Package 'pafr'

December 8, 2020

Title Read, Manipulate and Visualize 'Pairwise mApping Format' Data

Version 0.0.2

Description Provides functions to read, process and visualize pairwise sequence alignments in the 'PAF' format used by 'minimap2' and other whole-genome aligners. 'minimap2' is described by Li H. (2018) <doi:10.1093/bioinformatics/bty191>.

Depends R (>= 3.4.4), ggplot2,

Imports dplyr, tibble, stringr, rlang

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Suggests testthat, covr, knitr, ggpubr, rmarkdown, microbenchmark

RoxygenNote 7.1.1

URL https://dwinter.github.io/pafr/

BugReports https://github.com/dwinter/pafr/issues

VignetteBuilder knitr

NeedsCompilation no

Author David Winter [aut, cre] (<https://orcid.org/0000-0002-6165-0029>), Kate Lee [ctb] (<https://orcid.org/0000-0002-0886-3746>), Murray Cox [ctb] (<https://orcid.org/0000-0003-1936-0236>)

Maintainer David Winter <david.winter@gmail.com>

Repository CRAN

Date/Publication 2020-12-08 10:20:12 UTC

R topics documented:

`as_paf` *Coerce a data.frame or tibble into a pafr object*

Description

The main reason to use this function is speed up the process of reading in a large paf file that has no tags. Functions like read.table, read_delim (reader) and fread (data.table) can process a 12 column file more quickly than pafr's read_paf. If you you do not need tag data for your analyses or visualizations, it might make sense to use a fast reading function to get a 12 column data.frame, convert that data.frame into a 'pafr object with this function. The 'pafr' object can then work easily with the functions in this package.

Usage

```
as_paf(paf_data_frame)
```

Arguments

`paf_data_frame`	a data.frame object with 12 columns. Column names and types will be overwritten in the returned object

Value

a pafr object

See Also

read_paf

chrom_sizes *Extract the sizes of all sequences in a paf alignment*

Description

Extract the sizes of all sequences in a paf alignment

Usage

```
chrom_sizes(ali)
```

Arguments

ali	pafr or tibble containing the genome alignment (as returned by read_paf)

Value

list with two elements (tlens and qlens) Each element is a dataframe with one column of sequence names and another column containing the length of each sequence

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
chrom_sizes(ali)
```

dotplot *Generate a dot plot from a paf alignment*

Description

Generate a dot plot from a paf alignment

Usage

```
dotplot(
  ali,
  order_by = c("size", "qstart", "provided"),
  label_seqs = FALSE,
  dashes = TRUE,
  ordering = list(),
  alignment_colour = "black",
  xlab = "query",
  ylab = "target",
  line_size = 2
)
```

Arguments

ali	pafr or tibble containing the genome alignment (as returned by read_paf)
order_by	How the query and target sequences should be ordered in the dot plot. Option must be one of 'size' (smallest-to-largest), 'qstart' (query organised smallest to largest, target by first match in the query genome) or 'provided' (ordering as specified in the ordering argument)
label_seqs	boolean If TRUE, label centre of query and target sequences in margins of the dot plot
dashes	boolean If TRUE, add dashes to borders of query and target sequences in the dot plot
ordering	If order_by is set to TRUE, this variable should be a list with two elements specifying the order of query and then target sequences in the dot plot. This option is ignored if order_by is set to other values
alignment_colour	character The colour used to draw each aligned section in the dot plot (defaults to black)
xlab	character The x-axis label (defaults to 'query')
ylab	character The y-axis label (defaults to 'target')
line_size	The width of the line used to represent an alignment in the dot plot (defaults to 2)

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
dotplot(ali)
dotplot(ali) + theme_bw()
dotplot(ali, label_seqs=TRUE, order_by="qstart", alignment_colour="blue")
```

---

filter_secondary_alignments

Remove secondary alignments from a pafr alignment

Description

Remove secondary alignments from a pafr alignment

Usage

```
filter_secondary_alignments(ali, remove_inversions = FALSE)
```

Arguments

ali	Genomic alignment in pafr or tbl_df format, as returned by read_paf
remove_inversions	logical If TRUE, also remove inversions (tp flag 'I' or 'i') from the alignment

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
ali
filter_secondary_alignments(ali)
```

`Gb_lab` *Number formatters for scales in base pairs*

Description

For use with ggplot2

Usage

```
Gb_lab(x)

Mb_lab(x)

Kb_lab(x)
```

Arguments

x	The data (in base pairs) to be formatted as Gb, Mb or Kb

Value

A character vector with scale labels

Examples

```
## Not run:
ali <- read_paf(system.file("extdata", "fungi.paf", package="pafr"))
doplot(ali) + scale_x_continuous("Genomic position", label=Mb_lab)

## End(Not run)
```

`highlight_query` *Highlight segments of a query or target genome in a dot plot*

Description

This plot is intended to be used in conjunction with `link{dotplot}`. Adding `higlight_query` or `highlight_target` to a dotplot function call (see examples below) will add a rectangular 'highlight' corresponding to a particular genomic interval in the corresponding genome.

Usage

```
highlight_query(bed, fill = "yellow", colour = "black", alpha = 0.6)

highlight_target(bed, fill = "yellow", colour = "black", alpha = 0.6)
```

Arguments

bed	data.frame or tbl_df containing a bed file, as returned by read_bed. Should contain three columns named 'chrom', 'start' and 'end'
fill	character Fill colour for highlight segment
colour	character Outline colour for highlight segment
alpha	character Opacity ([0-1]) for highlight segment

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
cen <- read_bed(system.file("extdata", "Q_centro.bed", package="pafr"))
dotplot(ali) + highlight_query(cen)
interval <- data.frame(chrom="T_chr3", start=2000000, end=3000000)
dotplot(ali, label_seqs=TRUE) +
   highlight_target(interval)
```

---

plot_coverage *Plot the regions of one genome that are covered by alignments in a paf file*

Description

Each sequence in the focal genome is displayed as a rectangle, with regions covered by an alignment shaded as per the fill argument described below. Uncovered regions remain white.

Usage

```
plot_coverage(
  ali,
  target = TRUE,
  fill = "forestgreen",
  direct_label = TRUE,
  label_colour = "black",
  xlab = "Position in sequence",
  x_labeller = Mb_lab
)
```

Arguments

ali	alignment An alignment as read by read_paf
target	logical If TRUE, display coverage for the target genome; if FALSE, display coverage for the query
fill	character How to colour the alignment blocks. If the character provided is the name of a column in the alignment, this column will be passed to ggplot2 to shade alignment blocks. Otherwise, the character is treated as a single colour to be used for all alignment blocks.
direct_label	logical If TRUE, use geom_text to directly label the name of the focal sequences; if FALSE, no direct labels are drawn
label_colour	character Colour used for direct labels
xlab	string Name for the x-axis
x_labeller	function Function to be used to label the x-axis (defaults to Mb_lab

Details

Note that this function uses theme_coverage_plot to style the graph. Using another ggplot theme on the plot may produce unexpected results.

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
plot_coverage(ali)
plot_coverage(ali, fill='qname', direct_label=FALSE) +
   scale_fill_brewer(palette="Set1")
```

---

plot_synteny *Plot synteny between a query and target sequence in a PAF alignment*

Description

Plot synteny between a query and target sequence in a PAF alignment

Usage

```
plot_synteny(
  ali,
  q_chrom,
  t_chrom,
  centre = TRUE,
  rc = FALSE,
  xlab = "Position in query",
  ylab = "",
  x_labeller = Mb_lab
)
```

Arguments

ali	pafr or tibble containing the genome alignment (as returned by read_paf)
q_chrom	character Name for the query sequence
t_chrom	character Name for the target sequence
centre	logical If TRUE (default), adjust the position of the target sequence, so it is centred on the query. If not, both sequences start at position zero
rc	logical If TRUE, use the reverse and complement for the target sequence
xlab	string Name for the x-axis
ylab	string Name for the y-axis
x_labeller	Function to be used to label the x-axis

Value

A ggplot object that displays synteny between query and target sequences

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
long_ali <- subset(ali, alen > 1e4)
plot_synteny(long_ali, q_chrom="Q_chr3", t_chrom="T_chr4", centre=TRUE)
plot_synteny(long_ali, q_chrom="Q_chr5", t_chrom="T_chr5", centre=TRUE)
plot_synteny(long_ali, q_chrom="Q_chr5", t_chrom="T_chr5", centre=TRUE, rc=TRUE)
```

---

read_bed *Read genomic intervals in bed format*

Description

The first three columns of the file specified by file_name must contain data in the standard bed format (i.e., a genomic interval represented by 0-based half-open interval with seq-id, start and end position). These columns will be renamed to 'chrom', 'start' and 'end', respectively. Any other columns present in the data will be left unmodified.

Usage

```
read_bed(file_name, tibble = FALSE, ...)
```

Arguments

file_name	Path to the bed file to be read in
tibble	logical If TRUE, the genomic intervals are returned as a tidy tbl_df.
...	Other arguments passed to read.table

Details

The file is read into memory with `read.table`, with the argument `sep` set to `'\t'` and `stringsAsFactors` set to FALSE. All other arguments are left as default, but arguments can be passed from `read_bed` to `read.table`.

Value

Either a `data.frame` or a `tbl_df` with at least three columns named 'chrom', 'start' and 'end'

Examples

```
bed_path <- system.file("extdata", "Q_centro.bed", package="pafr")
centro <- read_bed(bed_path)
centro
# Can pass arguments to read.table
miss_two <- read_bed(bed_path, skip=2)
miss_two
```

---

read_paf *Read a genomic alignment in PAF format*

Description

See the package vignette for detailed information on the file format and its representation as an R object.

Usage

```
read_paf(file_name, tibble = FALSE, include_tags = TRUE)
```

Arguments

`file_name`	Path to the .paf file
`tibble`	logical If TRUE, the genomic alignments are returned as a tidy `tbl_df`
`include_tags`	logical if TRUE (default) read additional information about each alignment encoded as PAF tags. Setting this to FALSE will speed up parsing of paf alignments, specially those with large CIGAR strings/

Value

Either a `pafr` object, which acts as a `data.frame`, or a `tbl_df` containing information on genomic alignments. The contents of this table are described in detail in the pafr package vingette.

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
ali
```

theme_coverage_plot *A minimalistic ggplot2 theme designed for use with genome coverage plots*

Description

This theme is used as the default when plot_coverage is called, so you should usually only call this function to modify the appearance of the coverage plot.

Usage

```
theme_coverage_plot(facet_labs = TRUE, show_legend = TRUE)
```

Arguments

facet_labs	logical If TRUE (default), label sequences using the facet labels; if FALSE, sequences are labeled directly using geom_text
show_legend	logical If TRUE (default), label display any legend associated with the fill parameter of plot_coverage; if FALSE, do not display a legend

Examples

```
ali <- read_paf( system.file("extdata", "fungi.paf", package="pafr") )
plot_coverage(ali) + theme_coverage_plot(show_legend=FALSE)
```

Index