The Pviz User Guide

Renan Sauteraud*

April 25, 2023

Contents

1 Introduction

`Pviz` is an R package inspired by and depending on `Gviz`. It introduces new types of track and extends the existing ones in order to deal with amino-acid based data.

This package keeps most of the mechanics of `Gviz`, notably the use of `DisplayParameters` and the same plotting function: `plotTracks`. Therefore, the user is invited to refer to `Gviz` help pages and vignette for more information and examples.

As with any R package, it should first be loaded in the session

```
library(Pviz)
```

*rsautera@fhcrc.org

```
## Warning:  replacing previous import 'utils::findMatches' by 'S4Vectors::findMatches' when
loading 'AnnotationDbi'
```

2 Gviz tracks

`Pviz` extends and uses the most common classes of `Gviz` to make them easier to use with amino acid data. We removed the requirement for a genome and a chromosome when creating these tracks. Moreover, they support the functions defined in Pviz.

2.1 ATrack

`ATrack` extends Gviz's `AnnotationTrack` and behaves the same way. However, it does not require to specify a chromosome and a genome. Please refer to `Gviz` documentation for more details about `AnnotationTrack` and the available `DisplayParameters`.

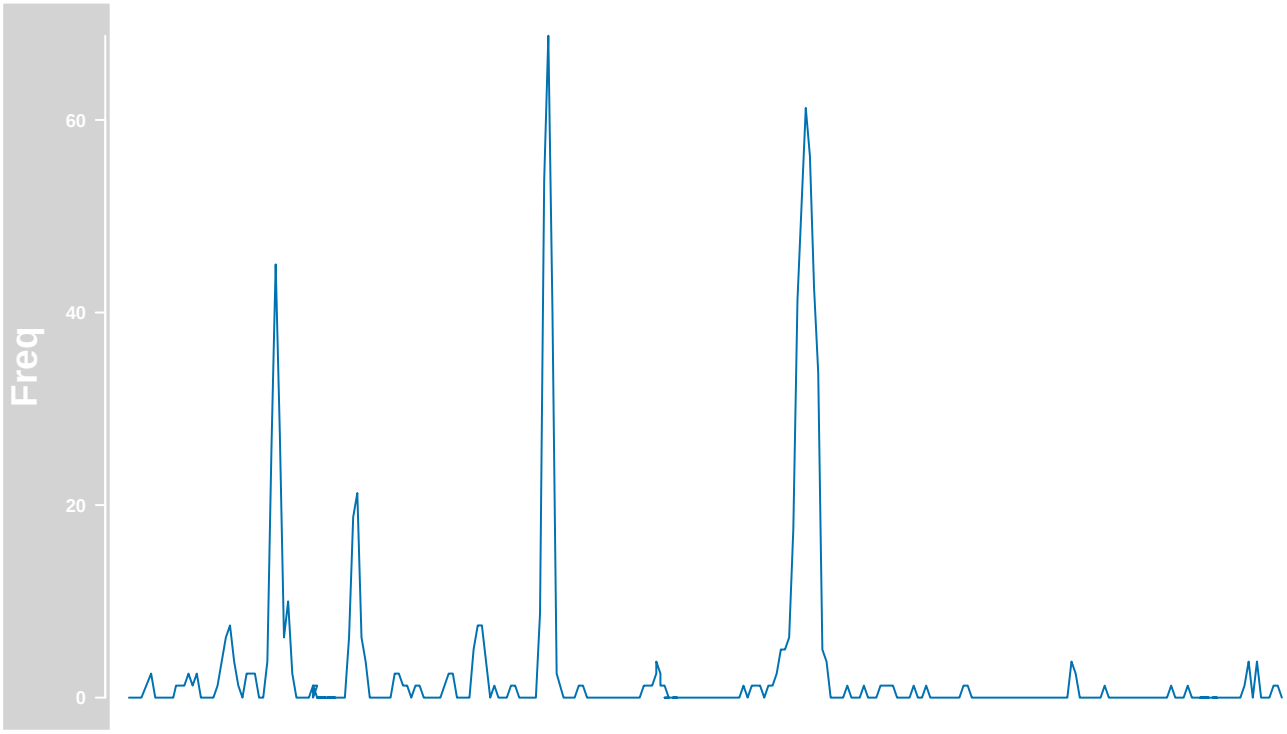
```
at<-ATrack(start = c(250, 480), end = c(320, 520), id = c("Anno1", "Anno2"),
           showFeatureId = TRUE, fontcolor = "black", name = "Annotations")
plotTracks(at, from = 1, to = 600)
```

2.2 DTrack

Naturally `DTrack` extends Gviz's `DataTrack`. Here again, please refer to `Gviz` documentation for details on how to use `DataTrack`.

Some example data are available in the data package `pepDat`. Frequency of antibody binding event in hxb2 enveloppe peptides.

```
library(pepDat)
data(restab_aggregate)
dt <- DTrack(data = restab_aggregate$group2, start = restab_aggregate$start,
             width=15, name="Freq", type = "l")
plotTracks(dt, from = 1, to = 850, type = "l")
```



3 Pviz new track types

`Pviz` introduce some new track types to deal with amino-acid based data. The new tracks look can be modified using the `DisplayParameters` and will most of the time offer the same options as the ones available for `Gviz` tracks.

3.1 ProteinAxisTrack

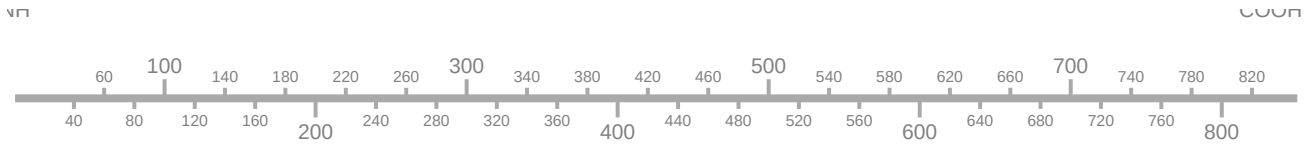
This track acts as a replacement for the `GenomeAxisTrack`. I comes with the same coloration, transparency and other customization options but loses the DNA representation for a simple segment.

```
pat<-ProteinAxisTrack()
plotTracks(pat, from = 1, to = 850)
```

Just like in `GenomeAxisTrack`, it is possible to use littleTicks to get a more precise scale. Moreover, because `Pviz`, has been made to deal with peptides and protein, the option addNC can display indicators

for N-term and C-term ends on the axis.

```
pat<-ProteinAxisTrack(addNC = TRUE, littleTicks = TRUE)
plotTracks(pat, from = 1, to = 850)
```



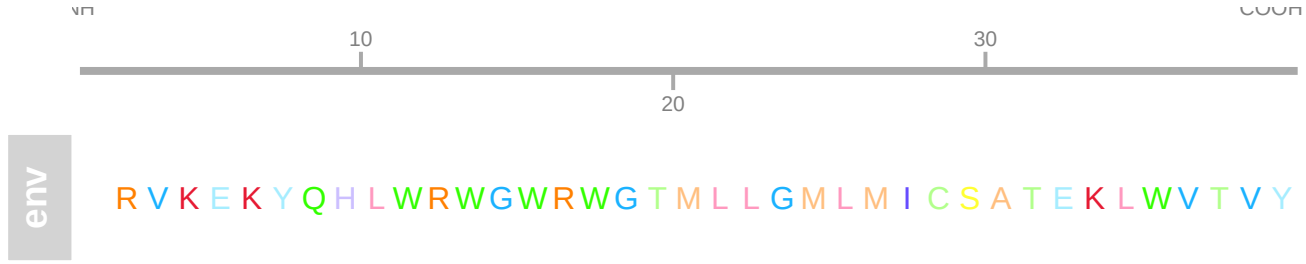
3.2 ProteinSequenceTrack

This new track simply displays a selected sequence. It can takes both `AAstring` or regular `character`.

Note that the first amino acid of the sequence should correspond to the first position of any other element you choose to display at the same time.

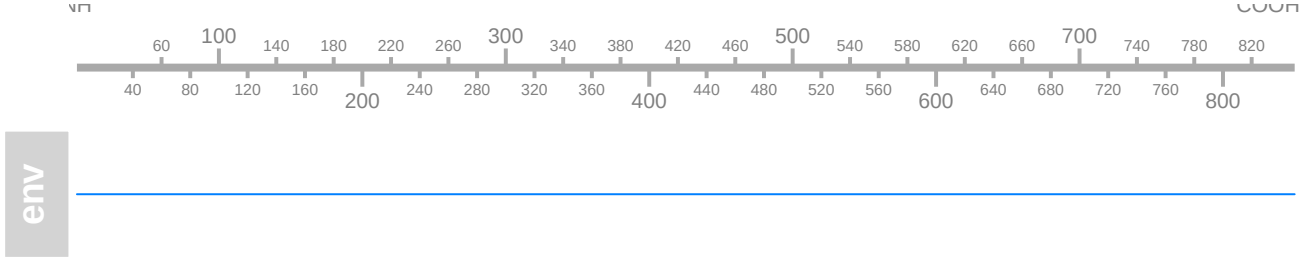
The previously loaded dataset also contains the sequence of the envelope of hxb2 to be used as an example. The peptide collections in `pepDat` contain reference sequence as metadata. Here hxb2 sequence is displayed.

```
data(pep_hxb2)
hxb2_seq <- metadata(pep_hxb2)$sequence
st <- ProteinSequenceTrack(sequence = hxb2_seq, name = "env")
plotTracks(trackList = c(pat, st), from = 1, to = 40)
```



The sequence track for proteins handles overplotting the same way it does it for nucleotides. If the plotting range becomes wider, only the color code will be displayed. Once it becomes too big to even show these colors, a straight line will be displaed. Naturally, the character size will also influence what can be displayed in the graphic window.

```
st <- ProteinSequenceTrack(sequence = hxb2_seq, name = "env", cex = 0.5)
plotTracks(trackList = c(pat, st), from = 1, to = 850)
```



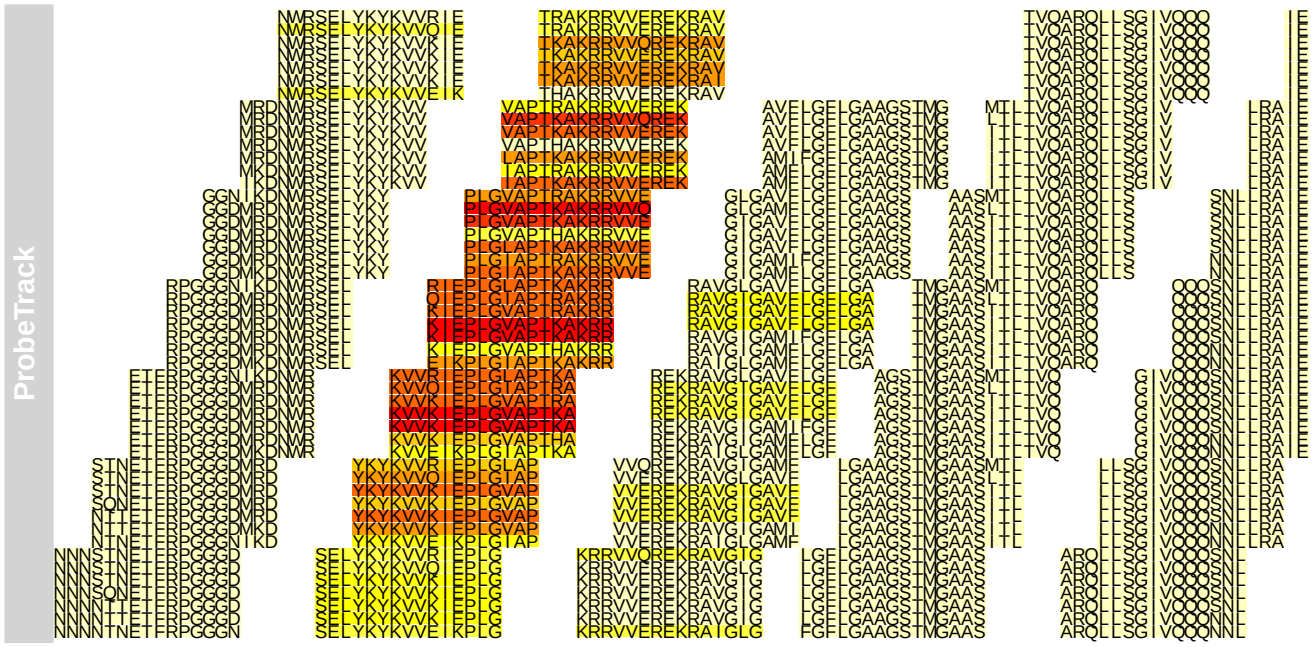
Although the character expansion has been set to less than 1. The ranges are still too wide for a correct display and only a straight line will be displayed.

3.3 ProbeTrack

This track is designed to display peptide microarray data. It draws each peptide relative to its position in the sequence and enclose them in rectangles colored depending on their frequency of binding event or intensity. It is useful to spot differences between clades at a specific position or get an overview of the regions with antibody binding activity, depending on the scale used while plotting.

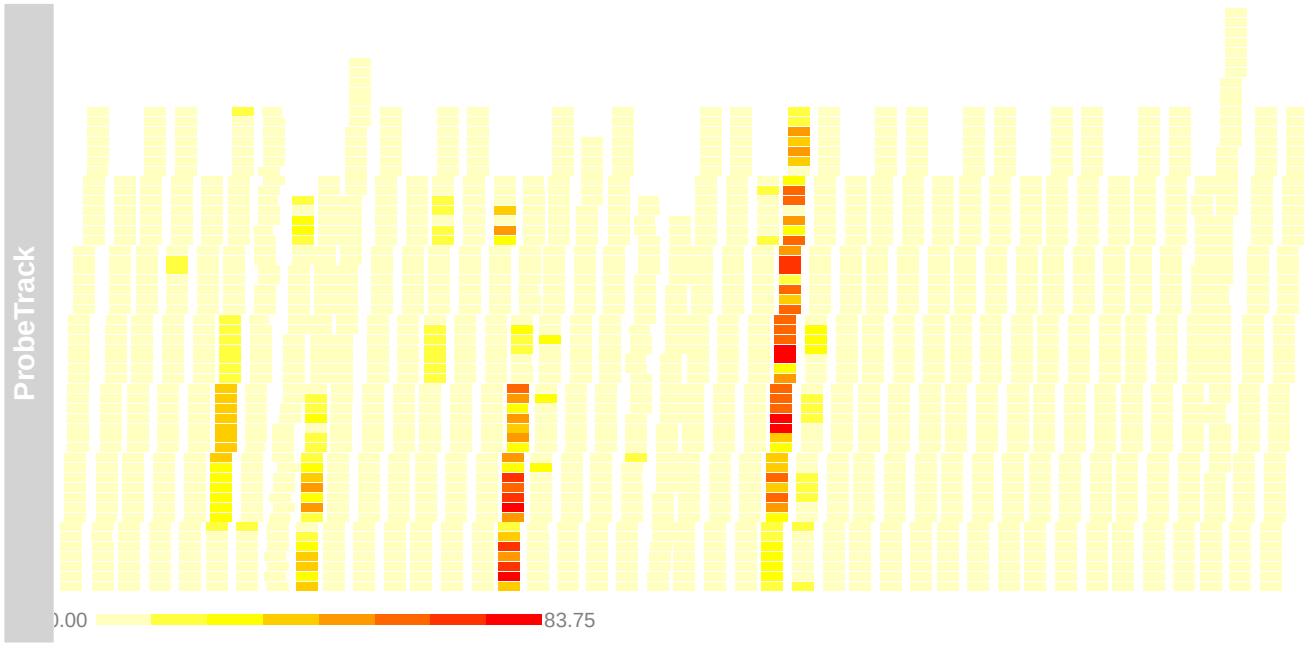
To create this track, the sequence of the peptides, their intensity or frequency and their starting position have to be passed as arguments. All three arguments should be of the same length. Here, the result of a peptide microarray analysis is used. This time with clade specific calls.

```
data(restab)
pt<-ProbeTrack(sequence = restab$peptide, intensity = restab$group2,
               probeStart = restab$start)
plotTracks(pt, from = 460, to = 560)
```



Unlike in `ProteinSequenceTrack`, the size of the characters in each peptide sequence depends on the plotting ranges (the user can still choose to change the size manually) and if the ranges become too wide, the characters will appear as dots or completely disappear instead of stacking on top of each other. While it loses the sequence information, it might be relevant to locate regions where peptides have high intensity/frequency.

```
plotTracks(pt, legend = TRUE)
```



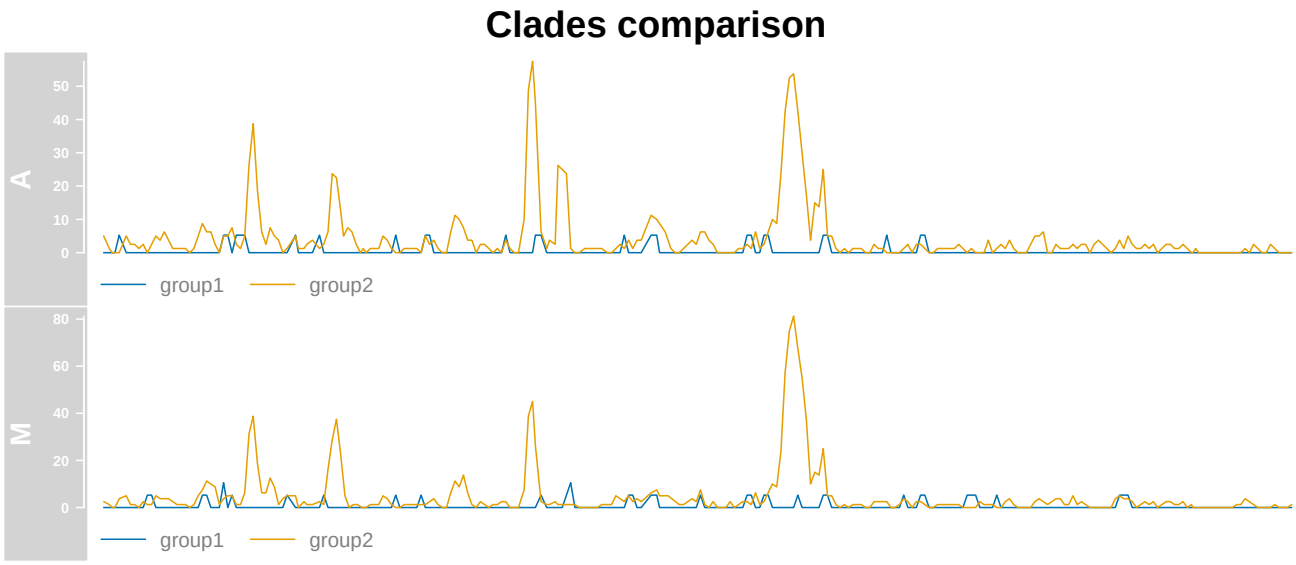
For a more explicit display, a legend has been implemented for this track and can be called during track creation or in the plotting function. The legend displays the scale of frequencies.

3.4 CladeTrack

Finally, after displaying all peptides, it is possible to look at clades of interest or compare the binding activity in different clades.

`CladeTrack` extends `DTrack` and adds a new constructor that can use the result of a peptide microarray analysis from `pepStat` to create the track. The display parameters are the same as in `DTrack` and `Gviz`'s `DataTrack`.

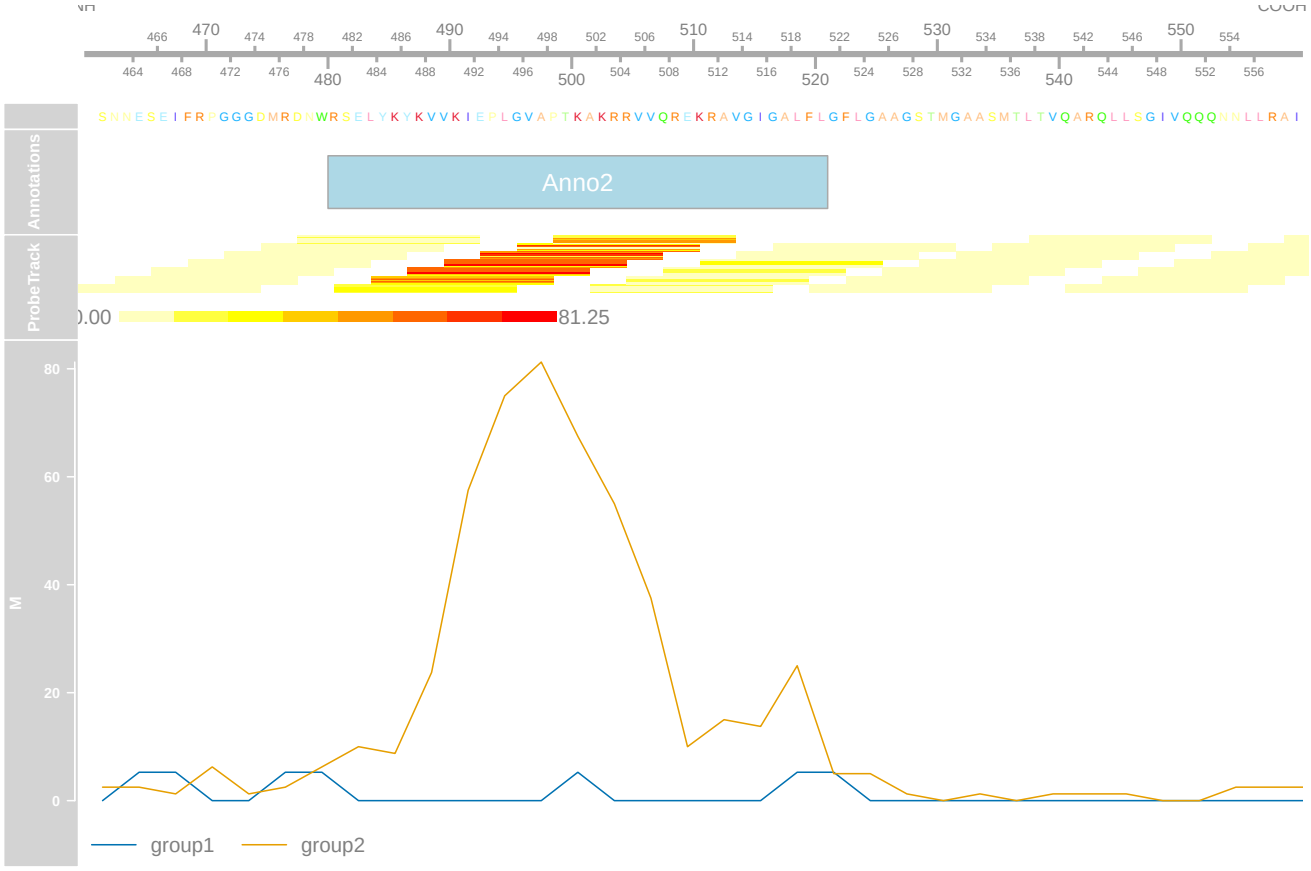
```
ctA <- CladeTrack(restab, clade = "A", type = "l")
ctM <- CladeTrack(restab, clade = "M", type = "l", legend = TRUE)
plotTracks(c(ctA, ctM), main = "Clades comparison", cex.main = 1.5)
```



4 Example of plot

Naturally, the interest of `Pviz`, just like its parent `Gviz` is the display of multiple tracks at once. Here is an example of what `Pviz` can render, using the tracks previously created.

```
pt <- ProbeTrack(sequence = restab$peptide, intensity = restab$group2,
                 probeStart = restab$start, cex=0, legend=TRUE)
plotTracks(trackList=c(pat, st, at, pt, ctM), from=460, to=560,
           type="l")
```



5 Summary plots

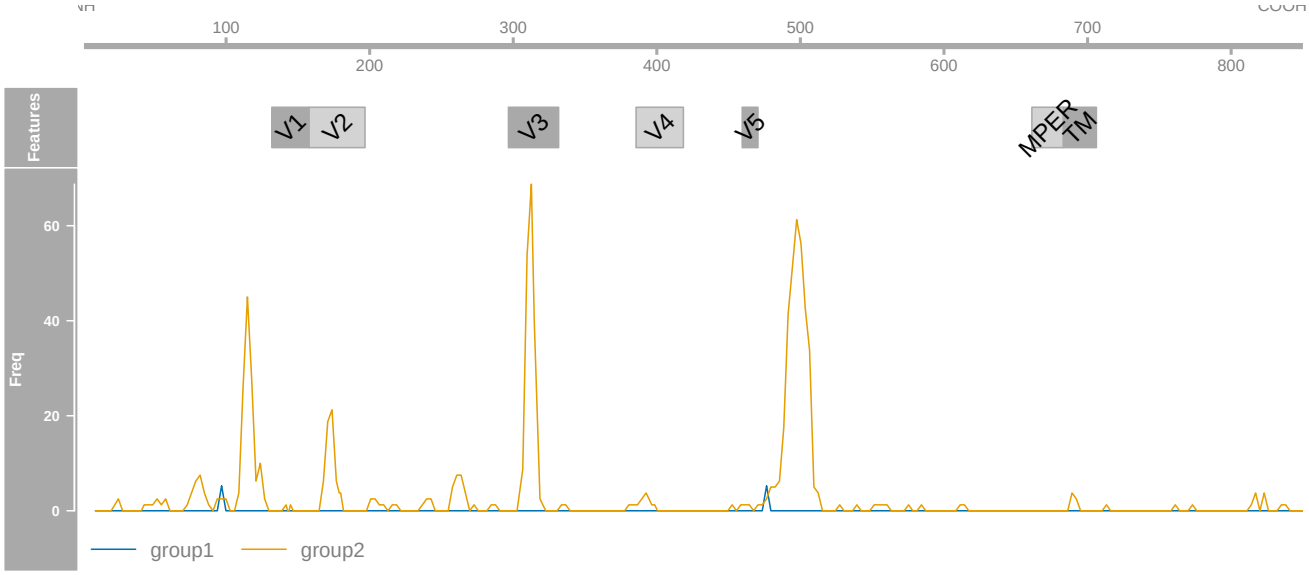
As part of the pipeline for peptide microarray analysis, the package provides two function to quickly dispplay the result of an experiment. These ploting functions come with a selected set of tracks as well as hard coded annotations for hxb2 enveloppe landmarks. The function are used as a display in the shinyApp that comes with `pepStat`.

More information regarding the plots and the analysis are available in the vignette of the `pepStat` package. The additional display parameters are passed directly to the `plotTracks` function.

5.1 plot_inter

First we can plot the result of an experiment:

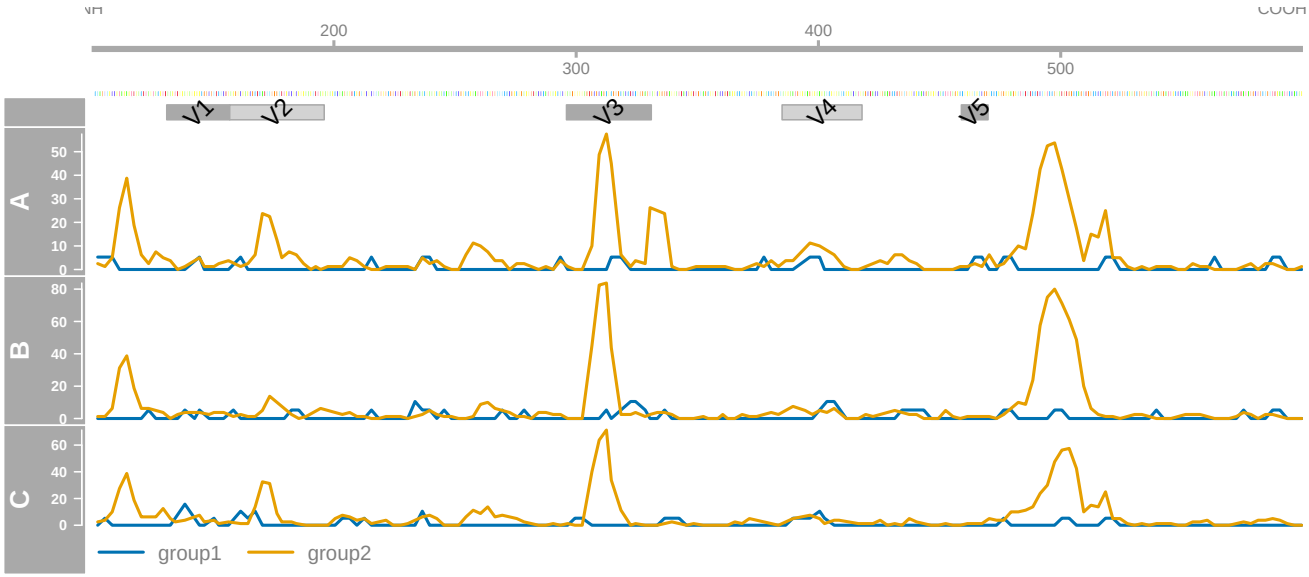
```
plot_inter(restab_aggregate)
```



5.2 plot_clade

Then redo a clade specific analysis and plot the results for clade of interest:

```
plot_clade(restab, clade = c("A", "B", "C"), sequence = hxb2_seq,
           from = 100, to = 600)
```



6 sessionInfo

```
sessionInfo()

## R version 4.3.0 RC (2023-04-18 r84287)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 22.04.2 LTS
##
## Matrix products: default
## BLAS:   /home/biocbuild/bbs-3.18-bioc/R/lib/libRblas.so
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_GB              LC_COLLATE=C
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## time zone: America/New_York
## tzcode source: system (glibc)
##
## attached base packages:
## [1] grid      stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
## [1] pepDat_1.19.0        Pviz_1.35.0          Gviz_1.45.0
## [4] GenomicRanges_1.53.0 GenomeInfoDb_1.37.0  IRanges_2.35.0
## [7] S4Vectors_0.39.0     BiocGenerics_0.47.0  knitr_1.42
##
## loaded via a namespace (and not attached):
##   [1] DBI_1.1.3                   bitops_1.0-7
##   [3] deldir_1.0-6                gridExtra_2.3
##   [5] biomaRt_2.57.0              rlang_1.1.0
##   [7] magrittr_2.0.3              biovizBase_1.49.0
##   [9] matrixStats_0.63.0          compiler_4.3.0
##  [11] RSQLite_2.3.1               GenomicFeatures_1.53.0
##  [13] png_0.1-8                   vctrs_0.6.2
##  [15] ProtGenerics_1.33.0         stringr_1.5.0
##  [17] pkgconfig_2.0.3             crayon_1.5.2
##  [19] fastmap_1.1.1               backports_1.4.1
##  [21] dbplyr_2.3.2                XVector_0.41.0
##  [23] utf8_1.2.3                  Rsamtools_2.17.0
```

```
##  [25] rmarkdown_2.21             bit_4.0.5
##  [27] xfun_0.39                  zlibbioc_1.47.0
##  [29] cachem_1.0.7               progress_1.2.2
##  [31] blob_1.2.4                 highr_0.10
##  [33] DelayedArray_0.27.0        BiocParallel_1.35.0
##  [35] jpeg_0.1-10                parallel_4.3.0
##  [37] prettyunits_1.1.1          cluster_2.1.4
##  [39] R6_2.5.1                   VariantAnnotation_1.47.0
##  [41] stringi_1.7.12             RColorBrewer_1.1-3
##  [43] rtracklayer_1.61.0         rpart_4.1.19
##  [45] Rcpp_1.0.10                SummarizedExperiment_1.31.0
##  [47] base64enc_0.1-3            Matrix_1.5-4
##  [49] nnet_7.3-18                tidyselect_1.2.0
##  [51] rstudioapi_0.14            dichromat_2.0-0.1
##  [53] yaml_2.3.7                 codetools_0.2-19
##  [55] curl_5.0.0                 lattice_0.21-8
##  [57] tibble_3.2.1               Biobase_2.61.0
##  [59] KEGGREST_1.41.0            evaluate_0.20
##  [61] foreign_0.8-84             BiocFileCache_2.9.0
##  [63] xml2_1.3.3                 Biostrings_2.69.0
##  [65] pillar_1.9.0               filelock_1.0.2
##  [67] MatrixGenerics_1.13.0      checkmate_2.1.0
##  [69] generics_0.1.3             RCurl_1.98-1.12
##  [71] ensembldb_2.25.0           hms_1.1.3
##  [73] ggplot2_3.4.2              munsell_0.5.0
##  [75] scales_1.2.1               glue_1.6.2
##  [77] Hmisc_5.0-1                lazyeval_0.2.2
##  [79] tools_4.3.0                interp_1.1-4
##  [81] BiocIO_1.11.0              data.table_1.14.8
##  [83] BSgenome_1.69.0            GenomicAlignments_1.37.0
##  [85] XML_3.99-0.14              latticeExtra_0.6-30
##  [87] AnnotationDbi_1.63.0       colorspace_2.1-0
##  [89] GenomeInfoDbData_1.2.10    htmlTable_2.4.1
##  [91] restfulr_0.0.15            Formula_1.2-5
##  [93] cli_3.6.1                  rappdirs_0.3.3
##  [95] fansi_1.0.4                dplyr_1.1.2
##  [97] AnnotationFilter_1.25.0    gtable_0.3.3
##  [99] digest_0.6.31              rjson_0.2.21
## [101] htmlwidgets_1.6.2          memoise_2.0.1
## [103] htmltools_0.5.5            lifecycle_1.0.3
## [105] httr_1.4.5                 bit64_4.0.5
```