

Package ‘ggchord’

August 24, 2026

Title Multi-Sequence Alignment Chord Diagram Visualization Tool

Version 0.8.0

Description A 'ggplot2'-based R package that visualizes multi-sequence alignment results as chord diagrams using layered grammar of graphics. Users build chord plots by stacking geom layers (geom_seq, geom_ribbon, geom_gene, geom_axis). Layout parameters are specified within each geom layer, following 'ggplot2' conventions. Homologous regions between query and subject sequences are intuitively displayed.

License MIT + file LICENSE

Depends R (>= 4.1.0)

Imports ggplot2 (>= 4.0.0), grDevices, grid

VignetteBuilder knitr

Suggests knitr, rmarkdown, qpdf, dplyr, plotly, testthat (>= 3.0.0)

Encoding UTF-8

LazyData true

URL <https://github.com/DangJem/ggchord>,
<https://dangjem.github.io/ggchord/>

BugReports <https://github.com/DangJem/ggchord/issues>

NeedsCompilation no

Author Jem Dang [aut, cre]

Maintainer Jem Dang <dangjem0730@gmail.com>

Config/roxygen2/version 8.0.0

Repository CRAN

Date/Publication 2026-08-24 08:20:19 UTC

Contents

+.ggchord	2
as.data.frame.ggchord_validation	3
clean_ggchord_data	3

coord_chord	5
deduplicate_ggchord_ribbons	6
filter_ggchord_ribbons	7
gene_data_example	8
geom_axis	9
geom_feature	10
geom_gene	12
geom_gene_label	13
geom_gene_label_repel	14
geom_ribbon	16
geom_ribbon_highlight	19
geom_seq	21
geom_seq_label	22
geom_seq_region	24
get_chord_layout	25
ggchord	26
ggplotly.ggchord	27
merge_ggchord_ribbons	28
print.ggchord_clean	29
read_blast	29
read_fasta_lengths	30
read_gff3	31
ribbon_data_example	32
seq_data_example	32
validate_ggchord_data	33
Index	35

<code>+.ggchord</code>	<i>Combine a ggchord plot with ggplot2 objects</i>
------------------------	--

Description

Supports stacking ggplot2 layers, lists of layers, scales, and themes onto a ggchord plot using the + operator.

Usage

```
## S3 method for class 'ggchord'
e1 + e2
```

Arguments

- e1 A ggchord object
- e2 A ggplot2 layer, a list of layers, a scale, or a theme

Value

A ggchord object

as.data.frame.ggchord_validation

Coerce a validation result to a flat data.frame

Description

Combines the errors and warnings tables into a single data.frame and adds a severity column, which is convenient for filtering, exporting or printing the full report programmatically.

Usage

```
## S3 method for class 'ggchord_validation'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	A "ggchord_validation" object.
row.names	Ignored.
optional	Ignored.
...	Ignored.

Value

A data.frame with columns table, category, row, column, message and severity.

clean_ggchord_data

Clean ggchord input data with explicit, report-driven policies

Description

Applies a conservative, predictable set of cleaning policies to the three ggchord tables and returns the cleaned copies plus a full report. The original data frames are never modified. Nothing is dropped silently: every change (including every dropped row) is recorded in report with the original row number, the reason, the original value(s) and the new value(s).

Usage

```
clean_ggchord_data(
  seq_data,
  ribbon_data = NULL,
  gene_data = NULL,
  unknown_id = c("drop", "error", "keep"),
  out_of_range = c("clip", "drop", "error", "keep"),
  reversed_interval = c("sort", "drop", "error", "keep"),
  invalid_pident = c("clip", "drop", "error", "keep"),
```

```

empty_annotation = c("keep", "drop", "replace"),
replacement_annotation = "unannotated"
)

```

Arguments

seq_data	data.frame/tibble, required. Must contain seq_id and length; used as the coordinate reference for clipping.
ribbon_data	data.frame/tibble, optional. Alignment results.
gene_data	data.frame/tibble, optional. Gene annotation data.
unknown_id	Character, default "drop". Policy for rows whose sequence ID is missing, empty or unknown: "drop" removes them, "error" stops with a message, "keep" leaves them unchanged.
out_of_range	Character, default "clip". Policy for coordinates outside [1, sequence length]: "clip" clamps them, "drop" removes the row, "error" stops, "keep" leaves them unchanged. After clipping, intervals that become invalid (degenerate) are removed explicitly and reported.
reversed_interval	Character, default "sort". Policy for intervals with start > end: "sort" swaps the endpoints so the feature draws stably (the original direction is recorded in the report), "drop" removes the row, "error" stops, "keep" leaves them unchanged.
invalid_pident	Character, default "clip". Policy for pident outside [0, 100]: "clip" clamps them, "drop" removes the row, "error" stops, "keep" leaves them unchanged.
empty_annotation	Character, default "keep". Policy for missing/empty anno in gene data: "replace" fills them with replacement_annotation, "drop" removes the row, "keep" leaves them unchanged.
replacement_annotation	Character, default "unannotated". Replacement annotation used when empty_annotation = "replace".

Value

A list with four components: seq_data, ribbon_data, gene_data (cleaned copies) and report (a data.frame with columns table, row (original row number), column, reason, original_value, new_value and action).

Examples

```

library(ggchord)
data(seq_data_example)
data(ribbon_data_example)
data(gene_data_example)

# Introduce a few typical problems
bad_r <- transform(ribbon_data_example,
                    qstart = pmin(qstart, 1),

```

```

      pident = pmin(pident, 150))
bad_g <- transform(gene_data_example,
                  anno = ifelse(seq_len(nrow(gene_data_example)) == 1,
                              NA_character_, anno))

out <- clean_ggchord_data(seq_data_example, bad_r, bad_g)
head(out$report)
# The cleaned tables are ready for ggchord()

p <- ggchord(out$seq_data, out$ribbon_data, out$gene_data) +
  geom_seq() + geom_ribbon() + geom_gene()

```

coord_chord	<i>Chord diagram coordinate system</i>
-------------	--

Description

A lightweight Coord. It creates placeholder coordinates in `ggchord()`, which are replaced at print time with the actual extents computed from the layout.

Usage

```
coord_chord(layout = NULL)
```

Arguments

layout	Chord layout object (passed internally by <code>ggchord()</code> , may be <code>NULL</code>)
--------	---

Value

A Coord object for `ggplot2` + composition

Examples

```

library(ggchord)
data(seq_data_example)
p <- ggchord(seq_data_example) + coord_chord() + geom_seq()
p

```

deduplicate_ggchord_ribbons

Deduplicate alignment ribbons

Description

Removes fully duplicated, coordinate-near-duplicated or highly overlapping alignment blocks within each (query, subject) pair and keeps the best representative per keep.

Usage

```
deduplicate_ggchord_ribbons(
  ribbon_data,
  tolerance = 0,
  by = c("exact", "coordinates", "overlap"),
  keep = c("best_pident", "longest", "first"),
  min_reciprocal_overlap = 0.9
)
```

Arguments

<code>ribbon_data</code>	data.frame in ribbon_data format.
<code>tolerance</code>	Numeric, default 0. Maximum absolute difference (in bp) allowed on each of qstart/qend/sstart/send for two rows to count as coordinate duplicates (used with <code>by = "coordinates"</code>).
<code>by</code>	Character, default "exact". "exact" removes rows with identical pair and coordinates; "coordinates" additionally treats rows within tolerance bp as duplicates; "overlap" treats blocks whose reciprocal overlap is at least <code>min_reciprocal_overlap</code> on both sequences as duplicates.
<code>keep</code>	Character, default "best_pident". Which representative to keep: "best_pident" (highest pident), "longest" (longest length) or "first" (first occurrence in the input).
<code>min_reciprocal_overlap</code>	Numeric (0-1), default 0.9. Reciprocal overlap threshold used with <code>by = "overlap"</code> .

Value

A list with data (deduplicated data frame with `source_rows` attribute) and report (n_input/n_kept/n_removed plus a data.frame of removed rows with `row`, `duplicate_of` and `reason`).

Examples

```
library(ggchord)
data(ribbon_data_example)
dup <- rbind(ribbon_data_example, ribbon_data_example[1, ])
out <- deduplicate_ggchord_ribbons(dup, by = "exact")
out$report
```

filter_ggchord_ribbons

Filter alignment ribbons before plotting

Description

Keeps the ribbon rows that satisfy all requested criteria and optionally sorts them. The returned data frame keeps every extra column and the original column order; the original row numbers are attached as the `source_rows` attribute.

Usage

```
filter_ggchord_ribbons(
  ribbon_data,
  seq_ids = NULL,
  min_pident = NULL,
  max_pident = NULL,
  min_length = NULL,
  max_evalue = NULL,
  min_bitscore = NULL,
  min_query_coverage = NULL,
  min_subject_coverage = NULL,
  keep_pairs = NULL,
  drop_self_links = TRUE,
  sort_by = NULL
)
```

Arguments

<code>ribbon_data</code>	data.frame with at least <code>qaccver</code> and <code>saccver</code> columns (normally the full <code>ribbon_data</code> format).
<code>seq_ids</code>	Optional character vector. Keep only rows where both the query and the subject are in <code>seq_ids</code> .
<code>min_pident</code> , <code>max_pident</code>	Optional numeric. Lower/upper bounds on <code>pident</code> .
<code>min_length</code>	Optional numeric. Lower bound on <code>length</code> .
<code>max_evalue</code>	Optional numeric. Upper bound on <code>evalue</code> (requires an <code>evalue</code> column).
<code>min_bitscore</code>	Optional numeric. Lower bound on <code>bitscore</code> (requires a <code>bitscore</code> column).
<code>min_query_coverage</code>	Optional numeric (0-100). Lower bound on the query coverage. Uses the <code>qcovs</code> column when present; otherwise computed from <code>qlen</code> and the query interval (requires <code>qlen</code>).
<code>min_subject_coverage</code>	Optional numeric (0-100). Lower bound on the subject coverage, computed from <code>slen</code> and the subject interval (requires an <code>slen</code> column).

keep_pairs	Optional data.frame/list/matrix describing an undirected set of sequence pairs. A data.frame or matrix with the first two columns used as query/subject IDs, or a list of length-2 character vectors. A row is kept when its query/subject pair matches any pair in either direction.
drop_self_links	Logical, default TRUE. Remove rows where qaccver == saccver.
sort_by	Optional character vector of column names. Prefix a name with "-" or "desc:" for descending order (e.g. c("pident", "-length")).

Value

A list with data (the filtered data frame, with source_rows attribute) and report (n_input/n_kept/n_removed, removed_by_reason, removed_rows and kept_rows).

Examples

```
library(ggchord)
data(ribbon_data_example)
out <- filter_ggchord_ribbons(
  ribbon_data_example,
  min_pident = 95,
  drop_self_links = TRUE,
  sort_by = c("pident", "-length")
)
out$report
```

gene_data_example	<i>Example gene annotation data</i>
-------------------	-------------------------------------

Description

Gene annotation data for ggchord demonstration (short genes have been filtered out)

Usage

```
gene_data_example
```

Format

A data frame containing the following columns:

- seq_id: Sequence ID
- start: Gene start position
- end: Gene end position
- strand: Strand direction (+/-)
- anno: Gene annotation category

geom_axis

*Add an axis layer***Description**

Draws axes for each sequence in the chord diagram (including axis lines, major/minor ticks, and labels). Axis parameters (spacing, tick count/length, label size/orientation, etc.) are specified here.

Usage

```
geom_axis(
  mapping = NULL,
  data = NULL,
  show_axis = NULL,
  axis_gap = NULL,
  axis_tick_major_number = NULL,
  axis_tick_major_length = NULL,
  axis_tick_minor_number = NULL,
  axis_tick_minor_length = NULL,
  axis_label_size = NULL,
  axis_label_offset = NULL,
  axis_label_orientation = NULL,
  axis_label_hide_overlaps = FALSE,
  show_legend = FALSE,
  ...
)
```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
show_axis	Logical. Whether to show the axis, default TRUE
axis_gap	Optional numeric/vector. Spacing between sequence and axis, default 0.05
axis_tick_major_number	Optional integer/vector. Number of major ticks, default 3
axis_tick_major_length	Optional numeric/vector. Major tick length ratio, default 0.02
axis_tick_minor_number	Optional integer/vector. Number of minor ticks, default 4
axis_tick_minor_length	Optional numeric/vector. Minor tick length ratio, default 0.01
axis_label_size	Optional numeric/vector. Tick label font size, default 3
axis_label_offset	Optional numeric/vector. Label offset ratio, default 2

axis_label_orientation	Optional character/numeric/vector. Label orientation, default "parallel". Accepted values: "horizontal" (text stays horizontal), "parallel" (text runs parallel to the axis, i.e. along the arc), "perpendicular" (text runs perpendicular to the axis, i.e. along the radial direction), or a numeric angle in degrees (ggplot2 convention: counter-clockwise from horizontal, in the final rendered plot space). A vector or named vector can be used to specify a different orientation per sequence.
axis_label_hide_overlaps	Logical, default FALSE. When TRUE, axis labels whose boxes would overlap the plot content (sequence arcs, genes, ribbons) or other axis labels are automatically hidden.
show_legend	Whether to show the legend, default FALSE (axes do not participate in legends)
...	Additional arguments passed to geom_path/geom_segment/geom_text

Value

A list of ggplot2 layers

Examples

```
library(ggchord)
data(seq_data_example)
p <- ggchord(seq_data_example) + geom_seq() + geom_axis()
p
```

geom_feature

Draw generic genomic features

Description

A thin, backwards-compatible convenience layer for CDS, tRNA, rRNA, repeat, CRISPR, promoter or user-defined features. It prepares a gene-compatible table from a type / category / label specification and reuses the proven geom_gene() geometry and scales.

Usage

```
geom_feature(
  data,
  type = "type",
  category = NULL,
  label = "label",
  feature_colors = NULL,
  feature_width = NULL,
  feature_offset = NULL,
  feature_order = NULL,
  show_legend = TRUE,
```

```

    legend_position = "right",
    ...
)

```

Arguments

data	data.frame with seq_id, start, end and strand; optional type, category and label.
type	Column name used as the feature type, default "type".
category	Optional column name used for colour grouping; defaults to type.
label	Optional column name used for annotation text; defaults to label when present, otherwise type.
feature_colors	Optional named color vector by feature value; unnamed vectors are recycled positionally.
feature_width	Optional numeric or named vector controlling feature width; passed to geom_gene(gene_width = ...).
feature_offset	Optional numeric or named vector controlling feature offset; passed to geom_gene(gene_offset = ...).
feature_order	Optional feature order for the legend.
show_legend	Logical. Show the feature legend, default TRUE.
legend_position	Position of the feature legend: "left", "right", "top", "bottom" or "inside".
...	Additional arguments passed to geom_gene().

Value

A list of ggplot2 layers

Examples

```

library(ggchord)
data(seq_data_example)
features <- data.frame(seq_id = "MT108731.1",
                      start = 1000, end = 4000,
                      strand = "+", type = "CDS")
p <- ggchord(seq_data_example) + geom_seq() + geom_feature(features)
p

```

geom_gene

*Add a gene arrow layer***Description**

Draws gene annotation arrows on the chord diagram. Gene layout parameters (offset, width, color scheme, etc.) are specified here. The gene fill scale is kept independent from the ribbon's fill scale via a separate internal aesthetic used by the ribbon layer.

Usage

```
geom_gene(
  mapping = NULL,
  data = NULL,
  gene_offset = NULL,
  gene_width = NULL,
  gene_color_scheme = NULL,
  gene_colors = NULL,
  gene_order = NULL,
  show_legend = TRUE,
  legend_position = "right",
  ...
)
```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
gene_offset	Optional numeric/vector/list. Radial offset of gene arrows, default 0.1
gene_width	Optional numeric/vector/list. Width of gene arrows, default 0.05
gene_color_scheme	Character. "strand" or "manual", default "strand"
gene_colors	Optional color vector. Fill color of gene arrows
gene_order	Optional character vector. Display order of genes in the legend
show_legend	Whether to show the legend, default TRUE
legend_position	Position of this layer's legend (the Strand or Gene Annotation legend): one of "left", "right", "top", "bottom" or "inside", default "right". Pass NULL to let the legend follow theme(legend.position = ...) together with the other legends.
...	Additional arguments passed to geom_polygon()

Value

A list of ggplot2 layers. To annotate the genes with their labels, add a [geom_gene_label\(\)](#) layer.

Examples

```
library(ggchord)
data(seq_data_example)
data(gene_data_example)
p <- ggchord(seq_data_example, gene_data = gene_data_example) +
  geom_seq() + geom_gene()
p
```

geom_gene_label	<i>Add a gene label layer</i>
-----------------	-------------------------------

Description

Draws the gene annotation labels on a chord diagram. This layer is independent from `geom_gene()`: add it after `geom_gene()` to annotate the gene arrows with their texts.

Usage

```
geom_gene_label(
  mapping = NULL,
  data = NULL,
  gene_label_size = NULL,
  gene_label_rotation = NULL,
  gene_label_radial_offset = 0.04,
  gene_label_circum_offset = NULL,
  gene_label_circum_limit = NULL,
  gene_label_wrap = NULL,
  show_legend = FALSE,
  ...
)
```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
gene_label_size	Numeric. Label font size, default 2.5
gene_label_rotation	Optional numeric/vector/list. Label rotation angle, default 0
gene_label_radial_offset	Optional numeric/vector/list. Radial offset of labels, default 0.04
gene_label_circum_offset	Optional numeric/vector/list. Circumferential offset of labels, default 0
gene_label_circum_limit	Optional logical/vector/list. Whether to limit circumferential offset, default TRUE

gene_label_wrap	Numeric or NULL, default NULL. When set, long gene annotations are wrapped at this many characters (e.g. 15), which makes the labels narrower and less prone to overlap.
show_legend	Whether to show the legend, default FALSE
...	Additional arguments passed to geom_text()

Details

Long annotations can be wrapped with `gene_label_wrap`. For automatic de-overlapping (with leader lines), use `geom_gene_label_repel()` instead.

Value

A list of ggplot2 layers. To let the labels avoid each other and the genes (with leader lines), use `geom_gene_label_repel()` instead.

Examples

```
library(ggchord)
data(seq_data_example)
data(gene_data_example)
p <- ggchord(seq_data_example, gene_data = gene_data_example) +
  geom_seq() + geom_gene() + geom_gene_label()
p
```

`geom_gene_label_repel` *Add a repelled gene label layer (ggrepel-style)*

Description

Like `geom_gene_label()`, but the labels are placed with a force-based simulation that pushes them away from the genes and from each other (similar to `ggrepel::geom_text_repel()`). Labels that move far enough from their anchor are connected to it with a leader line, and labels that still overlap too many others can be hidden.

Usage

```
geom_gene_label_repel(
  mapping = NULL,
  data = NULL,
  gene_label_size = NULL,
  gene_label_rotation = NULL,
  gene_label_radial_offset = NULL,
  gene_label_circum_offset = NULL,
  gene_label_circum_limit = NULL,
  gene_label_wrap = NULL,
  max_overlaps = Inf,
```

```

    box_padding = 0.25,
    point_padding = 0.1,
    min_segment_length = 0.05,
    force = 1,
    seed = 123,
    gene_label_orientation = "horizontal",
    gene_label_segment = "elbow",
    gene_label_side = "outside",
    gene_label_segment_linetype = "auto",
    show_legend = FALSE,
    ...
)

```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
gene_label_size	Numeric. Label font size, default 2.5
gene_label_rotation	Optional numeric/vector/list. Label rotation angle, default 0
gene_label_radial_offset	Optional numeric/vector/list. Radial offset of labels, default 0
gene_label_circum_offset	Optional numeric/vector/list. Circumferential offset of labels, default 0
gene_label_circum_limit	Optional logical/vector/list. Whether to limit circumferential offset, default TRUE
gene_label_wrap	Numeric or NULL, default NULL. When set, long gene annotations are wrapped at this many characters (e.g. 15).
max_overlaps	Numeric, default Inf. Hide labels that still overlap more than this many other labels after repulsion (ggrepel-style decluttering). Use a finite value to clean up crowded plots.
box_padding	Numeric, default 0.25. Extra padding around each label box (data units).
point_padding	Numeric, default 0.1. Extra padding around the anchor points (data units).
min_segment_length	Numeric, default 0.05. Labels that moved less than this distance (data units) from their anchor do not draw a leader line. Keep it small so that every label is connected to its gene.
force	Numeric, default 1. Strength of the repulsive forces.
seed	Numeric, default 123. Random seed for reproducibility.
gene_label_orientation	Character, default "horizontal". One of "arc" (text rotated along the sequence arc) or "horizontal" (all labels are drawn horizontally).

gene_label_segment	Character, default "elbow". Leader line style: a straight "line" from the gene to the label, or an L-shaped "elbow" (a short segment outward, then a horizontal segment to the label). Elbow segment lengths adapt to each label's position and text width, so labels can be placed freely.
gene_label_side	Character, default "outside". Which side of the arc the labels sit on. "auto" keeps the strand-based placement (same as before); "outside" moves labels that would be inside the chord (where they can overlap the ribbons) to the outside of their arc; "inside" does the opposite. Labels moved to the other side are connected with a dashed leader line (see gene_label_segment_linetype).
gene_label_segment_linetype	Character or numeric, default "auto". Leader-line linetype. "auto" draws solid lines, except for labels that were moved to the other side of their arc, which are drawn dashed. Any other valid ggplot2 linetype (e.g. "solid", "dashed", "dotted", or a numeric dash pattern) is used for all leader lines.
show_legend	Whether to show the legend, default FALSE
...	Additional arguments passed to geom_text()

Value

A list of ggplot2 layers (a leader-line layer and a text layer).

Examples

```
library(ggchord)
data(seq_data_example)
data(gene_data_example)
p <- ggchord(seq_data_example, gene_data = gene_data_example) +
  geom_seq() + geom_gene() + geom_gene_label_repel()
p
```

geom_ribbon

Add an alignment ribbon layer

Description

Draws colored ribbons corresponding to alignment results. Color scheme and spacing parameters are specified here.

Usage

```
geom_ribbon(
  mapping = NULL,
  data = NULL,
  ribbon_color_scheme = NULL,
  ribbon_colors = NULL,
```

```

    ribbon_color_by = NULL,
    ribbon_color_limits = NULL,
    ribbon_color_breaks = NULL,
    ribbon_color_name = NULL,
    ribbon_alpha = NULL,
    ribbon_alpha_by = NULL,
    ribbon_alpha_range = c(0.15, 0.9),
    ribbon_ctrl_point = NULL,
    ribbon_gap = NULL,
    alpha = NULL,
    ribbon_outline_color = "black",
    ribbon_outline_width = 0.05,
    ribbon_outline_linetype = 1,
    ribbon_outline_by = NULL,
    ribbon_outline_colors = NULL,
    ribbon_linetype_by = NULL,
    ribbon_linetypes = NULL,
    ribbon_direction = c("none", "alpha", "outline", "linetype"),
    ribbon_direction_colors = c(same = "black", reverse = "grey50"),
    ribbon_direction_linetypes = c(same = "solid", reverse = "dashed"),
    ribbon_direction_alpha = c(same = 1, reverse = 0.45),
    show_legend = TRUE,
    legend_position = "left",
    legend_key_width = NULL,
    legend_key_height = NULL,
    ...
)

```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
ribbon_color_scheme	Character. Color scheme "pident", "query", "subject" or "single", default "pident"
ribbon_colors	Optional color vector. Ribbon color parameters
ribbon_color_by	Optional character column name. When set, ribbon fill is mapped to a continuous colourbar for that numeric column instead of pident (e.g. "bitscore").
ribbon_color_limits	Optional numeric length-2 limits for ribbon_color_by.
ribbon_color_breaks	Optional numeric breaks for the ribbon_color_by colourbar.
ribbon_color_name	Optional legend title for the ribbon_color_by colourbar (defaults to the column name).
ribbon_alpha	Numeric (0-1). Ribbon transparency, default 0.35

ribbon_alpha_by Optional character column name. When set, alpha is scaled continuously from that numeric column.

ribbon_alpha_range Numeric length-2. Alpha range used by `ribbon_alpha_by`, default `c(0.15, 0.9)`.

ribbon_ctrl_point Optional vector/list. Bezier control points, default `c(0,0)`

ribbon_gap Optional numeric/vector. Spacing between sequences and ribbons, default 0.15

alpha Ribbon transparency (overrides `ribbon_alpha`), defaults to the value used in the layout

ribbon_outline_color Character. Color of the ribbon outline (border), default "black"

ribbon_outline_width Numeric. Line width of the ribbon outline, default 0.05

ribbon_outline_linetype Numeric or character. Line type of the ribbon outline, default 1 (solid); see `linetype` in `ggplot2` for options

ribbon_outline_by Optional discrete column name. When set, outline colour is mapped by that column and `ribbon_outline_colors` controls the palette.

ribbon_outline_colors Optional named color vector for `ribbon_outline_by`; unnamed vectors are recycled positionally.

ribbon_linetype_by Optional discrete column name. When set, outline linetype is mapped by that column and `ribbon_linetypes` controls the values.

ribbon_linetypes Optional named linetype vector for `ribbon_linetype_by`.

ribbon_direction Character. How to visually distinguish same- vs reverse-orientation alignments: "none", "alpha", "outline" or "linetype".

ribbon_direction_colors Named color vector with same and reverse entries, used when `ribbon_direction = "outline"`.

ribbon_direction_linetypes Named linetype vector with same and reverse entries, used when `ribbon_direction = "linetype"`.

ribbon_direction_alpha Named numeric vector with same and reverse entries, used when `ribbon_direction = "alpha"`.

show_legend Whether to show the legend, default TRUE

legend_position Position of this layer's legend (the `Identity()` colourbar): one of "left", "right", "top", "bottom" or "inside", default "left". Pass NULL to let the legend follow `theme(legend.position = ...)` together with the other legends.

legend_key_width	Optional width of the Identity(Accepts a grid unit, e.g. <code>unit(1, "cm")</code> , or a number interpreted as centimetres. Default NULL uses the ggplot2 default width.
legend_key_height	Optional height of the Identity(Accepts a grid unit, e.g. <code>unit(5, "cm")</code> , or a number interpreted as centimetres. Default NULL lets the vertical bar fill the available height (and uses a fixed height for horizontal legends).
...	Additional arguments passed to <code>geom_polygon()</code>

Value

A list of ggplot2 layers

Examples

```
library(ggchord)
data(seq_data_example)
data(ribbon_data_example)
p <- ggchord(seq_data_example, ribbon_data_example) +
  geom_seq() + geom_ribbon()
p
```

`geom_ribbon_highlight` *Highlight selected alignment ribbons*

Description

Draws a second polygon layer on top of selected ribbons so they can be emphasized without changing the underlying Identity(done with safe, explicit filters (row numbers, query/subject IDs, pident and length ranges) or a predicate function.

Usage

```
geom_ribbon_highlight(
  mapping = NULL,
  data = NULL,
  ribbon_ids = NULL,
  qaccver = NULL,
  saccver = NULL,
  min_pident = NULL,
  max_pident = NULL,
  min_length = NULL,
  max_length = NULL,
  predicate = NULL,
  highlight_color = "#E11D48",
  highlight_alpha = 0.8,
  highlight_outline_color = NULL,
```

```

    highlight_outline_width = 0.3,
    show_legend = FALSE,
    ...
  )

```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
ribbon_ids	Optional integer vector of original ribbon row numbers to highlight.
qaccver	Optional character vector; only ribbons whose query ID is in this set are highlighted.
saccver	Optional character vector; only ribbons whose subject ID is in this set are highlighted.
min_pident	Optional numeric. Minimum percent identity.
max_pident	Optional numeric. Maximum percent identity.
min_length	Optional numeric. Minimum alignment length.
max_length	Optional numeric. Maximum alignment length.
predicate	Optional function taking the ribbon data.frame and returning a logical vector with one element per row. Evaluated safely (no string parsing).
highlight_color	Character. Highlight fill colour, default "#E11D48".
highlight_alpha	Numeric (0-1). Highlight alpha, default 0.8.
highlight_outline_color	Optional outline colour, default NULL.
highlight_outline_width	Numeric. Outline width, default 0.3.
show_legend	Logical. Whether to show a legend, default FALSE.
...	Additional arguments passed to geom_polygon()

Value

A list of ggplot2 layers

Examples

```

library(ggchord)
data(seq_data_example)
data(ribbon_data_example)
p <- ggchord(seq_data_example, ribbon_data_example) +
  geom_seq() + geom_ribbon() + geom_ribbon_highlight(ribbon_ids = 1)
p

```

geom_seq

*Add a sequence arc layer***Description**

Draws arcs (or straight lines, depending on the curvature setting) representing sequences in the chord diagram. Sequence layout parameters (order, orientation, radius, curvature, colors, etc.) are specified here. Sequences can be visually grouped with an extra inter-group gap and optional group labels.

Usage

```
geom_seq(
  mapping = NULL,
  data = NULL,
  seq_order = NULL,
  seq_labels = NULL,
  seq_orientation = NULL,
  seq_gap = NULL,
  seq_radius = NULL,
  seq_curvature = NULL,
  seq_colors = NULL,
  seq_group = NULL,
  seq_group_gap = 0.08,
  seq_group_labels = TRUE,
  seq_group_label_radius = 1.35,
  seq_group_colors = NULL,
  linewidth = 1.2,
  show_legend = TRUE,
  legend_position = "right",
  ...
)
```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
seq_order	Optional character vector. Specifies the drawing order of sequences
seq_labels	Optional character vector or named vector. Sequence labels
seq_orientation	Optional numeric (1 or -1). Sequence orientation, default 1
seq_gap	Optional numeric. Gap proportion between sequences, default 0.03
seq_radius	Optional numeric (> 0). Sequence arc radius, default 1.0
seq_curvature	Optional numeric. Arc curvature (0=straight, 1=standard arc, >1=more curved), default 1.0

seq_colors	Optional color vector or named vector. Sequence colors
seq_group	Optional group specification. NULL disables grouping unless seq_data contains a seq_group column; a single string naming such a column uses that column; otherwise a single value, named vector, unnamed vector matching the sequences, or a list (see process_sequence_param()) is accepted.
seq_group_gap	Numeric, default 0.08. Extra gap proportion inserted between consecutive groups (in addition to seq_gap).
seq_group_labels	Logical or character, default TRUE. When TRUE the group names are drawn at the angular midpoint of each group; a named character vector can be used to override the group label text.
seq_group_label_radius	Numeric, default 1.35. Radial position of the group labels as a multiplier of the group's outermost sequence radius (same convention as seq_label_radius: 1 on the arc, > 1 outside).
seq_group_colors	Optional named color vector by group name (or an unnamed vector recycled positionally). Colours the group labels.
linewidth	Arc line width, default 1.2
show_legend	Whether to show the legend for this layer, default TRUE
legend_position	Position of this layer's legend (the Seq ID legend): one of "left", "right", "top", "bottom" or "inside", default "right". Pass NULL to let the legend follow theme(legend.position = ...) together with the other legends. Can also be set with theme(legend.position.seq = ...).
...	Additional arguments passed to geom_path()

Value

A list of ggplot2 layers

Examples

```
library(ggchord)
data(seq_data_example)
p <- ggchord(seq_data_example) + geom_seq()
p
```

geom_seq_label	<i>Add a sequence label layer</i>
----------------	-----------------------------------

Description

Places sequence labels at the midpoint of each sequence arc, radially offset from the arc. Labels can be styled via their radial offset, rotation, font size, text orientation and justification.

Usage

```
geom_seq_label(
  mapping = NULL,
  data = NULL,
  seq_label_radius = 1,
  seq_label_rotation = NULL,
  seq_label_size = NULL,
  seq_labels = NULL,
  seq_label_orientation = c("arc", "horizontal"),
  seq_label_hjust = NULL,
  seq_label_vjust = NULL,
  check_overlap = FALSE,
  show_legend = FALSE,
  ...
)
```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Default NULL (retrieved automatically from the layout)
seq_label_radius	Optional numeric/vector. Radial position of the labels as a multiplier of the sequence arc radius: 1 is on the arc, > 1 places the label outside (away from the chord center) and < 1 places it inside, default 1
seq_label_rotation	Optional numeric/vector. Additional label rotation (degrees) on top of the arc-aligned orientation, default NULL (0). Ignored when seq_label_orientation = "horizontal".
seq_label_size	Optional numeric/vector. Label font size, default NULL (3)
seq_labels	Optional character vector. Override the label texts (defaults to the sequence labels from geom_seq() or the sequence IDs)
seq_label_orientation	Character, default "arc". Label text orientation: "arc" rotates the text along the sequence arc (and keeps it readable), "horizontal" draws every label horizontally, extending away from the chord center.
seq_label_hjust	Optional numeric/vector. Horizontal justification of the labels, default NULL. The default arc orientation uses -0.2 so the text sits just inside the sequence; with seq_label_orientation = "horizontal" the justification is chosen automatically so the text extends away from the chord center.
seq_label_vjust	Optional numeric/vector. Vertical justification of the labels, default NULL (0.5).
check_overlap	Logical, default FALSE. When TRUE, labels that would overlap a previously drawn label are skipped (ggplot2's geom_text() option).
show_legend	Whether to show the legend, default FALSE
...	Additional arguments passed to geom_text()

Value

A list of ggplot2 layers

Examples

```
library(ggchord)
data(seq_data_example)
p <- ggchord(seq_data_example) + geom_seq() + geom_seq_label()
p
```

geom_seq_region	<i>Highlight regions along sequence arcs</i>
-----------------	--

Description

Draws rectangular bands on sequence arcs for one or more coordinate intervals. This is useful for marking loci, repeats, CRISPR arrays, or other user-defined regions without turning them into gene arrows.

Usage

```
geom_seq_region(
  mapping = NULL,
  data = NULL,
  regions = NULL,
  region_fill = "#F59E0B",
  region_color = "#B45309",
  region_alpha = 0.25,
  region_width = 0.08,
  region_offset = 0,
  region_side = c("inside", "outside", "auto"),
  show_legend = FALSE,
  ...
)
```

Arguments

mapping	Default NULL (uses pre-computed data)
data	Optional data.frame; an alias for regions when regions is NULL.
regions	data.frame with at least seq_id, start, end; optional label, category and color.
region_fill	Character. Default fill colour for regions, default "#F59E0B".
region_color	Character. Outline colour, default "#B45309".
region_alpha	Numeric (0-1). Region alpha, default 0.25.
region_width	Numeric. Band width in chord radius units, default 0.08.

region_offset	Numeric. Radial offset from the sequence arc, default 0.
region_side	Character. "inside", "outside" or "auto"; "auto" places regions inside the chord when possible.
show_legend	Logical. Whether to show a legend for category colours, default FALSE.
...	Additional arguments passed to geom_polygon()

Value

A list of ggplot2 layers

Examples

```
library(ggchord)
data(seq_data_example)
regions <- data.frame(seq_id = "MT108731.1",
                      start = 1000, end = 4000,
                      color = "orange")
p <- ggchord(seq_data_example) + geom_seq() + geom_seq_region(regions)
p
```

get_chord_layout

Get the chord layout from the package environment

Description

Returns the most recently computed chord layout (after the plot was built, e.g. via print() or ggplot_build()). This is useful for building custom layers or annotations on top of the chord geometry.

Usage

```
get_chord_layout()
```

Value

A chord layout list containing the computed geometry (sequence arcs, ribbon polygons, gene arrows, axis elements, extremes, colors, etc.)

Examples

```
library(ggchord)
data(seq_data_example)
data(ribbon_data_example)
p <- ggchord(seq_data_example, ribbon_data_example) + geom_seq() + geom_ribbon()
invisible(ggplot2::ggplot_build(p))
names(get_chord_layout())$seq_arcs
```

ggchord	<i>ggchord: layered multi-sequence alignment chord diagrams for ggplot2</i>
---------	---

Description

ggchord visualizes multi-sequence alignment results using ggplot2's layered grammar. The ggchord() constructor handles data validation and global settings; the geom_* layers are stacked as needed, each responsible for its own layout parameters and visual rendering. The layout is computed lazily when the plot is built (e.g. via print(), ggsave(), or ggplot_build()).

Usage

```
ggchord(
  seq_data,
  ribbon_data = NULL,
  gene_data = NULL,
  title = NULL,
  rotation = 45,
  panel_margin = 0,
  show_legend = TRUE,
  debug = FALSE,
  validate = c("warn", "error", "none")
)
```

Arguments

seq_data	data.frame/tibble, required. Basic sequence information
ribbon_data	data.frame/tibble, optional. Alignment results
gene_data	data.frame/tibble, optional. Gene annotation data
title	Character. Main title of the plot, default NULL
rotation	Numeric. Global rotation angle (degrees), default 45
panel_margin	Optional numeric/list. Panel margin, default 0
show_legend	Logical. Whether to show legends, default TRUE
debug	Logical. Whether to output debug information, default FALSE
validate	Character, default "warn". How to run the structured input-data validation (see validate_ggchord_data): "warn" emits a single summary warning when the data has problems and caches the full report on the plot object (p\$ggchord\$validation); "error" stops on severe problems; "none" skips the diagnostic validation (the cheap structural checks that prevent crashes are still performed).

Value

A ggchord object (inherits from ggplot) to which geom_* layers can be added with +

Examples

```
library(ggchord)
data(seq_data_example)
data(ribbon_data_example)
data(gene_data_example)

p <- ggchord(
  seq_data = seq_data_example,
  ribbon_data = ribbon_data_example,
  gene_data = gene_data_example
) +
  geom_seq() +
  geom_ribbon() +
  geom_gene() +
  geom_axis()
print(p)
```

`ggplotly.ggchord`*Convert a ggchord plot to a plotly object*

Description

`ggchord` provides an S3 method for `plotly::ggplotly()` so that chord diagrams built with `ggchord()` (including plots that combine the ribbon and the gene layers) can be converted to interactive **plotly** charts. The conversion first renders the computed geometry with standard `ggplot2` layers whose colors are pre-mapped, then delegates to `plotly::ggplotly()`.

Usage

```
## S3 method for class 'ggchord'
ggplotly(p, ...)
```

Arguments

<code>p</code>	A <code>ggchord</code> plot object created with <code>ggchord()</code> .
<code>...</code>	Additional arguments passed to <code>plotly::ggplotly()</code> .

Value

A `plotly` object.

merge_ggchord_ribbons *Merge adjacent or overlapping alignment blocks of the same sequence pair*

Description

Merges alignment blocks that belong to the same (query, subject) pair, are adjacent ($\text{gap} \leq \text{max_gap}$) or overlapping on both sequences, and are compatible. The merged pident is weighted by alignment length. Merging is deliberately conservative: blocks whose merged query and subject spans would be inconsistent (unequal), whose pident differs by more than `min_pident_difference`, or whose orientation differs (when `require_same_orientation`) are left unmerged.

Usage

```
merge_ggchord_ribbons(
  ribbon_data,
  max_gap = 0,
  min_pident_difference = 0,
  require_same_orientation = TRUE,
  group_by = c("qaccver", "saccver")
)
```

Arguments

<code>ribbon_data</code>	data.frame in ribbon_data format.
<code>max_gap</code>	Numeric, default 0. Maximum gap (in bp) allowed between two blocks on both sequences for them to be merged.
<code>min_pident_difference</code>	Numeric, default 0. When > 0 , two blocks are only merged when their pident difference is $\leq$ this value.
<code>require_same_orientation</code>	Logical, default TRUE. Only merge blocks whose query/subject direction is the same (both ascending or both descending, i.e. collinear or both inverted in the same way).
<code>group_by</code>	Character vector, default <code>c("qaccver", "saccver")</code> . Columns used to identify the same sequence pair.

Value

A list with data (merged data frame with `source_rows` attribute) and report (data.frame with `output_row`, `from_rows` and `n_merged` for every output row).

Examples

```
library(ggchord)
# Two adjacent, collinear blocks of the same pair -> one merged block
rb <- data.frame(
```

```

qaccver = c("A", "A"), saccver = c("B", "B"),
length = c(100, 100), pident = c(95, 97),
qstart = c(1, 101), qend = c(100, 200),
sstart = c(501, 601), send = c(600, 700)
)
out <- merge_ggchord_ribbons(rb, max_gap = 0)
out$data
out$report

```

```
print.ggchord_clean
```

Print a cleaned ggchord data result

Description

Shows the dimensions of the cleaned tables and the first rows of the change report.

Usage

```
## S3 method for class 'ggchord_clean'
print(x, ...)
```

Arguments

x	A "ggchord_clean" object.
...	Ignored.

Value

The object invisibly.

```
read_blast
```

Read one or more BLAST tabular output files into ribbon_data format

Description

‘file’ reads a single BLAST tabular output file; ‘files’ reads multiple files at once and combines them. ‘files’ accepts a character vector of literal paths and/or wildcard patterns (e.g. “examples/blastn/*.o7”).

Usage

```

read_blast(
  file = NULL,
  files = NULL,
  format = c("auto", "outfmt6", "outfmt7", "custom"),
  col_names = NULL,
  comment = "#",
  ...
)

```

Arguments

file	Optional path to a single BLAST tabular output file.
files	Optional character vector of BLAST tabular output files (literal paths and/or wildcard patterns). All matched files are read and combined into one data.frame.
format	Character. "auto" (default) detects the column layout from the number of columns; "outfmt6" / "outfmt7" require the standard 12/17-column layouts; "custom" requires 'col_names'.
col_names	Optional character vector naming the columns in the file, used with 'format = "custom"' or to override auto-detection.
comment	Character comment character, default "#" (BLAST outfmt 7 header lines start with '#').
...	Additional arguments passed to [utils::read.table()] (e.g. 'na.strings').

Value

A data.frame with the required ribbon columns first ('qaccver', 'saccver', 'length', 'pident', 'qstart', 'qend', 'sstart', 'send') followed by any preserved optional columns.

Examples

```
library(ggchord)
blast_file <- tempfile(fileext = ".o7")
writeLines(c(
  "# BLASTN 2.13.0+",
  "seqA seqB 98.5 1200 18 0 1 1200 1 1200 1e-180 2400"
), blast_file)
read_blast(blast_file)
```

read_fasta_lengths	<i>Read one or more FASTA files into seq_data format</i>
--------------------	--

Description

'file' reads a single FASTA file; 'files' reads multiple files at once and combines them. 'files' accepts a character vector of literal paths and/or wildcard patterns (e.g. "examples/fasta/*.fna").

Usage

```
read_fasta_lengths(file = NULL, files = NULL, header_delim = NULL)
```

Arguments

file	Optional path to a single FASTA file.
files	Optional character vector of FASTA files (literal paths and/or wildcard patterns). All matched files are read and combined.
header_delim	Optional character. When given, each header is split at every occurrence of this delimiter and only the first piece is kept.

Value

A data.frame with columns 'seq_id' and 'length'.

Examples

```
library(ggchord)
fasta <- tempfile(fileext = ".fna")
writeLines(c(">seqA some description", "ACGTACGTACGTACGT", "ACGTACGT",
            ">seqB", "TTTTGGGG"), fasta)
read_fasta_lengths(fasta)
```

read_gff3	<i>Read one or more GFF3 files into gene_data format</i>
-----------	--

Description

'file' reads a single GFF3 file; 'files' reads multiple files at once and combines them. 'files' accepts a character vector of literal paths and/or wildcard patterns (e.g. "examples/gff3/*.gff3").

Usage

```
read_gff3(
  file = NULL,
  files = NULL,
  feature_types = "CDS",
  anno_from = c("product", "Name", "gene", "ID"),
  unstranded = c("plus", "drop")
)
```

Arguments

file	Optional path to a single GFF3 file.
files	Optional character vector of GFF3 files (literal paths and/or wildcard patterns). All matched files are read and combined.
feature_types	Character vector of feature types to keep, default "CDS".
anno_from	Character vector of GFF3 attribute keys, tried in order to fill the 'anno' column.
unstranded	Character, default "plus".

Value

A data.frame with 'seq_id', 'start', 'end', 'strand', 'anno' followed by 'type', 'source', 'score', 'phase' and 'attributes'.

Examples

```
library(ggchord)
gff <- tempfile(fileext = ".gff3")
writeLines(c(
  "##gff-version 3",
  paste0("seqA source CDS 101 500 . + 0 ",
    "ID=cds1;product=hypothetical protein")
), gff)
read_gff3(gff)
```

ribbon_data_example	<i>Example alignment data</i>
---------------------	-------------------------------

Description

Alignment data for ggchord demonstration (length >= 100)

Usage

```
ribbon_data_example
```

Format

A data frame containing standard alignment columns (qaccver, saccver, pident, etc.)

seq_data_example	<i>Example sequence data</i>
------------------	------------------------------

Description

Sequence length data for ggchord demonstration

Usage

```
seq_data_example
```

Format

A data frame containing columns: seq_id, length

 validate_ggchord_data *Validate ggchord input data before plotting*

Description

Performs structured validation of seq_data, ribbon_data and gene_data so that problems can be found, understood and fixed before plotting. The result is a "ggchord_validation" object with a valid flag, errors (severe problems that make the plot misleading), warnings (drawable but noteworthy issues), per-category counts, a data summary, the original row numbers of every problem, and a list of automatically fixable issues.

Usage

```
validate_ggchord_data(
  seq_data,
  ribbon_data = NULL,
  gene_data = NULL,
  strict = FALSE,
  check_coordinates = TRUE,
  check_duplicates = TRUE,
  check_self_links = TRUE
)
```

Arguments

seq_data	data.frame/tibble, required. Basic sequence information (columns seq_id, length).
ribbon_data	data.frame/tibble, optional. Alignment results (columns qaccver, saccver, length, pident, qstart, qend, sstart, send).
gene_data	data.frame/tibble, optional. Gene annotation data (columns seq_id, start, end, strand, anno).
strict	Logical. When TRUE, stop with an error as soon as any severe problem is found. When FALSE (default), return the full diagnostic report without stopping.
check_coordinates	Logical, default TRUE. Whether to check that ribbon/gene coordinates stay inside [1, sequence length].
check_duplicates	Logical, default TRUE. Whether to look for fully duplicated, near-duplicated and highly overlapping records.
check_self_links	Logical, default TRUE. Whether to flag alignment rows where qaccver == saccver.

Value

A "ggchord_validation" object (a list) with at least:

valid Logical: TRUE when there are no severe errors.

`errors` data.frame of severe issues (table, category, row, column, message).
`warnings` data.frame of non-severe issues (same columns).
`summary` Per-category counts (table, category, severity, n).
`data_summary` Counts of sequences/ribbons/genes, unknown IDs, out-of-range rows, etc.
`invalid_rows` Named list of original row numbers per problem category.
`cleanable` data.frame of fixable issues with suggested actions.
`print()` and `summary()` methods are provided.

Examples

```
library(ggchord)
data(seq_data_example)
data(ribbon_data_example)
data(gene_data_example)

res <- validate_ggchord_data(seq_data_example, ribbon_data_example,
                             gene_data_example)

res$valid
print(res)
summary(res)

# Introduce a problem: unknown sequence ID in the ribbons
bad <- transform(ribbon_data_example, saccver = "NOT_A_SEQUENCE")
v <- validate_ggchord_data(seq_data_example, bad)
v$invalid_rows$ribbon_unknown_id
```

Index

* datasets

- gene_data_example, [8](#)
- ribbon_data_example, [32](#)
- seq_data_example, [32](#)
- +.ggchord, [2](#)
- as.data.frame.ggchord_validation, [3](#)
- clean_ggchord_data, [3](#)
- coord_chord, [5](#)
- deduplicate_ggchord_ribbons, [6](#)
- filter_ggchord_ribbons, [7](#)
- gene_data_example, [8](#)
- geom_axis, [9](#)
- geom_feature, [10](#)
- geom_gene, [12](#), [13](#)
- geom_gene_label, [12](#), [13](#), [14](#)
- geom_gene_label_repel, [14](#), [14](#)
- geom_ribbon, [16](#)
- geom_ribbon_highlight, [19](#)
- geom_seq, [21](#)
- geom_seq_label, [22](#)
- geom_seq_region, [24](#)
- get_chord_layout, [25](#)
- ggchord, [26](#)
- ggplotly.ggchord, [27](#)
- merge_ggchord_ribbons, [28](#)
- print.ggchord_clean, [29](#)
- read_blast, [29](#)
- read_fasta_lengths, [30](#)
- read_gff3, [31](#)
- ribbon_data_example, [32](#)
- seq_data_example, [32](#)
- validate_ggchord_data, [26](#), [33](#)