ggbio: Extends the grammar of graphics to genomic data and object-oriented visualization tool.

Tengfei Yin

Iowa State University

July 25, 2012

Outline

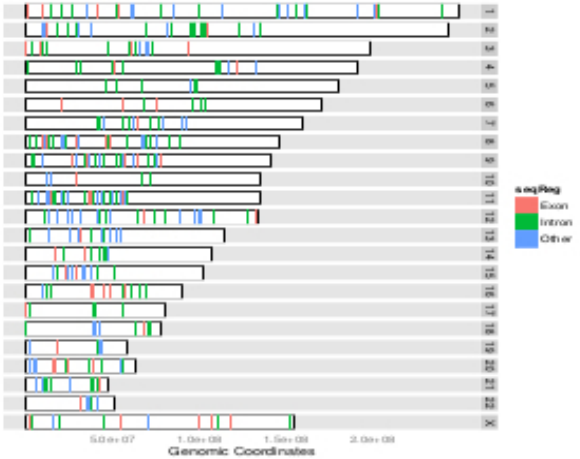
Outline

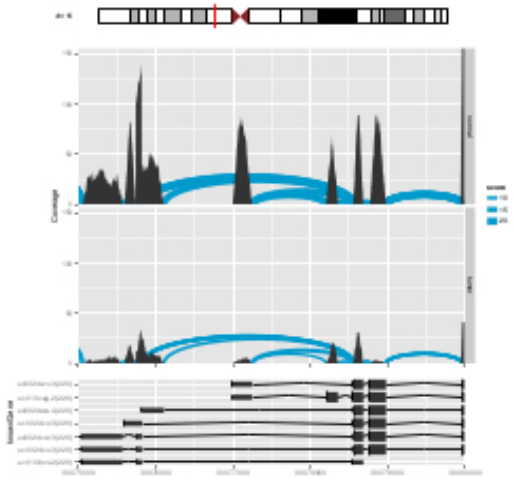
Outline

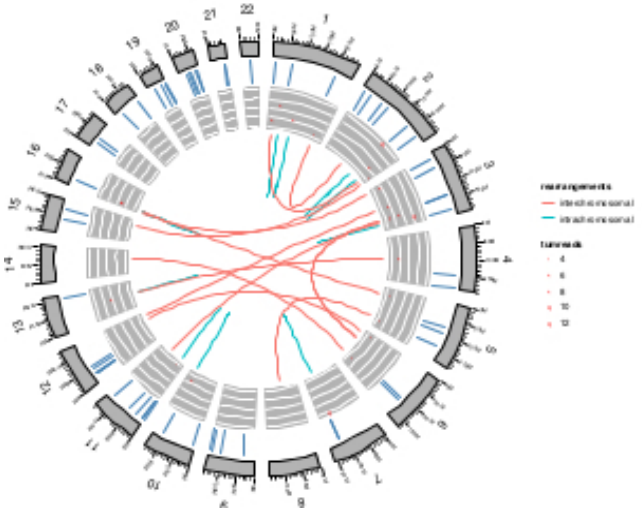
Motivation

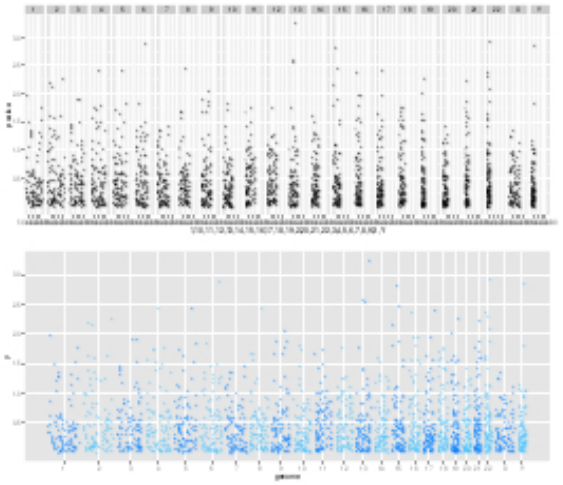
- Explore the data in different ways more efficient, flexible.
- Generalize genomic visualization frame work and object-oriented visualization.
- Elegant graphics for publications.
- Modularize components to facilitate construction of high level graphics.

What can ggbio does?









Outline

Package release/development

- Bioconductor is the *ONLY* way to get released/devel version packages for ggbio.
- Github development is stopped for developing ggbio.
- Package attached vignettes are recommended.
- On-line documentation will be changed soon to document examples.

Update

- R: >=2.15.0
- ggbio: >=1.5.10
- biovizBase: >=1.5.4

Use following code to update if you are no using AMI.

```
> library(BiocInstaller)
> useDevel(TRUE)
> biocLite("ggbio")
```

Rstudio(AMI) problem

If you can not load packages in AMI, please try

workaround

```
> source("~/.Rprofile")
```

How to get workshop code

- https://github.com/tengfei/slides
- Find main.R file under bioc2012/ggbio.

How to get vignette and example code

```
> help.start()
> ## or
> browseVignettes(package = "ggbio")
> ## browseVignettes may not work in AMI.
```

Down load pdf or copy R code.

Outline

- Demo basically from tutorial for seq data.
- Goal:
  - Overview of top 50 different binding peaks cross two samples.
  - Tracks for a particular region
    - One genomic features track.
    - One coverage track faceted by samples.
    - One single chromosome ideogram track.

Getting the ChIP-seq sample data

```
> library(chipseq)
> data(cstest)
> ## estimate fragment length
> fraglen <- estimate.mean.fraglen(cstest$ctcf)
> fraglen[!is.na(fraglen)]

   chr10    chr11    chr12
179.6794 172.4884 181.6732

> ctcf.ext <- resize(cstest$ctcf, width = 200)
> cov.ctcf <- coverage(ctcf.ext)
> gfp.ext <- resize(cstest$gfp, width = 200)
> cov.gfp <- coverage(gfp.ext)
```

Finding different binding peaks

```
> peakCutoff(cov.ctcf, fdr = 0.0001)

[1] 6.959837

> peaks.ctcf <- slice(cov.ctcf, lower = 8)
> peaks.gfp <- slice(cov.gfp, lower = 8)
> peakSummary <- diffPeakSummary(peaks.gfp, peaks.ctcf)
> peakSummary <-within(peakSummary,
+     {
+       diffs <- asinh(sums2) - asinh(sums1)
+       resids <- (diffs - median(diffs)) / mad(diffs)
+       up <- resids > 2
+       down <- resids < -2
+       change <- ifelse(up, "up",
+                        ifelse(down, "down", "flat"))
+     })
> pks <- as(peakSummary, 'GRanges')
```

GRanges object

- Container to store genomic interval and meta data.
- Serve as a foundamental and critical unit in *ggbio*

```
> head(pks)

GRanges with 6 ranges and 10 elementMetadata cols:
      seqnames             ranges strand |  comb.max     sums1     sums2
         <Rle>          <IRanges>  <Rle> | <integer> <integer> <integer>
  [1]    chr10 [3012955, 3013135]      * |        11         0      1799
  [2]    chr10 [3234799, 3234895]      * |        10         0       896
  [3]    chr10 [3270012, 3270297]      * |        20       160      4030
  [4]    chr10 [3277662, 3277832]      * |        13         0      1668
  [5]    chr10 [3277848, 3277859]      * |         8         0        96
  [6]    chr10 [3460857, 3460973]      * |        11       117      1047
         maxs1     maxs2      change     diffs      down     resids        up
     <integer> <integer> <character> <numeric> <logical>  <numeric> <logical>
  [1]         0        11        flat  8.188133     FALSE  0.4823571     FALSE
  [2]         0        10        flat  7.491088     FALSE  0.2876319     FALSE
  [3]         1        19        flat  3.226338     FALSE -0.9037602     FALSE
  [4]         0        13        flat  8.112528     FALSE  0.4612361     FALSE
  [5]         0         8        flat  5.257522     FALSE -0.3363326     FALSE
  [6]         1        10        flat  2.191492     FALSE -1.1928527     FALSE
  ---
  seqlengths:
           chr1         chr2         chr3 ...  chrY_random chrUn_random
             NA           NA           NA ...           NA           NA
```

Remove redundant seqnames and add seqlengths

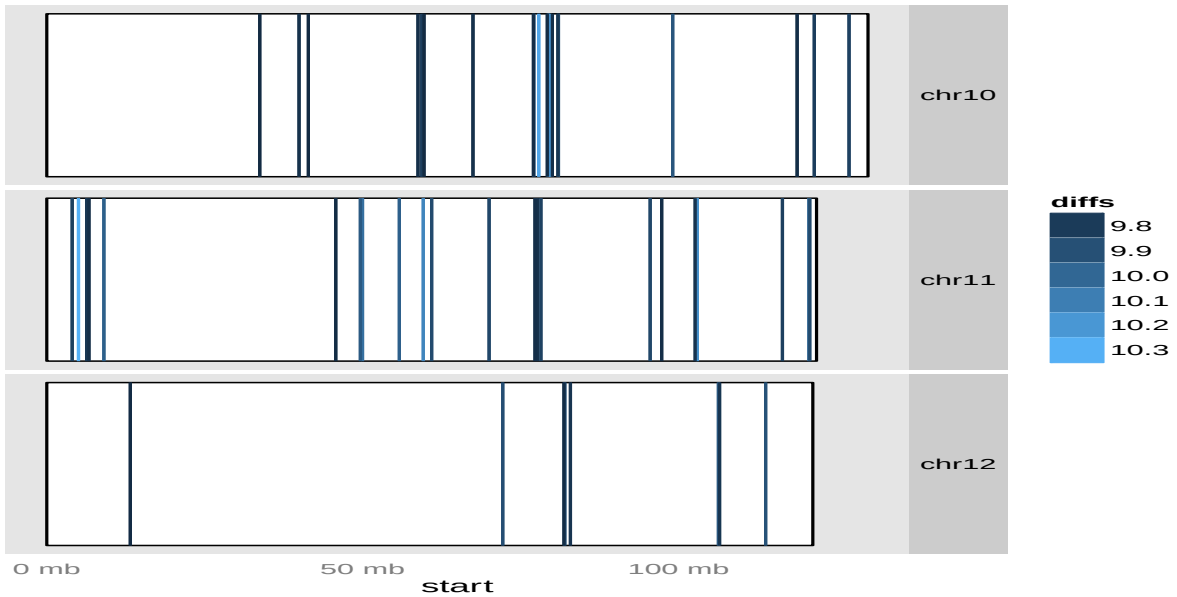
```
> chrs <- unique(as.character(seqnames(pks)))
> library(GenomicRanges)
> pks <- keepSeqlevels(pks, chrs)
> seqlengths(pks) <- seqlengths(cstest)[chrs]
> head(pks)

GRanges with 6 ranges and 10 elementMetadata cols:
      seqnames             ranges strand |  comb.max     sums1     sums2
         <Rle>          <IRanges>  <Rle> | <integer> <integer> <integer>
  [1]    chr10 [3012955, 3013135]      * |        11         0      1799
  [2]    chr10 [3234799, 3234895]      * |        10         0       896
  [3]    chr10 [3270012, 3270297]      * |        20       160      4030
  [4]    chr10 [3277662, 3277832]      * |        13         0      1668
  [5]    chr10 [3277848, 3277859]      * |         8         0        96
  [6]    chr10 [3460857, 3460973]      * |        11       117      1047
         maxs1     maxs2      change     diffs      down     resids        up
     <integer> <integer> <character> <numeric> <logical>  <numeric> <logical>
  [1]         0        11        flat  8.188133     FALSE  0.4823571     FALSE
  [2]         0        10        flat  7.491088     FALSE  0.2876319     FALSE
  [3]         1        19        flat  3.226338     FALSE -0.9037602     FALSE
  [4]         0        13        flat  8.112528     FALSE  0.4612361     FALSE
  [5]         0         8        flat  5.257522     FALSE -0.3363326     FALSE
  [6]         1        10        flat  2.191492     FALSE -1.1928527     FALSE
  ---
  seqlengths:
       chr10     chr11     chr12
   129993255 121843856 121257530
```

Overview of the top 50 different binding peaks

- layout_karyogram: stack all the chromosome
- seqlength required.

```
> pks <- pks[order(values(pks)$diffs,
+                  decreasing = TRUE)][1:50]
> library(ggbio)
> autoplot(pks, layout = "karyogram",
+          aes(color = diffs, fill = diffs))
```



Define a region to dig into

- Pick up a region of interest.
- That's a region which contains differential peaks.

```
> wh <- GRanges("chr11", IRanges(3062594, 3092593))
```

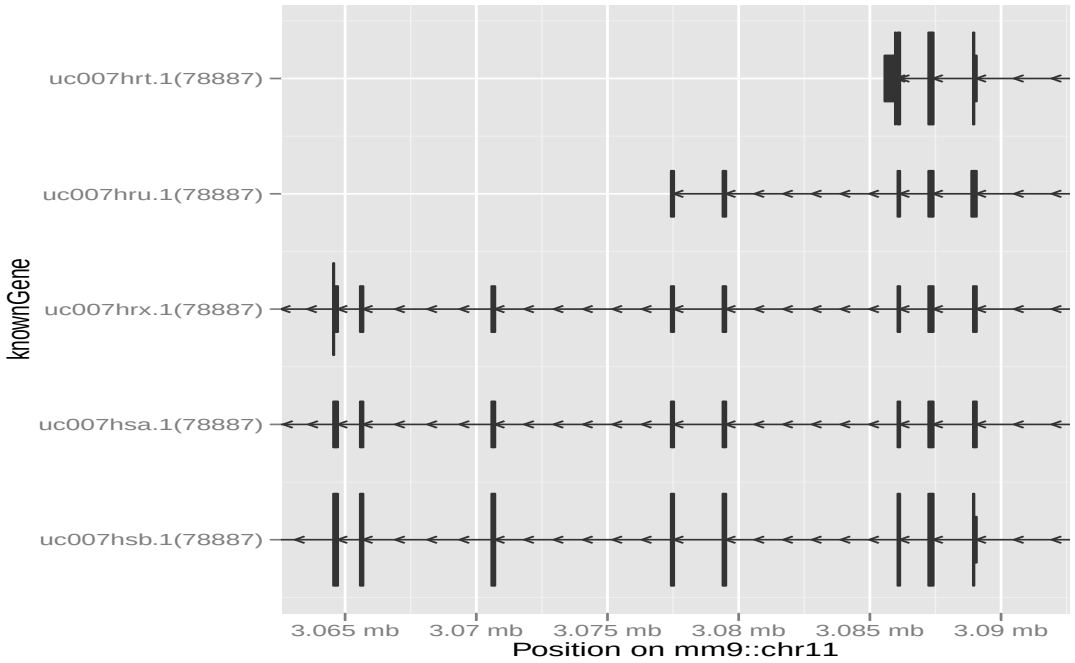
Genomic features track.

Coverage for particular region is ready, let's create a genomic feature track.

- Object: *TranscriptDb*.
- Function: `autoplot, stat_gene`.

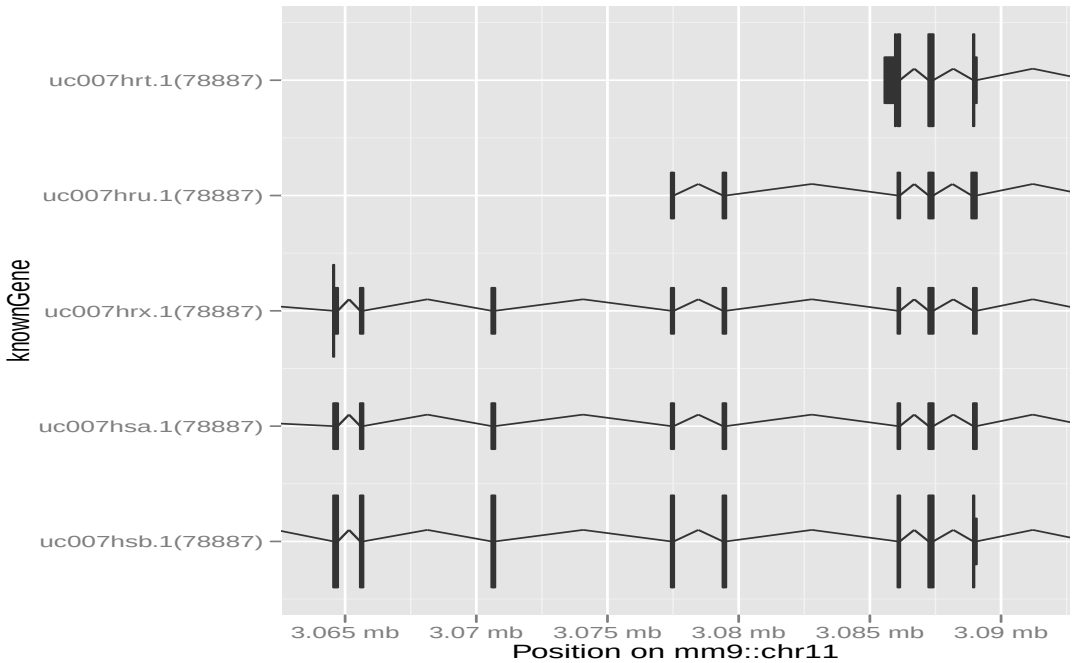
Genomic features track(cont).

```
> library(TxDb.Mmusculus.UCSC.mm9.knownGene)
> txdb <- TxDb.Mmusculus.UCSC.mm9.knownGene
> library(ggbio)
> p.gene <- autoplot(txdb, which = wh)
> p.gene
```



Genomic features track(cont).

```
> p.gene.c <- autoplot(txdb, which = wh,
+                      gap.geom = "chevron")
> p.gene.c
```



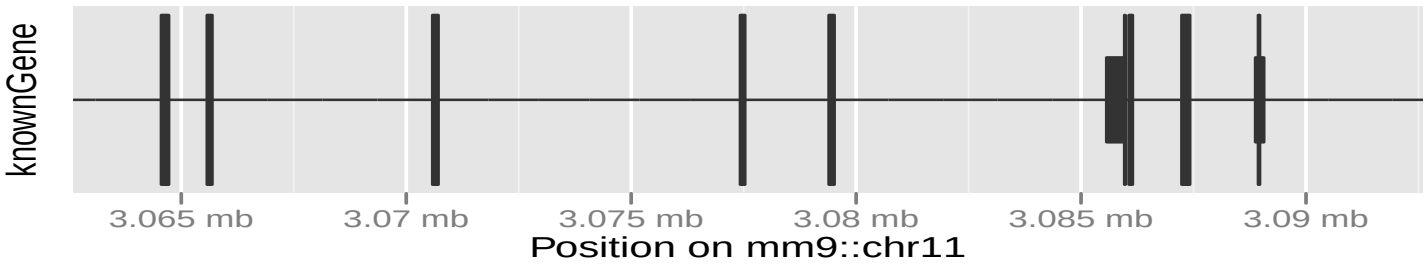
Your turn

Task

- Try to use stat = 'reduce' to see what happened.

Genomic features track(cont).

```
> p.gene.r <- autoplot(txdb, which = wh,
+                      stat = "reduce")
> library(grid)
> print(p.gene.r, vp = viewport(height = 0.3, width = 1))
```



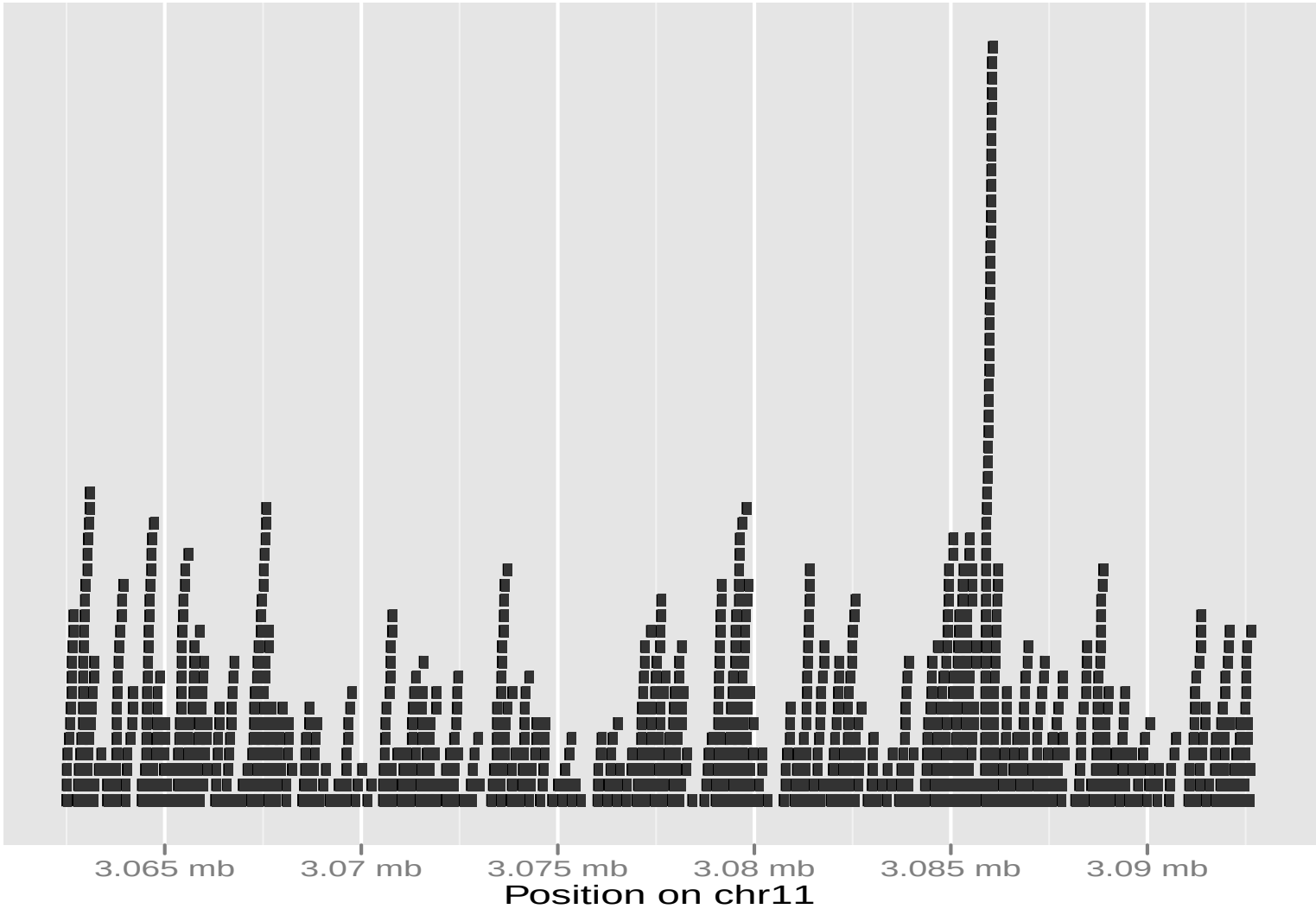
Get the example ChIP-seq data.

```
> ## shorter name
> cstest.s <- stack(cstest)
> cstest.s <- resize(cstest.s, width = 200)
> chipseq <- subsetByOverlaps(cstest.s, wh)
> head(chipseq)

GRanges with 6 ranges and 1 elementMetadata col:
      seqnames              ranges strand | sample
         <Rle>           <IRanges>  <Rle> |  <Rle>
  [1]    chr11 [3062550, 3062749]      + |   ctcf
  [2]    chr11 [3062594, 3062793]      + |   ctcf
  [3]    chr11 [3062732, 3062931]      + |   ctcf
  [4]    chr11 [3062768, 3062967]      + |   ctcf
  [5]    chr11 [3062839, 3063038]      + |   ctcf
  [6]    chr11 [3062844, 3063043]      + |   ctcf
  ---
  seqlengths:
          chr1         chr2         chr3 ...  chrY_random chrUn_random
     197195432    181748087    159599783 ...     58682461      5900358
```

Default geom for GRanges object

```
> autoplot(chipseq)
```

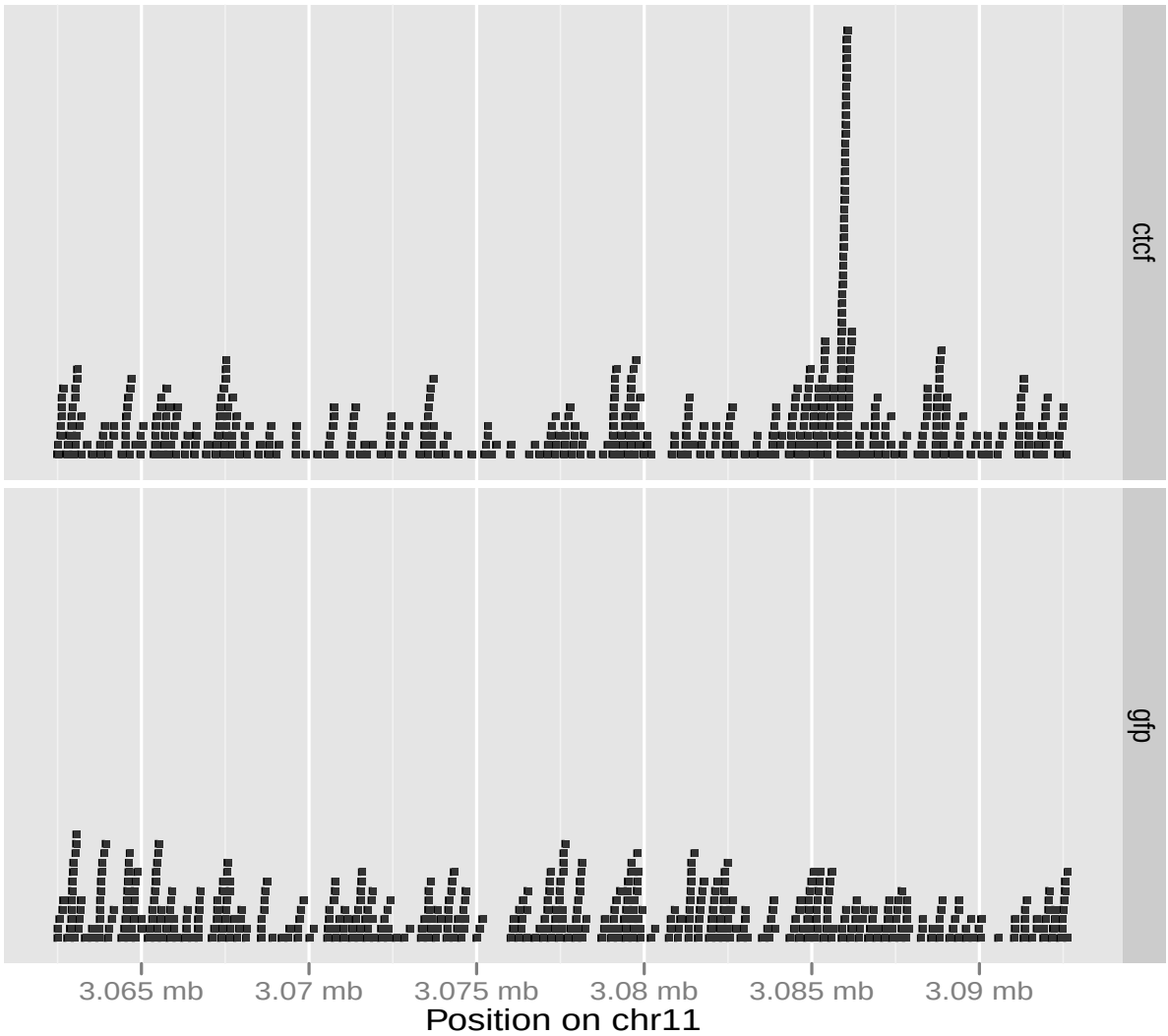


Understand GoG - Facets

- Default is always faceted by chromosome names by column. `.s̃eqnames`
- You can facet the plots by other factors such as samples. `sample s̃eqnames`

```
> autoplot(chipseq, facets = sample ~ .)
> ## equivalent to
> ## autoplot(gr) +  facet_grid(sample ~ .)
```

Facets by sample



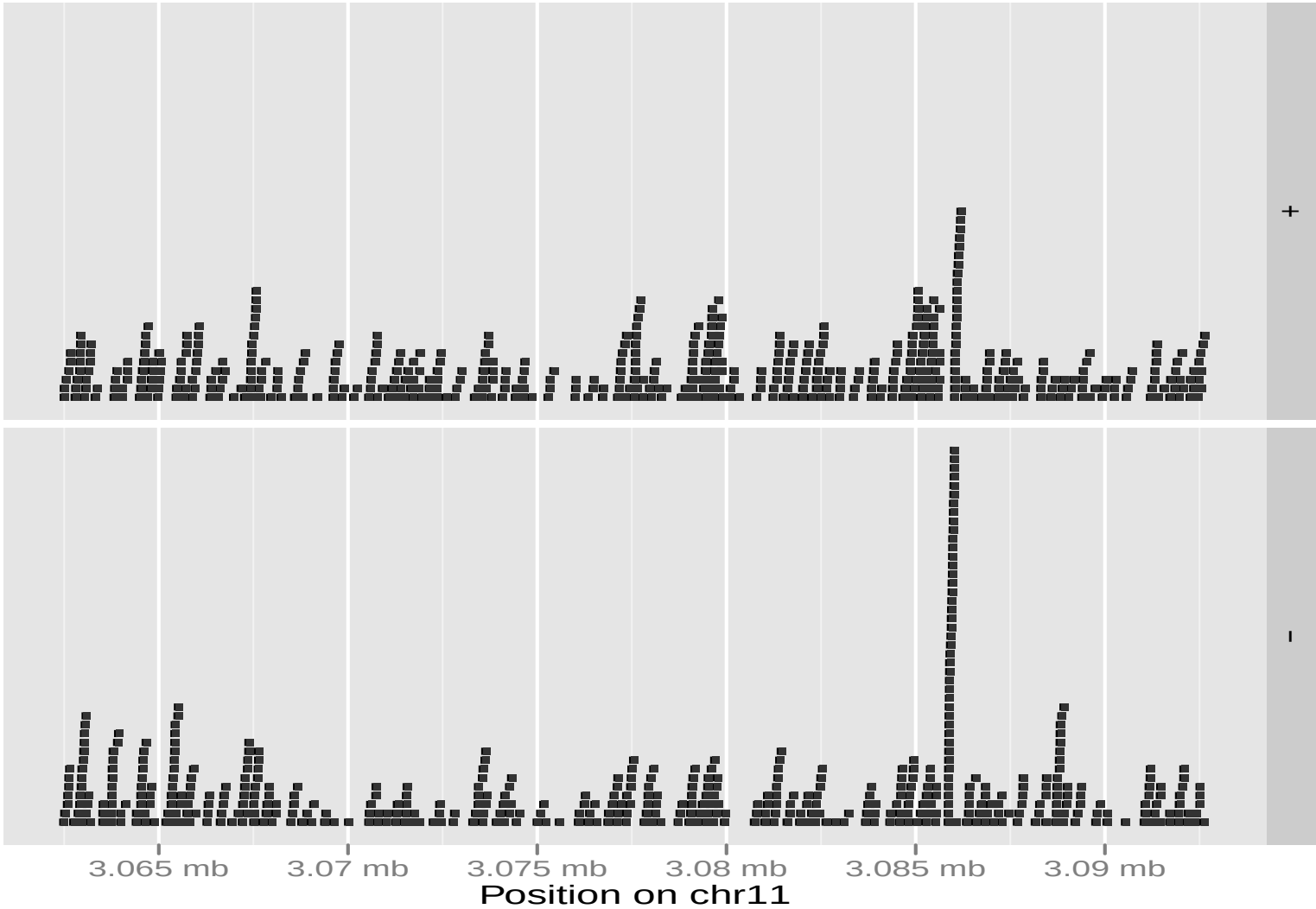
Your turn

Task

- Faceted by strand and indicated as the row.

Solution

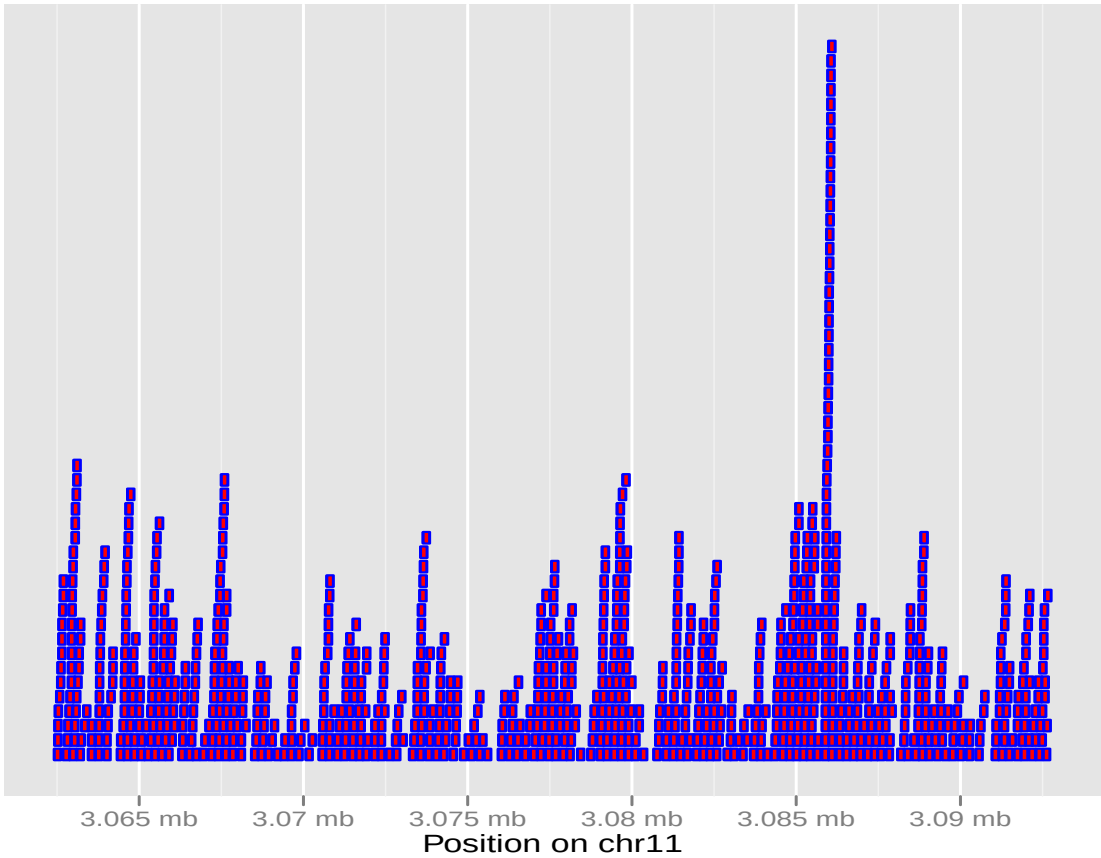
```
> autoplot(chipseq, facets = strand ~ .)
```



Understand GoG - Fixed aesthetics

- Fixed color/fill/size.., just passed in `autoplot`.

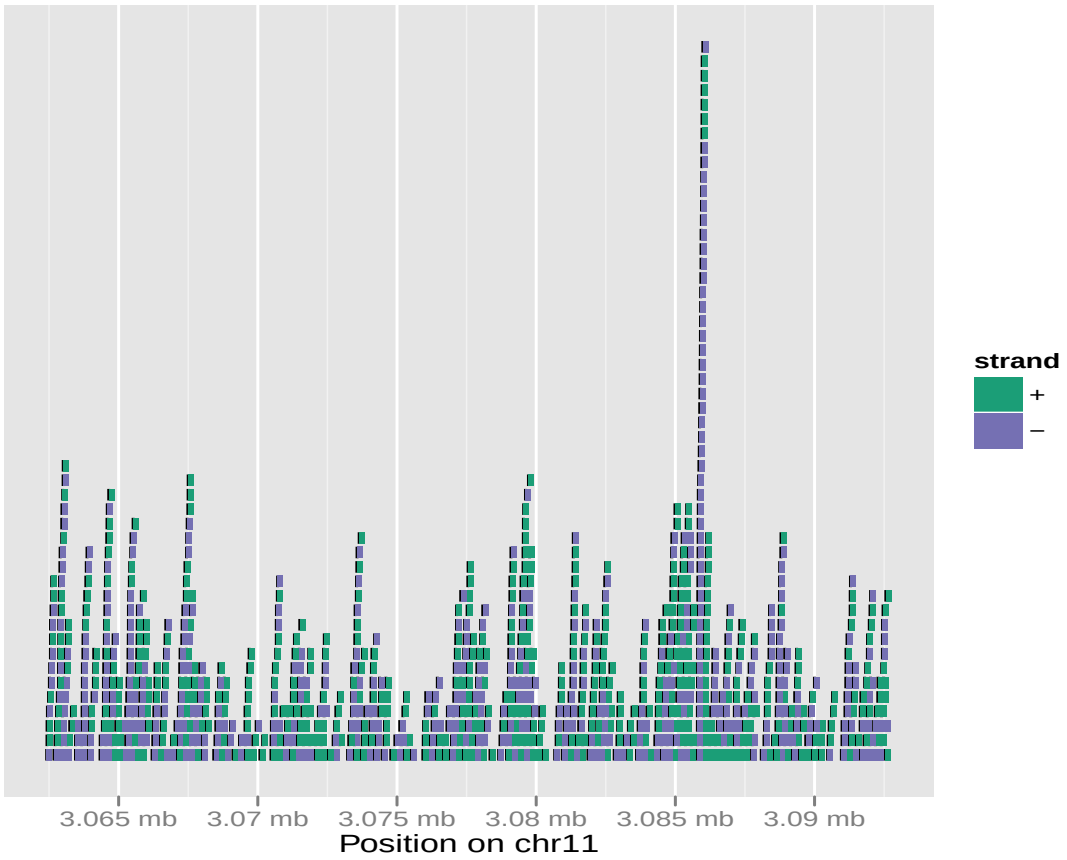
```
> autoplot(chipseq, color = "blue", fill = "red")
```



Understand GoG - Aesthetics mapping

- To map variables into aesthetics, use function `aes()`.
- Variable inside `aes()`, don't quote.

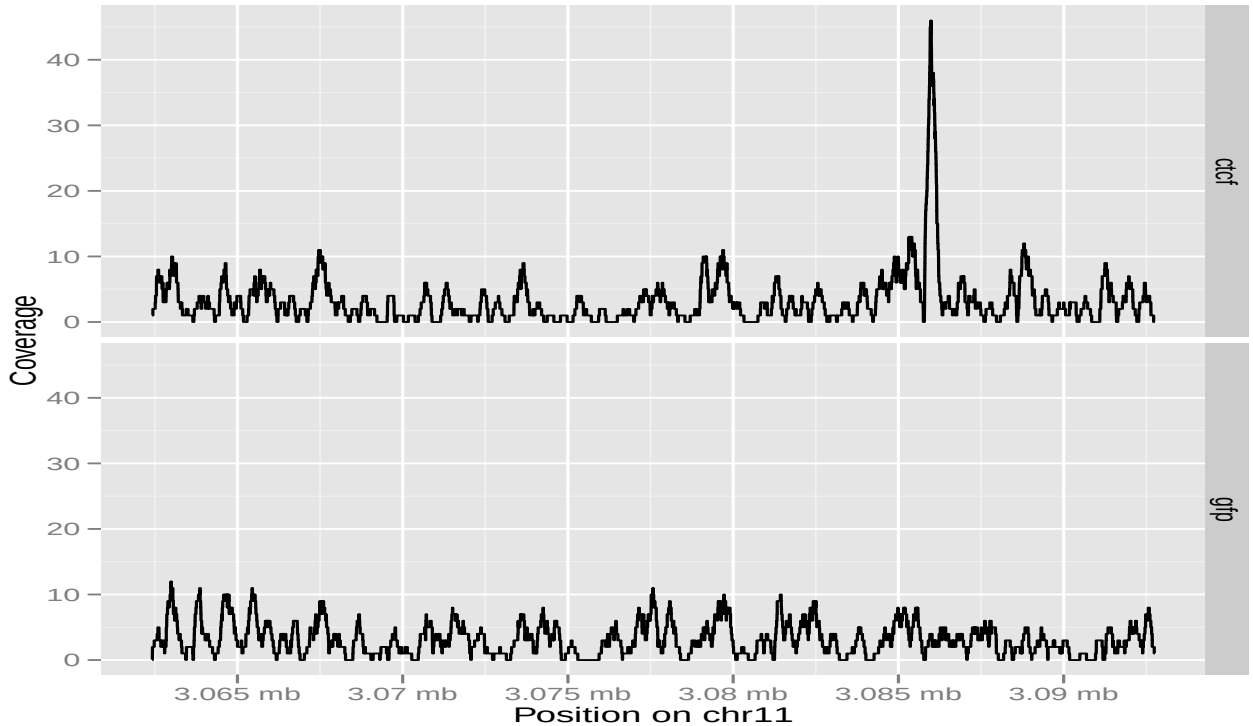
```
> autoplot(chipseq, aes(fill = strand))
```



Understand GoG - Statistical transformation(stat)

- Statistical transformation on the object to make new data model.
- key argument `stat`. Every object/class has different set of stats.
- Usually every stat has a default geom, for example, coverage's default geom is 'line'

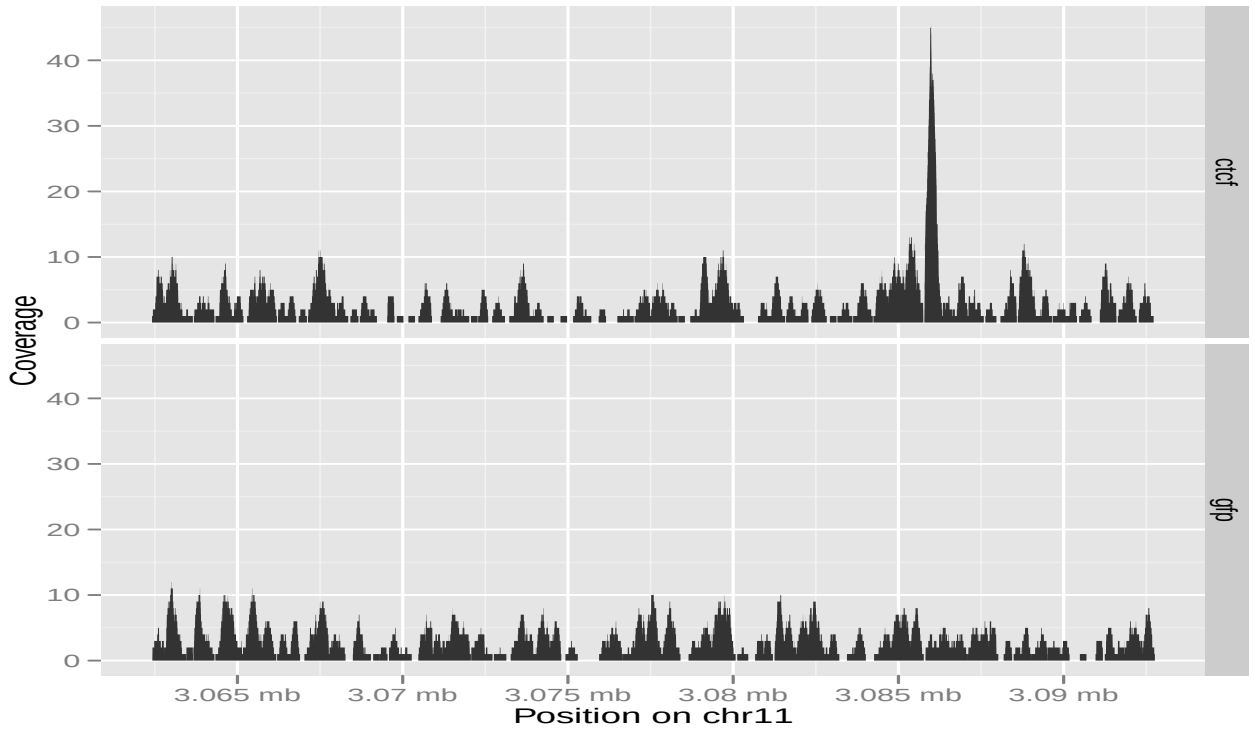
```
> autoplot(chipseq, stat = "coverage",
+          facets = sample ~ .)
```



Understand GoG - Geometric object(geom)

- Geometric object to represent the element(interval).
- key argument geom.
- Every geom has a default statistical transformation(stat).

```
> autoplot(chipseq, stat = "coverage",
+          geom = "area", facets = sample ~ .)
```



How to check the geom/stat for GRanges.

- Usually can check help manuals for particular object.
- For `GRanges`, following code can show you something defined within *ggbio*.
- When geom is other than those defined in *ggbio*, it is treated as a data.frame.

```
> ggbio:::.ggbio.geom

[1] "rect"       "chevron"    "alignment" "arrowrect" "arrow"
[7] "arch"       "bar"

> ggbio:::.ggbio.stat

[1] "identity"  "coverage"  "stepping"  "aggregate" "table"
[7] "mismatch"  "reduce"
```

Your turn

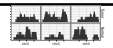
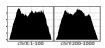
Task

- Try different sets of geom and stats.
- Save the faceted coverage plot for future use.

Extended grammar of graphics table I

Comp	name	usage	icon
geom	geom_rect	rectangle	
	geom_segment	segment	
	geom_chevron	chevron	
	geom_arrow	arrow	
	geom_arch	arches	
	geom_bar	bar	
	geom_alignment	alignment (gene)	
stat	stat_coverage	coverage (of reads)	
	stat_mismatch	mismatch pileup for alignments	
	stat_aggregate	aggregate in sliding window	
	stat_stepping	avoid overplotting	
	stat_gene	consider gene structure	
	stat_table	tabulate ranges	
	stat_identity	no change	

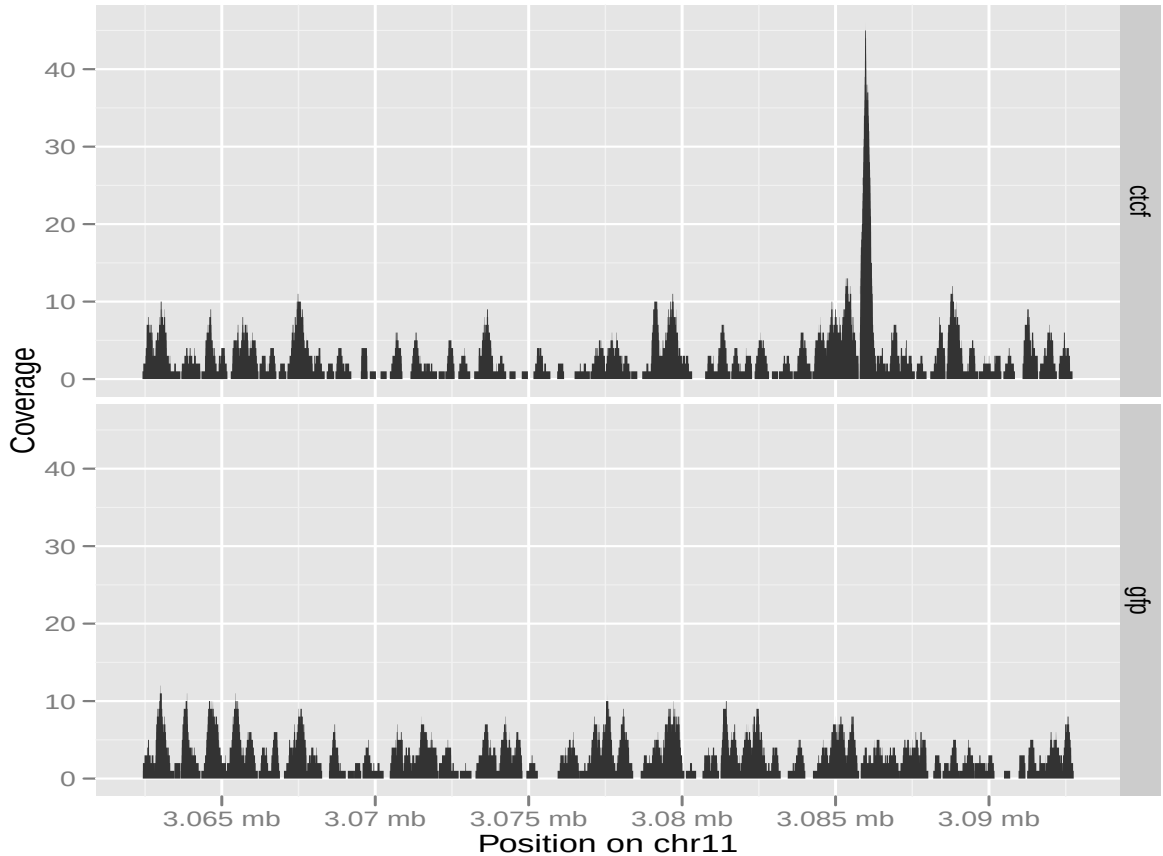
Extended grammar of graphics table II

coord	linear	ggplot2 linear but facet by chromosome	
	genome	put everything on genomic coordinates	
	truncate gaps	compact view by shrinking gaps	
layout	track	stacked tracks	
	karyogram	karyogram display	
	circle	circular	
faceting	formula	facet by formula	
	ranges	facet by ranges	
scale	not extended	*ggplot2*default	

Extended grammar of graphics table III

Save the coverage plot

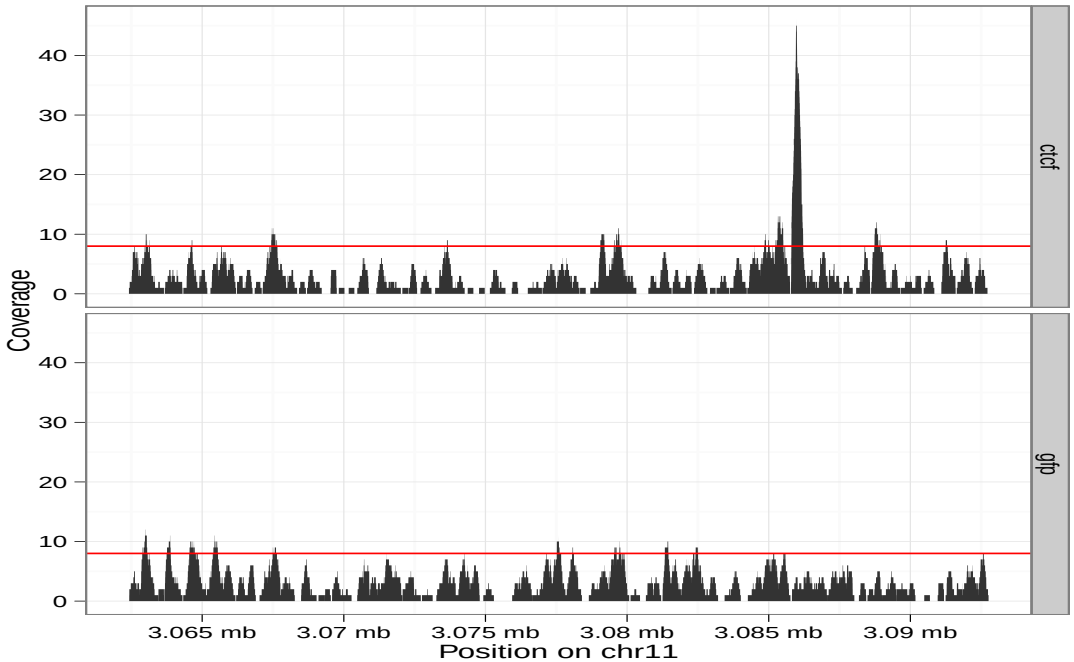
```
> p.cov <- autoplot(chipseq, stat = "coverage",
+                   facets = sample ~ ., geom = "area")
> p.cov
```



Additive ability

- Use '+" method to add on more layers or revise an existing graphic.

```
> p.cov + theme_bw() +
+   geom_hline(yintercept = 8, color = "red")
```

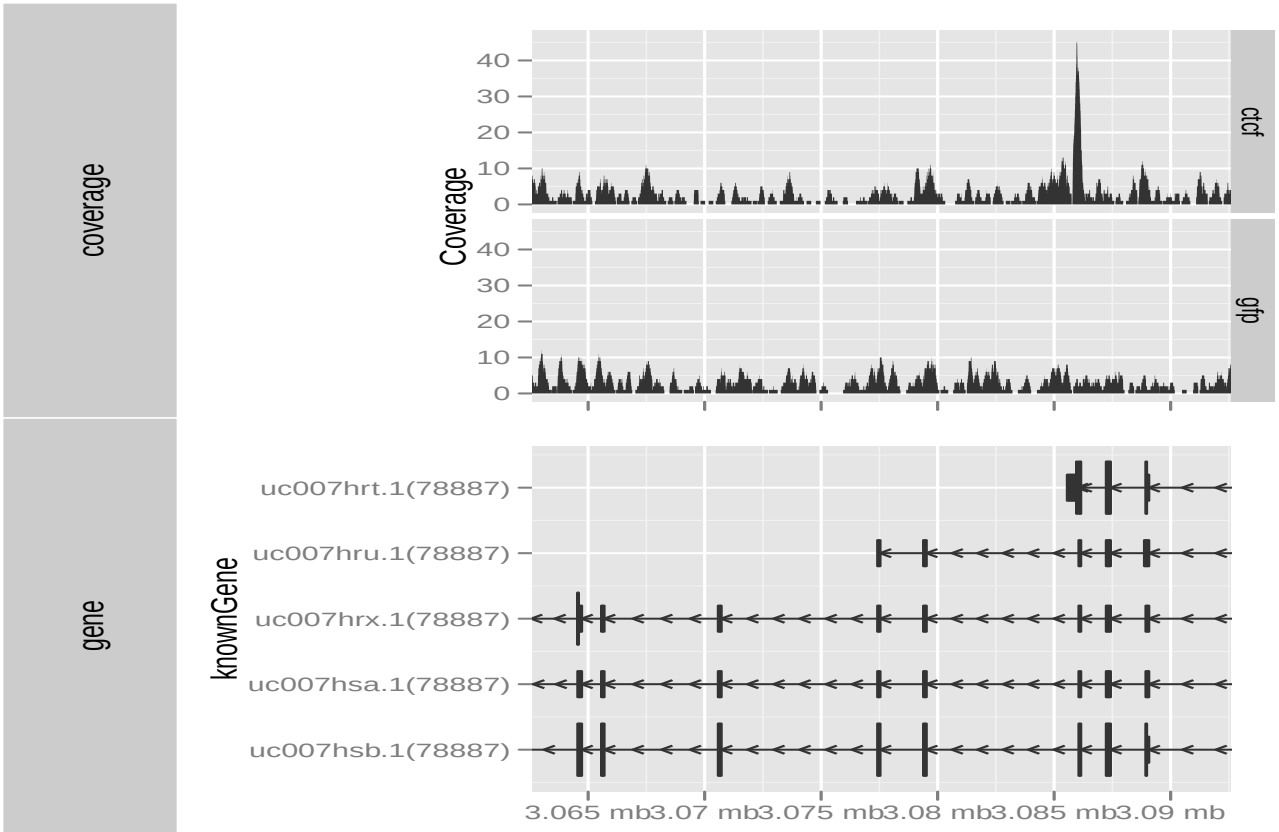


Tracks

- Post-graphic arrangement.
- Align any *ggplot2* or *ggbio* plot with numeric x.
- So you can construct graphics by *ggplot2* yourself, and *ggbio* will do the alignment for you.

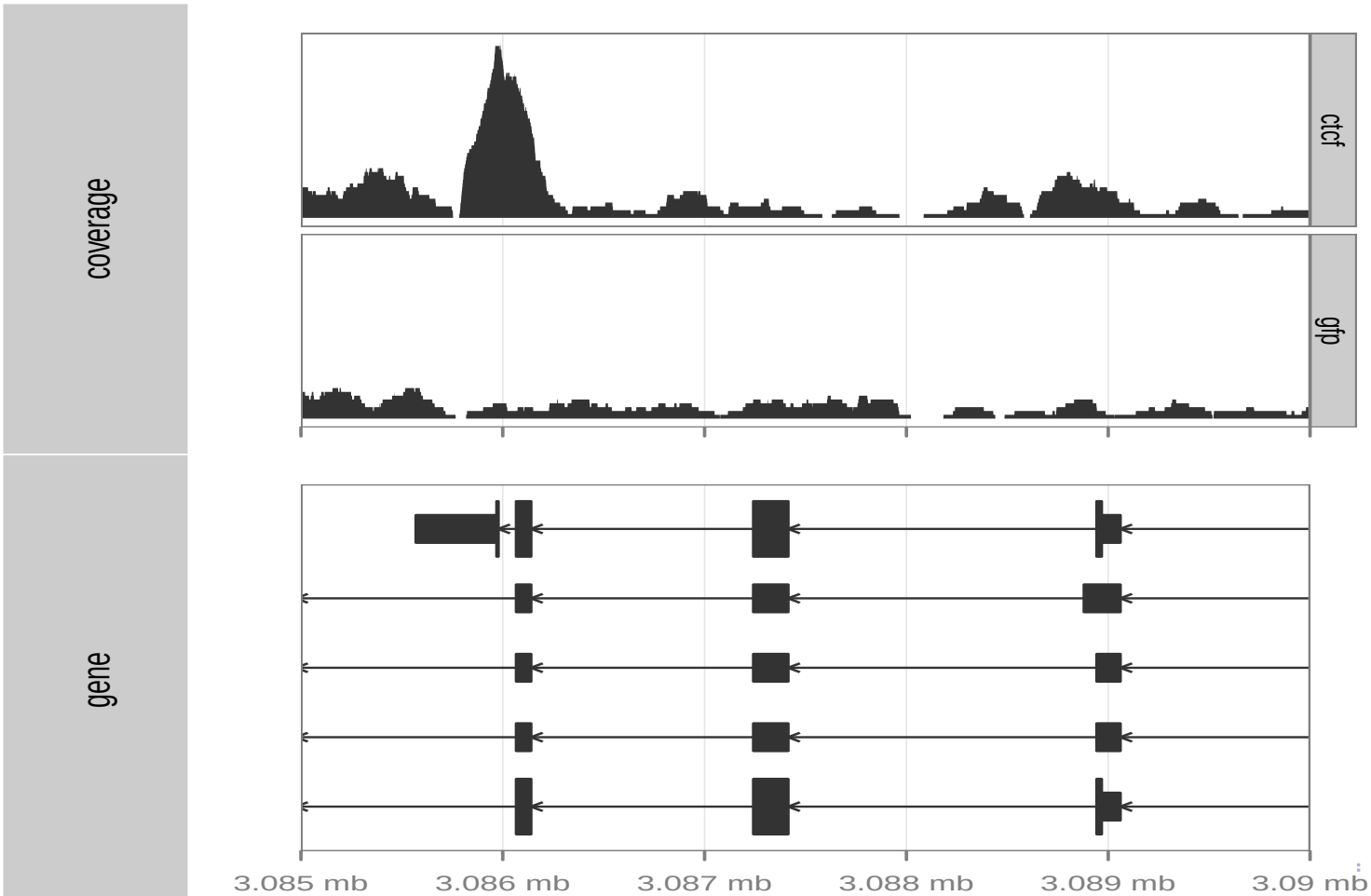
Building tracks

```
> tks <-tracks("coverage" = p.cov,
+              "gene" = p.gene, xlim = wh,
+               xlim.change = c(TRUE, TRUE))
> tks
```



Zoom to a small region

```
> tks <- tks + coord_cartesian(xlim  = c(3.085e6, 3.09e6)) +
+   theme_alignment()
> tks
```

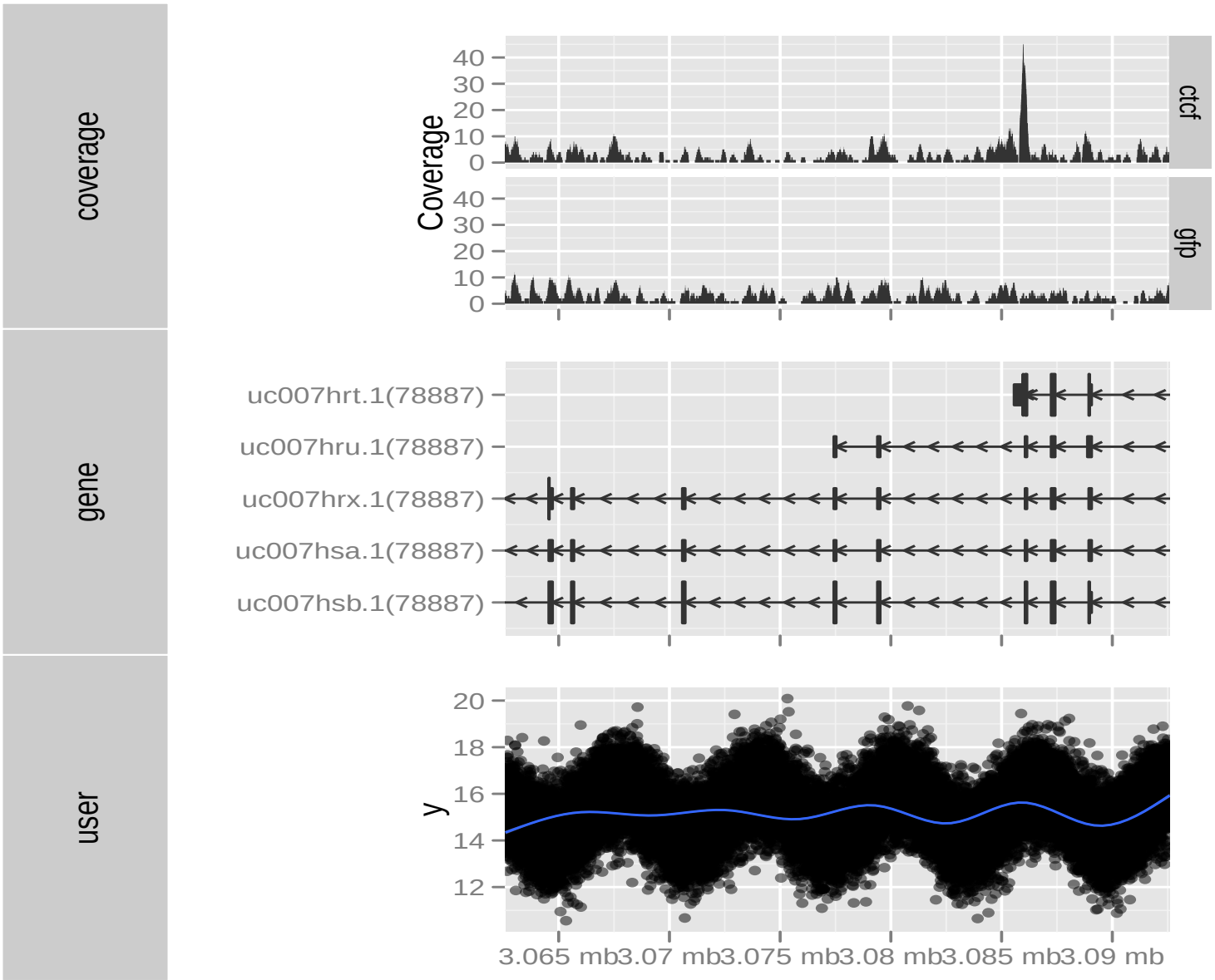


Plugin your own graphics

- If you are a *ggplot2* guru, just create your own graphics.

```
> simul <- seq(from = start(wh), to = end(wh), by = 1)
> df <- data.frame(x = simul, y = sin(simul/1000) +  log(simul
> p <- qplot(data = df, x = x, y = y, geom = "point", alpha =
> tracks("coverage" = p.cov,
+        "gene" = p.gene + ylab(""),
+        "user" = p + scale_x_sequnit("mb"),
+        xlim = wh)
```

plug-in your own data



Get ideogram

```
> library(rtracklayer)
> ## require network
> head(ucscGenomes())

       db species      date                                name
1    hg19   Human Feb. 2009 Genome Reference Consortium GRCh37
2    hg18   Human Mar. 2006                 NCBI Build 36.1
3    hg17   Human  May 2004                   NCBI Build 35
4    hg16   Human Jul. 2003                   NCBI Build 34
5 felCat4     Cat Dec. 2008                NHGRI catChrV17e
6 felCat3     Cat Mar. 2006          Broad Institute Release 3

> library(biovizBase)
> ## getIdeogram()
> obj <- getIdeogram("hg19")
> obj <- getIdeogram("hg19", cytoband = FALSE)
```

Your turn

Task

- Get mouse ideogram. keyword 'mm9'.
- Might be a little harder.

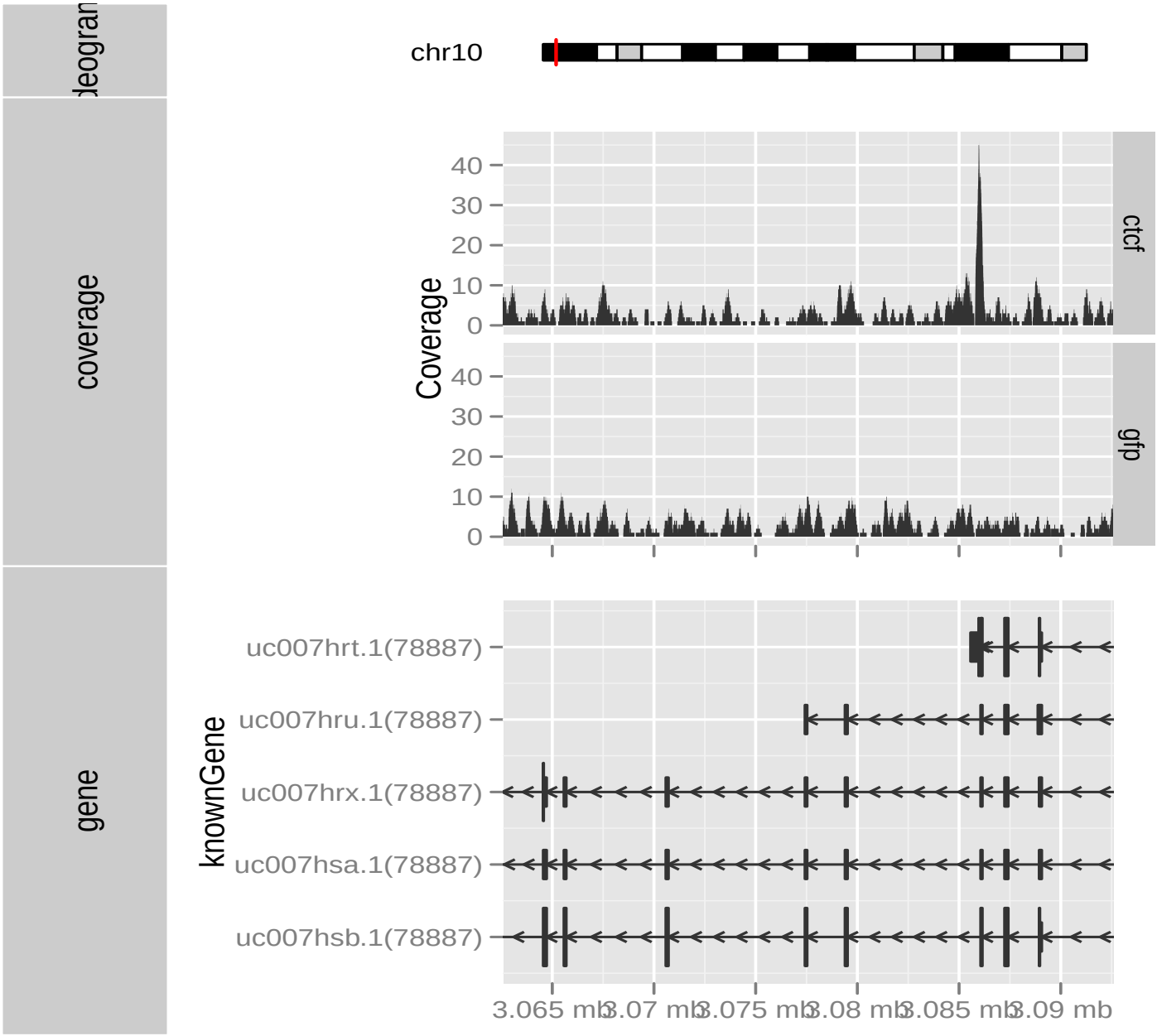
Add the ideogram cytoband.

```
> library(biovizBase)
> mm9 <- getIdeogram("mm9")
> cyto.def <- getOption("biovizBase")$cytobandColor
> cyto.new <- c(cyto.def, c(gpos33 = "grey80", gpos66 = "grey6
> p.ideo <- plotIdeogram(mm9, "chr10", zoom = c(start(wh),end(
+   scale_fill_manual(values = cyto.new)
> print(p.ideo)
```

Binding an unchanged track.

- Unchanged track never zoom in/out or synchronize.

```
> tks <- tracks("ideogram" = p.ideo, "coverage" = p.cov,
+               "gene" = p.gene,
+               xlim = wh,
+               xlim.change = c(FALSE, TRUE, TRUE),
+               heights = c(1, 5, 5))
> tks
```

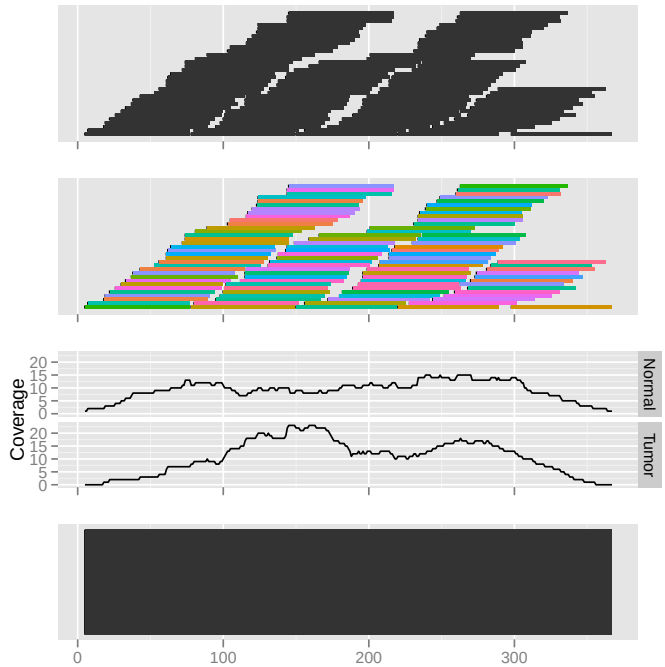

ideogram
coverage
gene
chr10
Coverage
40
30
20
10
0
ctcf
gfp
knownGene
uc007hrt.1(78887)
uc007hru.1(78887)
uc007hrx.1(78887)
uc007hsa.1(78887)
uc007hsb.1(78887)
3.065 mb
3.07 mb
3.075 mb
3.08 mb
3.085 mb
3.09 mb

Object supported(overview).

GRanges				

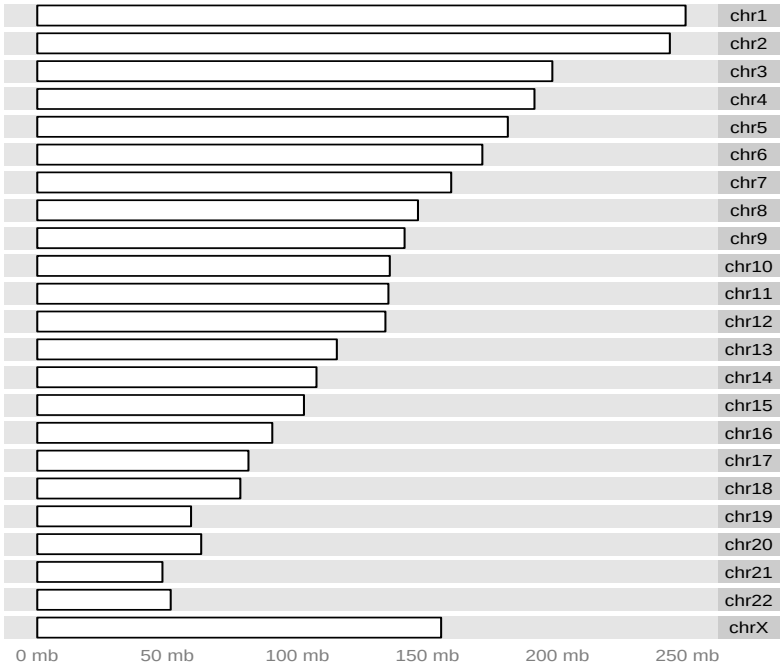
Object supported(overview).

GRanges	*IRanges*			



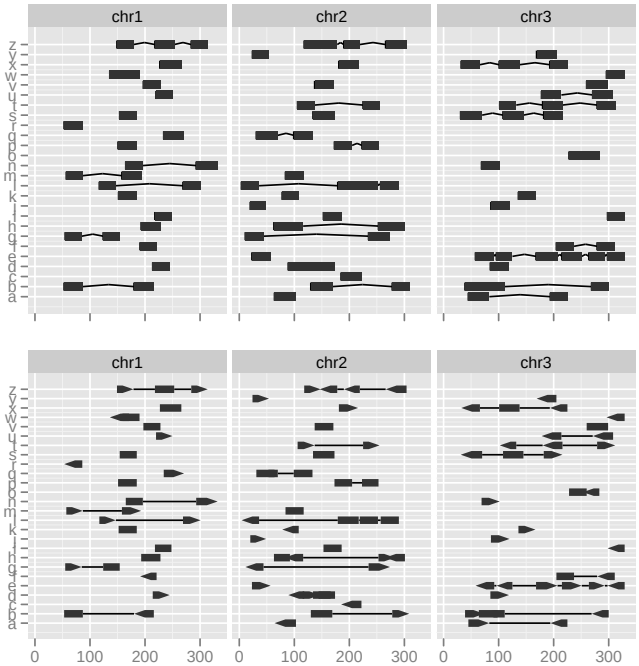
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*		



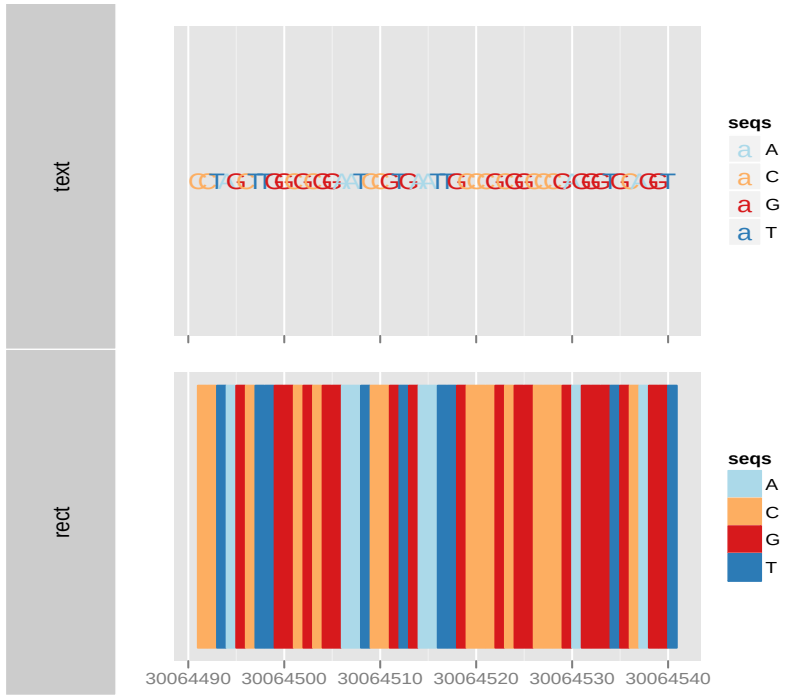
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	



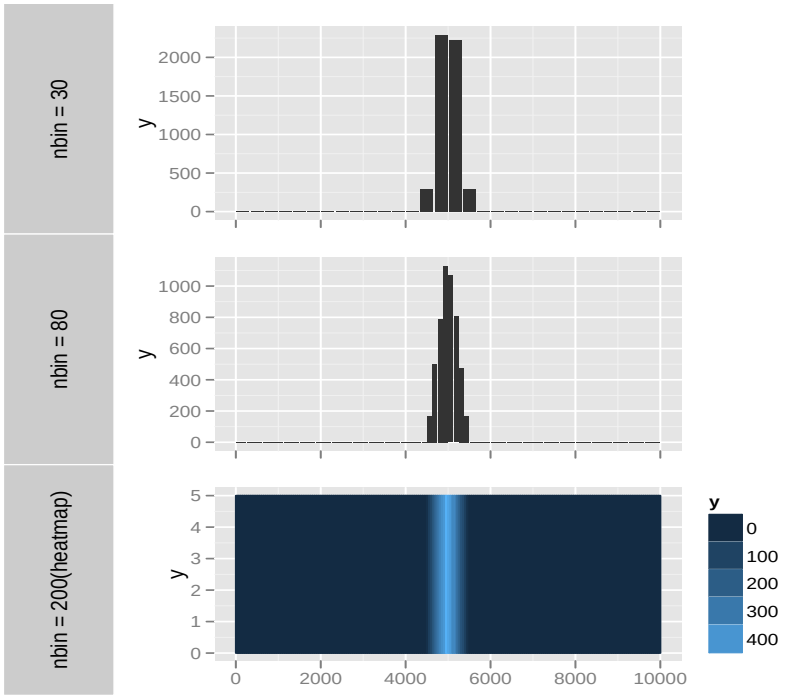
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*



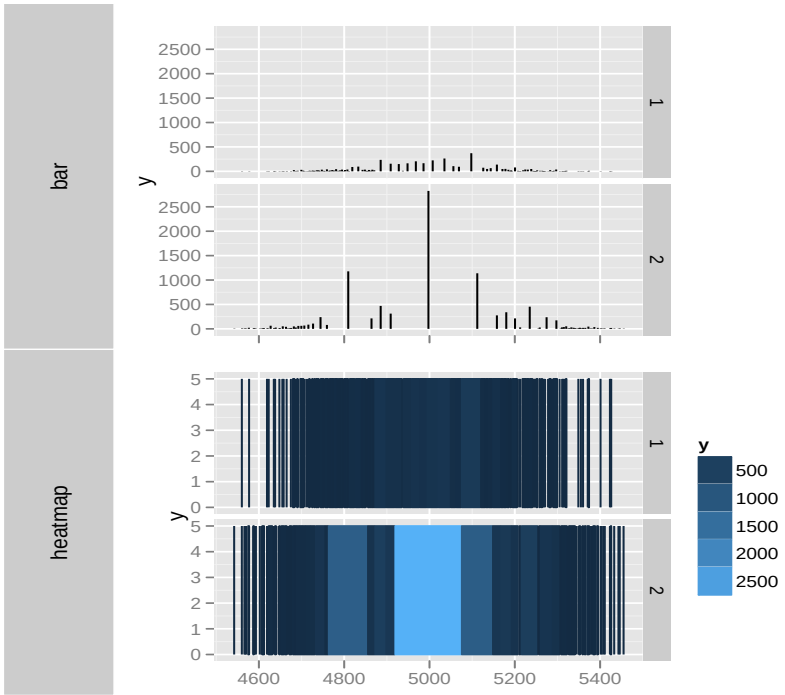
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle				



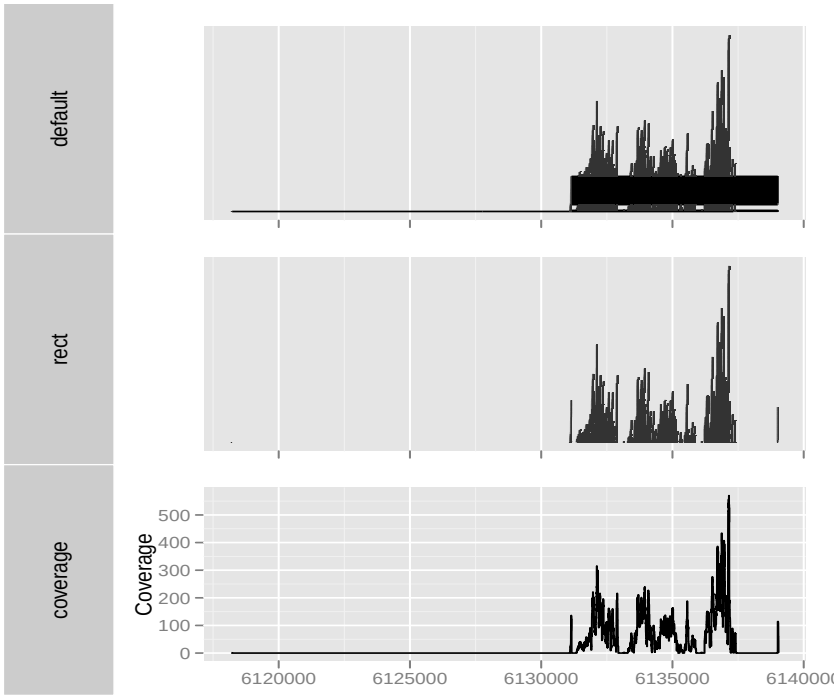
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*			



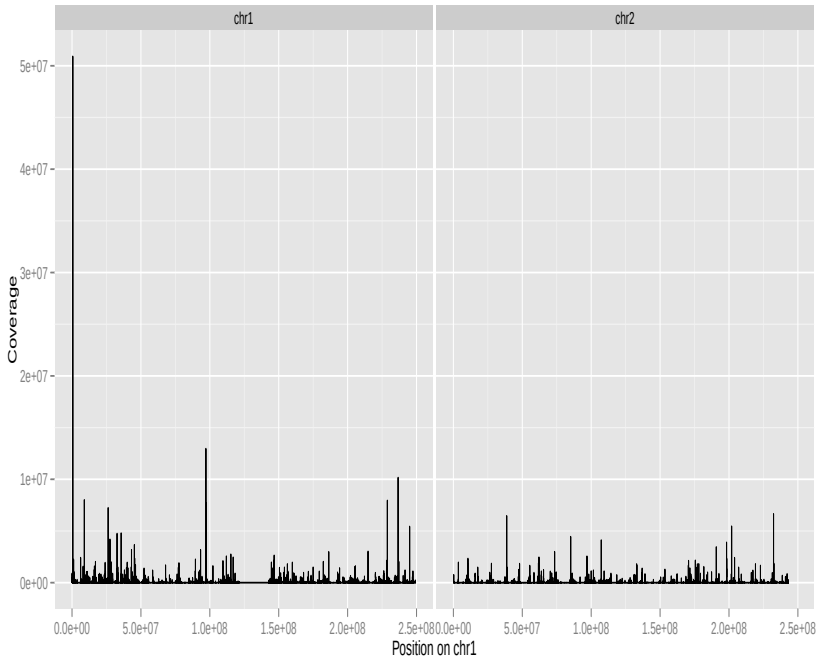
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*		



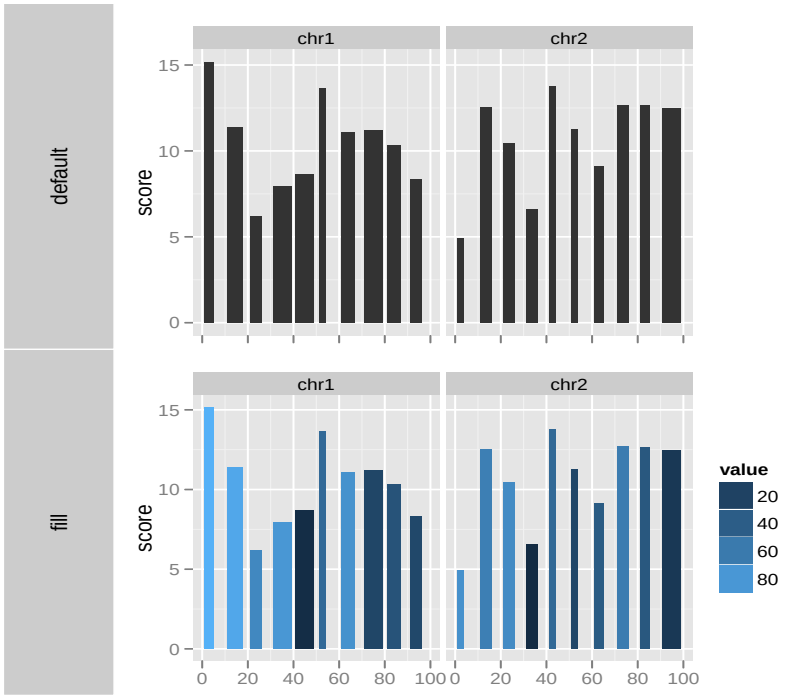
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	



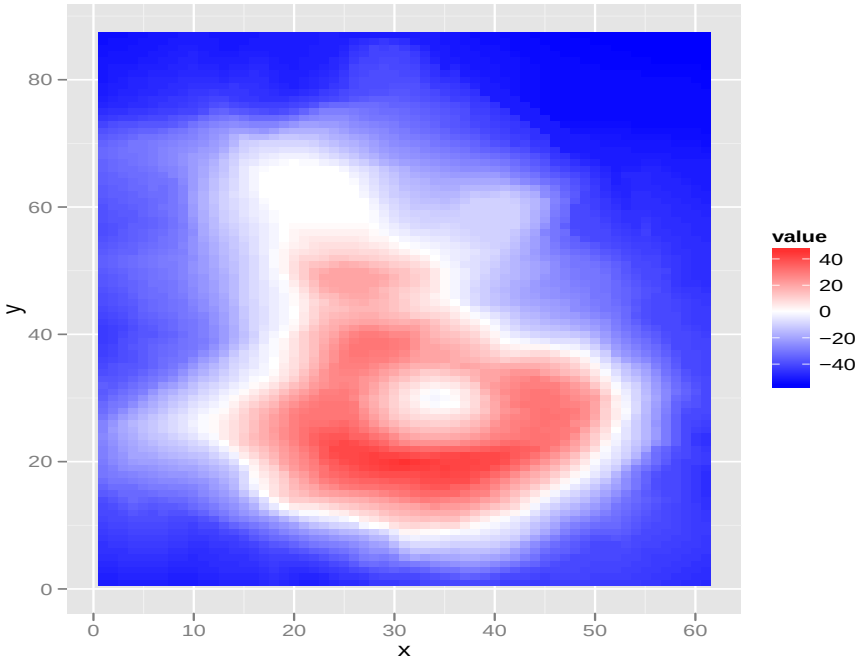
Object supported(overview).

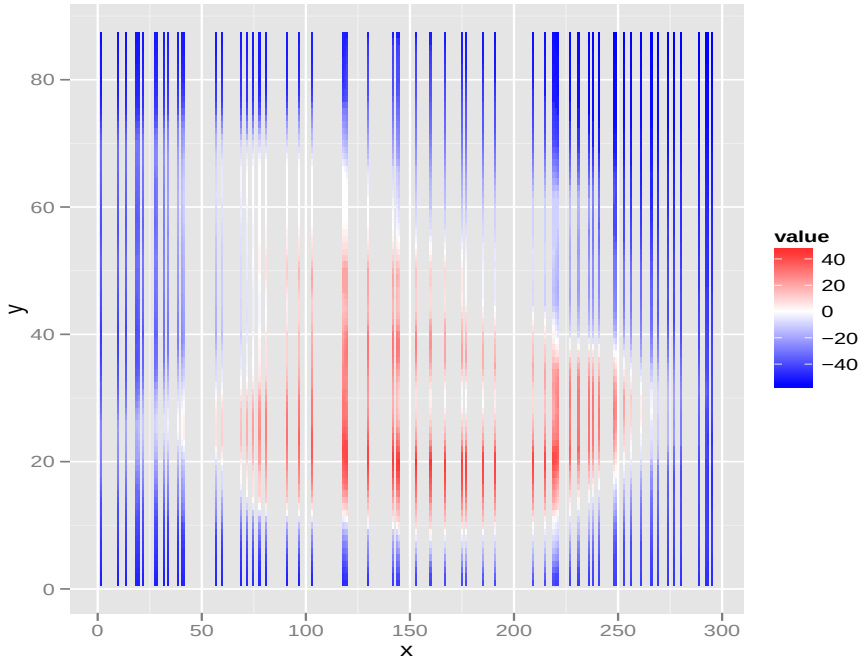
GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	*character*



Object supported(overview).

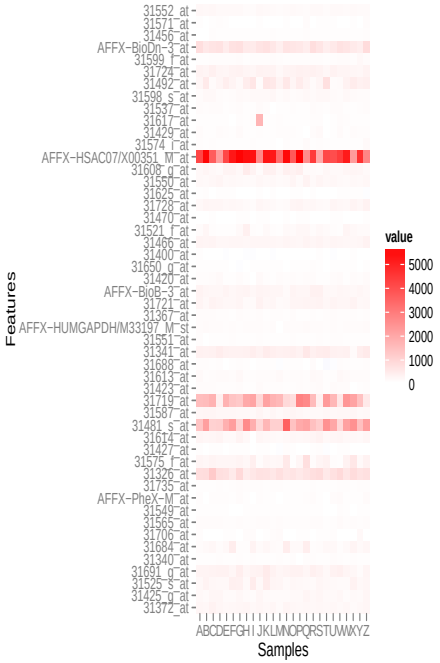
GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	*character*
matrix				

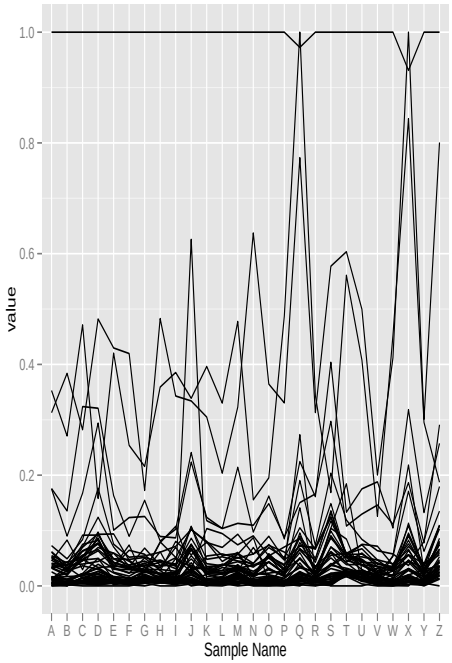


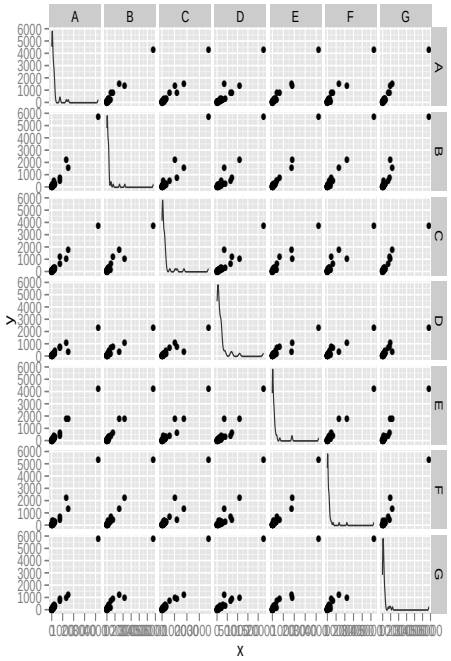


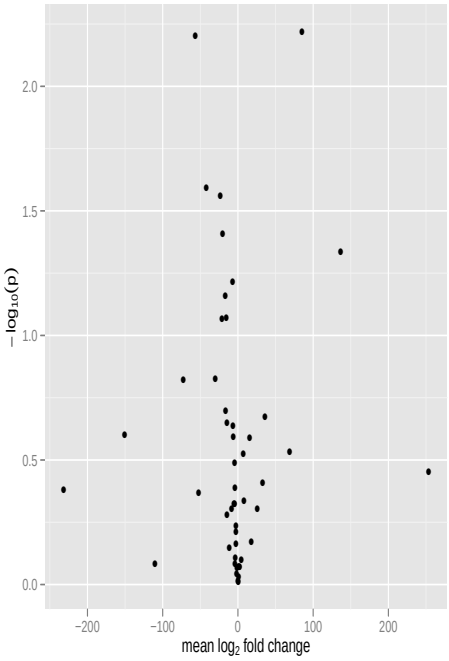
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	*character*
matrix	*ExpressionSet*			



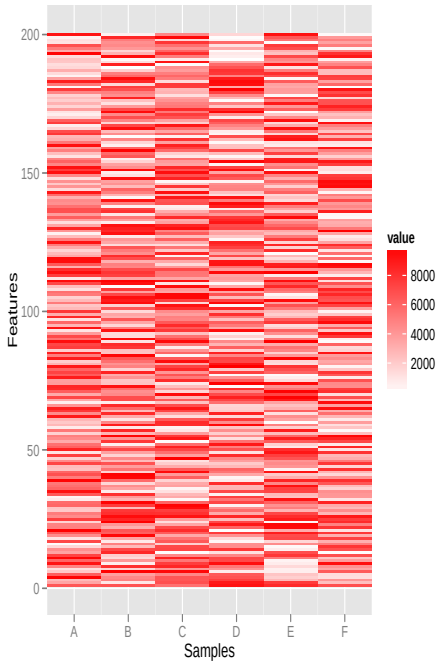


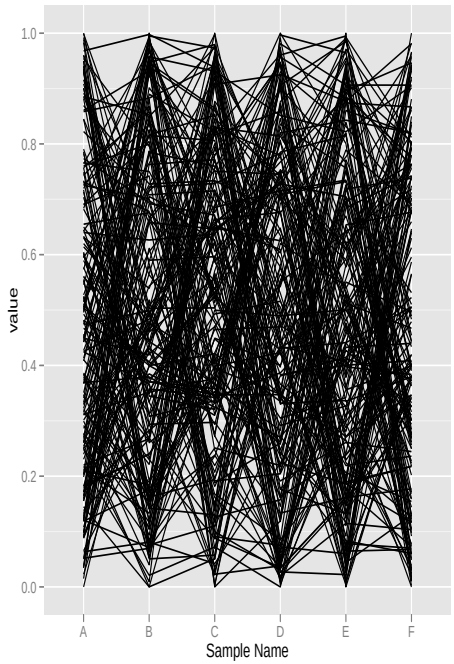


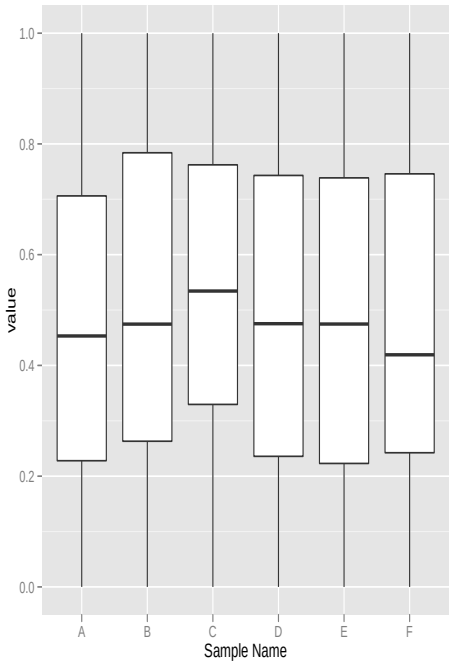


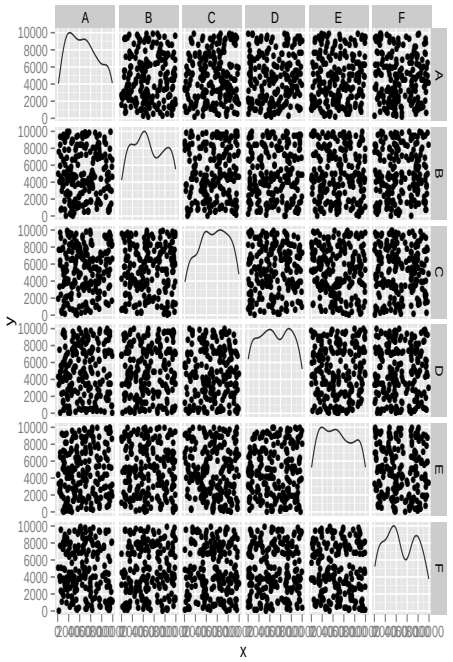
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	*character*
matrix	*ExpressionSet*	*SummarizedExperiment*		



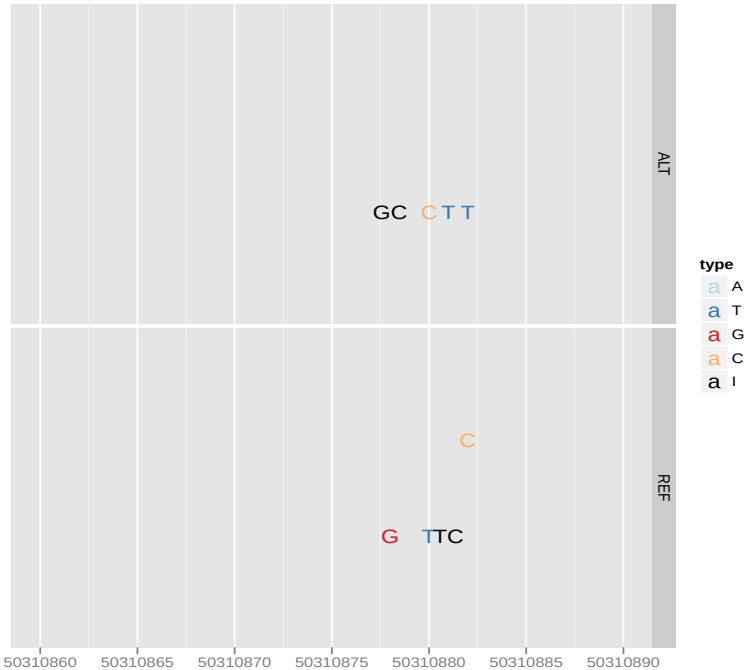






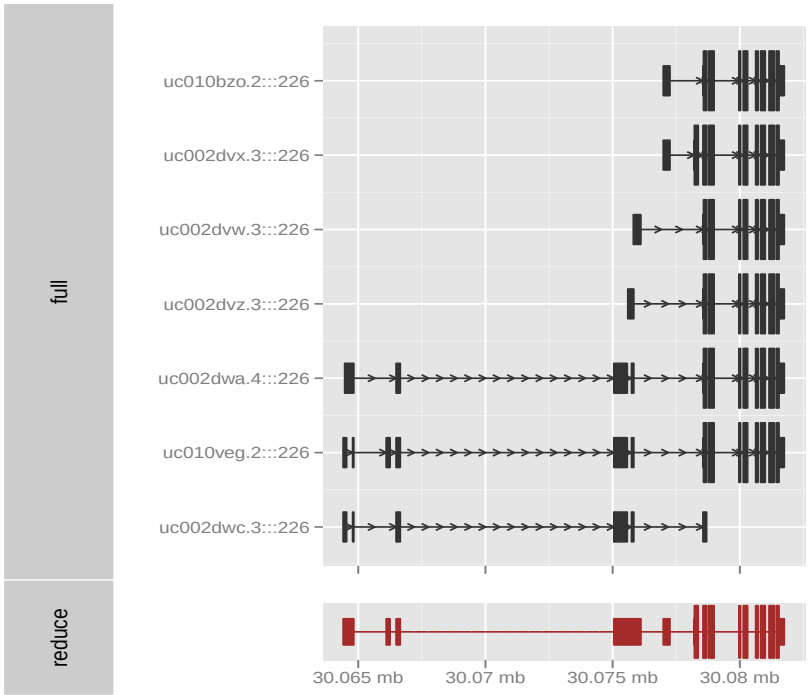
Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	*character*
matrix	*ExpressionSet*	*SummarizedExperiment*	*VCF*	



Object supported(overview).

GRanges	*IRanges*	*Seqinfo*	*GRangesList*	*BSgenome*
Rle	*RleList*	*GappedAlignment*	*BamFile*	*character*
matrix	*ExpressionSet*	*SummarizedExperiment*	*VCF*	*TranscriptDb*



Outline

Task

- Create point/scale/text/segment track in layout circle.
- Show rearrangement as links in the inner circle.

Rules keep in mind.

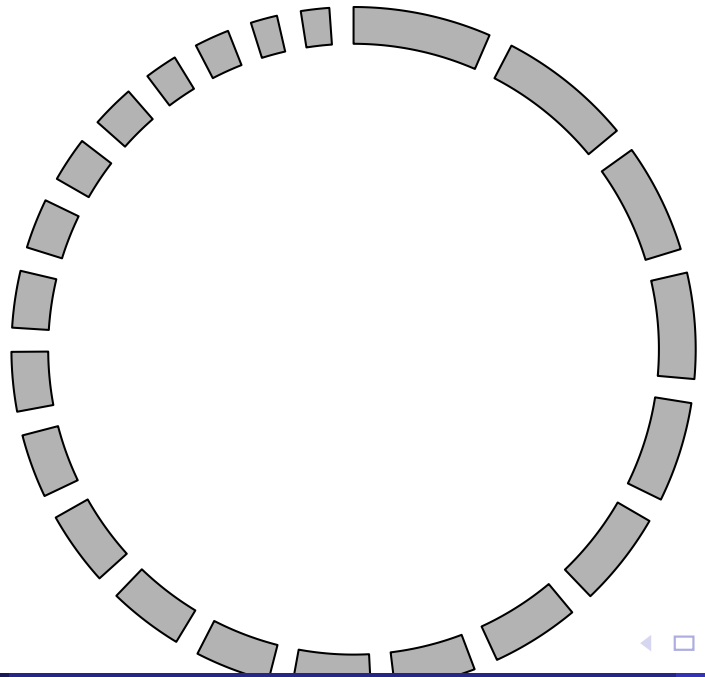
- All data must have the same seqlevels and seqlengths definition.
- Don't forget to attach data in `layout_circle`.

Processing the data

```
> crc1 <- system.file("extdata", "crc1-missense.csv", package = "biovizBase")
> crc1 <- read.csv(crc1)
> library(GenomicRanges)
> mut.gr <- with(crc1,GRanges(Chromosome, IRanges(Start_position, End_position),
+                             strand = Strand))
> values(mut.gr) <- subset(crc1, select = -c(Start_position, End_position, Chromosome))
> data("hg19Ideogram", package = "biovizBase")
> seqs <- seqlengths(hg19Ideogram)
> ## subset_chr
> chr.sub <- paste("chr", 1:22, sep = "")
> ## levels tweak
> seqlevels(mut.gr) <- c(chr.sub, "chrX")
> mut.gr <- keepSeqlevels(mut.gr, chr.sub)
> seqs.sub <- seqs[chr.sub]
> ## remove wrong position
> bidx <- end(mut.gr) <= seqs.sub[match(as.character(seqnames(mut.gr)),
+                names(seqs.sub))]
> mut.gr <- mut.gr[which(bidx)]
> ## assign_seqlengths
> seqlengths(mut.gr) <- seqs.sub
> ## reanme to shorter names
> new.names <- as.character(1:22)
> names(new.names) <- paste("chr", new.names, sep = "")
> mut.gr.new <- renameSeqlevels(mut.gr, new.names)
```

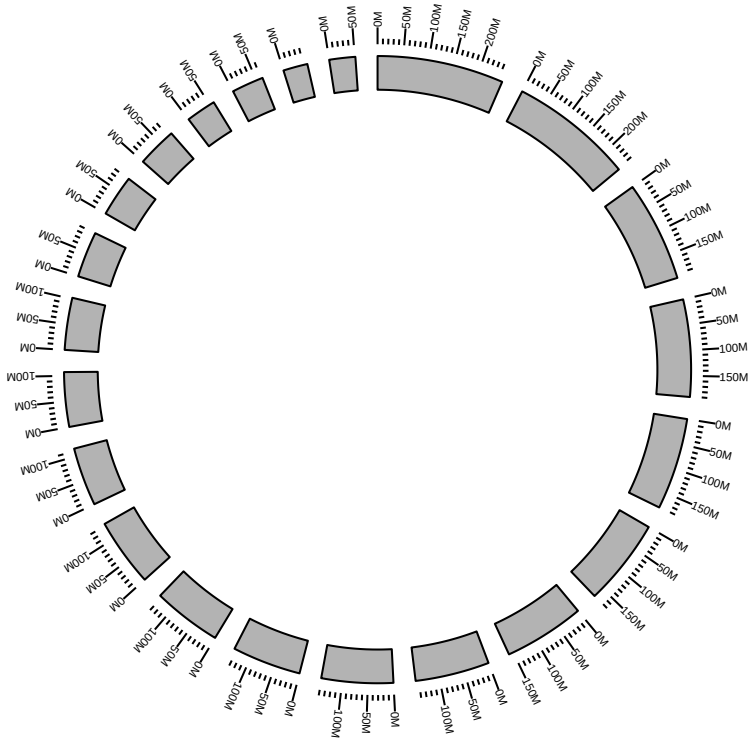
Create an ideogram track

```
> hg19Ideo <- hg19Ideogram
> hg19Ideo <- keepSeqlevels(hg19Ideogram, chr.sub)
> hg19Ideo <- renameSeqlevels(hg19Ideo, new.names)
> p <- ggplot() + layout_circle(hg19Ideo, geom = "ideo",
+           fill = "gray70", radius = 30, trackWidth = 4)
> p
```



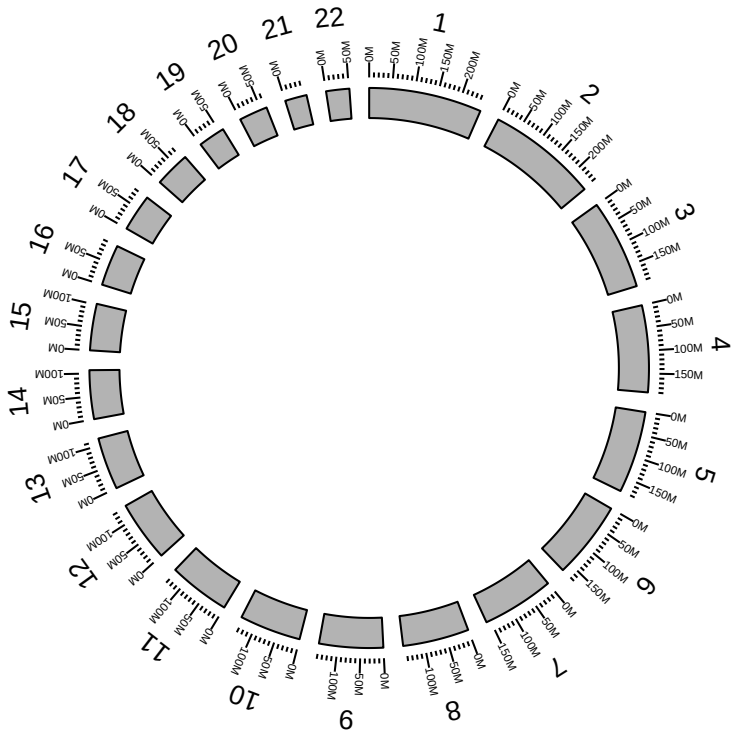
Create an scale track

```
> p <- p + layout_circle(hg19Ideo, geom = "scale",
+                        size = 2, radius = 35, trackWidth = 2)
> p
```



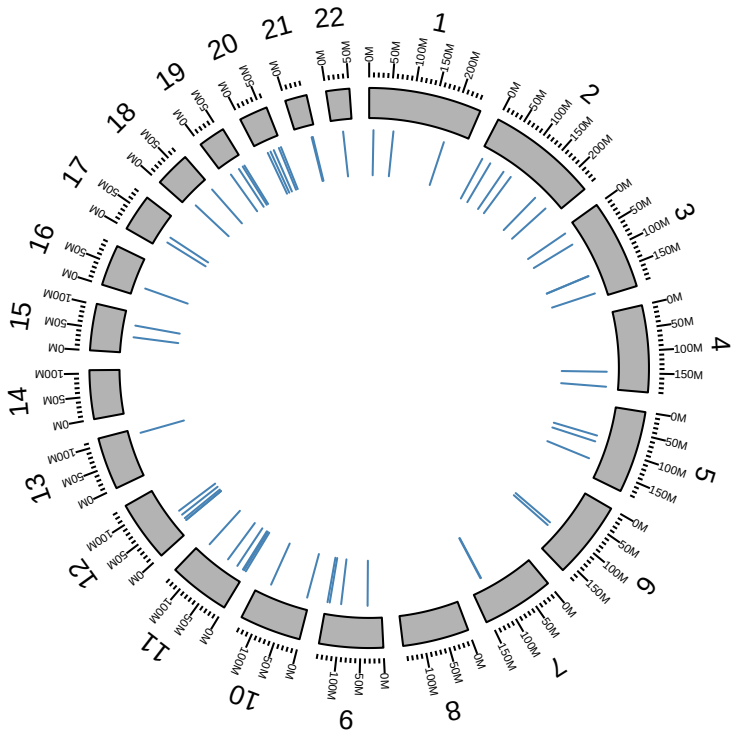
Create an text track

```
> p <- p + layout_circle(hg19Ideo, geom = "text",
+                        aes(label = seqnames), vjust = 0,
+                        radius = 38, trackWidth = 7)
> p
```



Create an mutation track

```
> p <- p + layout_circle(mut.gr, geom = "rect",
+                        color = "steelblue",
+                        radius = 23 ,trackWidth = 6)
> p
```

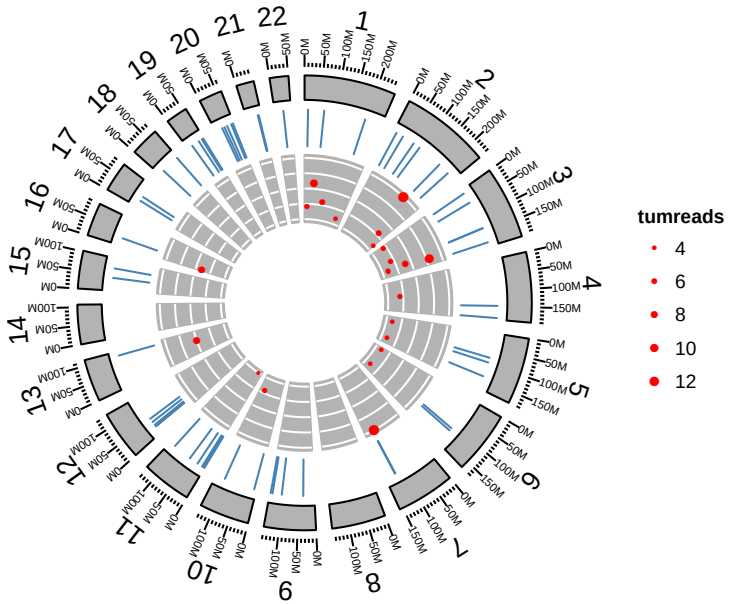


Create an mutation track

```
> rearr  <- read.csv(system.file("extdata",
+                                "crc-rearrangment.csv", package = "biovizBase"))
> ## start position
> gr1 <- with(rearr, GRanges(chr1, IRanges(pos1, width = 1)))
> ## end position
> gr2 <- with(rearr, GRanges(chr2, IRanges(pos2, width = 1)))
> ## add extra column
> nms <- colnames(rearr)
> .extra.nms <- setdiff(nms, c("chr1", "chr2", "pos1", "pos2"))
> values(gr1) <- rearr[,.extra.nms]
> ## remove out-of-limits data
> seqs <- as.character(seqnames(gr1))
> .mx <- seqlengths(hg19Ideo)[seqs]
> idx1 <- start(gr1) > .mx
> seqs <- as.character(seqnames(gr2))
> .mx <- seqlengths(hg19Ideo)[seqs]
> idx2 <- start(gr2) > .mx
> idx <- !idx1 & !idx2
> gr1 <- gr1[idx]
> seqlengths(gr1) <- seqlengths(hg19Ideo)
> gr2 <- gr2[idx]
> seqlengths(gr2) <- seqlengths(hg19Ideo)
> values(gr1)$to.gr <- gr2
> ## rename to gr
> gr <- gr1
> values(gr)$rearrangements <- ifelse(as.character(seqnames(gr))
+               == as.character(seqnames((values(gr)$to.gr))),
+              "intrachromosomal", "interchromosomal")
> gr.crc1 <- gr[values(gr)$individual == "CRC-1"]
```

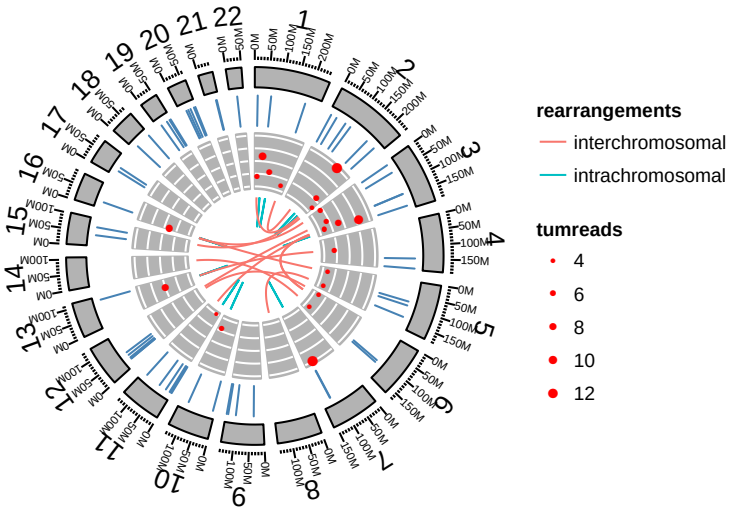
Create an point track

```
> p <- p + layout_circle(gr.crc1, geom = "point",
+           aes(y = score, size = tumreads), color = "red",
+           radius = 12 ,trackWidth = 10, grid = TRUE) +
+   scale_size(range = c(1, 2.5))
> p
```



Create a link track

```
> p <- p + layout_circle(gr.crc1, geom = "link",
+                        linked.to = "to.gr",
+                        aes(color = rearrangements),
+                        radius = 10 ,trackWidth = 1)
> p
```



Outline

Future study

- Support more core data model in Bioconductor.
- More elegant theme for tracks.
- Keep improving with new ggplot2 development
- More powerful tracks function, may accept lattice graphics.

Issue/bug report

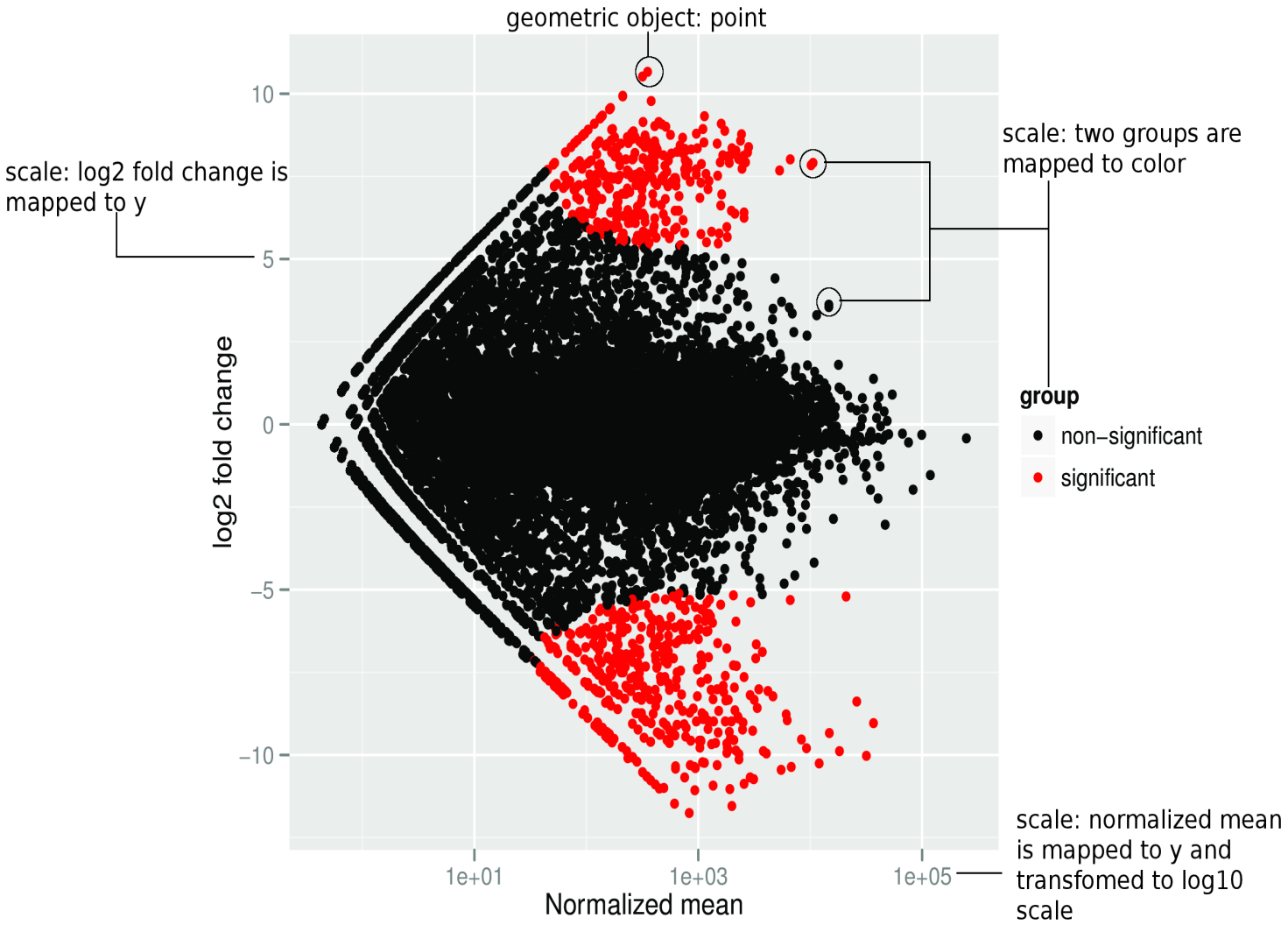
- File an issue report at github(recommended).
- Send me en email.
- Send on bioconductor mailing list.

Acknowledgment

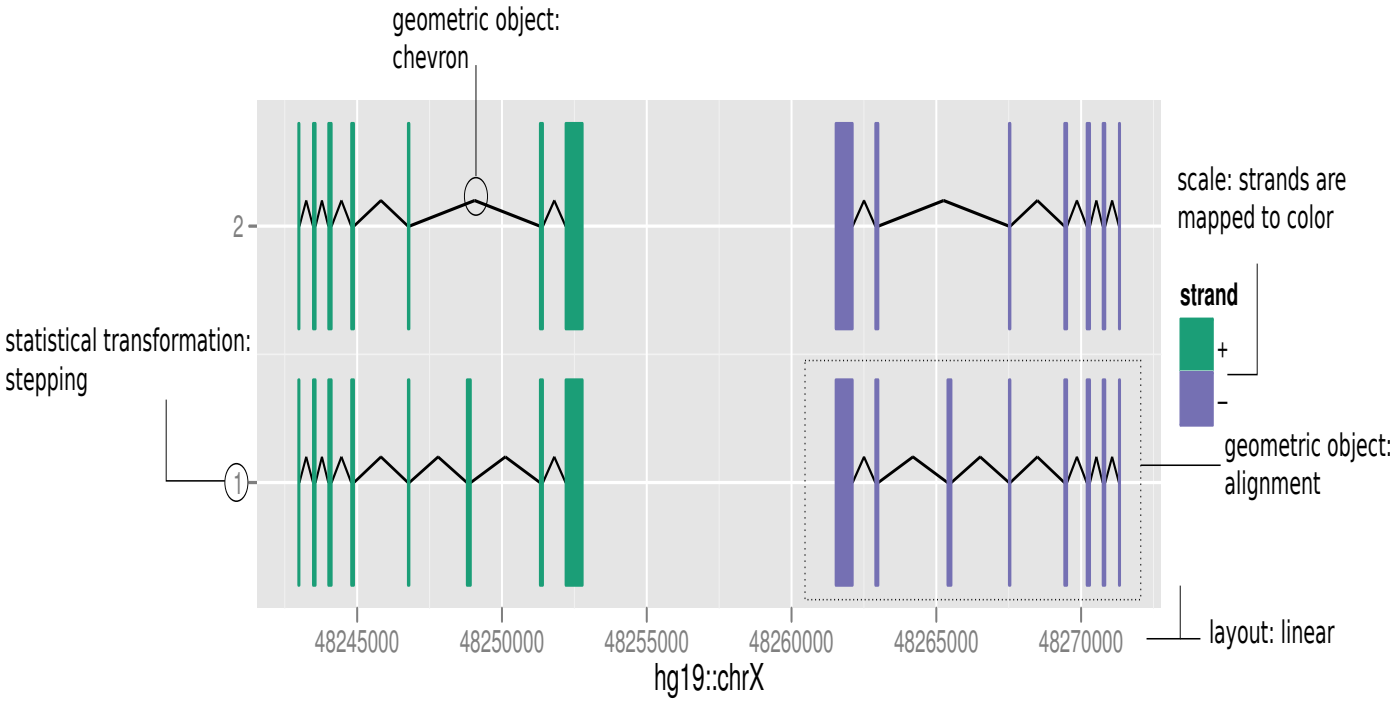
- Michael Lawrence, Dianne Cook
- Genentech

Thank you !!!

What is grammar of graphics(GoG)



GoG in Genomic World.



Pipeline in ggbio