# Package ‘ggalluvial’

April 16, 2020

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Alluvial Plots in 'ggplot2'</td>
</tr>
<tr>
<td>Version</td>
<td>0.11.3</td>
</tr>
<tr>
<td>Date</td>
<td>2020-03-29</td>
</tr>
<tr>
<td>Maintainer</td>
<td>Jason Cory Brunson <a href="mailto:cornelioid@gmail.com">cornelioid@gmail.com</a></td>
</tr>
<tr>
<td>Depends</td>
<td>R (&gt;= 3.3), ggplot2 (&gt;= 2.2)</td>
</tr>
<tr>
<td>Imports</td>
<td>stats, dplyr (&gt;= 0.7), tidyr (&gt;= 0.7), lazyeval, rlang, tidyselect</td>
</tr>
<tr>
<td>Suggests</td>
<td>grid, alluvial, testthat, knitr, babynames, sessioninfo, mgrepel, ggfittext (&gt;= 0.6), vdiffr (&gt;= 0.2)</td>
</tr>
<tr>
<td>License</td>
<td>GPL-3</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://corybrunson.github.io/ggalluvial/">http://corybrunson.github.io/ggalluvial/</a></td>
</tr>
<tr>
<td>BugReports</td>
<td><a href="https://github.com/corybrunson/ggalluvial/issues">https://github.com/corybrunson/ggalluvial/issues</a></td>
</tr>
<tr>
<td>VignetteBuilder</td>
<td>knitr</td>
</tr>
<tr>
<td>RoxygenNote</td>
<td>7.1.0</td>
</tr>
<tr>
<td>NeedsCompilation</td>
<td>no</td>
</tr>
<tr>
<td>Author</td>
<td>Jason Cory Brunson [aut, cre]</td>
</tr>
<tr>
<td>Repository</td>
<td>CRAN</td>
</tr>
<tr>
<td>Date/Publication</td>
<td>2020-04-16 14:10:02 UTC</td>
</tr>
</tbody>
</table>
Description

Alluvial plots consist of multiple horizontally-distributed columns (axes) representing factor variables, vertical divisions (strata) of these axes representing these variables’ values; and splines (alluvial flows) connecting vertical subdivisions (lodes) within strata of adjacent axes representing subsets or amounts of observations that take the corresponding values of the corresponding variables. This function checks a data frame for either of two types of alluvial structure:

Usage

```r
is_lodes_form(
  data,
  key,
  value,
  id,
  weight = NULL,
  logical = TRUE,
  silent = FALSE
)
```

```r
is_alluvia_form(
  data,
  ...,
  axes = NULL,
  weight = NULL,
  logical = TRUE,
  silent = FALSE
)
```
to_lodes_form(
data,
..., axes = NULL,
key = "x",
value = "stratum",
id = "alluvium",
diffuse = FALSE,
discern = FALSE)
)
to_alluvia_form(data, key, value, id, distill = FALSE)

Arguments

data A data frame.
key, value, id In to_lodes_form, handled as in tidyr::gather() and used to name the new axis (key), stratum (value), and alluvium (identifying) variables. In to_alluvia_form, handled as in tidyr::spread() and used to identify the fields of data to be used as the axis (key), stratum (value), and alluvium (identifying) variables.
weight Optional field of data, handled using rlang::enquo(), to be used as heights or depths of the alluvia or lodes.
logical Defunct. Whether to return a logical value or a character string indicating the type of alluvial structure ("none", "lodes", or "alluvia").
silent Whether to print messages.
... Used in is_alluvia_form and to_lodes_form as in dplyr::select() to determine axis variables, as an alternative to axes. Ignored when axes is provided.
axes In *_alluvia_form, handled as in dplyr::select() and used to identify the field(s) of data to be used as axes.
diffuse Fields of data, handled using tidyselect::vars_select(), to merge into the reshaped data by id. They must be a subset of the axis variables. Alternatively, a logical value indicating whether to merge all (TRUE) or none (FALSE) of the axis variables.
discern Logical value indicating whether to suffix values of the variables used as axes that appear at more than one variable in order to distinguish their factor levels. This forces the levels of the combined factor variable value to be in the order of the axes.
distill A logical value indicating whether to include variables, other than those passed to key and value, that vary within values of id. Alternatively, a function (or its name) to be used to distill each such variable to a single value. In addition to existing functions, distill accepts the character values "first" (used if distill is TRUE), "last", and "most" (which returns the modal value).
Details

• One row per lode, wherein each row encodes a subset or amount of observations having a specific profile of axis values, a key field encodes the axis, a value field encodes the value within each axis, and an id column identifies multiple lodes corresponding to the same subset or amount of observations. is_lodes_form tests for this structure.

• One row per alluvium, wherein each row encodes a subset or amount of observations having a specific profile of axis values and a set of axes of fields encodes its values at each axis variable. is_alluvia_form tests for this structure.

to_lodes_form takes a data frame with several designated variables to be used as axes in an alluvial plot, and reshapes the data frame so that the axis variable names constitute a new factor variable and their values comprise another. Other variables’ values will be repeated, and a row-grouping variable can be introduced. This function invokes tidyr::gather().

to_alluvia_form takes a data frame with axis and axis value variables to be used in an alluvial plot, and reshape the data frame so that the axes constitute separate variables whose values are given by the value variable. This function invokes tidyr::spread().

See Also

Other alluvial data manipulation: self-adjoin

Examples

# Titanic data in alluvia format
titanic_alluvia <- as.data.frame(Titanic)
head(titanic_alluvia)
is_alluvia_form(titanic_alluvia, 
weight = "Freq")

# Titanic data in lodes format
titanic_lodes <- to_lodes_form(titanic_alluvia, 
key = "x", value = "stratum", id = "alluvium", 
axes = 1:4)
head(titanic_lodes)
is_lodes_form(titanic_lodes, 
key = "x", value = "stratum", id = "alluvium", 
weight = "Freq")

# again in lodes format, this time diffusing the `Class` variable
titanic_lodes2 <- to_lodes_form(titanic_alluvia, 
key = variable, value = value, 
id = passenger, 
1:3, diffuse = Class)
head(titanic_lodes2)
is_lodes_form(titanic_lodes2, 
key = variable, value = value, id = passenger, 
weight = Freq)

# curriculum data in lodes format
data(majors)
head(majors)
is_lodes_form(majors, 
key = "semester", value = "curriculum", id = "student")
# curriculum data in alluvia format
majors_alluvia <- to_alluvia_form(majors,
                               key = "semester", value = "curriculum",
                               id = "student")

head(majors_alluvia)
is_alluvia_form(majors_alluvia, tidyselect::starts_with("CURR"))

# distill variables that vary within `id` values
set.seed(1)
majors$hypo_grade <- LETTERS[sample(5, size = nrow(majors), replace = TRUE)]
majors_alluvia2 <- to_alluvia_form(majors,
                                key = "semester", value = "curriculum",
                                id = "student",
                                distill = "most")

head(majors_alluvia2)

# options to distinguish strata at different axes
gg <- ggplot(majors_alluvia,
             aes(axis1 = CURR1, axis2 = CURR7, axis3 = CURR13))

gg +
  geom_alluvium(aes(fill = as.factor(student)), width = 2/5, discern = TRUE) +
  geom_stratum(width = 2/5, discern = TRUE) +
  geom_text(stat = "stratum", discern = TRUE, infer.label = TRUE)

gg +
  geom_alluvium(aes(fill = as.factor(student)), width = 2/5, discern = FALSE) +
  geom_stratum(width = 2/5, discern = FALSE) +
  geom_text(stat = "stratum", discern = FALSE, infer.label = TRUE)

# warning when inappropriate
ggplot(majors[majors$semester %in% paste0("CURR", c(1, 7, 13)), ],
       aes(x = semester, stratum = curriculum, alluvium = student,
           label = curriculum)) +
  geom_alluvium(aes(fill = as.factor(student)), width = 2/5, discern = TRUE) +
  geom_stratum(width = 2/5, discern = TRUE) +
  geom_text(stat = "stratum", discern = TRUE)

---

**geom_alluvium**

**Alluvia across strata**

**Description**

`geom_alluvium` receives a dataset of the horizontal (x) and vertical (y, ymin, ymax) positions of the **lodes** of an alluvial plot, the intersections of the alluvia with the strata. It plots both the lodes themselves, using `geom_lode()`, and the flows between them, using `geom_flow()`.

**Usage**

```r
geom_alluvium(
  mapping = NULL,
  data = NULL,
  stat = "alluvium",
```

position = "identity",
width = 1/3,
knot.pos = 1/6,
na.rm = FALSE,
show.legend = NA,
inherit.aes = TRUE,
... )

Arguments

mapping Set of aesthetic mappings created by `aes()` or `aes()`. If specified and `inherit.aes` = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

data The data to be displayed in this layer. There are three options:
If NULL, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.
A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.
A function will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x, 10)`).

stat The statistical transformation to use on the data; override the default.

position Position adjustment, either as a string, or the result of a call to a position adjustment function.

width Numeric; the width of each stratum, as a proportion of the distance between axes. Defaults to 1/3.

knot.pos The horizontal distance between a stratum (width/2 from its axis) and the knot of the x-spline, as a proportion of the separation between strata. Defaults to 1/6.

na.rm Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).

show.legend logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.

inherit.aes If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.

... Additional arguments passed to `ggplot2::layer()`.

Aesthetics

`geom_alluvium`, `geom_flow`, `geom_lode`, and `geom_stratum` understand the following aesthetics (required aesthetics are in bold):

• x
geom_alluvium

- y
- ymin
- ymax
- alpha
- colour
- fill
- linetype
- size
- group

group is used internally; arguments are ignored.

Defunct parameters

The previously defunct parameters axis_width and ribbon_bend have been discontinued. Use width and knot.pos instead.

See Also

ggplot2::layer() for additional arguments and stat_alluvium() and stat_flow() for the corresponding stats.

Other alluvial geom layers: geom_flow(), geom_lode(), geom_stratum()

Examples

# basic
ggplot(as.data.frame(Titanic),
       aes(y = Freq,
           axis1 = Class, axis2 = Sex, axis3 = Age,
           fill = Survived)) +
  geom_alluvium() +
  scale_x_discrete(limits = c("Class", "Sex", "Age"))

gg <- ggplot(alluvial::Refugees,
              aes(y = refugees, x = year, alluvium = country))
# time series bump chart
gg + geom_alluvium(aes(fill = country, colour = country),
                   width = 1/4, alpha = 2/3, decreasing = FALSE)
# time series line plot of refugees data, sorted by country
gg + geom_alluvium(aes(fill = country, colour = country),
                   decreasing = NA, width = 0, knot.pos = 0)
Description

`geom_flow` receives a dataset of the horizontal \((x)\) and vertical \((y, ymin, ymax)\) positions of the lodes of an alluvial plot, the intersections of the alluvia with the strata. It reconfigures these into alluvial segments connecting pairs of corresponding lodes in adjacent strata and plots filled x-splines between each such pair, using a provided knot position parameter `knot.pos`, and filled rectangles at either end, using a provided `width`.

Usage

```r
geom_flow(
  mapping = NULL,
  data = NULL,
  stat = "flow",
  position = "identity",
  width = 1/3,
  knot.pos = 1/6,
  aes.flow = "forward",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

- `mapping`: Set of aesthetic mappings created by `aes()` or `aes()`. If specified and `inherit.aes` = `TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
- `data`: The data to be displayed in this layer. There are three options:
  - If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.
  - A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.
  - A function will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A function can be created from a `formula` (e.g., `~ head(.x,10)`).
- `stat`: The statistical transformation to use on the data; override the default.
- `position`: Position adjustment, either as a string, or the result of a call to a position adjustment function.
- `width`: Numeric; the width of each stratum, as a proportion of the distance between axes. Defaults to 1/3.
**geom_flow**

- **knot.pos** - The horizontal distance between a stratum (width/2 from its axis) and the knot of the x-spline, as a proportion of the separation between strata. Defaults to 1/6.
- **aes.flow** - Character; how inter-lode flows assume aesthetics from lodes. Options are "forward" and "backward".
- **na.rm** - Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).
- **show.legend** - Logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.
- **inherit.aes** - If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.
- **...** - Additional arguments passed to `ggplot2::layer()`.

**Aesthetics**

`geom_alluvium`, `geom_flow`, `geom_lode`, and `geom_stratum` understand the following aesthetics (required aesthetics are in bold):

- **x**
- **y**
- **ymin**
- **ymax**
- **alpha**
- **colour**
- **fill**
- **linetype**
- **size**
- **group**

`group` is used internally; arguments are ignored.

**Defunct parameters**

The previously defunct parameters axis_width and ribbon_bend have been discontinued. Use `width` and `knot.pos` instead.

**See Also**

- `ggplot2::layer()` for additional arguments and `stat_alluvium()` and `stat_flow()` for the corresponding stats.
- Other alluvial geom layers: `geom_alluvium()`, `geom_lode()`, `geom_stratum()`
Examples

```r
# use of strata and labels
geom_lode
ggplot(as.data.frame(Titanic),
  aes(y = Freq,
      axis1 = Class, axis2 = Sex, axis3 = Age)) +
  geom_flow() +
  scale_x_discrete(limits = c("Class", "Sex", "Age")) +
  geom_stratum() + geom_text(stat = "stratum", infer.label = TRUE) +
  ggtitle("Alluvial plot of Titanic passenger demographic data")

# use of facets
geom_lode
ggplot(as.data.frame(Titanic),
  aes(y = Freq,
      axis1 = Class, axis2 = Sex)) +
  geom_flow(aes(fill = Age), width = .4) +
  geom_stratum(width = .4) +
  geom_text(stat = "stratum", infer.label = TRUE, size = 3) +
  scale_x_discrete(limits = c("Class", "Sex")) +
  facet_wrap(~ Survived, scales = "fixed")

# time series alluvia of WorldPhones data
wph <- as.data.frame(as.table(WorldPhones))
names(wph) <- c("Year", "Region", "Telephones")
geom_lode
ggplot(wph,
  aes(x = Year, alluvium = Region, y = Telephones)) +
  geom_flow(aes(fill = Region, colour = Region), width = 0)

# rightward flow aesthetics for vaccine survey data
data(vaccinations)
levels(vaccinations$response) <- rev(levels(vaccinations$response))
geom_lode
ggplot(vaccinations,
  aes(x = survey, stratum = response, alluvium = subject,
      y = freq, fill = response, label = round(a, 3))) +
  geom_lode() + geom_flow() +
  geom_stratum(alpha = 0) +
  geom_text(stat = "stratum")
```

Description

`geom_alluvium` receives a dataset of the horizontal (x) and vertical (y, ymin, ymax) positions of the lodes of an alluvial plot, the intersections of the alluvia with the strata. It plots rectangles for these lodes of a provided width.

Usage

```r
geom_lode()
```
geom_lode

mapping = NULL,
data = NULL,
stat = "alluvium",
position = "identity",
width = 1/3,
na.rm = FALSE,
show.legend = NA,
inherit.aes = TRUE,
...
)

Arguments

mapping  Set of aesthetic mappings created by aes() or aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data  The data to be displayed in this layer. There are three options:
If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot().
A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created.
A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x,10)).
stat  The statistical transformation to use on the data; override the default.
position  Position adjustment, either as a string, or the result of a call to a position adjustment function.
width  Numeric; the width of each stratum, as a proportion of the distance between axes. Defaults to 1/3.
na.rm  Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).
show.legend  logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes  If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. borders().
...
Additional arguments passed to ggplot2::layer().

Aesthetics

geom_alluvium, geom_flow, geom_lode, and geom_stratum understand the following aesthetics (required aesthetics are in bold):

* x
geom_lode

- y
- ymin
- ymax
- alpha
- colour
- fill
- linetype
- size
- group

group is used internally; arguments are ignored.

Defunct parameters

The previously defunct parameters axis_width and ribbon_bend have been discontinued. Use width and knot.pos instead.

See Also

ggplot2::layer() for additional arguments and stat_alluvium() and stat_stratum() for the corresponding stats.

Other alluvial geom layers: geom_alluvium(), geom_flow(), geom_stratum()

Examples

# one axis
ggplot(as.data.frame(Titanic),
       aes(y = Freq,
           axis = Class)) +
geom_lode(aes(fill = Class, alpha = Survived)) +
scale_x_discrete(limits = c("Class")) +
scale_alpha_manual(values = c(.25, .75))

gg <- ggplot(as.data.frame(Titanic),
             aes(y = Freq,
                 axis1 = Class, axis2 = Sex, axis3 = Age,
                 fill = Survived))
# alluvia and lodes
gg + geom_alluvium() + geom_lode()
# lodes as strata
gg + geom_alluvium() +
geom_stratum(stat = "alluvium")
**geom_stratum**

**Strata at axes**

**Description**

`geom_stratum` receives a dataset of the horizontal (x) and vertical (y, ymin, ymax) positions of the strata of an alluvial plot. It plots rectangles for these strata of a provided width.

**Usage**

```r
gem_stratum(
    mapping = NULL,
    data = NULL,
    stat = "stratum",
    position = "identity",
    show.legend = NA,
    inherit.aes = TRUE,
    width = 1/3,
    na.rm = FALSE,
    ...
)
```

**Arguments**

- `mapping` Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

- `data` The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x, 10)`).

- `stat` The statistical transformation to use on the data; override the default.

- `position` Position adjustment, either as a string, or the result of a call to a position adjustment function.

- `show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

- `inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.
width Numeric; the width of each stratum, as a proportion of the distance between axes. Defaults to 1/3.
na.rm Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).
... Additional arguments passed to `ggplot2::layer()`.

Aesthetics

`geom_alluvium`, `geom_flow`, `geom_lode`, and `geom_stratum` understand the following aesthetics (required aesthetics are in bold):

- x
- y
- ymin
- ymax
- alpha
- colour
- fill
- linetype
- size
- group

`group` is used internally; arguments are ignored.

Defunct parameters

The previously defunct parameters `axis_width` and `ribbon_bend` have been discontinued. Use `width` and `knot.pos` instead.

See Also

`ggplot2::layer()` for additional arguments and `stat_stratum()` for the corresponding stat.

Other alluvial geom layers: `geom_alluvium()`, `geom_flow()`, `geom_lode()`

Examples

```r
# full axis width
ggplot(as.data.frame(Titanic),
       aes(y = Freq,
           axis1 = Class, axis2 = Sex, axis3 = Age, axis4 = Survived)) +
geom_stratum(width = 1) + geom_text(stat = "stratum", infer.label = TRUE) +
scale_x_discrete(limits = c("Class", "Sex", "Age", "Survived"))

# use of facets
ggplot(as.data.frame(Titanic),
       aes(y = Freq,
           axis1 = Class, axis2 = Sex)) +
```
geom_flow(aes(fill = Survived)) +
geom_stratum() + geom_text(stat = "stratum", infer.label = TRUE) +
scale_x_discrete(limits = c("Class", "Sex")) +
facet_wrap(~ Age, scales = "free_y")

---

**Description**

These functions control the order of lodes within strata in an alluvial diagram. They are invoked by `stat_alluvium()` and can be passed to the `lode.guidance` parameter.

**Usage**

- `lode_zigzag(n, i)`
- `lode_zagzig(n, i)`
- `lode_forward(n, i)`
- `lode_rightward(n, i)`
- `lode_backward(n, i)`
- `lode_leftward(n, i)`
- `lode_frontback(n, i)`
- `lode_rightleft(n, i)`
- `lode_backfront(n, i)`
- `lode_leftright(n, i)`

**Arguments**

- `n` Numeric, a positive integer
- `i` Numeric, a positive integer at most `n`

**Details**

Each function orders the numbers 1 through `n`, starting at index `i`. The choice of function made in `stat_alluvium()` determines the order in which the other axes contribute to the sorting of lodes within each index axis. After starting at `i`, the functions order the remaining axes as follows:

- **zigzag**: Zigzag outward from `i`, starting in the outward direction
self-adjoin

- **zigzag**: Zigzag outward from \( i \), starting in the inward direction
- **forward**: Increasing order (alias rightward)
- **backward**: Decreasing order (alias leftward)
- **frontback**: Proceed forward from \( i \) to \( n \), then backward to \( 1 \) (alias rightleft)
- **backfront**: Proceed backward from \( i \) to \( 1 \), then forward to \( n \) (alias leftright)

---

**majors**

<table>
<thead>
<tr>
<th>Student curricula across semesters</th>
</tr>
</thead>
</table>

**Description**

This data set follows the major curricula of 10 students across 8 academic semesters. The data were kindly contributed by Dario Bonaretti.

**Format**

An alluvial data frame in lodes form.

---

self-adjoin

| Adjoin a dataset to itself |

**Description**

This function binds a dataset to itself along adjacent pairs of a key variable. It is invoked by `geom_flow()` to convert data in lodes form to something similar to alluvia form.

**Usage**

```r
self_adjoin(
  data,
  key,
  by = NULL,
  link = NULL,
  keep.x = NULL,
  keep.y = NULL,
  suffix = c(".x", ".y")
)
```
self-adjoin

Arguments

- **data**: A data frame in long form (repeated measures data; see alluvial-data).
- **key**: Column of data indicating sequential collection; handled as in tidyr::spread().
- **by**: Character vector of variables to self-adjoin by; passed to dplyr::join functions.
- **link**: Character vector of variables to adjoin. Will be replaced by pairs of variables suffixed by suffix.
- **keep.x**, **keep.y**: Character vector of variables to associate with the first (respectively, second) copy of data after adjoining. These variables can overlap with each other but cannot overlap with by or link.
- **suffix**: Suffixes to add to the adjoined link variables; passed to dplyr::join functions.

Details

self_adjoin invokes dplyr::join functions in order to convert a dataset with measures along a discrete key variable into a dataset consisting of column bindings of these measures (by any by variables) along adjacent values of key.

See Also

Other alluvial data manipulation: alluvial-data

Examples

```r
# self-adjoin 'majors' data
data(majors)
major_changes <- self_adjoin(majors, key = semester,
   by = "student", link = c("semester", "curriculum"))
major_changes$change <- major_changes$curriculum.x == major_changes$curriculum.y
head(major_changes)

# self-adjoin 'vaccinations' data
data(vaccinations)
vaccination_steps <- self_adjoin(vaccinations, key = survey,
   by = "subject", link = c("survey", "response"),
   keep.x = c("freq", "a"))
head(vaccination_steps)
vaccination_steps <- self_adjoin(vaccinations, key = survey,
   by = "subject", link = c("survey", "response"),
   keep.x = c("freq", "a"), keep.y = "a")
head(vaccination_steps)
```
stat_alluvium  Alluvial positions

Description

Given a dataset with alluvial structure, `stat_alluvium` calculates the centroids (x and y) and heights (ymin and ymax) of the lodes, the intersections of the alluvia with the strata. It leverages the group aesthetic for plotting purposes (for now).

Usage

```
stat_alluvium(
mapping = NULL,
data = NULL,
geom = "alluvium",
position = "identity",
decreasing = ggalluvial_opt("decreasing"),
reverse = ggalluvial_opt("reverse"),
absolute = ggalluvial_opt("absolute"),
discern = FALSE,
negate.strata = NULL,
aggregate.y = NULL,
cement.alluvia = ggalluvial_opt("cement.alluvia"),
lode.guidance = ggalluvial_opt("lode.guidance"),
lode.ordering = ggalluvial_opt("lode.ordering"),
aes.bind = ggalluvial_opt("aes.bind"),
infer.label = FALSE,
min.y = NULL,
max.y = NULL,
na.rm = FALSE,
show.legend = NA,
inherit.aes = TRUE,
...)
```

Arguments

- **mapping**: Set of aesthetic mappings created by `aes()` or `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

- **data**: The data to be displayed in this layer. There are three options:
  - If NULL, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.
  - A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.
A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(x,10)`).

**geom**
The geometric object to use display the data; override the default.

**position**
Position adjustment, either as a string, or the result of a call to a position adjustment function.

**decreasing**
Logical; whether to arrange the strata at each axis in the order of the variable values (NA, the default), in ascending order of totals (largest on top, FALSE), or in descending order of totals (largest on bottom, TRUE).

**reverse**
Logical; if decreasing is NA, whether to arrange the strata at each axis in the reverse order of the variable values, so that they match the order of the values in the legend. Ignored if decreasing is not NA. Defaults to TRUE.

**absolute**
Logical; if some cases or strata are negative, whether to arrange them (respecting decreasing and reverse) using negative or absolute values of y.

**discern**
Passed to `to_lodes_form()` if data is in alluvia format.

**negate.strata**
A vector of values of the stratum aesthetic to be treated as negative (will ignore missing values with a warning).

**aggregate.y**
Deprecated alias for `cement.alluvia`.

**cement.alluvia**
Logical value indicating whether to aggregate y values over equivalent alluvia before computing lode and flow positions. Alternatively, a function (or its name) to combine the labels (if any) of equivalent alluvia, similar to the `distill` parameter of `to_alluvia_form()`). If set to TRUE, defaults to function `[dplyr::first()]`.

**lode.guidance**
The function to prioritize the axis variables for ordering the lodes within each stratum, or else a character string identifying the function. Character options are "zigzag", "frontback", "backfront", "forward", and "backward" (see lode-guidance-functions).

**lode.ordering**
A list (of length the number of axes) of integer vectors (each of length the number of rows of data) or NULL entries (indicating no imposed ordering), or else a numeric matrix of corresponding dimensions, giving the preferred ordering of alluvia at each axis. This will be used to order the lodes within each stratum by sorting the lodes first by stratum and then by the provided vectors.

**aes.bind**
At what grouping level, if any, to prioritize differentiation aesthetics when ordering the lodes within each stratum. Defaults to "none" (no aesthetic binding) with intermediate option "flows" to bind aesthetics after stratifying by axes linked to the index axis (the one adjacent axis in `stat_flow()`; all remaining axes in `stat_alluvium()` and strongest option "alluvia" to bind aesthetics after stratifying by the index axis but before stratifying by linked axes (only available for `stat_alluvium()`). Stratification by any axis is done with respect to the strata at that axis, after separating positive and negative strata, consistent with the values of decreasing, reverse, and absolute. Thus, if "none", then lode orderings will not depend on aesthetic variables. All aesthetic variables are used, in the order in which they are specified in aes().

**infer.label**
Logical; whether to assign the stratum or alluvium variable to the label aesthetic. Defaults to FALSE, and requires that no label aesthetic is assigned. This parameter is intended only for uses in which the data are in alluvia form and are therefore converted to lode form before the statistical transformation.
min.y Numeric; bounds on the heights of the strata to be rendered. Use these bounds to exclude strata outside a certain range, for example when labeling strata using `ggplot2::geom_text()`.

max.y Numeric; bounds on the heights of the strata to be rendered. Use these bounds to exclude strata outside a certain range, for example when labeling strata using `ggplot2::geom_text()`.

na.rm Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).

show.legend logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.

inherit.aes If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.

... Additional arguments passed to `ggplot2::layer()`.

Aesthetics

`stat_alluvium`, `stat_flow`, and `stat_stratum` require one of two sets of aesthetics:

- x and at least one of alluvium and stratum
- any number of axis[0-9]* (axis1, axis2, etc.)

Use x, alluvium, and/or stratum for data in lodes format and axis[0-9]* for data in alluvia format (see `alluvial-data`). Arguments to parameters inconsistent with the format will be ignored. Additionally, each `stat_*()` accepts the following optional aesthetics:

- y
- group
- label

y controls the heights of the alluvia and may be aggregated across equivalent observations. group is used internally; arguments are ignored. label is used to label the strata or lodes and must take a unique value across the observations within each stratum or lode. Often the same variable will be passed to label as to the corresponding alluvial aesthetic (stratum or alluvium).

These and any other aesthetics are aggregated as follows: Numeric aesthetics, including y, are summed. Character and factor aesthetics, including label, are assigned to strata or lodes provided they take unique values across the observations within each (and are otherwise assigned NA).

Package options

`stat_stratum`, `stat_alluvium`, and `stat_flow` order strata and lodes according to the values of several parameters, which must be held fixed across every layer in an alluvial plot. These package-specific options set global values for these parameters that will be defaulted to when not manually set:

- `ggalluvial.decreasing` (each `stat_*`): defaults to NA.
- `ggalluvial.reverse` (each `stat_*`): defaults to TRUE.
- `ggalluvial.absolute` (each stat_*): defaults to TRUE.
- `ggalluvial.cement.alluvia` (stat_alluvium): defaults to FALSE.
- `ggalluvial.lode.guidance` (stat_alluvium): defaults to "zigzag".
- `ggalluvial.lode.ordering` (stat_alluvium): defaults to NULL.
- `ggalluvial.aes.bind` (stat_alluvium and stat_flow): defaults to "none".

See `base::options()` for how to use options.

**Defunct parameters**

The previously defunct parameters `weight` and `aggregate.wts` have been discontinued. Use `y` and `cement.alluvia` instead.

**See Also**

`ggplot2::layer()` for additional arguments and `geom_alluvium()`, `geom_lode()`, and `geom_flow()` for the corresponding geoms.

Other alluvial stat layers: `stat_flow()`, `stat_stratum()`

**Examples**

```r
# illustrate positioning
ggplot(as.data.frame(Titanic),
       aes(y = Freq,
           axis1 = Class, axis2 = Sex, axis3 = Age,
           color = Survived)) +
stat_stratum(geom = "errorbar") +
geom_line(stat = "alluvium") +
stat_alluvium(geom = "pointrange") +
geom_text(stat = "stratum", infer.label = TRUE) +
scale_x_discrete(limits = c("Class", "Sex", "Age"))

# lode ordering examples
gg <- ggplot(as.data.frame(Titanic),
             aes(y = Freq,
                  axis1 = Class, axis2 = Sex, axis3 = Age)) +
geom_stratum() + geom_text(stat = "stratum", infer.label = TRUE) +
scale_x_discrete(limits = c("Class", "Sex", "Age"))

# use of lode controls
gg + geom_flow(aes(fill = Survived, alpha = Sex), stat = "alluvium",
               lode.guidance = "forward")

# prioritize aesthetic binding
gg + geom_flow(aes(fill = Survived, alpha = Sex), stat = "alluvium",
               aes.bind = TRUE, lode.guidance = "forward")

# use of lode ordering
lode_ord <- replicate(n = 3, expr = sample(x = 32), simplify = FALSE)
print(lode_ord)

gg + geom_flow(aes(fill = Survived, alpha = Sex), stat = "alluvium",
               lode.ordering = lode_ord)

# fixed lode ordering across axes
```
```r
stat_alluvium

gg + geom_flow(aes(fill = Survived, alpha = Sex), stat = "alluvium",
   lode.ordering = lode_orders[[1]])
# use of custom luide guidance function
lode_custom <- function(n, i) {
  stopifnot(n == 3)
  switch(
    i,
    "1" = 1:3,
    "2" = c(2, 3, 1),
    "3" = 3:1
  )
}
gg + geom_flow(aes(fill = Survived, alpha = Sex), stat = "alluvium",
   aes.bind = TRUE, lode.guidance = lode_custom)
data(majors)
# omit missing elements & reverse the 'y' axis
ggplot(majors,
   aes(x = semester, stratum = curriculum, alluvium = student, y = 1)) +
ggplot(majors,
   aes(x = semester, stratum = curriculum, alluvium = student, fill = curriculum)) +
geom_stratum() +
geom_stratum(aes(fill = curriculum), color = NA, na.rm = TRUE) +
theme_bw() +
scale_y_reverse()
# alluvium cementation examples
gg <- ggplot(majors,
   aes(x = semester, stratum = curriculum, alluvium = student, fill = curriculum)) +
geom_stratum()
# diagram with outlined alluvia and labels
gg + geom_flow(stat = "alluvium", color = "black") +
gg + geom_flow(stat = "alluvium", color = "black", cement.alluvia = TRUE) +
gg + geom_flow(stat = "alluvium", color = "black", cement.alluvia = TRUE)
# cemented diagram with default label cementation
gg + geom_flow(stat = "alluvium", color = "black", cement.alluvia = TRUE) +
gg + geom_flow(stat = "alluvium", color = "black", cement.alluvia = TRUE)
# cemented diagram with custom label cementation
gg + geom_flow(stat = "alluvium", color = "black", cement.alluvia = TRUE) +
gg + geom_flow(stat = "alluvium", color = "black", cement.alluvia = TRUE)
# irregular spacing between axes of a continuous variable
data(Refugees, package = "alluvial")
refugees_sub <- subset(Refugees, year %in% c(2003, 2005, 2010, 2013))
ggplot(data = refugees_sub,
   aes(x = year, y = refugees, alluvium = country)) +
ggplot(data = refugees_sub,
   aes(x = year, y = refugees, alluvium = country)) +
ggplot(data = refugees_sub,
   aes(x = year, y = refugees, alluvium = country)) +
ggplot(data = refugees_sub,
   aes(x = year, y = refugees, alluvium = country)) +
ggplot(data = refugees_sub,
   aes(x = year, y = refugees, alluvium = country)) +
ggplot(data = refugees_sub,
   aes(x = year, y = refugees, alluvium = country)) +
geom_alluvium(aes(fill = country),
   alpha = .75, decreasing = FALSE, knot.pos = 1) +
geom_stratum(aes(stratum = country), decreasing = FALSE, width = 1/2) +
theme_bw() +
scale_fill_brewer(type = "qual", palette = "Set3")
```
## Not run:
data(babynames, package = "babynames")
# a discontiguous alluvium
bn <- subset(babynames, prop >= .01 & sex == "F" & year > 1962 & year < 1968)
ggplot(data = bn,
       aes(x = year, alluvium = name, y = prop)) +
  geom_alluvium(aes(fill = name, color = name == "Tammy"),
                decreasing = TRUE, show.legend = FALSE) +
  scale_color_manual(values = c("#00000000", "#0000000"))
# filling in missing zeros
bn2 <- merge(bn,
             expand.grid(year = unique(bn$year), name = unique(bn$name)),
             all = TRUE)
bn2$prop[is.na(bn2$prop)] <- 0
ggplot(data = bn2,
       aes(x = year, alluvium = name, y = prop)) +
  geom_alluvium(aes(fill = name, color = name == "Tammy"),
                decreasing = TRUE, show.legend = FALSE) +
  scale_color_manual(values = c("#00000000", "#0000000"))
## End(Not run)

# use negative y values to encode deaths versus survivals
titanic <- as.data.frame(Titanic)
titanic <- transform(titanic, Lives = Freq * (-1) ^ (Survived == "No"))
ggplot(subset(titanic, Class != "Crew"),
       aes(axis1 = Class, axis2 = Sex, axis3 = Age, y = Lives)) +
  geom_alluvium(aes(alpha = Survived, fill = Class), absolute = FALSE) +
  geom_stratum(absolute = FALSE) +
  geom_text(stat = "stratum", infer.label = TRUE, absolute = FALSE) +
  scale_x_discrete(limits = c("Class", "Sex", "Age"), expand = c(.1, .05)) +
  scale_alpha_discrete(range = c(.25, .75), guide = FALSE)

---

**stat_flow**

<table>
<thead>
<tr>
<th>Flow positions</th>
</tr>
</thead>
</table>

**Description**

Given a dataset with alluvial structure, `stat_flow` calculates the centroids (x and y) and heights (ymin and ymax) of the flows between each pair of adjacent axes.

**Usage**

```r
call_for_user <- function()
stat_flow(
  mapping = NULL,
  data = NULL,
  geom = "flow",
  position = "identity",
  decreasing = ggalluvial_opt("decreasing"),
```
reverse = ggalluvial_opt("reverse"),
absolute = ggalluvial_opt("absolute"),
discern = FALSE,
negate.strata = NULL,
aes.bind = ggalluvial_opt("aes.bind"),
infer.label = FALSE,
min.y = NULL,
max.y = NULL,
na.rm = FALSE,
show.legend = NA,
inherit.aes = TRUE,
... )

Arguments

- **mapping**: Set of aesthetic mappings created by `aes()` or `aes()`. If specified and `inherit.aes` = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

- **data**: The data to be displayed in this layer. There are three options:
  - If NULL, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.
  - A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.
  - A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x,10)`).

- **geom**: The geometric object to use display the data; override the default.

- **position**: Position adjustment, either as a string, or the result of a call to a position adjustment function.

- **decreasing**: Logical; whether to arrange the strata at each axis in the order of the variable values (NA, the default), in ascending order of totals (largest on top, FALSE), or in descending order of totals (largest on bottom, TRUE).

- **reverse**: Logical; if `decreasing` is NA, whether to arrange the strata at each axis in the reverse order of the variable values, so that they match the order of the values in the legend. Ignored if `decreasing` is not NA. Defaults to TRUE.

- **absolute**: Logical; if some cases or strata are negative, whether to arrange them (respecting decreasing and reverse) using negative or absolute values of y.

- **discern**: Passed to `to_lodes_form()` if data is in alluvia format.

- **negate.strata**: A vector of values of the stratum aesthetic to be treated as negative (will ignore missing values with a warning).

- **aes.bind**: At what grouping level, if any, to prioritize differentiation aesthetics when ordering the lodes within each stratum. Defaults to "none" (no aesthetic binding) with intermediate option "flows" to bind aesthetics after stratifying by axes linked to the index axis (the one adjacent axis in `stat_flow()`) all remaining
axes in `stat_alluvium()` and strongest option "alluvia" to bind aesthetics after stratifying by the index axis but before stratifying by linked axes (only available for `stat_alluvium()`). Stratification by any axis is done with respect to the strata at that axis, after separating positive and negative strata, consistent with the values of decreasing, reverse, and absolute. Thus, if "none", then lode orderings will not depend on aesthetic variables. All aesthetic variables are used, in the order in which they are specified in `aes()`.

**infer.label** Logical; whether to assign the stratum or alluvium variable to the label aesthetic. Defaults to FALSE, and requires that no label aesthetic is assigned. This parameter is intended only for uses in which the data are in alluva form and are therefore converted to lode form before the statistical transformation.

**min.y** Numeric; bounds on the heights of the strata to be rendered. Use these bounds to exclude strata outside a certain range, for example when labeling strata using `ggplot2::geom_text()`.

**max.y** Numeric; bounds on the heights of the strata to be rendered. Use these bounds to exclude strata outside a certain range, for example when labeling strata using `ggplot2::geom_text()`.

**na.rm** Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).

**show.legend** logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.

**inherit.aes** If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.

**...** Additional arguments passed to `ggplot2::layer()`.

### Aesthetics

`stat_alluvium`, `stat_flow`, and `stat_stratum` require one of two sets of aesthetics:

- `x` and at least one of `alluvium` and `stratum`
- any number of `axis[0-9]`* (axis1, axis2, etc.)

Use `x`, `alluvium`, and/or `stratum` for data in lodes format and `axis[0-9]`* for data in alluvia format (see `alluvial-data`). Arguments to parameters inconsistent with the format will be ignored. Additionally, each `stat_*()` accepts the following optional aesthetics:

- `y`
- `group`
- `label`

`y` controls the heights of the alluvia and may be aggregated across equivalent observations. `group` is used internally; arguments are ignored. `label` is used to label the strata or lodes and must take a unique value across the observations within each stratum or lode. Often the same variable will be passed to `label` as to the corresponding alluvial aesthetic (stratum or alluvium).

These and any other aesthetics are aggregated as follows: Numeric aesthetics, including `y`, are summed. Character and factor aesthetics, including `label`, are assigned to strata or lodes provided they take unique values across the observations within each (and are otherwise assigned NA).
Package options

`stat_stratum`, `stat_alluvium`, and `stat_flow` order strata and lodes according to the values of several parameters, which must be held fixed across every layer in an alluvial plot. These package-specific options set global values for these parameters that will be defaulted to when not manually set:

- `ggalluvial.decreasing` (each `stat_*`): defaults to `NA`.
- `ggalluvial.reverse` (each `stat_*`): defaults to `TRUE`.
- `ggalluvial.absolute` (each `stat_*`): defaults to `TRUE`.
- `ggalluvial.cement.alluvia` (`stat_alluvium`): defaults to `FALSE`.
- `ggalluvial.lode.guidance` (`stat_alluvium`): defaults to "zigzag".
- `ggalluvial.lode.ordering` (`stat_alluvium`): defaults to `NULL`.
- `ggalluvial.aes.bind` (`stat_alluvium` and `stat_flow`): defaults to "none".

See `base::options()` for how to use options.

Defunct parameters

The previously defunct parameters `weight` and `aggregate.wts` have been discontinued. Use `y` and `cement.alluvia` instead.

See Also

`ggplot2::layer()` for additional arguments and `geom_alluvium()` and `geom_flow()` for the corresponding geoms.

Other alluvial stat layers: `stat_alluvium()`, `stat_stratum()`

Examples

```r
# illustrate positioning
ggplot(as.data.frame(Titanic),
  aes(y = Freq,
      axis1 = Class, axis2 = Sex, axis3 = Age,
      color = Survived)) +
  stat_stratum(geom = "errorbar") +
  geom_line(stat = "flow") +
  stat_flow(geom = "pointrange") +
  geom_text(aes(label = response), stat = "stratum") +
  scale_x_discrete(limits = c("Class", "Sex", "Age"))

# alluvium--flow comparison
data(vaccinations)
gg <- ggplot(vaccinations,
  aes(x = survey, stratum = response, alluvium = subject,
      y = freq, fill = response)) +
  geom_stratum(alpha = .5) +
  geom_text(aes(label = response), stat = "stratum")
# rightward alluvial aesthetics for vaccine survey data
gg + geom_flow(stat = "alluvium", lode.guidance = "forward")
```
# memoryless flows for vaccine survey data

```r
gg + geom_flow()
```

# size filter examples

```r
gg <- ggplot(vaccinations,
aes(y = freq,
    x = survey, stratum = response, alluvium = subject,
    fill = response, label = response)) +
stat_stratum(alpha = .5) +
gem_text(stat = "stratum")
```

# omit small flows

```r
gg + geom_flow(min.y = 50)
```

# omit large flows

```r
gg + geom_flow(max.y = 100)
```

# negate missing entries

```r
ggplot(vaccinations,
aes(y = freq,
    x = survey, stratum = response, alluvium = subject,
    fill = response, label = response,
    alpha = response != "Missing")) +
stat_stratum(negate.strata = "Missing") +
gem_flow(negate.strata = "Missing") +
gem_text(stat = "stratum", alpha = 1, negate.strata = "Missing") +
scale_alpha_discrete(range = c(.2, .6)) +
guides(alpha = FALSE)
```

# aesthetics that vary between and within strata

data(vaccinations)

```r
vaccinations$subgroup <- LETTERS[1:2][rbinom(
n = length(unique(vaccinations$subject)), size = 1, prob = .5
) + 1][vaccinations$subject]
```

```r
ggplot(vaccinations,
aes(x = survey, stratum = response, alluvium = subject,
    y = freq, fill = response, label = response)) +
gem_flow(aes(alpha = subgroup)) +
scale_alpha_discrete(range = c(1/3, 2/3)) +
gelem_stratum(alpha = .5) +
gem_text(stat = "stratum")
```

# can even set aesthetics that vary both ways

```r
ggplot(vaccinations,
aes(x = survey, stratum = response, alluvium = subject,
    y = freq, label = response)) +
gem_flow(aes(fill = interaction(response, subgroup)), aes.bind = "alluvia") +
scale_alpha_discrete(range = c(1/3, 2/3)) +
gelem_stratum(alpha = .5) +
gem_text(stat = "stratum")
```

---

**stat_stratum**

**Stratum positions**
Description

Given a dataset with alluvial structure, `stat_stratum` calculates the centroids (x and y) and heights (ymin and ymax) of the strata at each axis.

Usage

```r
stat_stratum(
  mapping = NULL,
  data = NULL,
  geom = "stratum",
  position = "identity",
  decreasing = ggalluvial_opt("decreasing"),
  reverse = ggalluvial_opt("reverse"),
  absolute = ggalluvial_opt("absolute"),
  discern = FALSE,
  negate.strata = NULL,
  infer.label = FALSE,
  label.strata = NULL,
  min.y = NULL,
  max.y = NULL,
  min.height = NULL,
  max.height = NULL,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

- **mapping**: Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes` = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.

- **data**: The data to be displayed in this layer. There are three options:

  - If NULL, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

  A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

  A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x,10)`).

- **geom**: The geometric object to use display the data; override the default.

- **position**: Position adjustment, either as a string, or the result of a call to a position adjustment function.
**stat_stratum**

- **decreasing** Logical; whether to arrange the strata at each axis in the order of the variable values (NA, the default), in ascending order of totals (largest on top, FALSE), or in descending order of totals (largest on bottom, TRUE).

- **reverse** Logical; if decreasing is NA, whether to arrange the strata at each axis in the reverse order of the variable values, so that they match the order of the values in the legend. Ignored if decreasing is not NA. Defaults to TRUE.

- **absolute** Logical; if some cases or strata are negative, whether to arrange them (respecting decreasing and reverse) using negative or absolute values of y.

- **discern** Passed to `to_lodes_form()` if data is in alluvia format.

- **negate.strata** A vector of values of the stratum aesthetic to be treated as negative (will ignore missing values with a warning).

- **infer.label** Logical; whether to assign the stratum or alluvium variable to the label aesthetic. Defaults to FALSE, and requires that no label aesthetic is assigned. This parameter is intended only for uses in which the data are in alluva form and are therefore converted to lode form before the statistical transformation.

- **label.strata** Deprecated; alias for infer.label.

- **min.y, max.y** Numeric; bounds on the heights of the strata to be rendered. Use these bounds to exclude strata outside a certain range, for example when labeling strata using `ggplot2::geom_text()`.

- **min.height, max.height** Deprecated aliases for min.y and max.y.

- **na.rm** Logical: if FALSE, the default, NA lodes are not included; if TRUE, NA lodes constitute a separate category, plotted in grey (regardless of the color scheme).

- **show.legend** logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.

- **inherit.aes** If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.

- **...** Additional arguments passed to `ggplot2::layer()`.

### Aesthetics

`stat_alluvium`, `stat_flow`, and `stat_stratum` require one of two sets of aesthetics:

- **x** and at least one of **alluvium** and **stratum**
- any number of **axis[0-9]*** (axis1, axis2, etc.)

Use **x**, **alluvium**, and/or **stratum** for data in lodes format and **axis[0-9]*** for data in alluvia format (see `alluvial-data`). Arguments to parameters inconsistent with the format will be ignored. Additionally, each `stat_*()` accepts the following optional aesthetics:

- **y**
- **group**
- **label**
y controls the heights of the alluvia and may be aggregated across equivalent observations. group is used internally; arguments are ignored. label is used to label the strata or lodes and must take a unique value across the observations within each stratum or lode. Often the same variable will be passed to label as to the corresponding alluvial aesthetic (stratum or alluvium).

These and any other aesthetics are aggregated as follows: Numeric aesthetics, including y, are summed. Character and factor aesthetics, including label, are assigned to strata or lodes provided they take unique values across the observations within each (and are otherwise assigned NA).

**Package options**

stat_stratum, stat_alluvium, and stat_flow order strata and lodes according to the values of several parameters, which must be held fixed across every layer in an alluvial plot. These package-specific options set global values for these parameters that will be defaulted to when not manually set:

- ggalluvial.decreasing (each stat_*): defaults to NA.
- ggalluvial.reverse (each stat_*): defaults to TRUE.
- ggalluvial.absolute (each stat_*): defaults to TRUE.
- ggalluvial.cement.alluvia (stat_alluvium): defaults to FALSE.
- ggalluvial.lode.guidance (stat_alluvium): defaults to "zigzag".
- ggalluvial.lode.ordering (stat_alluvium): defaults to NULL.
- ggalluvial.aes.bind (stat_alluvium and stat_flow): defaults to "none".

See `base::options()` for how to use options.

**Defunct parameters**

The previously defunct parameters weight and aggregate.wts have been discontinued. Use y and cement.alluvia instead.

**See Also**

`ggplot2::layer()` for additional arguments and `geom_stratum()` for the corresponding geom.

Other alluvial stat layers: `stat_alluvium()`, `stat_flow()`

**Examples**

```r
# only `stratum` assignment is necessary to generate strata
data(vaccinations)
ggplot(vaccinations,
    aes(y = freq,
        x = survey, stratum = response,
        fill = response)) +
    stat_stratum(width = .5)

# lode data, positioning with y labels
ggplot(vaccinations,
    aes(y = freq,
        x = survey, stratum = response, alluvium = subject,
        label = subject)) +
    stat_stratum() +
    scale_y_continuous(vjust = .5, limits = c(0, 1))
```
vaccinations

Influenza vaccination rates

Description

This data set is aggregated from a longitudinal sample from three RAND American Life Panel surveys that included questions on influenza vaccination. The data were kindly contributed by Raffaele Vardavas.
vaccinations

Format

An alluvial data frame in lodes form.

Source

RAND American Life Panel [https://alpdata.rand.org/](https://alpdata.rand.org/)
Index

aes(), 6, 8, 11, 13, 18, 24, 28
aes_(), 6, 8, 11, 13, 18, 24, 28
alluvial-data, 2
base::options(), 21, 26, 30
borders(), 6, 9, 11, 13, 20, 25, 29
dplyr::join, 17
dplyr::select(), 3
fortify(), 6, 8, 11, 13, 18, 24, 28
geom_alluvium, 5, 9, 12, 14
geom_alluvium(), 21, 26
geom_flow, 7, 8, 12, 14
geom_flow(), 5, 16, 21, 26
ggplot(), 6, 8, 11, 13, 18, 24, 28
ggplot2::geom_text(), 20, 25, 29
is_alluvia_form(alluvial-data), 2
is_lodes_form(alluvial-data), 2
lode_guidancefunctions, 15
lode_backfront
  (lode_guidancefunctions), 15
lode_backward
  (lode_guidancefunctions), 15
lode_forward(lode_guidancefunctions), 15
lode_frontback
  (lode_guidancefunctions), 15
lode_leftright
  (lode_guidancefunctions), 15
lode_leftward
  (lode_guidancefunctions), 15
lode_rightleft
  (lode_guidancefunctions), 15
lode_rightward
  (lode_guidancefunctions), 15
lode_zagzig(lode_guidancefunctions), 15
lode_zigzag(lode_guidancefunctions), 15
majors, 16
rlang::enquo(), 3
self_adjoin, 16
self_adjoin(self_adjoin), 16
stat_alluvium, 18, 26, 30
stat_alluvium(), 7, 9, 12, 15
stat_flow, 21, 23, 30
stat_flow(), 7, 9
stat_stratum, 21, 26, 27
stat_stratum(), 12, 14
tidyr::gather(), 3, 4
tidyr::spread(), 3, 4, 17
tidyselect::vars_select(), 3
to_alluvia_form(alluvial-data), 2
to_alluvia_form(), 19
to_lodes_form(alluvial-data), 2
to_lodes_form(), 19, 24, 29
vaccinations, 31

33