grep('APOB',genes$gene_name)
```

```
##  [1]  4158 11412 21081 30303 37547 37548 37549 37550 37552 37553 37555 37557
## [13] 43461 47915
```

```
genes[grep('APOB',genes$gene_name),]
```

```
##         chr     start       end strand            genetype    gene_name
## 4158   chr1 183646275 183653316      -       protein_coding      APOBEC4
## 11412 chr12   7649400   7665908      -       protein_coding      APOBEC1
## 21081 chr16  28494643  28498970      +       protein_coding        APOBR
## 30303  chr2  21001429  21044073      -       protein_coding         APOB
## 37547 chr22  38952741  38992778      +       protein_coding     APOBEC3A
## 37548 chr22  38982347  38992804      +       protein_coding     APOBEC3B
## 37549 chr22  38991559  38998209      -               lncRNA APOBEC3B-AS1
## 37550 chr22  39014257  39020352      +       protein_coding     APOBEC3C
## 37552 chr22  39021113  39033277      +       protein_coding     APOBEC3D
## 37553 chr22  39040604  39055972      +       protein_coding     APOBEC3F
## 37555 chr22  39077067  39087743      +       protein_coding     APOBEC3G
## 37557 chr22  39097224  39104067      +       protein_coding     APOBEC3H
## 43461  chr4 170099818 170100104      + processed_pseudogene   APOBEC3AP1
## 47915  chr6  41053304  41064511      +       protein_coding      APOBEC2
```




https://doi.org/10.1186/s13059-016-1023-z

# karyoploteR

https://bernatgel.github.io/karyoploter_tutorial/

https://www.bioconductor.org/packages/devel/bioc/vignettes/karyoploteR/inst/doc/karyoploteR.html

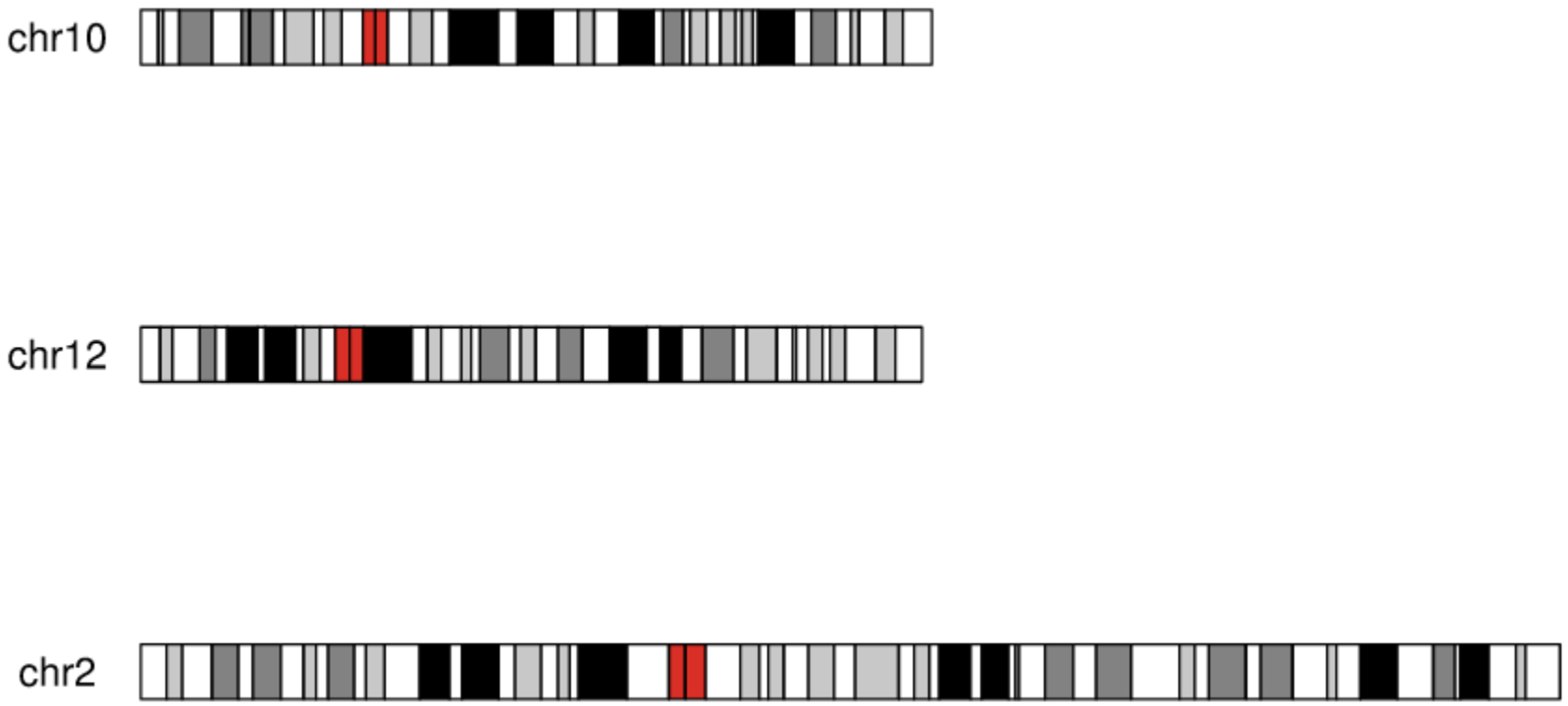
```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("karyoploteR")
```

```
library(karyoploteR)
```

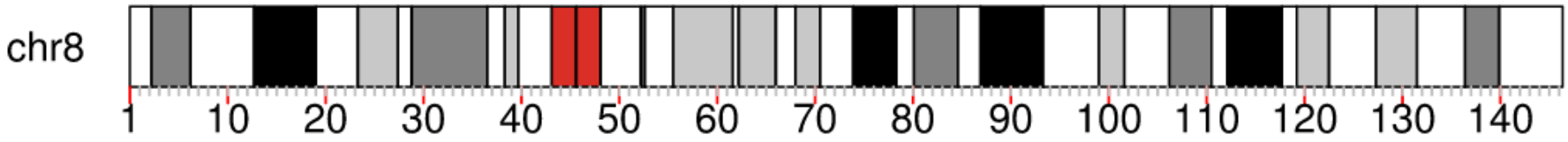
```
kp <- plotKaryotype()
```


По умолчанию – hg19, есть другие геномы, можно подгрузить «свой» геном

```
kp <- plotKaryotype(genome = "hg19", chromosomes=c("chr10", "chr12", "chr2"))
```

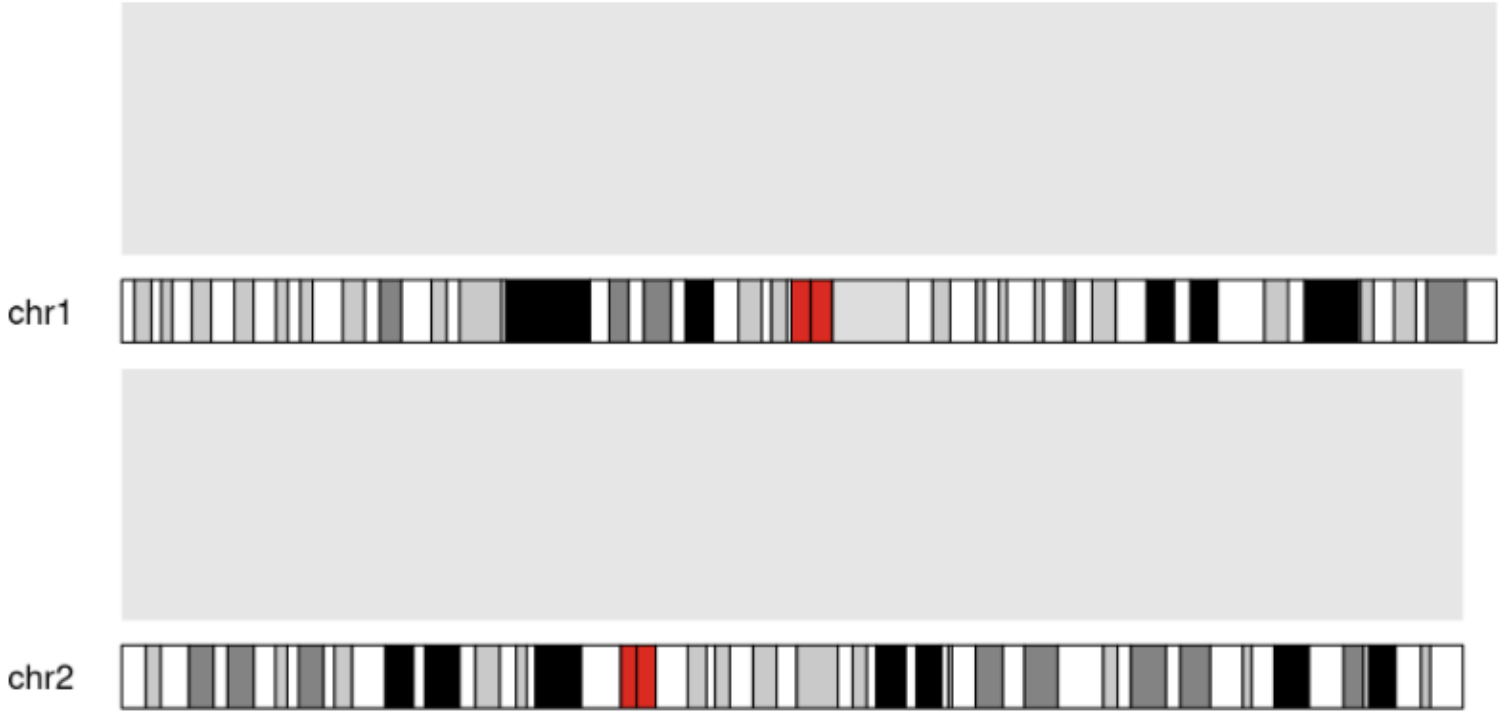


```
kp <- plotKaryotype(chromosomes="chr8", plot.type = 2)

kpAddBaseNumbers(kp, tick.dist = 10000000, tick.len = 10, tick.col="red", cex=1,
                 minor.tick.dist = 1000000, minor.tick.len = 5, minor.tick.col = "gray")
```



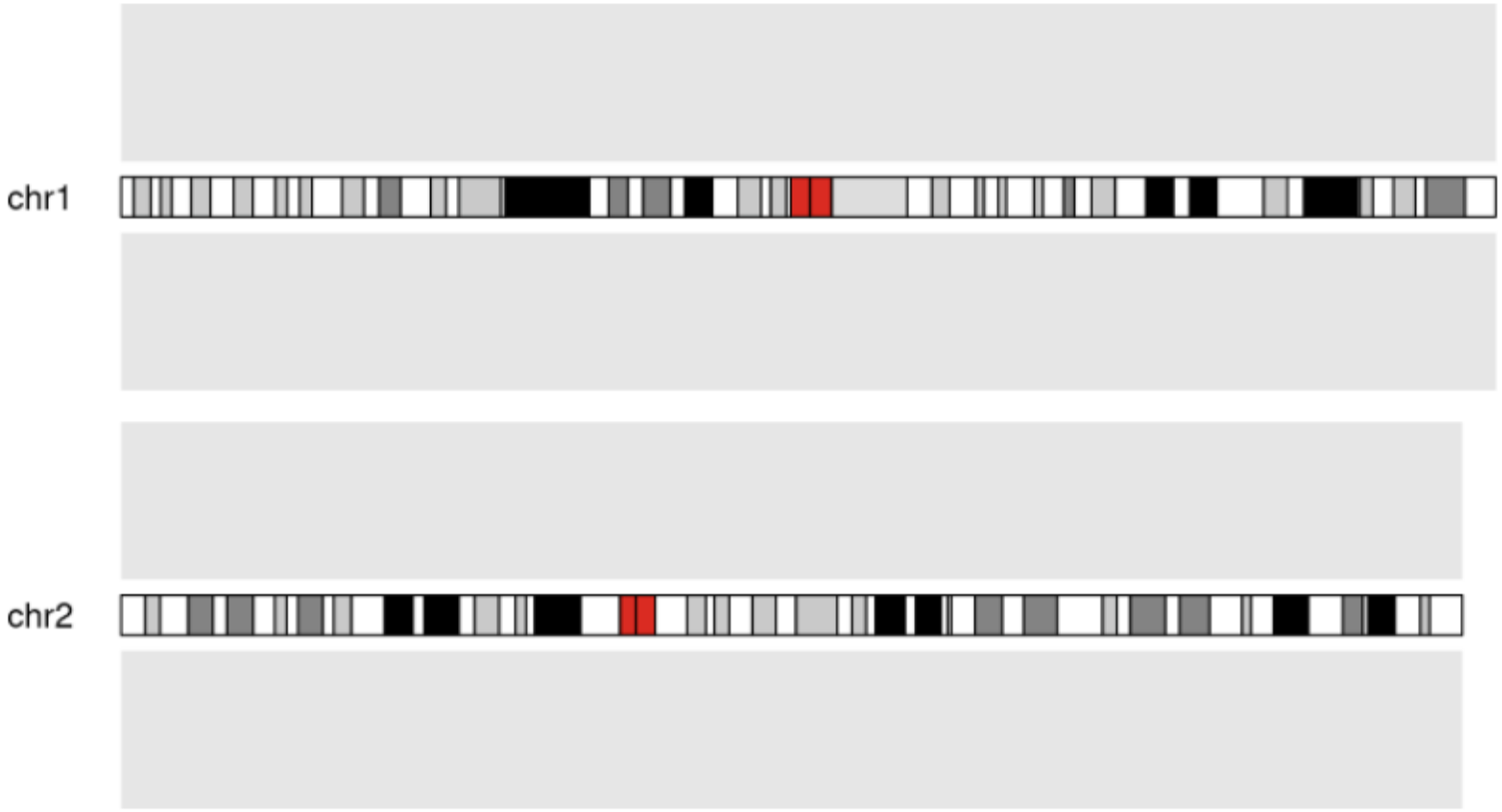
# plot.type: 1-7

```
kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 1)
kpDataBackground(kp, data.panel = 1)
```



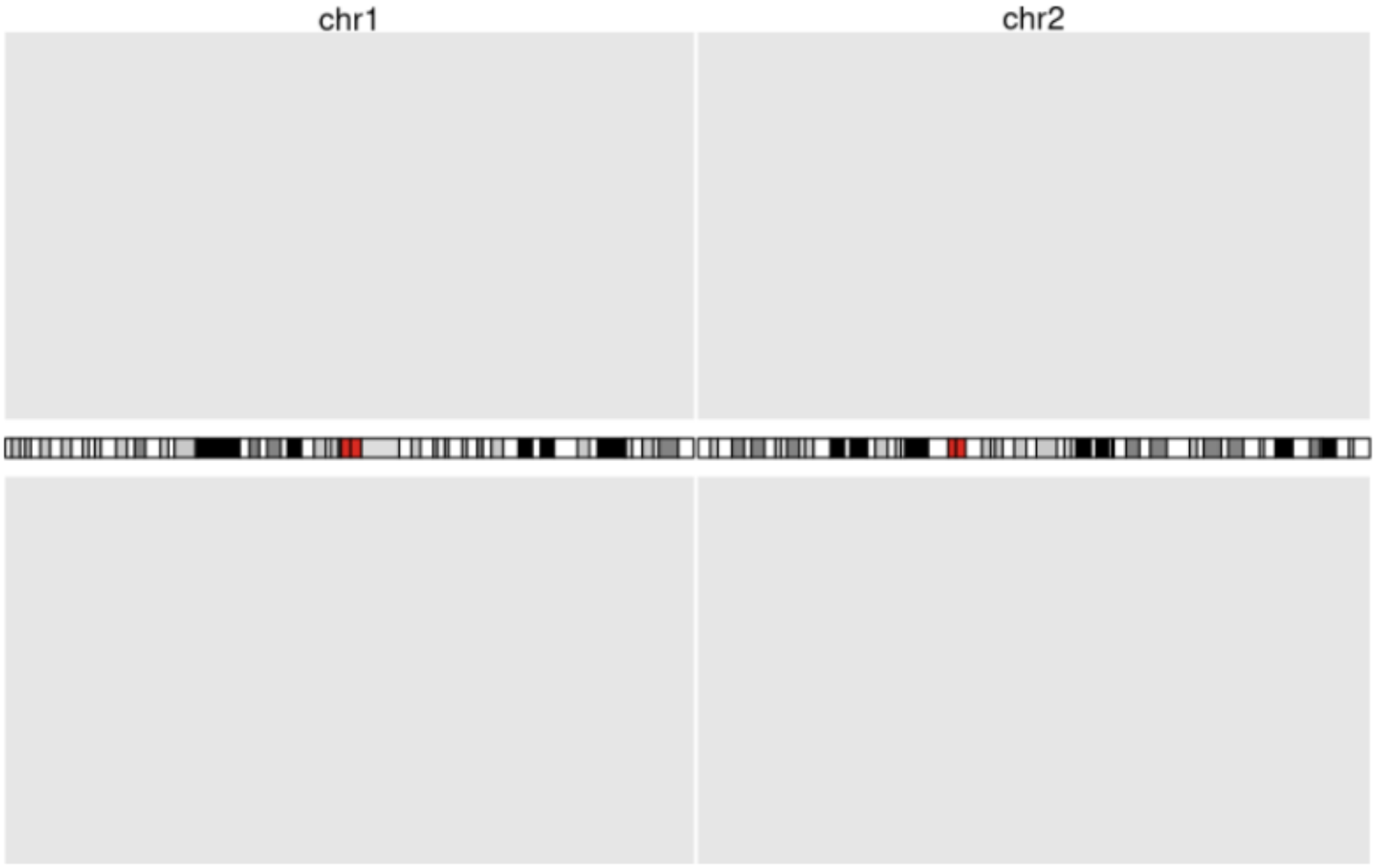
# plot.type: 1-7

```
kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 2)
kpDataBackground(kp, data.panel = 1)
kpDataBackground(kp, data.panel = 2)
```



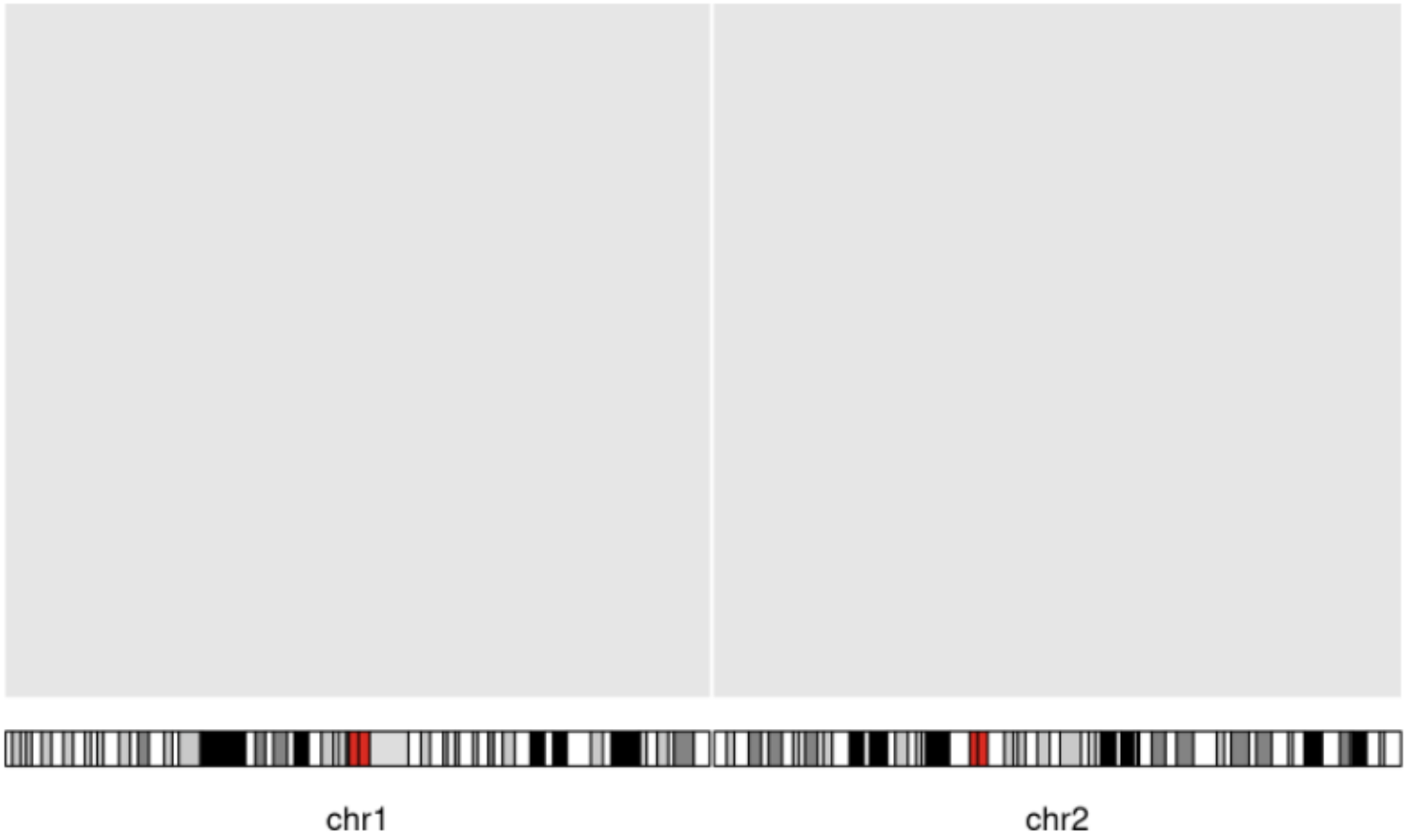
# plot.type: 1-7

```
kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 3)
kpDataBackground(kp, data.panel = 1)
kpDataBackground(kp, data.panel = 2)
```



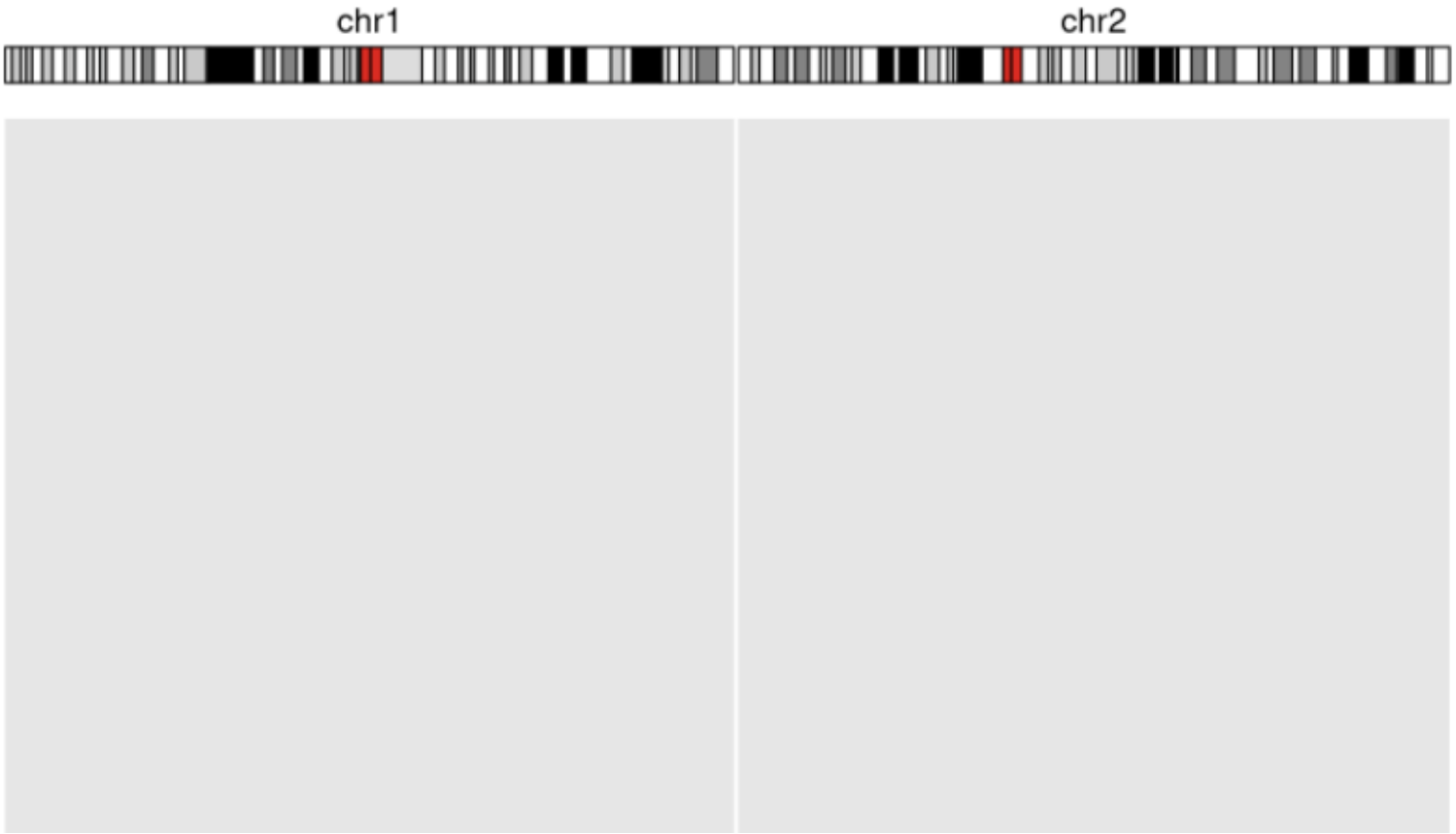
# plot.type: 1-7

```
kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 4)
kpDataBackground(kp, data.panel = 1)
```



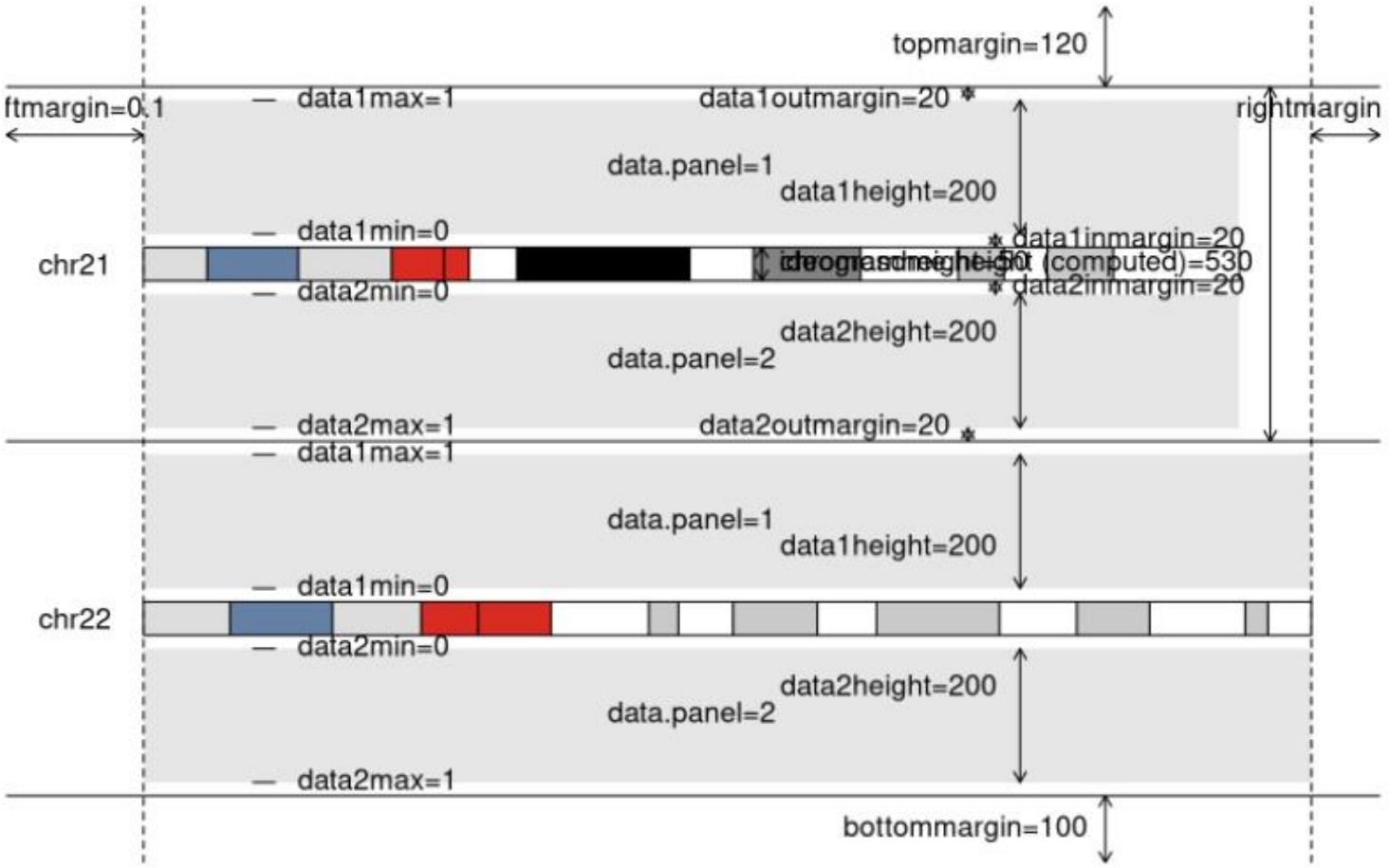
# plot.type: 1-7

```
kp <- plotKaryotype(chromosomes = c("chr1", "chr2"), plot.type = 5)
kpDataBackground(kp, data.panel = 1)
```

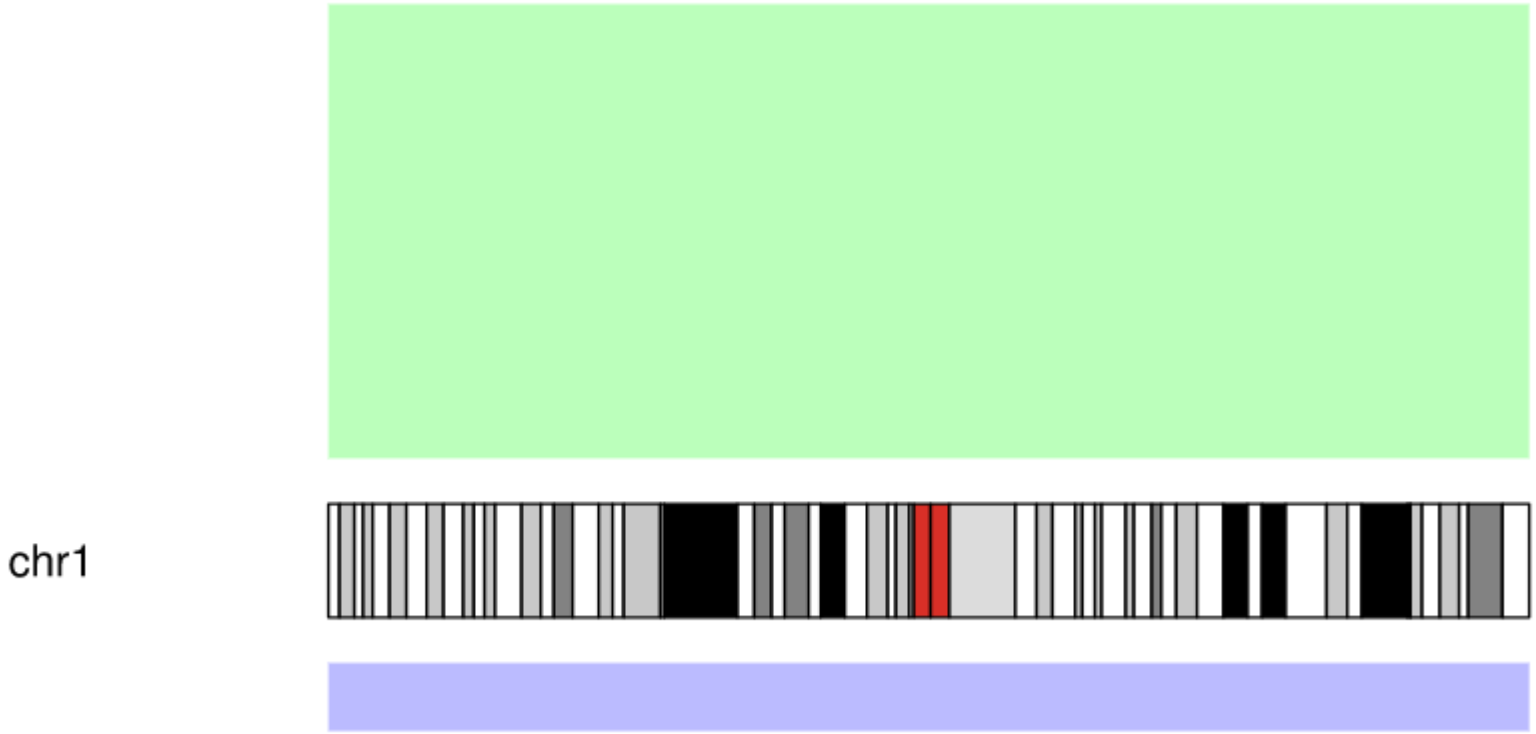


# Настройки параметров графика

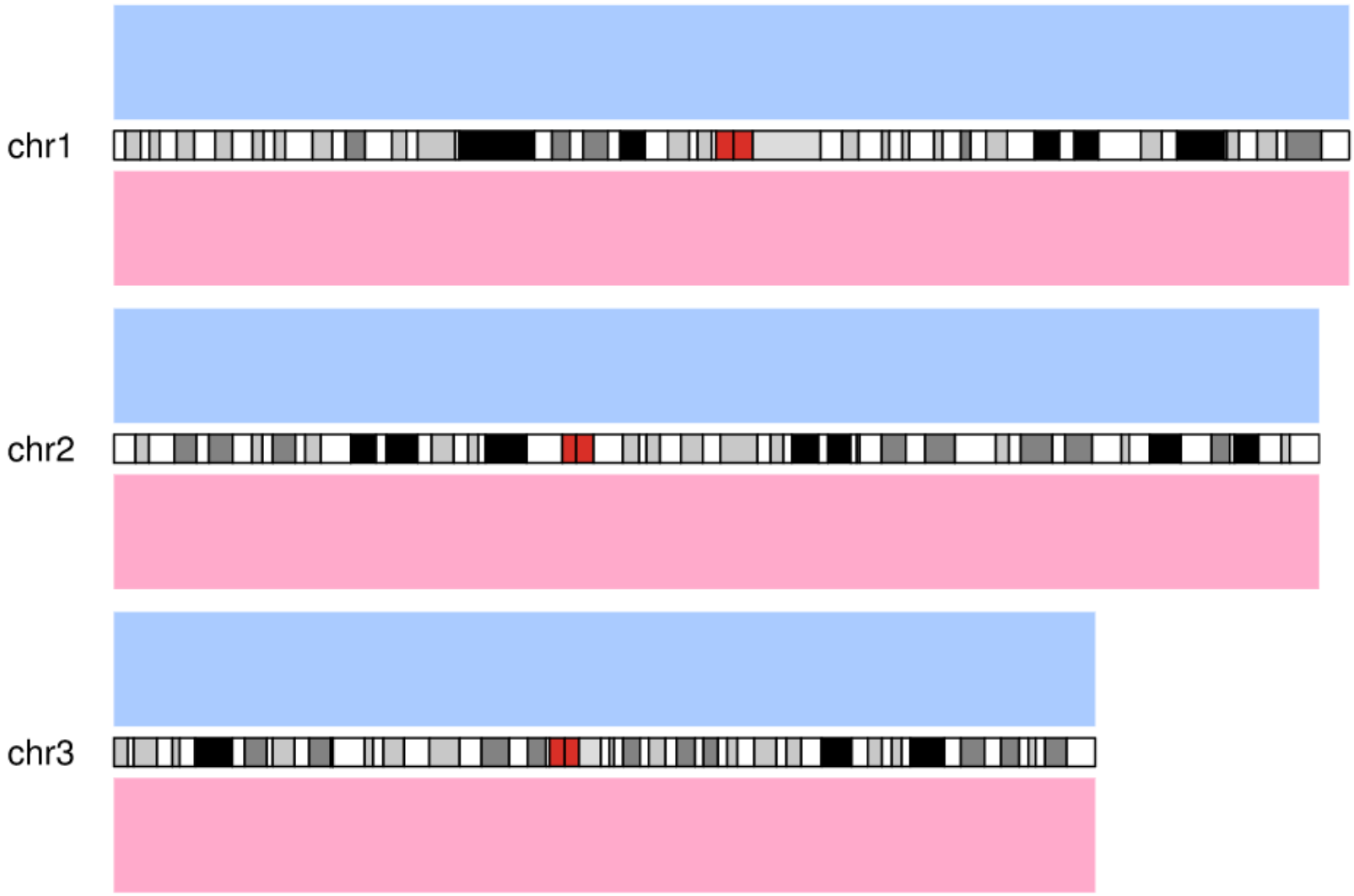
```
plotDefaultPlotParams(plot.type=2)
```



```
pp <- getDefaultPlotParams(plot.type=2)
pp$leftmargin <- 0.3
pp$data2height <- 30
kp <- plotKaryotype(chromosomes=c("chr1"), plot.type=2, plot.params = pp)
kpDataBackground(kp, data.panel = 1, color = "#BBFFBB")
kpDataBackground(kp, data.panel = 2, color = "#BBBBFF")
```



```
kp <- plotKaryotype(plot.type=2, chromosomes = c("chr1", "chr2", "chr3"))
kpDataBackground(kp, data.panel = 1, col="#AACBFF")
kpDataBackground(kp, data.panel = 2, col="#FFAACB")
```

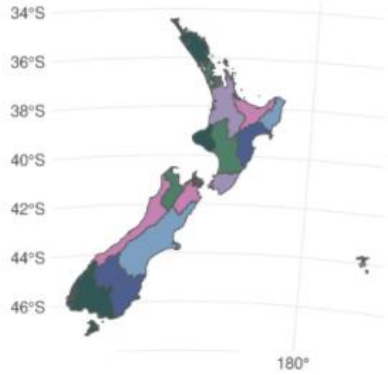


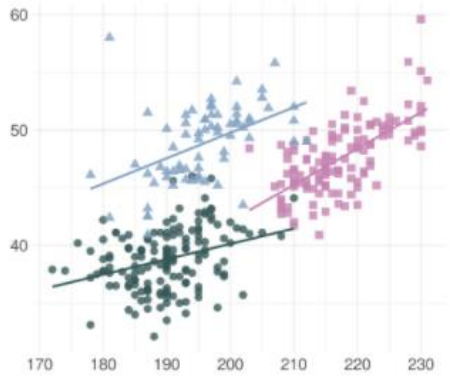
# Manu

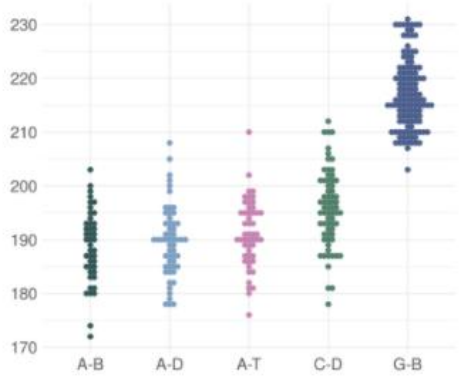
https://g-thomson.github.io/Manu/index.html

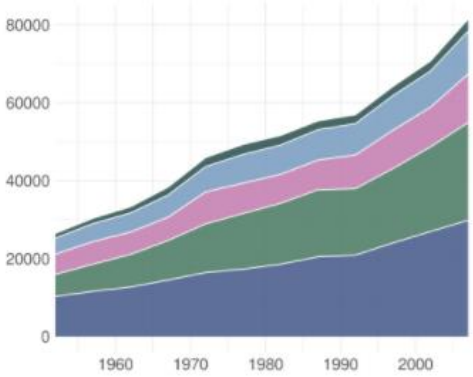
## Kererū - *Hemiphaga novaeseelandiae* - NZ wood pigeon

```
c("#325756", "#7d9fc2", "#C582B2", "#51806a", "#4d5f8e", "#A092B7")
```

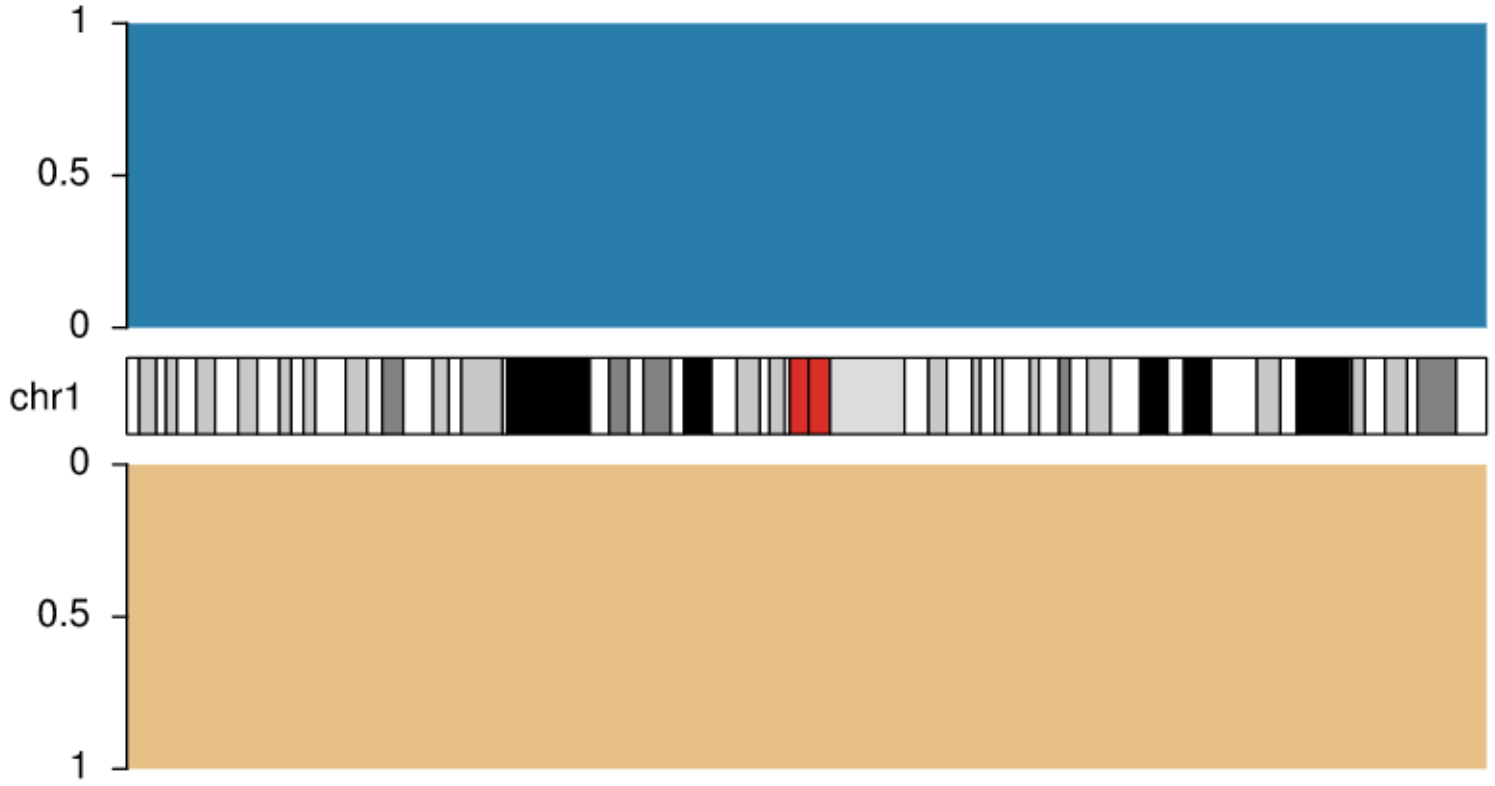


```
kp <- plotKaryotype(plot.type=2, chromosomes = "chr1")
kpDataBackground(kp, data.panel = 1, col="#287DAB")
kpDataBackground(kp, data.panel = 2, col="#E5BF86")
kpAxis(kp, data.panel=1)
kpAxis(kp, data.panel=2)
```



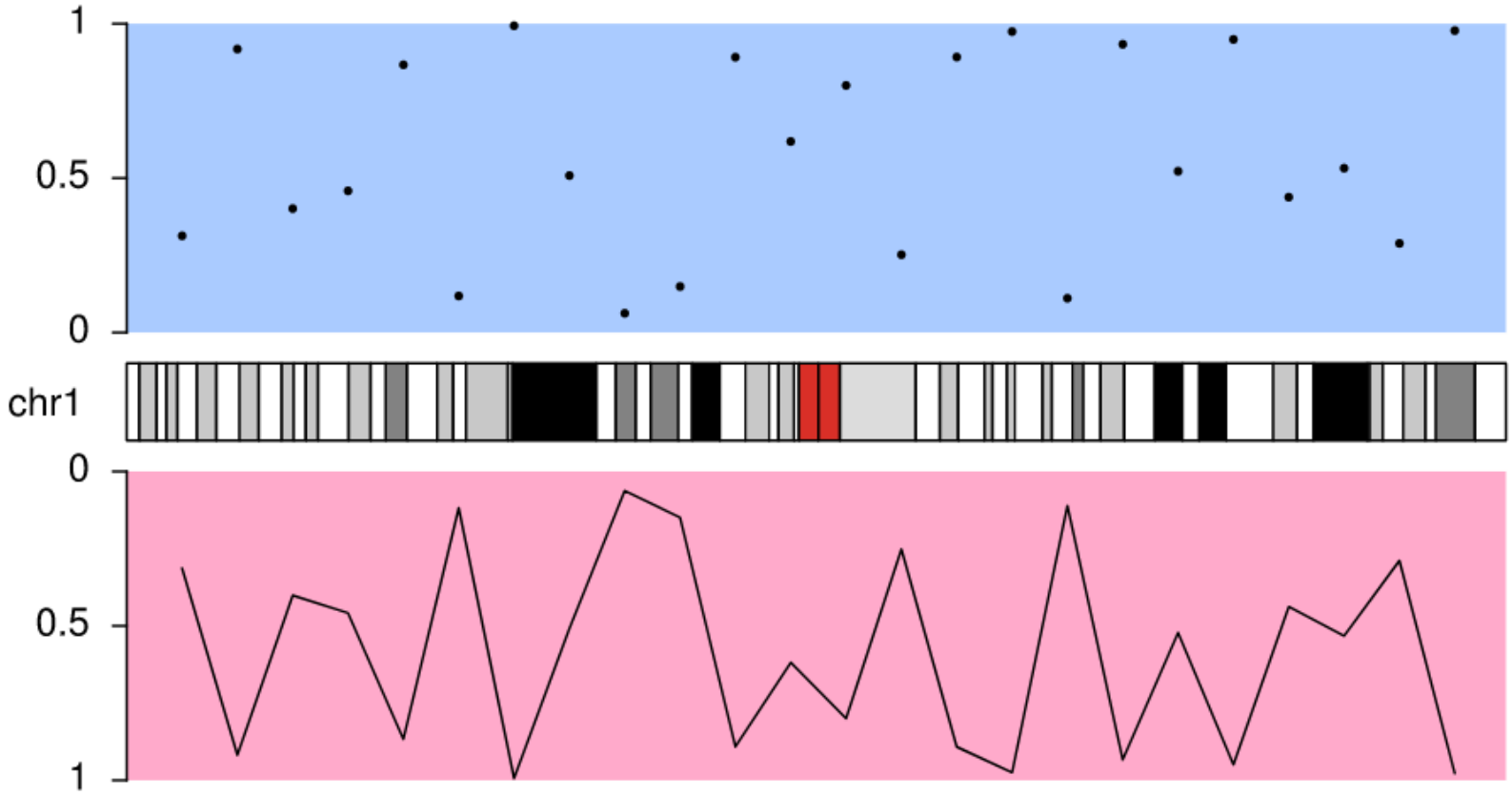
```r
x <- 1:24*10e6 #one data point every 10 milion bases (10e6)
y <- runif(n = 24, min = 0, max = 1) #random y values
x
```

```
##  [1] 1.0e+07 2.0e+07 3.0e+07 4.0e+07 5.0e+07 6.0e+07 7.0e+07 8.0e+07 9.0e+07
## [10] 1.0e+08 1.1e+08 1.2e+08 1.3e+08 1.4e+08 1.5e+08 1.6e+08 1.7e+08 1.8e+08
## [19] 1.9e+08 2.0e+08 2.1e+08 2.2e+08 2.3e+08 2.4e+08
```

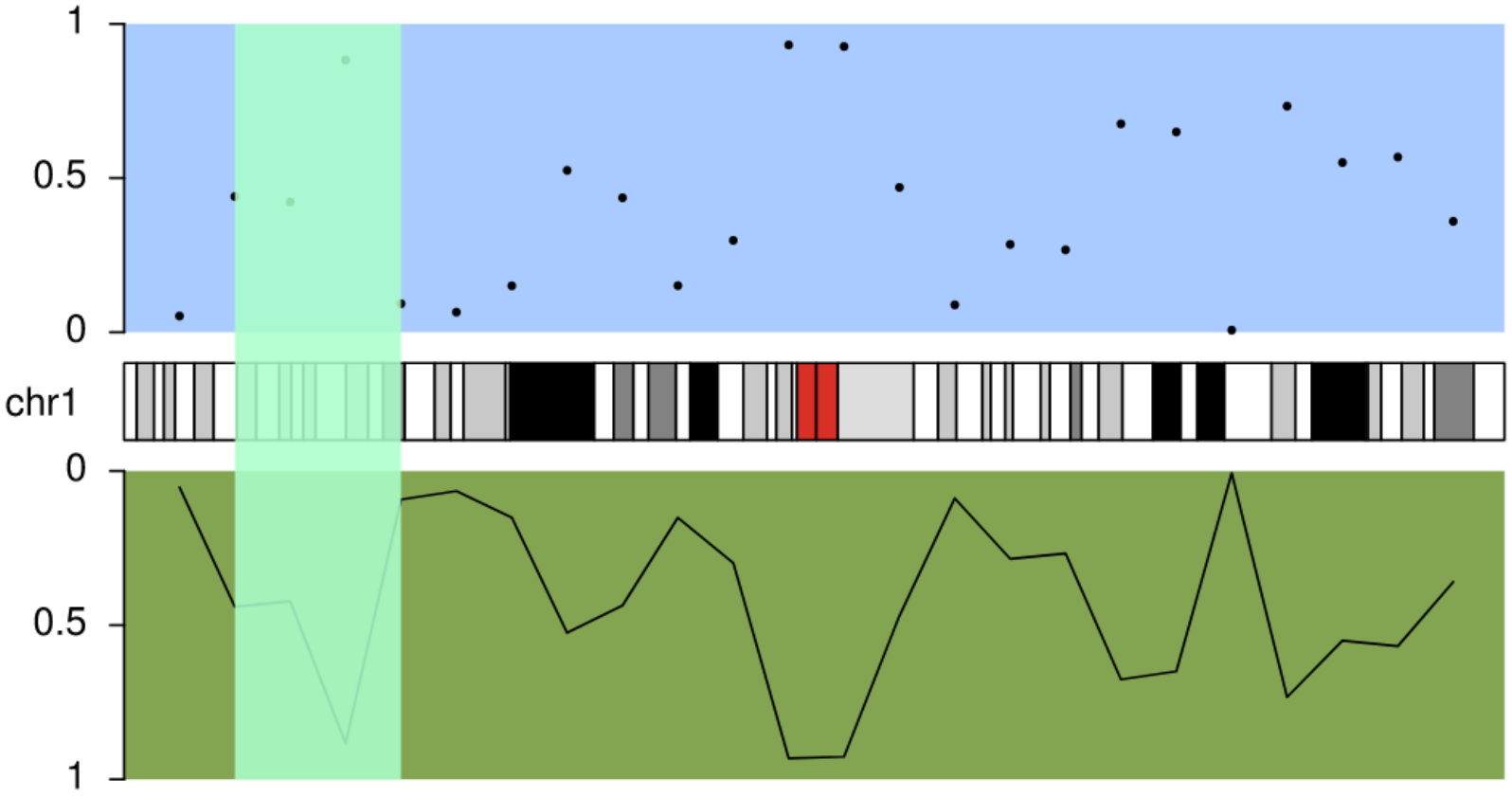
```r
y
```

```
##  [1] 0.31300794 0.91777443 0.40119072 0.45863823 0.86683937 0.11825779
##  [7] 0.99337023 0.50813481 0.06221574 0.14889436 0.89134963 0.61853153
## [13] 0.80002054 0.25188531 0.89193119 0.97436305 0.11084408 0.93324720
## [19] 0.52221278 0.94898547 0.43816815 0.53217114 0.28849970 0.97758928
```

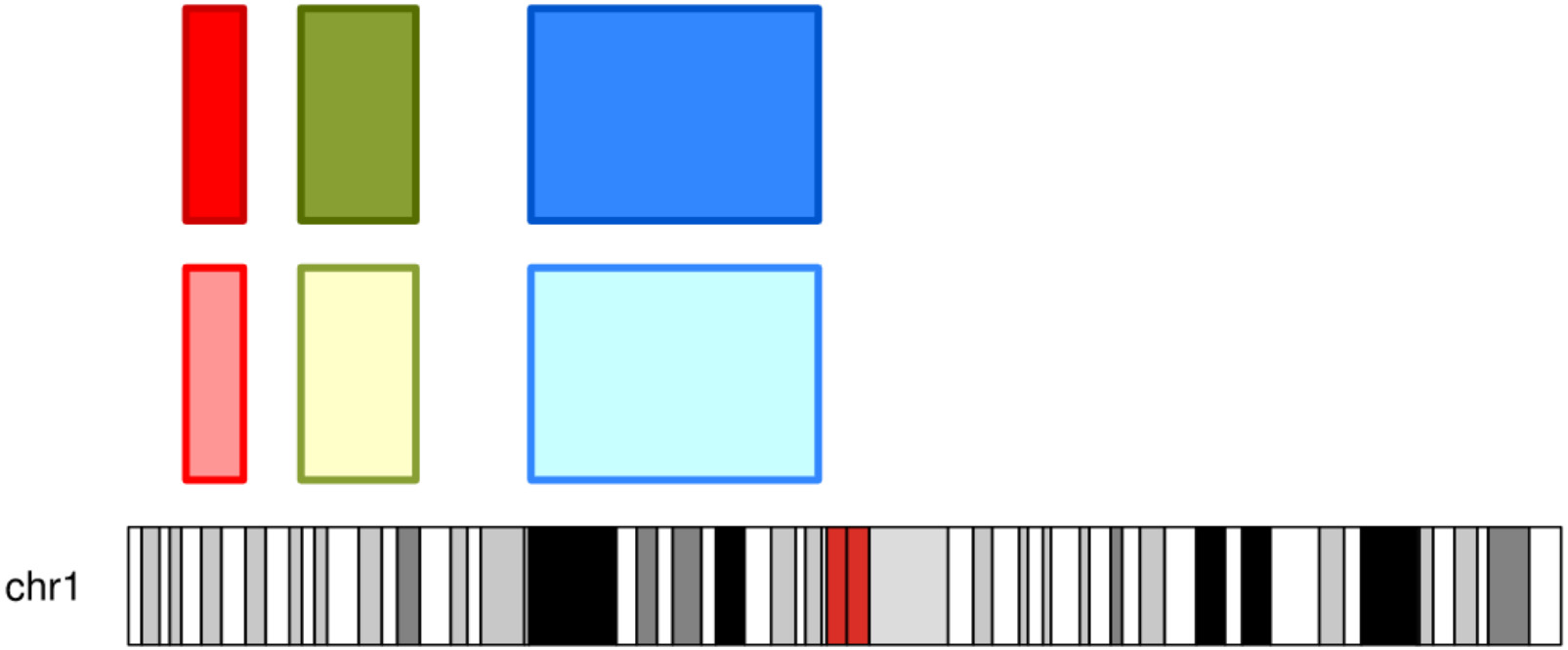
```
kp <- plotKaryotype(plot.type=2, chromosomes = "chr1")
kpDataBackground(kp, data.panel = 1, col="#AACBFF")
kpDataBackground(kp, data.panel = 2, col="#FFAACB")
kpPoints(kp, chr="chr1", x=x, y=y, data.panel = 1)
kpLines(kp, chr="chr1", x=x, y=y, data.panel = 2)
kpAxis(kp, data.panel=1)
kpAxis(kp, data.panel=2)
```



```
kp <- plotKaryotype(plot.type=2, chromosomes = "chr1")
kpDataBackground(kp, data.panel = 1, col="#AACBFF")
kpDataBackground(kp, data.panel = 2, col="#83A552")
kpPoints(kp, chr="chr1", x=x, y=y, data.panel = 1)
kpLines(kp, chr="chr1", x=x, y=y, data.panel = 2)
kpAxis(kp, data.panel=1)
kpAxis(kp, data.panel=2)

kpRect(kp, chr="chr1", x0=20e6, x1=50e6, y0=0, y1=1, col="#AAFFCBDD", data.panel="all", border=NA)
```



```
regs <- toGRanges(c("chr1:10000000-20000000",
                    "chr1:30000000-50000000",
                    "chr1:70000000-120000000"))

colors <- c("red", "#889F34", lighter(rainbow(n = 18)[12], 50))

kp <- plotKaryotype(chromosomes = "chr1")
kpPlotRegions(kp, data=regs, r0=0, r1=0.45, col = lighter(colors), border=colors, lwd=3)
kpPlotRegions(kp, data=regs, r0=0.55, r1=1, col = colors, border=darker(colors, 50), lwd=3)
```



```
regs <- toGRanges(c("chr1:10000000-20000000",
                    "chr1:30000000-50000000",
                    "chr1:70000000-120000000"))
regs
```

```
## GRanges object with 3 ranges and 0 metadata columns:
##     seqnames              ranges strand
##        <Rle>           <IRanges>  <Rle>
##   1     chr1  10000000-20000000      *
##   2     chr1  30000000-50000000      *
##   3     chr1 70000000-120000000      *
##   -------
##   seqinfo: 1 sequence from an unspecified genome; no seqlengths
```

# IRanges & GRanges

http://www.biostat.jhsph.edu/~khansen/IRangesLecture.pdf
http://bioconductor.org/packages/2.4/bioc/vignettes/IRanges/inst/doc/IRangesOverview.pdf
https://bioconductor.org/packages/release/bioc/vignettes/GenomicRanges/inst/doc/GenomicRangesIntroduction.html

IRanges – содержит координаты начала, конца интервала и любые другие столбцы

GRanges – дополнительно хранит название хромосомы и цепь

# IRanges & GRanges

```
library(IRanges)
```

```
intervals = IRanges(start = c(1,3,5), end = c(3,5,7))
intervals
```

```
## IRanges object with 3 ranges and 0 metadata columns:
##           start       end     width
##       <integer> <integer> <integer>
##   [1]         1         3         3
##   [2]         3         5         3
##   [3]         5         7         3
```

# IRanges & GRanges

```
Gintervals = GRanges(seqnames = "chr1", strand = c("+", "-", "+"), ranges = intervals)
Gintervals
```

```
## GRanges object with 3 ranges and 0 metadata columns:
##       seqnames    ranges strand
##          <Rle> <IRanges>  <Rle>
##   [1]     chr1       1-3      +
##   [2]     chr1       3-5      -
##   [3]     chr1       5-7      +
##   -------
##   seqinfo: 1 sequence from an unspecified genome; no seqlengths
```

```
kp <- plotKaryotype(plot.type = 4, ideogram.plotter = NULL, labels.plotter = NULL)
```

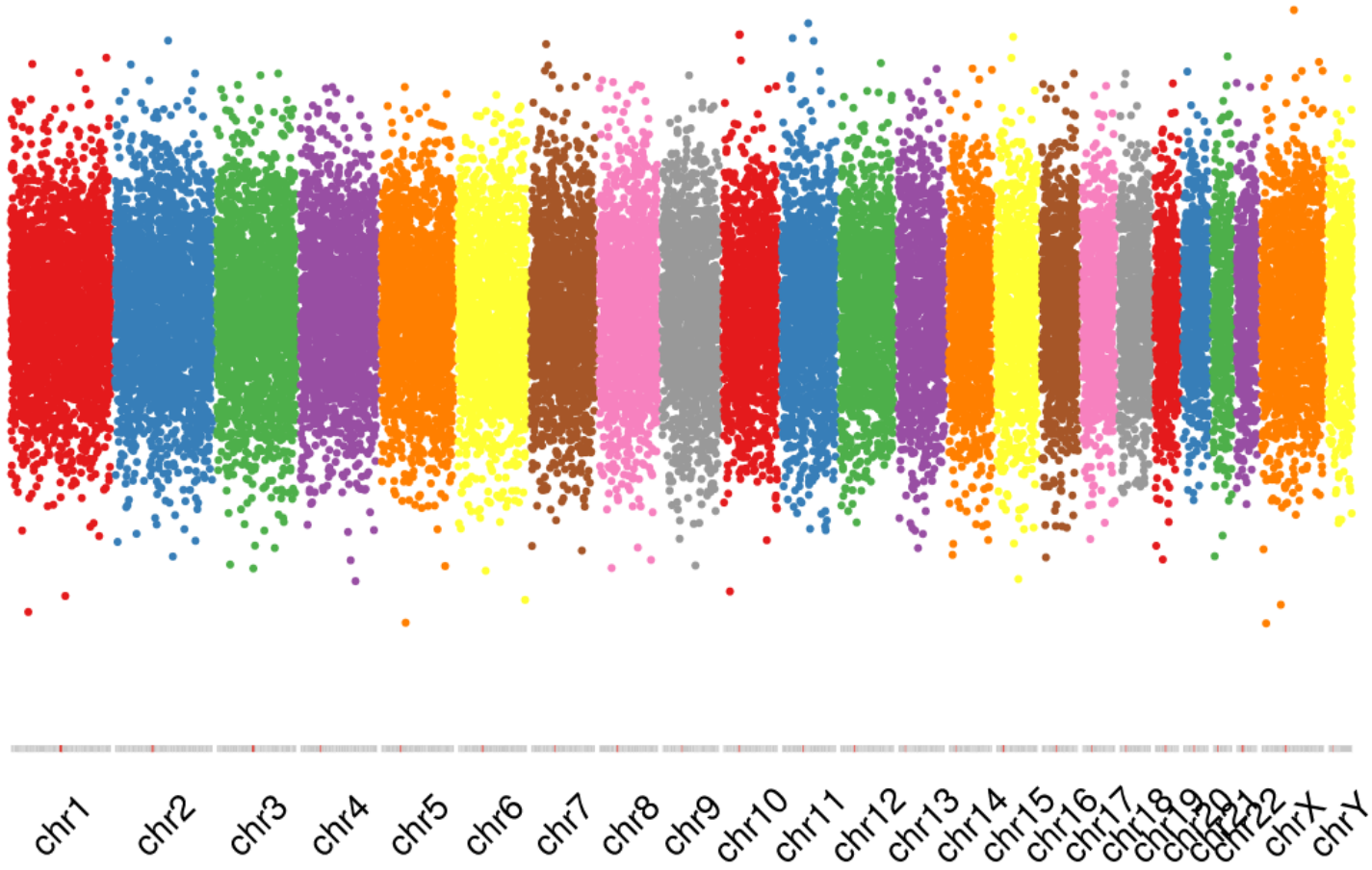
```
points <- unlist(tileGenome(kp$chromosome.lengths, tilewidth = 100e3))
points
```

```
## GRanges object with 30980 ranges and 0 metadata columns:
##           seqnames            ranges strand
##              <Rle>         <IRanges>  <Rle>
##       [1]     chr1           1-100000      *
##       [2]     chr1      100001-199999      *
##       [3]     chr1      200000-299998      *
##       [4]     chr1      299999-399998      *
##       [5]     chr1      399999-499997      *
##       ...      ...                ...    ...
##   [30976]     chrY 58873571-58973569      *
##   [30977]     chrY 58973570-59073569      *
##   [30978]     chrY 59073570-59173568      *
##   [30979]     chrY 59173569-59273567      *
##   [30980]     chrY 59273568-59373566      *
##   -------
##   seqinfo: 24 sequences from an unspecified genome
```

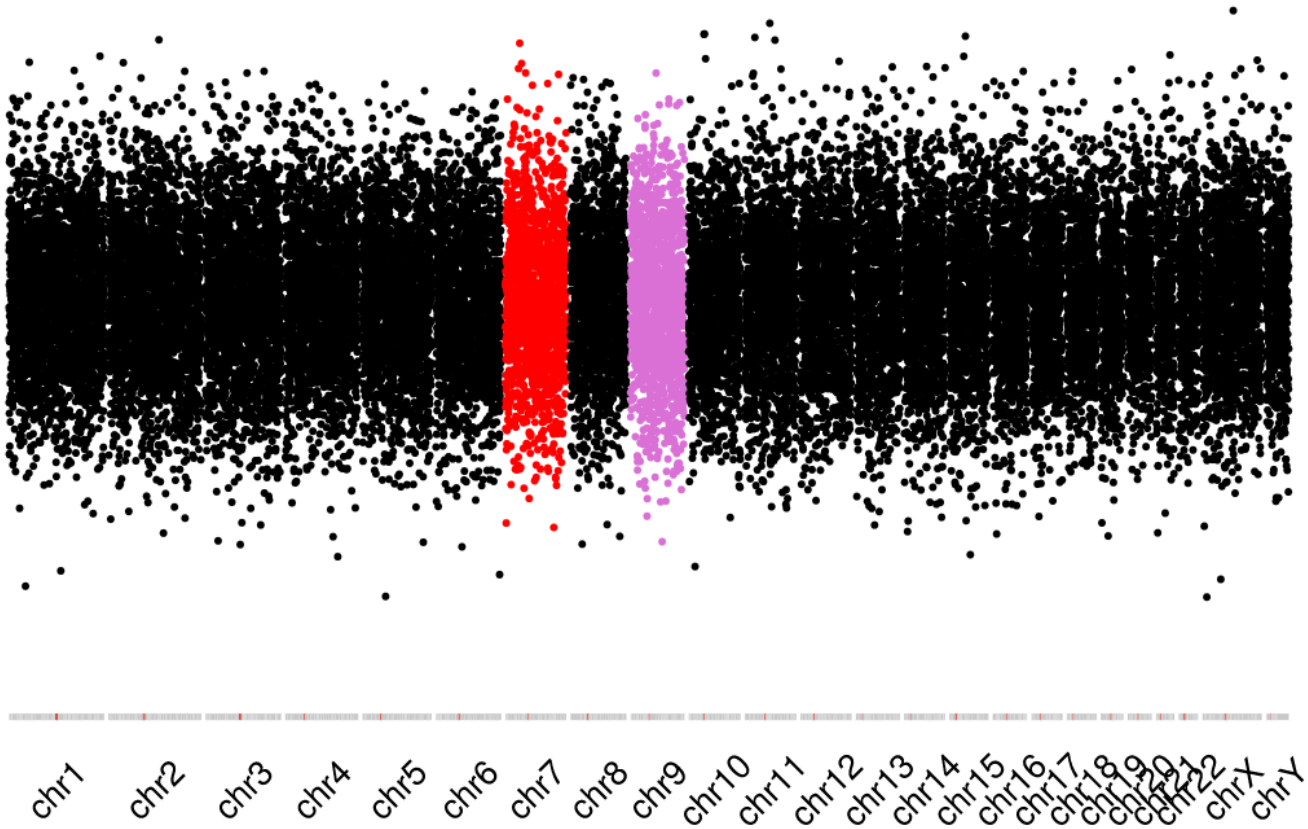
```
points$y <- rnorm(n = length(points), mean = 0.5, sd = 0.1)
points
```

```
## GRanges object with 30980 ranges and 1 metadata column:
##           seqnames              ranges strand |                 y
##              <Rle>           <IRanges>  <Rle> |         <numeric>
##       [1]     chr1             1-100000      * |  0.56607708815408
##       [2]     chr1        100001-199999      * | 0.615970501578412
##       [3]     chr1        200000-299998      * | 0.488076976023875
##       [4]     chr1        299999-399998      * | 0.357143516464714
##       [5]     chr1        399999-499997      * |  0.44497205230684
##       ...      ...                 ...    ... .               ...
##   [30976]     chrY 58873571-58973569      * |  0.37564289134434
##   [30977]     chrY 58973570-59073569      * | 0.496087736648997
##   [30978]     chrY 59073570-59173568      * |  0.54466226288011
##   [30979]     chrY 59173569-59273567      * | 0.601041564899096
##   [30980]     chrY 59273568-59373566      * | 0.431082944575043
##   -------
##   seqinfo: 24 sequences from an unspecified genome
```

```
kp <- plotKaryotype(plot.type = 4, ideogram.plotter = NULL, labels.plotter = NULL)
kpAddCytobandsAsLine(kp)
kpAddChromosomeNames(kp, srt=45)

kpPoints(kp, data = points, col=colByChr(points, colors = "brewer.set1"))
```



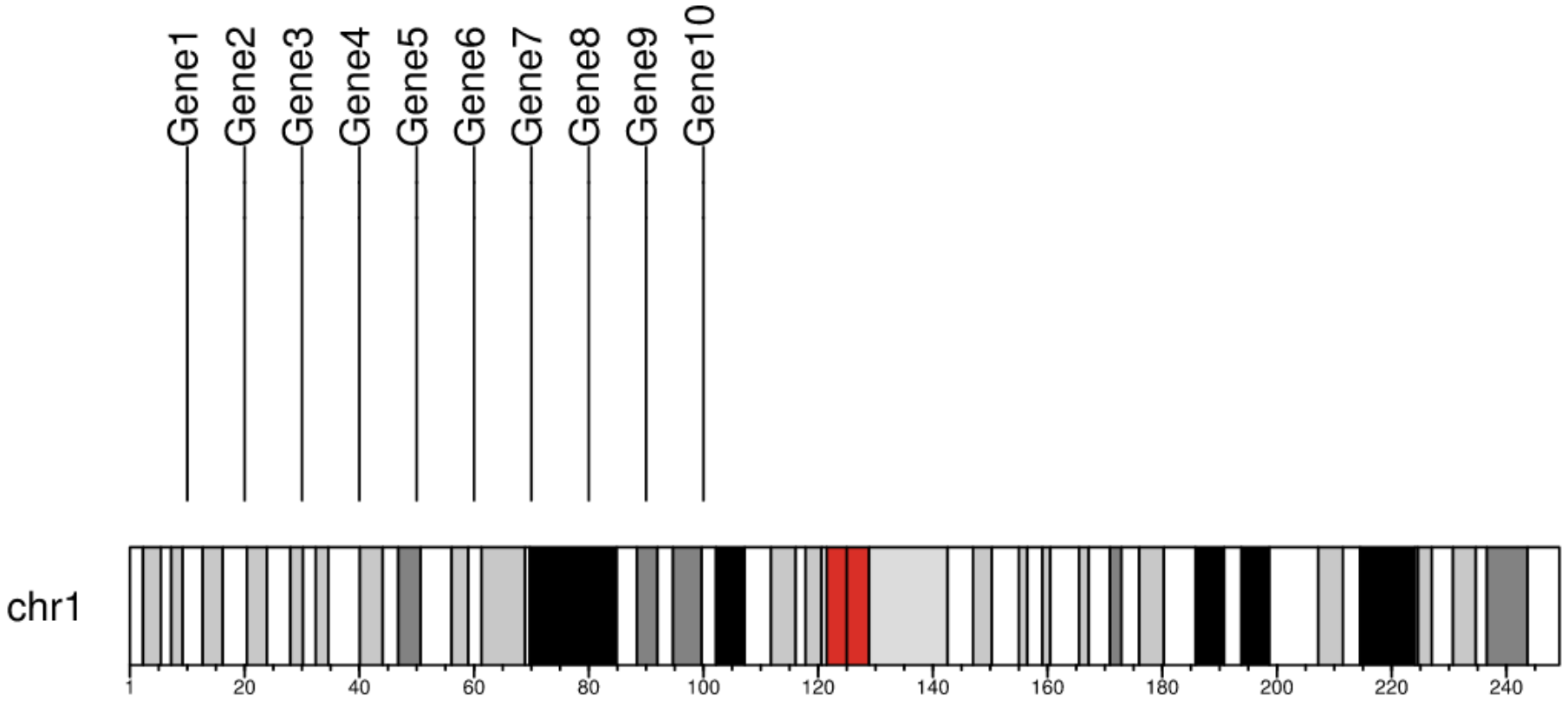
```
cols <- c(chr7="red", chr9="orchid")

kp <- plotKaryotype(plot.type = 4, ideogram.plotter = NULL, labels.plotter = NULL)
kpAddCytobandsAsLine(kp)
kpAddChromosomeNames(kp, srt=45)

kpPoints(kp, data = points, col=colByChr(points, colors = cols))
```



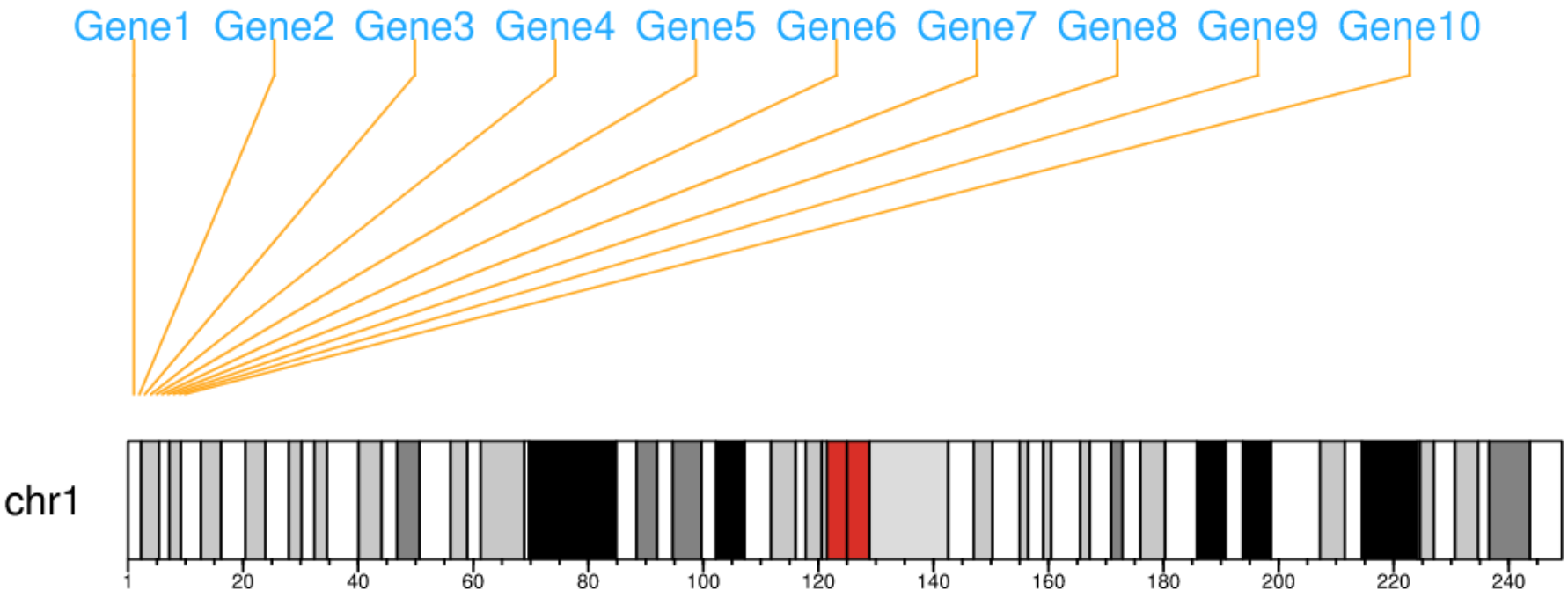
```r
markers <- data.frame(chr=rep("chr1", 10), pos=(1:10*10e6), labels=paste0("Gene", 1:10))
markers
```

```
##     chr   pos labels
## 1  chr1 1e+07  Gene1
## 2  chr1 2e+07  Gene2
## 3  chr1 3e+07  Gene3
## 4  chr1 4e+07  Gene4
## 5  chr1 5e+07  Gene5
## 6  chr1 6e+07  Gene6
## 7  chr1 7e+07  Gene7
## 8  chr1 8e+07  Gene8
## 9  chr1 9e+07  Gene9
## 10 chr1 1e+08 Gene10
```

```
kp <- plotKaryotype(chromosomes="chr1")
kpAddBaseNumbers(kp)
kpPlotMarkers(kp, chr=markers$chr, x=markers$pos, labels=markers$labels)
```



```r
markers <- data.frame(chr=rep("chr1", 10), pos=(1:10*1e6), labels=paste0("Gene", 1:10))

kp <- plotKaryotype(chromosomes="chr1")
kpAddBaseNumbers(kp)
kpPlotMarkers(kp, chr=markers$chr, x=markers$pos, labels=markers$labels,
              text.orientation = "horizontal", marker.parts = c(0, 0.9, 0.1),
              line.color = "#FFAA22", label.color = "#22AAFF",
              label.dist = 0.01, max.iter = 1000)
```



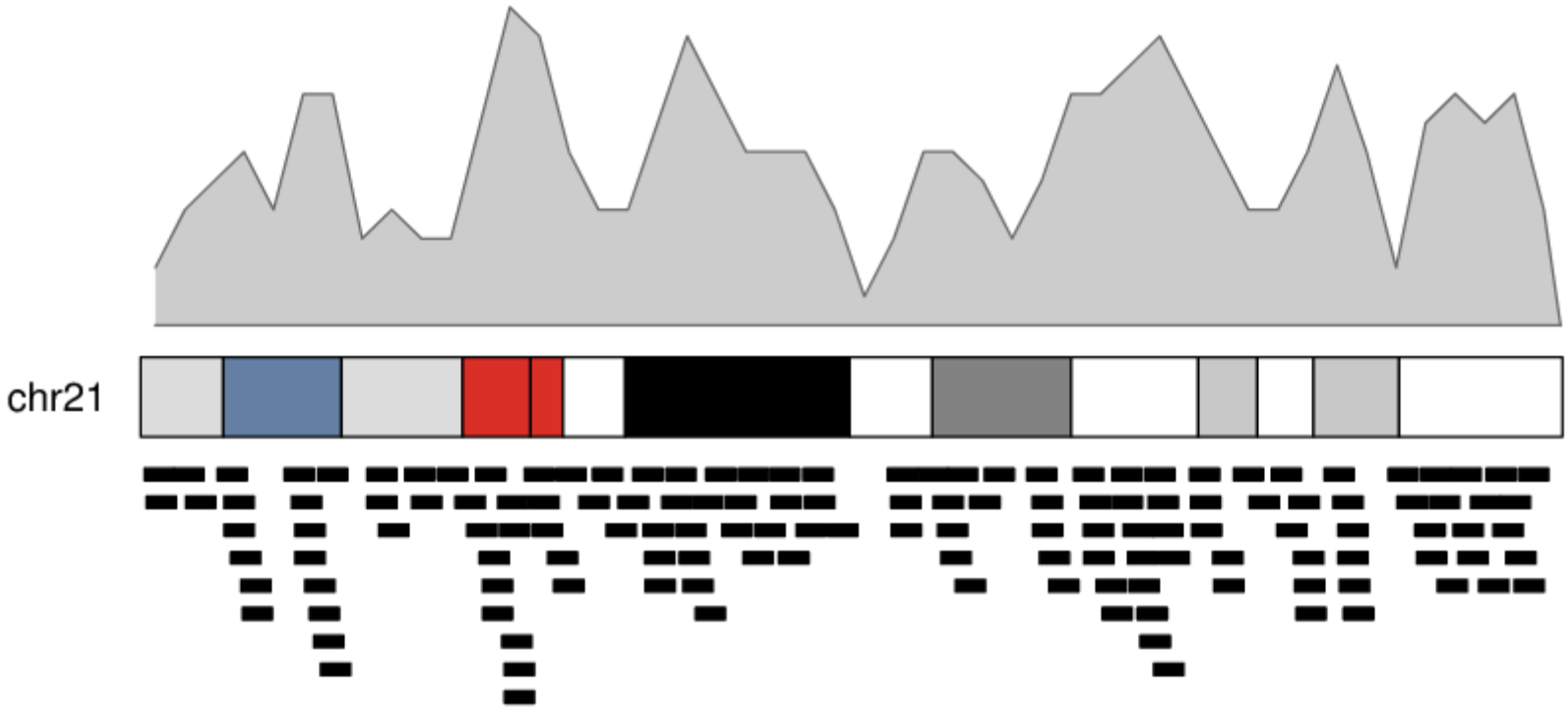
```r
library(BSgenome.Hsapiens.UCSC.hg19)
```

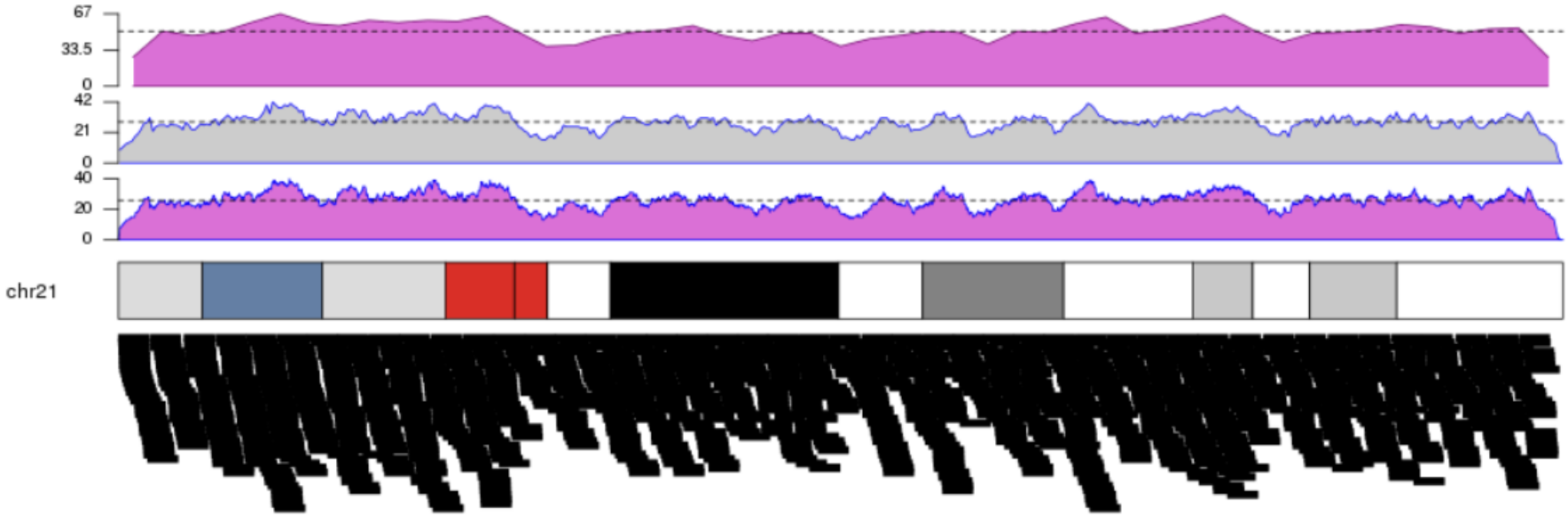
```r
regions <- createRandomRegions(nregions=10000, length.mean = 1e6, mask=NA, non.overlapping = FALSE)
```

```r
regions
```

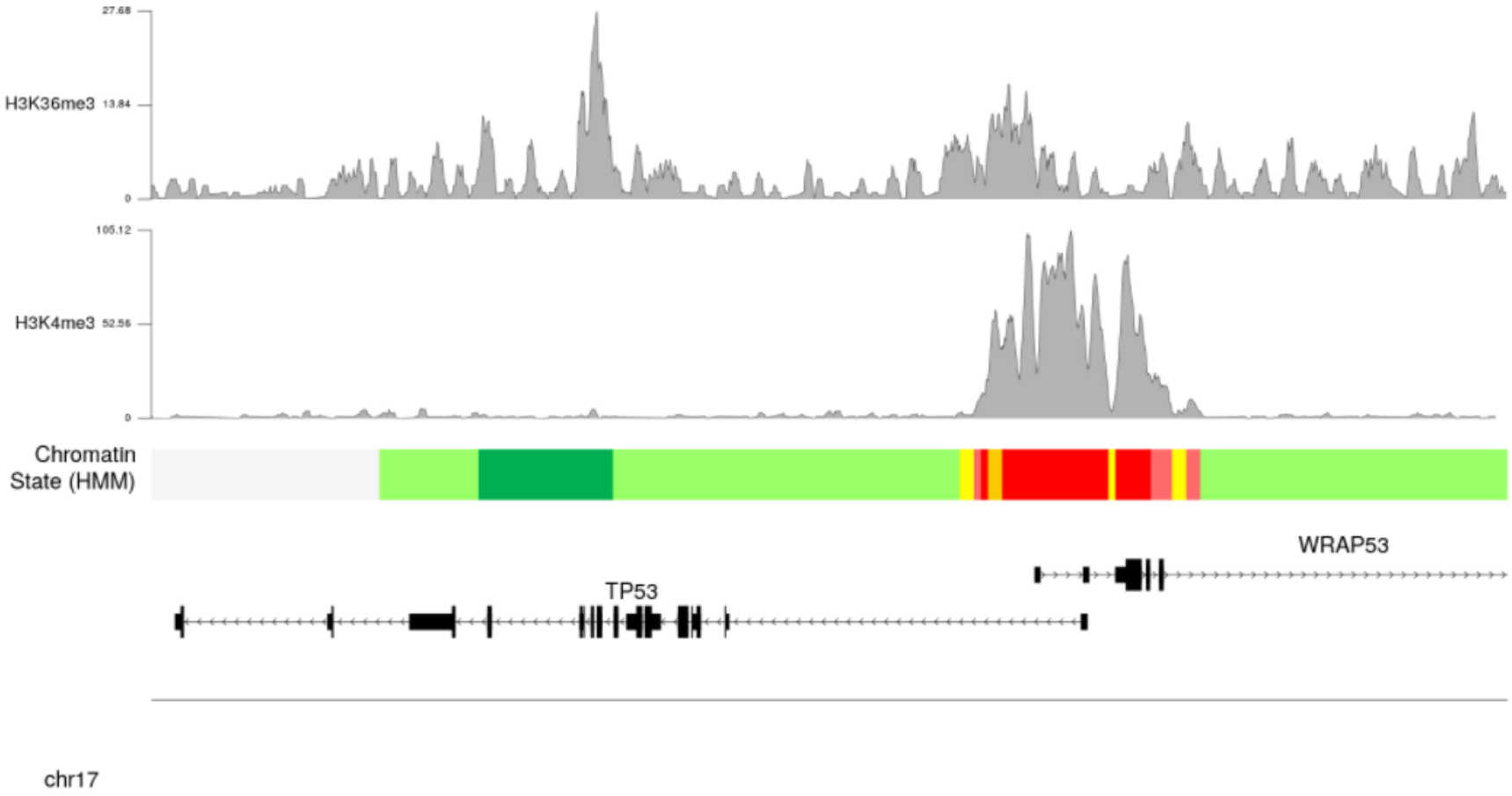
```
## GRanges object with 10000 ranges and 0 metadata columns:
##           seqnames              ranges strand
##              <Rle>           <IRanges>  <Rle>
##       [1]    chr10   37181327-38181308      *
##       [2]     chrY   54537740-55537738      *
##       [3]     chr6 161761847-162761800      *
##       [4]     chrX   40630528-41630528      *
##       [5]     chr8   65066326-66066302      *
##       ...      ...                 ...    ...
##    [9996]     chr3 105958555-106958530      *
##    [9997]    chr12 124535264-125535255      *
##    [9998]    chr12   68334781-69334784      *
##    [9999]     chr4   28886866-29886886      *
##   [10000]    chr12   73143491-74143508      *
##   -------
##   seqinfo: 93 sequences from an unspecified genome; no seqlengths
```

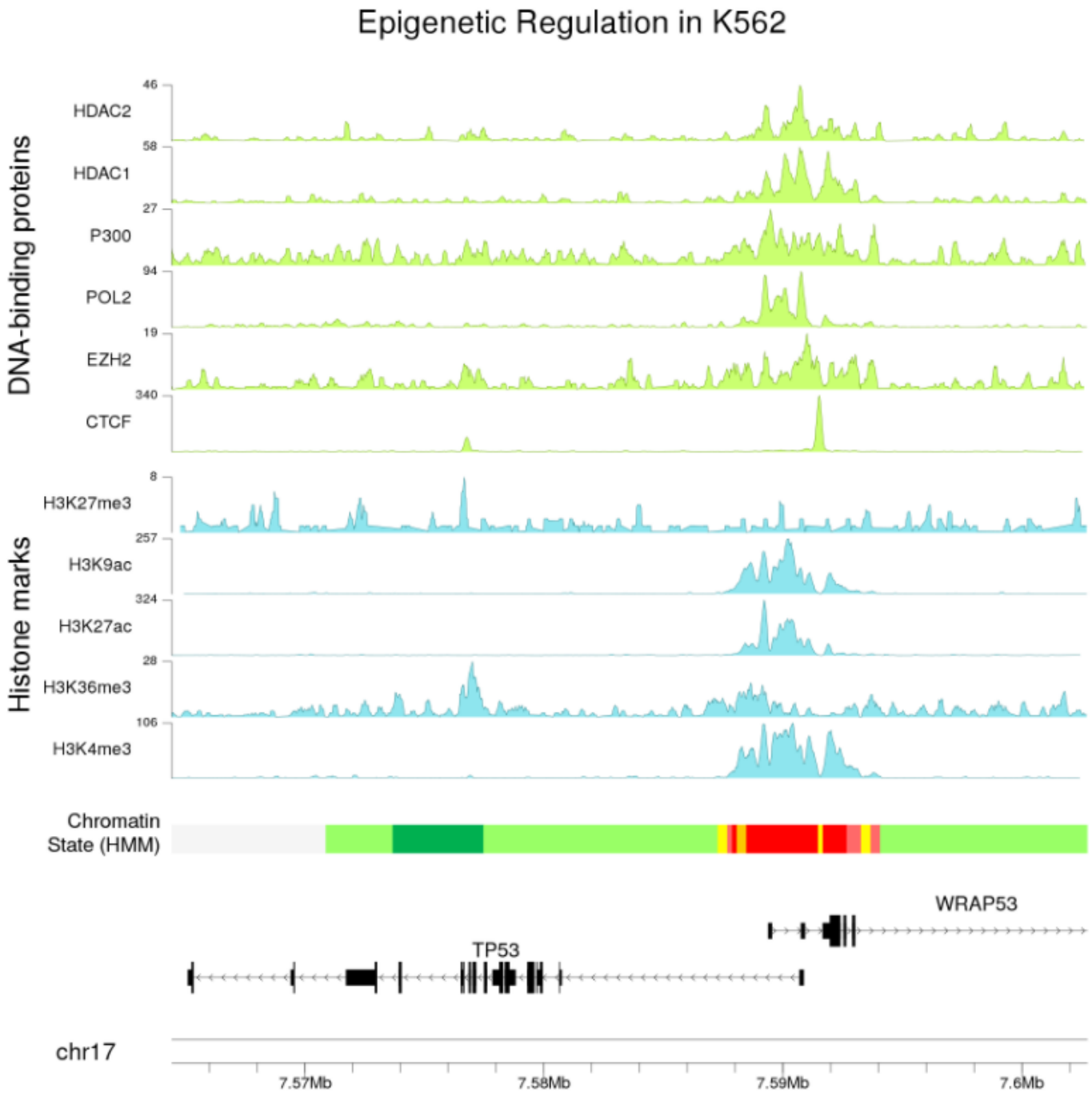
```
kp <- plotKaryotype(plot.type=2, chromosomes = "chr21")
kpPlotDensity(kp, data=regions)
kpPlotRegions(kp, data=regions, data.panel=2)
```






+ понимает разные форматы файлов

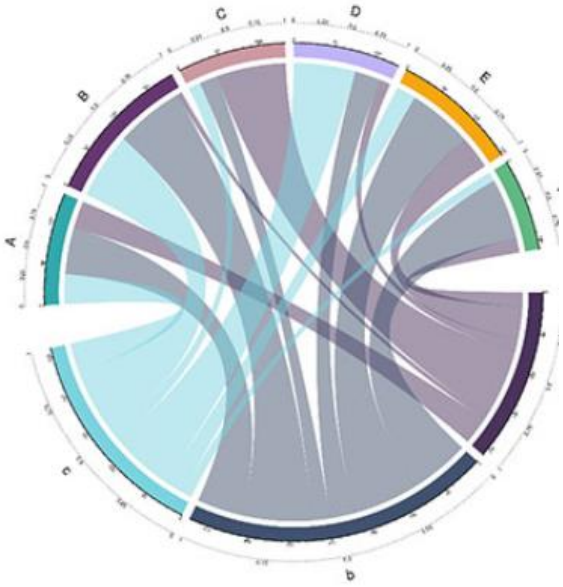

27.68
H3K36me3 13.84
0
105.12
H3K4me3 52.56
0
Chromatin State (HMM)
WRAP53
TP53
chr17

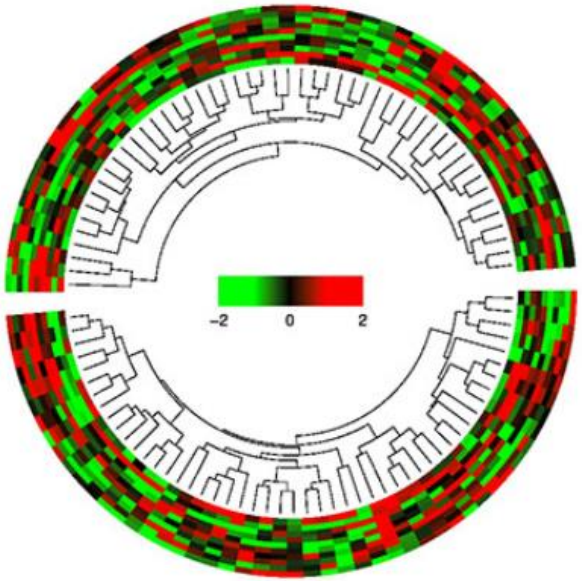

Epigenetic Regulation in K562
DNA-binding proteins
HDAC2
HDAC1
P300
POL2
EZH2
CTCF
46
58
27
94
19
340
Histone marks
H3K27me3
H3K9ac
H3K27ac
H3K36me3
H3K4me3
8
257
324
28
106
Chromatin State (HMM)
WRAP53
TP53
chr17
7.57Mb
7.58Mb
7.59Mb
7.6Mb

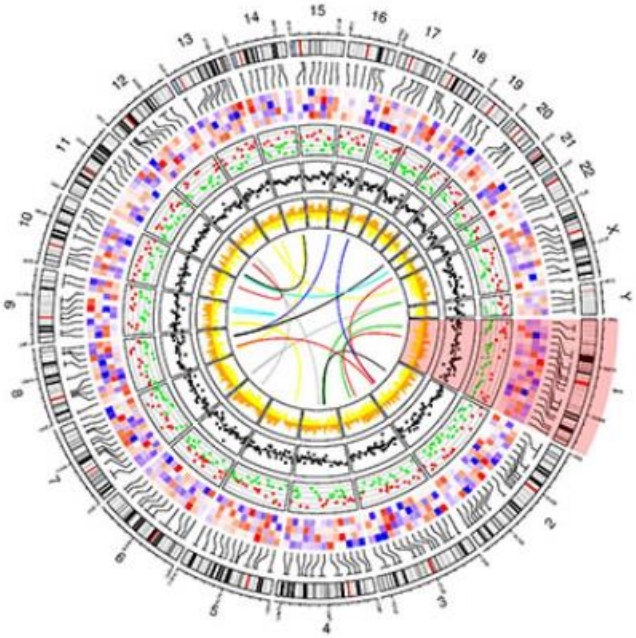
# circlize & BioCircos

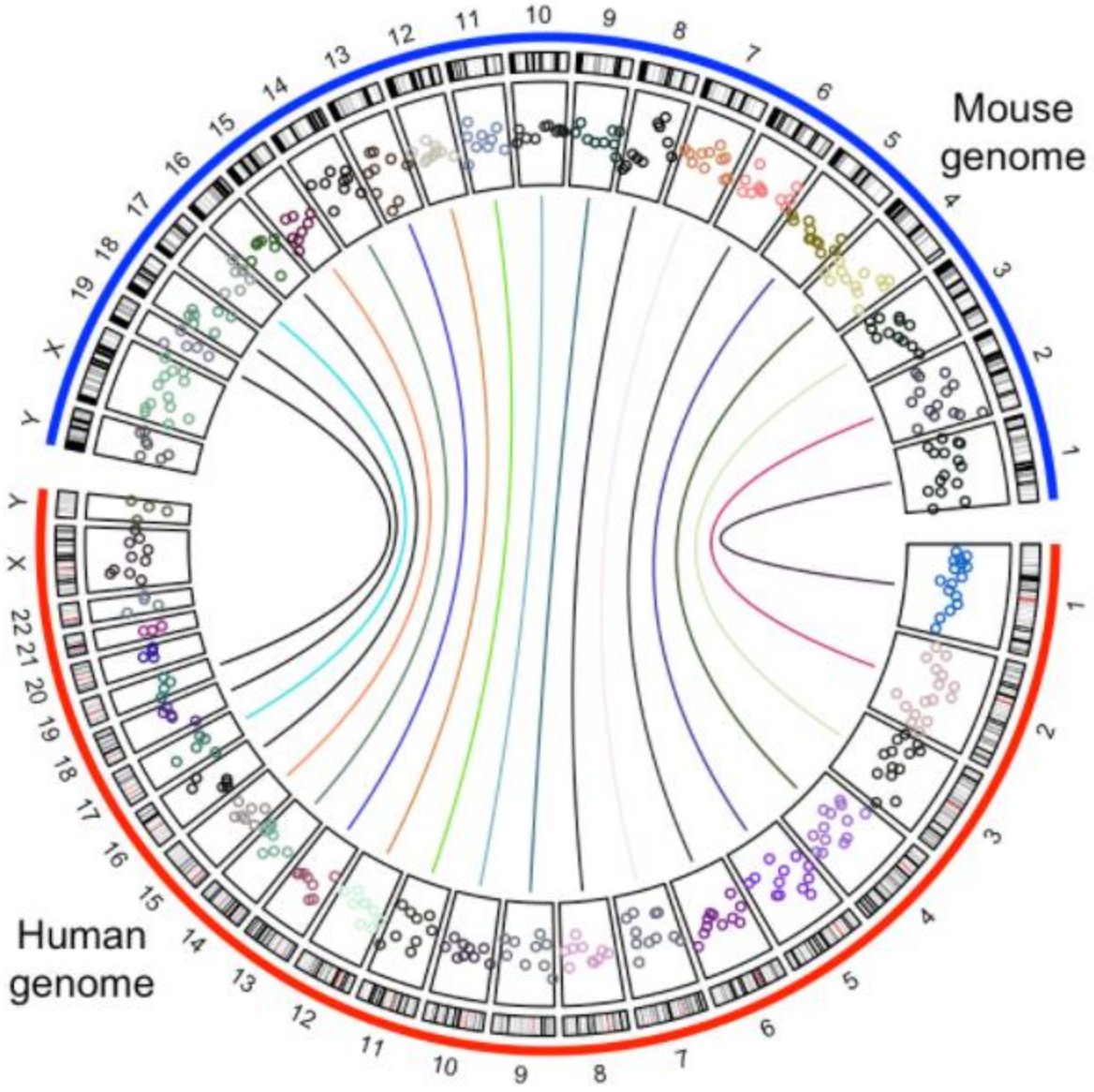
https://jokergoo.github.io/circlize_book/book/

https://cran.r-project.org/web/packages/BioCircos/vignettes/BioCircos.html




Mouse genome
Human genome

# valr

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5506536/
https://rpubs.com/jayhesselberth/valr

# bedtools

https://bedtools.readthedocs.io/en/latest/

# valr

- `read_bed()` : read a BED3+ file
- `read_bed12()` : read a BED12 file
- `read_bedgraph()` : read a bedGraph file
- `read_genome()` : read a UCSC “chrom size” file
- `read_vcf()` : read the Variant Call Format
- `read_bigwig()` : read UCSC bigWig files

# valr - bed

```
read_bed(system.file('extdata', '3fields.bed.gz', package = 'valr'))
#> # A tibble: 10 x 3
#>    chrom  start    end
#>    <chr>  <int>  <int>
#> 1   chr1  11873  14409
#> 2   chr1  14361  19759
#> 3   chr1  14406  29370
#> 4   chr1  34610  36081
#> 5   chr1  69090  70008
#> 6   chr1 134772 140566
#> 7   chr1 321083 321115
#> 8   chr1 321145 321207
#> 9   chr1 322036 326938
#> 10  chr1 327545 328439
```

# valr - bed

```
read_bed(n_fields = 6, system.file('extdata', '6fields.bed.gz', package = 'valr'))
#> # A tibble: 10 x 6
#>    chrom  start    end         name score strand
#>    <chr>  <int>  <int>        <chr> <chr>  <chr>
#> 1   chr1  11873  14409      DDX11L1     3      +
#> 2   chr1  14361  19759       WASH7P    10      -
#> 3   chr1  14406  29370       WASH7P     7      -
#> 4   chr1  34610  36081      FAM138F     3      -
#> 5   chr1  69090  70008        OR4F5     1      +
#> 6   chr1 134772 140566    LOC729737     3      -
#> 7   chr1 321083 321115     DQ597235     1      +
#> 8   chr1 321145 321207     DQ599768     1      +
#> 9   chr1 322036 326938 LOC100133331     3      +
#> 10  chr1 327545 328439    LOC388312     1      +
```

# valr - bed

```
read_bed(n_fields = 6, system.file('extdata', '6fields.bed.gz', package = 'valr'))
#> # A tibble: 10 x 6
#>    chrom  start    end         name score strand
#>    <chr>  <int>  <int>        <chr> <chr>  <chr>
#> 1   chr1  11873  14409      DDX11L1     3      +
#> 2   chr1  14361  19759       WASH7P    10      -
#> 3   chr1  14406  29370       WASH7P     7      -
#> 4   chr1  34610  36081      FAM138F     3      -
#> 5   chr1  69090  70008        OR4F5     1      +
#> 6   chr1 134772 140566    LOC729737     3      -
#> 7   chr1 321083 321115     DQ597235     1      +
#> 8   chr1 321145 321207     DQ599768     1      +
#> 9   chr1 322036 326938 LOC100133331     3      +
#> 10  chr1 327545 328439    LOC388312     1      +
```

# valr - bedGraph

```
read_bedgraph(system.file('extdata', 'test.bg.gz', package = 'valr'))
#> # A tibble: 4 x 4
#>   chrom    start      end value
#>   <chr>    <int>    <int> <dbl>
#> 1 chr19 49302000 49302300 -1.00
#> 2 chr19 49302300 49302600 -0.75
#> 3 chr19 49302600 49302900 -0.50
#> 4 chr19 49302900 49303200 -0.25
```

# valr – практически полностью повторяет bedtools

## Single set operations

These methods operate on a single set of intervals:

- `bed_sort()` : order intervals
- `bed_cluster()` : Cluster (but don't merge) overlapping/nearby intervals.
- `bed_complement()` : extract intervals *not* represented by an interval file.
- `bed_merge()` : combine overlapping and nearby intervals into a single interval.
- `bed_flank()` : Generate new flanking intervals
- `bed_slop()` : Expand the size of input intervals
- `bed_shift()` : Shift the coordinates of an input set, bounded by a genome

Sort | Cluster | Complement | Merge | Flank | Slop | Shift

# valr – практически полностью повторяет bedtools

## Multiple set operations

These methods compare two sets of intervals:

- `bed_intersect()` : find overlapping intervals
- `bed_map()` : apply a function to selected columns for overlapping intervals
- `bed_subtract()` : Remove intervals based on overlaps between two files
- `bed_window()` : Find overlapping intervals within a window
- `bed_closest()` : find the closest intervals independent of overlaps

Intersection | Map | Subtract | Window | Closest

# valr – практически полностью повторяет bedtools

## Randomzing intervals

`valr` provides methods for creating new random intervals or permutations of existing intervals:

- `bed_random` generates random intervals from an input `genome`.
- `bed_shuffle` shuffles coordinates given a set of input intervals.
- Random sampling of input intervals is done with `dplyr`.

Random | Sample | Shuffle

# valr – практически полностью повторяет bedtools

## Randomzing intervals

`valr` provides methods for creating new random intervals or permutations of existing intervals:

- `bed_random` generates random intervals from an input `genome`.
- `bed_shuffle` shuffles coordinates given a set of input intervals.
- Random sampling of input intervals is done with `dplyr`.

Random | Sample | Shuffle