Package 'ggridges'

August 27, 2025

Type Package

Title Ridgeline Plots in 'ggplot2'

Version 0.5.7

Description Ridgeline plots provide a convenient way of visualizing changes in distributions over time or space. This package enables the creation of such plots in 'ggplot2'.

URL https://wilkelab.org/ggridges/

BugReports https://github.com/wilkelab/ggridges/issues

Depends R (>= 3.2)

Imports ggplot2 (>= 3.5.0), grid (>= 3.0.0), scales (>= 0.4.1), withr (>= 2.1.1)

License GPL-2 | file LICENSE

LazyData true

Suggests covr, dplyr, patchwork, ggplot2movies, forcats, knitr, rmarkdown, testthat, vdiffr

VignetteBuilder knitr

Collate 'data.R' 'ggridges.R' 'geoms.R' 'geomsv.R' 'geoms-gradient.R' 'geom-density-line.R' 'position.R' 'scale-cyclical.R' 'scale-point.R' 'scale-vline.R' 'stats.R' 'theme.R' 'utils_ggplot2.R' 'utils.R'

RoxygenNote 7.3.2

Encoding UTF-8

NeedsCompilation no

Author Claus O. Wilke [aut, cre] (ORCID: <https://orcid.org/0000-0002-7470-9261>)

Maintainer Claus O. Wilke <wilke@austin.utexas.edu>

Repository CRAN

Date/Publication 2025-08-27 05:10:01 UTC

Contents

`Aus_athletes` *Australian athletes*

Description

This dataset is equivalent to `ais` from the `DAAG` package.

Usage

```
Aus_athletes
```

Format

An object of class `data.frame` with 202 rows and 13 columns.

References

Telford, R.D. and Cunningham, R.B. 1991. Sex, sport and body-size dependency of hematology in highly trained athletes. Medicine and Science in Sports and Exercise 23: 788-794.

Examples

```
# none yet
```

Catalan_elections *Results from Catalan regional elections (1980-2015)*

Description

Data from Catalan regional elections for 949 municipalities, from 11 elections spanning the years 1980-2015. The data was obtained and processed from Idescat.cat by Marc Belzunces (Twitter: @marcbeldata).

Usage

```
Catalan_elections
```

Format

A tibble with 20764 rows and 4 variables:

`Municipality`

`Year`

`Option` The voter option; either "Indy" or "Unionist"

`Percent` The percentage of the voters choosing the given option

geom_density_line *Smoothed density estimates drawn with a ridgeline rather than area*

Description

This function is a drop-in replacement for ggplot2's `ggplot2::geom_density()`. The only difference is that the geom draws a ridgeline (line with filled area underneath) rather than a polygon.

Usage

```
geom_density_line(
  mapping = NULL,
  data = NULL,
  stat = "density",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

mapping	Set of aesthetic mappings created by 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 for this layer. When using a geom_*() function to construct a layer, the stat argument can be used the override the default coupling between geoms and stats. The stat argument accepts the following: • A Stat ggproto subclass, for example StatCount. • A string naming the stat. To give the stat as a string, strip the function name of the stat_ prefix. For example, to use stat_count(), give the stat as "count". • For more information and other ways to specify the stat, see the layer stat documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following: • The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
...	Other arguments passed on to layer()'s params argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the position argument, or aesthetics that are required can *not* be passed through .... Unknown arguments that are not part of the 4 categories below are ignored. • Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, colour = "red" or linewidth = 3. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the params. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.

- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the geom part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`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()`.

See Also

See `ggplot2::geom_density()`.

Examples

```
library(ggplot2)
ggplot(diamonds, aes(carat)) +
  geom_density_line()

ggplot(diamonds, aes(carat)) +
  geom_density_line(adjust = 1/5)
ggplot(diamonds, aes(carat)) +
  geom_density_line(adjust = 5)

ggplot(diamonds, aes(depth, colour = cut)) +
  geom_density_line(alpha = 0.5) +
  xlim(55, 70)
ggplot(diamonds, aes(depth, fill = cut, colour = cut)) +
  geom_density_line(alpha = 0.1) +
  xlim(55, 70)
```

---

geom_density_ridges *Create ridgeline plot*

Description

geom_density_ridges arranges multiple density plots in a staggered fashion, as in the cover of the famous Joy Division album Unknown Pleasures.

Usage

```
geom_density_ridges(
  mapping = NULL,
  data = NULL,
  stat = "density_ridges",
  position = "points_sina",
  panel_scaling = TRUE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)

geom_density_ridges2(
  mapping = NULL,
  data = NULL,
  stat = "density_ridges",
  position = "points_sina",
  panel_scaling = TRUE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. 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.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
panel_scaling	If TRUE, the default, relative scaling is calculated separately for each panel. If FALSE, relative scaling is calculated globally.

na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
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.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.
...	other arguments passed on to ggplot2::layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like color = "red" or linewidth = 3. They may also be parameters to the paired geom/stat.

Details

By default, this geom calculates densities from the point data mapped onto the x axis. If density calculation is not wanted, use stat="identity" or use geom_ridgeline. The difference between geom_density_ridges and geom_ridgeline is that geom_density_ridges will provide automatic scaling of the ridgelines (controlled by the scale aesthetic), whereas geom_ridgeline will plot the data as is. Note that when you set stat="identity", the height aesthetic must be provided.

Note that the default stat_density_ridges makes joint density estimation across all datasets. This may not generate the desired result when using faceted plots. As an alternative, you can set stat = "density" to use ggplot2::stat_density. In this case, it is required to add the aesthetic mapping height = after_stat(density) (see examples).

Aesthetics

Required aesthetics are in bold.

- x
- y
- weight Optional case weights passed to stats::density to calculate a weighted density estimate
- group Defines the grouping. Not needed if a categorical variable is mapped onto y, but needed otherwise. Will typically be the same variable as is mapped to y.
- height The height of each ridgeline at the respective x value. Automatically calculated and provided by stat_density_ridges if the default stat is not changed.
- scale A scaling factor to scale the height of the ridgelines relative to the spacing between them. A value of 1 indicates that the maximum point of any ridgeline touches the baseline right above, assuming even spacing between baselines.
- rel_min_height Lines with heights below this cutoff will be removed. The cutoff is measured relative to the overall maximum, so rel_min_height=0.01 would remove everything that is 1\ ridgelines. Default is 0, so nothing is removed. alpha
- colour, fill, group, alpha, linetype, linewidth, as in geom_ridgeline.
- point_shape, point_colour, point_size, point_fill, point_alpha, point_stroke, as in geom_ridgeline.

Examples

```
library(ggplot2)

# set the `rel_min_height` argument to remove tails
ggplot(iris, aes(x = Sepal.Length, y = Species)) +
  geom_density_ridges(rel_min_height = 0.005) +
  scale_y_discrete(expand = c(0.01, 0)) +
  scale_x_continuous(expand = c(0.01, 0)) +
  theme_ridges()

# set the `scale` to determine how much overlap there is among the plots
ggplot(diamonds, aes(x = price, y = cut)) +
  geom_density_ridges(scale = 4) +
  scale_y_discrete(expand = c(0.01, 0)) +
  scale_x_continuous(expand = c(0.01, 0)) +
  theme_ridges()

# the same figure with colors, and using the ggplot2 density stat
ggplot(diamonds, aes(x = price, y = cut, fill = cut, height = after_stat(density))) +
  geom_density_ridges(scale = 4, stat = "density") +
  scale_y_discrete(expand = c(0.01, 0)) +
  scale_x_continuous(expand = c(0.01, 0)) +
  scale_fill_brewer(palette = 4) +
  theme_ridges() + theme(legend.position = "none")

# use geom_density_ridges2() instead of geom_density_ridges() for solid polygons
ggplot(iris, aes(x = Sepal.Length, y = Species)) +
  geom_density_ridges2() +
  scale_y_discrete(expand = c(0.01, 0)) +
  scale_x_continuous(expand = c(0.01, 0)) +
  theme_ridges()
```

---

geom_ridgeline *Plot a ridgeline (line with filled area underneath)*

Description

Plots the sum of the `y` and `height` aesthetics versus `x`, filling the area between `y` and `y + height` with a color. Thus, the data mapped onto `y` and onto height must be in the same units. If you want relative scaling of the heights, you can use `geom_density_ridges` with `stat = "identity"`.

Usage

```
geom_ridgeline(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
```

```
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. 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.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
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.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.
...	other arguments passed on to ggplot2::layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like color = "red" or linewidth = 3. They may also be parameters to the paired geom/stat.

Details

In addition to drawing ridgelines, this geom can also draw points if they are provided as part of the dataset. The stat stat_density_ridges() takes advantage of this option to generate ridgeline plots with overlaid jittered points.

Aesthetics

Required aesthetics are in bold.

- x
- y
- height Height of the ridgeline, measured from the respective y value. Assumed to be positive, though this is not required.
- group Defines the grouping. Required when the dataset contains multiple distinct ridgelines. Will typically be the same variable as is mapped to y.
- scale A scaling factor to scale the height of the ridgelines. A value of 1 indicates that the heights are taken as is. This aesthetic can be used to convert height units into y units.

- `min_height` A height cutoff on the drawn ridgelines. All values that fall below this cutoff will be removed. The main purpose of this cutoff is to remove long tails right at the baseline level, but other uses are possible. The cutoff is applied before any height scaling is applied via the `scale` aesthetic. Default is 0, so negative values are removed.
- `colour` Color of the ridgeline
- `fill` Fill color of the area under the ridgeline
- `alpha` Transparency level of `fill`. Not applied to `color`. If you want transparent lines, you can set their color as RGBA value, e.g. #FF0000A0 for partially transparent red.
- `group` Grouping, to draw multiple ridgelines from one dataset
- `linetype` Linetype of the ridgeline
- `linewidth` Line thickness
- `point_shape`, `point_colour`, `point_size`, `point_fill`, `point_alpha`, `point_stroke` Aesthetics applied to points drawn in addition to ridgelines.

Examples

```
library(ggplot2)

d <- data.frame(x = rep(1:5, 3), y = c(rep(0, 5), rep(1, 5), rep(3, 5)),
                height = c(0, 1, 3, 4, 0, 1, 2, 3, 5, 4, 0, 5, 4, 4, 1))
ggplot(d, aes(x, y, height = height, group = y)) + geom_ridgeline(fill="lightblue")
```

`geom_ridgeline_gradient` *Plot ridgelines and ridgeline plots with fill gradients along the x axis*

Description

The geoms `geom_ridgeline_gradient` and `geom_density_ridges_gradient` work just like `geom_ridgeline` and `geom_density_ridges` except that the `fill` aesthetic can vary along the x axis. Because filling with color gradients is fraught with issues, these geoms should be considered experimental. Don't use them unless you really need to. Note that due to limitations in R's graphics system, transparency (`alpha`) has to be disabled for gradient fills.

Usage

```
geom_ridgeline_gradient(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  gradient_lwd = 0.5,
  show.legend = NA,
```

```
  inherit.aes = TRUE,
  ...
)

geom_density_ridges_gradient(
  mapping = NULL,
  data = NULL,
  stat = "density_ridges",
  position = "points_sina",
  panel_scaling = TRUE,
  na.rm = TRUE,
  gradient_lwd = 0.5,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. 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.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
gradient_lwd	A parameter to needed to remove rendering artifacts inside the rendered gradients. Should ideally be 0, but often needs to be around 0.5 or higher.
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.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.
...	other arguments passed on to ggplot2::layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like color = "red" or linewidth = 3. They may also be parameters to the paired geom/stat.
panel_scaling	Argument only to geom_density_ridges_gradient. If TRUE, the default, relative scaling is calculated separately for each panel. If FALSE, relative scaling is calculated globally.

Examples

```
library(ggplot2)

# Example for `geom_ridgeline_gradient()`
d <- data.frame(
  x = rep(1:5, 3) + c(rep(0, 5), rep(0.3, 5), rep(0.6, 5)),
  y = c(rep(0, 5), rep(1, 5), rep(3, 5)),
  height = c(0, 1, 3, 4, 0, 1, 2, 3, 5, 4, 0, 5, 4, 4, 1)
)
ggplot(d, aes(x, y, height = height, group = y, fill = factor(x+y))) +
  geom_ridgeline_gradient() +
  scale_fill_viridis_d(direction = -1) +
  theme(legend.position = 'none')

# Example for `geom_density_ridges_gradient()`
ggplot(lincoln_weather, aes(x = `Mean Temperature [F]`, y = `Month`, fill = stat(x))) +
  geom_density_ridges_gradient(scale = 3, rel_min_height = 0.01) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_discrete(expand = c(0, 0)) +
  scale_fill_viridis_c(name = "Temp. [F]", option = "C") +
  coord_cartesian(clip = "off") +
  labs(title = 'Temperatures in Lincoln NE in 2016') +
  theme_ridges(font_size = 13, grid = TRUE) +
  theme(axis.title.y = element_blank())
```

---

geom_vridgeline *Plot a vertical ridgeline (ridgeline rotated 90 degrees)*

Description

Plots the sum of the `x` and `width` aesthetics versus `y`, filling the area between `x` and `x + width` with a color. Just like `geom_ridgeline()`, but with y and x replaced.

Usage

```
geom_vridgeline(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. 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.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
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.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.
...	other arguments passed on to ggplot2::layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like color = "red" or linewidth = 3. They may also be parameters to the paired geom/stat.

Aesthetics

Required aesthetics are in bold.

- x
- y
- width Width of the ridgeline, measured from the respective x value. Assumed to be positive, though this is not required.
- group Defines the grouping. Required when the dataset contains multiple distinct ridgelines. Will typically be the same variable as is mapped to x.
- scale A scaling factor to scale the widths of the ridgelines. A value of 1 indicates that the widths are taken as is. This aesthetic can be used to convert width units into x units.
- min_width A width cutoff on the drawn ridgelines. All values that fall below this cutoff will be removed. The main purpose of this cutoff is to remove long tails right at the baseline level, but other uses are possible. The cutoff is applied before any width scaling is applied via the scale aesthetic. Default is 0, so negative values are removed.
- color Color of the ridgeline
- fill Fill color of the area under the ridgeline
- alpha Transparency level of fill. Not applied to color. If you want transparent lines, you can set their color as RGBA value, e.g. #FF0000A0 for partially transparent red.
- group Grouping, to draw multiple ridgelines from one dataset
- linetype Linetype of the ridgeline
- linewidth Line thickness

Examples

```
library(ggplot2)

d <- data.frame(y = rep(1:5, 3), x = c(rep(0, 5), rep(1, 5), rep(3, 5)),
                width = c(0, 1, 3, 4, 0, 1, 2, 3, 5, 4, 0, 5, 4, 4, 1))
ggplot(d, aes(x, y, width = width, group = x)) + geom_vridgeline(fill="lightblue")

ggplot(iris, aes(x=Species, y=Sepal.Width, width = after_stat(density), fill=Species)) +
  geom_vridgeline(stat="ydensity", trim=FALSE, alpha = 0.85, scale = 2)
```

---

lincoln_weather *Weather in Lincoln, Nebraska in 2016.*

Description

A dataset containing weather information from Lincoln, Nebraska, from 2016. Originally downloaded from Weather Underground by Austin Wehrwein, http://austinwehrwein.com/. The variables are listed below. Most are self-explanatory. Max, mean, and min measurements are calculated relative to the specific day of measurement.

Usage

```
lincoln_weather
```

Format

A tibble with 366 rows and 24 variables:

`CST` Day of the measurement

`Max Temperature [F]`

`Mean Temperature [F]`

`Min Temperature [F]`

`Max Dew Point [F]`

`Mean Dew Point [F]`

`Min Dewpoint [F]`

`Max Humidity`

`Mean Humidity`

`Min Humidity`

`Max Sea Level Pressure [In]`

`Mean Sea Level Pressure [In]`

`Min Sea Level Pressure [In]`

`Max Visibility [Miles]`

`Mean Visibility [Miles]`

`Min Visibility [Miles]`

`Max Wind Speed [MPH]`

`Mean Wind Speed[MPH]`

`Max Gust Speed [MPH]`

`Precipitation [In]`

`CloudCover`

`Events` Specific weather events, such as rain, snow, or fog

`WindDir [Degrees]`

`Month` The month in which the measurement was taken

`position_points_jitter` *Randomly jitter the points in a ridgeline plot*

Description

This is a position adjustment specifically for `geom_density_ridges()` and related geoms. It only jitters the points drawn by these geoms, if any. If no points are present, the plot remains unchanged. The effect is similar to `ggplot2::position_jitter()`: points are randomly shifted up and down and/or left and right.

Usage

```
position_points_jitter(
  width = 0,
  height = 0.2,
  yoffset = 0,
  adjust_vlines = FALSE,
  seed = NULL
)
```

Arguments

`width`	Width for horizontal jittering. By default set to 0.
`height`	Height for vertical jittering, applied in both directions (up and down). By default 0.2.
`yoffset`	Vertical offset applied in addition to jittering.
`adjust_vlines`	If `TRUE`, adjusts vertical lines (as are drawn for quantile lines, for example) to align with the point cloud.
`seed`	Random seed. If set to NULL, the current random number generator is used. If set to NA, a new random random seed is generated. If set to a number, this number is used as seed for jittering only.

See Also

Other position adjustments for ridgeline plots: position_points_sina, position_raincloud

Examples

```
library(ggplot2)

# default jittered points
ggplot(iris, aes(x = Sepal.Length, y = Species)) +
  geom_density_ridges(jittered_points = TRUE, position = "points_jitter", alpha = 0.7)

# simulating a rug
ggplot(iris, aes(x = Sepal.Length, y = Species)) +
 geom_density_ridges(jittered_points = TRUE, point_shape = '|', alpha = 0.7, point_size = 2,
                     position = position_points_jitter(width = 0.02, height = 0))
```

---

position_points_sina *Randomly distribute points in a ridgeline plot between baseline and ridgeline*

Description

This is a position adjustment specifically for geom_density_ridges() and related geoms. It only jitters the points drawn by these geoms, if any. If no points are present, the plot remains unchanged. The effect is similar to a sina plot: Points are randomly distributed to fill the entire shaded area representing the data density.

Usage

```
position_points_sina(rel_min = 0.02, rel_max = 0.98, seed = NULL)
```

Arguments

rel_min	The relative minimum value at which a point can be placed.
rel_max	The relative maximum value at which a point can be placed.
seed	See position_points_jitter.

See Also

Other position adjustments for ridgeline plots: position_points_jitter, position_raincloud

Examples

```
library(ggplot2)

ggplot(iris, aes(x = Sepal.Length, y = Species)) +
  geom_density_ridges(jittered_points = TRUE, position = "points_sina", alpha = 0.7)
```

position_raincloud *Create a cloud of randomly jittered points below a ridgeline plot*

Description

This is a position adjustment specifically for `geom_density_ridges()` and related geoms. It only jitters the points drawn by these geoms, if any. If no points are present, the plot remains unchanged. The effect is similar to `position_points_jitter()`, only that by default the points lie all underneath the baseline of each individual ridgeline.

Usage

```
position_raincloud(
  width = 0,
  height = 0.4,
  ygap = 0.05,
  adjust_vlines = FALSE,
  seed = NULL
)
```

Arguments

`width`	Width for horizontal jittering. By default set to 0.
`height`	Total height of point cloud. By default 0.4.
`ygap`	Vertical gap between ridgeline baseline and point cloud.
`adjust_vlines`	If `TRUE`, adjusts vertical lines (as are drawn for quantile lines, for example) to align with the point cloud.
`seed`	Random seed. See `position_points_jitter`.

Details

The idea for this position adjustment comes from Micah Allen's work on raincloud plots (Allen et al. 2021).

References

Allen, M., Poggiali, D., Whitaker, K., Marshall, T. R., van Langen, J., Kievit, R. A. (2021) Raincloud plots: a multi-platform tool for robust data visualization [version 2; peer review: 2 approved]. Wellcome Open Res 4:63.

See Also

Other position adjustments for ridgeline plots: `position_points_jitter`, `position_points_sina`

Examples

```
library(ggplot2)

ggplot(iris, aes(x = Sepal.Length, y = Species)) +
  geom_density_ridges(jittered_points = TRUE, position = "raincloud", alpha = 0.7)
```

`scale_cyclical` *Create a discrete scale that cycles between values*

Description

The readability of ridgeline plots can often be improved by alternating between fill colors and other aesthetics. The various cyclical scales make it easy to create plots with this feature, simply map your grouping variable to the respective aesthetic (e.g., `fill`) and then use `scale_fill_cyclical` to define the fill colors between you want to alternate. Note that the cyclical scales do not draw legends by default, because the legends will usually be wrong unless the labels are properly adjusted. To draw legends, set the `guide` argument to `"legend"`, as shown in the examples.

Usage

```
scale_colour_cyclical(..., values)

scale_fill_cyclical(..., values)

scale_alpha_cyclical(..., values)

scale_linetype_cyclical(..., values)

scale_size_cyclical(..., values)

scale_linewidth_cyclical(..., values)
```

Arguments

`...`	Common discrete scale parameters: `name`, `breaks`, `labels`, `na.value`, `limits` and `guide`. See ggplot2::discrete_scale for more details.
`values`	The aesthetic values that the scale should cycle through, e.g. colors if it is a scale for the color or fill aesthetic.

Examples

```
library(ggplot2)

# By default, scale_cyclical sets `guide = "none"`, i.e., no legend
# is drawn
ggplot(diamonds, aes(x = price, y = cut, fill = cut)) +
  geom_density_ridges(scale = 4) +
```

```
  scale_fill_cyclical(values = c("#3030D0", "#9090F0"))

# However, legends can be turned on by setting `guide = "legend"`
ggplot(diamonds, aes(x = price, y = cut, fill = cut)) +
  geom_density_ridges(scale = 4) +
  scale_fill_cyclical(values = c("#3030D0", "#9090F0"),
                      guide = "legend", name = "Fill colors",
                      labels = c("dark blue", "light blue"))

# Cyclical scales are also available for the various other aesthetics
ggplot(diamonds, aes(x = price, y = cut, fill = cut,
                     color = cut, linewidth = cut,
                     alpha = cut, linetype = cut)) +
  geom_density_ridges(scale = 4, fill = "blue") +
  scale_fill_cyclical(values = c("blue", "green")) +
  scale_color_cyclical(values = c("black", "white")) +
  scale_alpha_cyclical(values = c(0.4, 0.8)) +
  scale_linewidth_cyclical(values = c(2, 1)) +
  scale_linetype_cyclical(values = c(1, 2))
```

`scale_point` *Scales for point aesthetics*

Description

These are various scales that can be applied to point aesthetics, such as `point_color`, `point_fill`, `point_size`. The individual scales all have the same usage as existing standard ggplot2 scales, only the name differs.

See Also

See `scale_vline_color_hue()` for specific scales for vline aesthetics and `ggplot2::scale_discrete_manual()` for a general discrete scale.

Examples

```
library(ggplot2)

# default scales
ggplot(iris, aes(x=Sepal.Length, y=Species, fill = Species)) +
  geom_density_ridges(
    aes(
      point_color = Species, point_fill = Species,
      point_shape = Species
    ),
    alpha = .4, jittered_points = TRUE
  ) +
  theme_ridges()
```

```
# modified scales
ggplot(iris, aes(x=Sepal.Length, y=Species, fill = Species)) +
  geom_density_ridges(
    aes(
      point_color = Species, point_fill = Species,
      point_shape = Species
    ),
    alpha = .4, point_alpha = 1,
    jittered_points = TRUE
  ) +
  scale_fill_hue(l = 50) +
  scale_point_color_hue(l = 20) +
  scale_point_fill_hue(l = 70) +
  scale_discrete_manual("point_shape", values = c(21, 22, 23)) +
  theme_ridges()
```

`scale_vline` *Scales for vline aesthetics*

Description

These are various scales that can be applied to vline aesthetics, such as `vline_color`, `vline_width`, `vline_linetype`. The individual scales all have the same usage as existing standard ggplot2 scales, only the name differs.

See Also

See `scale_point_color_hue()` for specific scales for point aesthetics and `ggplot2::scale_discrete_manual()` for a general discrete scale.

Examples

```
library(ggplot2)

# default scales
ggplot(iris, aes(x=Sepal.Length, y=Species, fill = Species, color = Species)) +
  geom_density_ridges(
    aes(vline_color = Species, vline_linetype = Species),
    alpha = .4, quantile_lines = TRUE
  ) +
  theme_ridges()

# modified scales
ggplot(iris, aes(x=Sepal.Length, y=Species, fill = Species, color = Species)) +
  geom_density_ridges(
    aes(vline_color = Species),
    alpha = .4, quantile_lines = TRUE
  ) +
  scale_fill_hue(l = 50) +
  scale_vline_color_hue(l = 30) +
  theme_ridges()
```

stat_binline *Stat for histogram ridgeline plots*

Description

Works like `stat_bin` except that the output is a ridgeline describing the histogram rather than a set of counts.

Usage

```
stat_binline(
  mapping = NULL,
  data = NULL,
  geom = "density_ridges",
  position = "identity",
  ...,
  binwidth = NULL,
  bins = NULL,
  center = NULL,
  boundary = NULL,
  breaks = NULL,
  closed = c("right", "left"),
  pad = TRUE,
  draw_baseline = TRUE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `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 geom to use for drawing.

position — A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following:

- The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position.
- A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter".
- For more information and other ways to specify the position, see the layer position documentation.

... — Other arguments passed on to layer()'s params argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the position argument, or aesthetics that are required can *not* be passed through .... Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, colour = "red" or linewidth = 3. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the params. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a stat_*() function, the ... argument can be used to pass on parameters to the geom part of the layer. An example of this is stat_density(geom = "area", outline.type = "both"). The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a geom_*() function, the ... argument can be used to pass on parameters to the stat part of the layer. An example of this is geom_area(stat = "density", adjust = 0.5). The stat's documentation lists which parameters it can accept.
- The key_glyph argument of layer() may also be passed on through .... This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

binwidth — The width of the bins. Can be specified as a numeric value or as a function that calculates width from unscaled x. Here, "unscaled x" refers to the original x values in the data, before application of any scale transformation. When specifying a function along with a grouping structure, the function will be called once per group. The default is to use the number of bins in bins, covering the range of the data. You should always override this value, exploring multiple widths to find the best to illustrate the stories in your data.

The bin width of a date variable is the number of days in each time; the bin width of a time variable is the number of seconds.

bins — Number of bins. Overridden by binwidth. Defaults to 30.

center, boundary — bin position specifiers. Only one, center or boundary, may be specified for a single plot. center specifies the center of one of the bins. boundary specifies the boundary between two bins. Note that if either is above or below the range of

the data, things will be shifted by the appropriate integer multiple of `binwidth`. For example, to center on integers use `binwidth = 1` and `center = 0`, even if `0` is outside the range of the data. Alternatively, this same alignment can be specified with `binwidth = 1` and `boundary = 0.5`, even if `0.5` is outside the range of the data.

`breaks` Alternatively, you can supply a numeric vector giving the bin boundaries. Overrides `binwidth`, `bins`, `center`, and `boundary`.

`closed` One of `"right"` or `"left"` indicating whether right or left edges of bins are included in the bin.

`pad` If `TRUE`, adds empty bins at either end of x. This ensures that the binline always goes back down to 0. Defaults to `TRUE`.

`draw_baseline` If `FALSE`, removes lines along 0 counts. Defaults to `TRUE`.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`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()`.

Examples

```
library(ggplot2)

ggplot(iris, aes(x = Sepal.Length, y = Species, group = Species, fill = Species)) +
  geom_density_ridges(stat = "binline", bins = 20, scale = 2.2) +
  scale_y_discrete(expand = c(0, 0)) +
  scale_x_continuous(expand = c(0, 0)) +
  coord_cartesian(clip = "off") +
  theme_ridges()

ggplot(iris, aes(x = Sepal.Length, y = Species, group = Species, fill = Species)) +
  stat_binline(bins = 20, scale = 2.2, draw_baseline = FALSE) +
  scale_y_discrete(expand = c(0, 0)) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_fill_grey() +
  coord_cartesian(clip = "off") +
  theme_ridges() +
  theme(legend.position = 'none')

library(ggplot2movies)
ggplot(movies[movies$year>1989,], aes(x = length, y = year, fill = factor(year))) +
   stat_binline(scale = 1.9, bins = 40) +
   scale_x_continuous(limits = c(1, 180), expand = c(0, 0)) +
   scale_y_reverse(expand = c(0, 0)) +
   scale_fill_viridis_d(begin = 0.3, option = "B") +
   coord_cartesian(clip = "off") +
   labs(title = "Movie lengths 1990 - 2005") +
```

```
   theme_ridges() +
   theme(legend.position = "none")

count_data <- data.frame(
  group = rep(letters[1:5], each = 10),
  mean = rep(1:5, each = 10)
)
count_data$group <- factor(count_data$group, levels = letters[5:1])
count_data$count <- rpois(nrow(count_data), count_data$mean)

ggplot(count_data, aes(x = count, y = group, group = group)) +
  geom_density_ridges2(
    stat = "binline",
    aes(fill = group),
    binwidth = 1,
    scale = 0.95
  ) +
  geom_text(
    stat = "bin",
    aes(y = group + 0.9*stat(count/max(count)),
    label = ifelse(stat(count) > 0, stat(count), "")),
    vjust = 1.2, size = 3, color = "white", binwidth = 1
  ) +
  scale_x_continuous(breaks = c(0:12), limits = c(-.5, 13), expand = c(0, 0)) +
  scale_y_discrete(expand = c(0, 0)) +
  scale_fill_cyclical(values = c("#0000B0", "#7070D0")) +
  guides(y = "none") +
  coord_cartesian(clip = "off") +
  theme_ridges(grid = FALSE)
```

`stat_density_ridges` *Stat for density ridgeline plots*

Description

This stat is the default stat used by `geom_density_ridges`. It is very similar to ggplot2::stat_density, however there are a few differences. Most importantly, the density bandwidth is chosen across the entire dataset.

Usage

```
stat_density_ridges(
  mapping = NULL,
  data = NULL,
  geom = "density_ridges",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
```

```
  bandwidth = NULL,
  from = NULL,
  to = NULL,
  jittered_points = FALSE,
  quantile_lines = FALSE,
  calc_ecdf = FALSE,
  quantiles = 4,
  quantile_fun = quantile,
  n = 512,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. 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.
geom	The geometric object to use to display the data.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
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.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them.
bandwidth	Bandwidth used for density calculation. If not provided, is estimated from the data.
from, to	The left and right-most points of the grid at which the density is to be estimated, as in stats::density(). If not provided, these are estimated from the data range and the bandwidth.
jittered_points	If TRUE, carries the original point data over to the processed data frame, so that individual points can be drawn by the various ridgeline geoms. The specific position of these points is controlled by various position objects, e.g. position_points_sina() or position_raincloud().
quantile_lines	If TRUE, enables the drawing of quantile lines. Overrides the calc_ecdf setting and sets it to TRUE.
calc_ecdf	If TRUE, stat_density_ridges calculates an empirical cumulative distribution function (ecdf) and returns a variable ecdf and a variable quantile. Both can be mapped onto aesthetics via stat(ecdf) and stat(quantile), respectively.

quantiles	Sets the number of quantiles the data should be broken into. Used if either calc_ecdf = TRUE or quantile_lines = TRUE. If quantiles is an integer then the data will be cut into that many equal quantiles. If it is a vector of probabilities then the data will cut by them.
quantile_fun	Function that calculates quantiles. The function needs to accept two parameters, a vector x holding the raw data values and a vector probs providing the probabilities that define the quantiles. Default is quantile.
n	The number of equally spaced points at which the density is to be estimated. Should be a power of 2. Default is 512.
...	other arguments passed on to ggplot2::layer(). These are often aesthetics, used to set an aesthetic to a fixed value, like color = "red" or linewidth = 3. They may also be parameters to the paired geom/stat.

Examples

```
library(ggplot2)

# Examples of coloring by ecdf or quantiles
ggplot(iris, aes(x = Sepal.Length, y = Species, fill = factor(stat(quantile)))) +
  stat_density_ridges(
    geom = "density_ridges_gradient",
    calc_ecdf = TRUE,
    quantiles = 5
  ) +
  scale_fill_viridis_d(name = "Quintiles") +
  theme_ridges()

ggplot(iris,
  aes(
    x = Sepal.Length, y = Species, fill = 0.5 - abs(0.5-stat(ecdf))
  )) +
  stat_density_ridges(geom = "density_ridges_gradient", calc_ecdf = TRUE) +
  scale_fill_viridis_c(name = "Tail probability", direction = -1) +
  theme_ridges()

ggplot(iris,
  aes(
    x = Sepal.Length, y = Species, fill = factor(stat(quantile))
  )) +
  stat_density_ridges(
    geom = "density_ridges_gradient",
    calc_ecdf = TRUE, quantiles = c(0.025, 0.975)
  ) +
  scale_fill_manual(
    name = "Probability",
    values = c("#FF0000A0", "#A0A0A0A0", "#0000FFA0"),
    labels = c("(0, 0.025]", "(0.025, 0.975]", "(0.975, 1]")
  ) +
  theme_ridges()
```

theme_ridges *A custom theme specifically for use with ridgeline plots*

Description

This theme has some special modifications that make ridgeline plots look better, such as properly aligned y axis labels. It can draw plots with and without background grids (see examples).

Usage

```
theme_ridges(
  font_size = 14,
  font_family = "",
  line_size = 0.5,
  grid = TRUE,
  center_axis_labels = FALSE
)
```

Arguments

font_size	Overall font size. Default is 14.
font_family	Default font family.
line_size	Default line size.
grid	If TRUE (default), a background grid is drawn. If FALSE, background is left empty.
center_axis_labels	If TRUE, axis labels are drawn centered. If FALSE (default), axis lables are drawn right/top-aligned.

Value

The theme.

Examples

```
library(ggplot2)

# Example with background grid
ggplot(iris, aes(x = Sepal.Length, y = Species, group = Species)) +
  geom_density_ridges(rel_min_height = 0.005) +
  scale_y_discrete(expand = c(0.01, 0)) +
  scale_x_continuous(expand = c(0.01, 0)) +
  theme_ridges()

# Example without background grid
ggplot(iris, aes(x = Sepal.Length, y = Species, group = Species)) +
  geom_density_ridges() +
```

```
scale_y_discrete(expand = c(0.01, 0)) +
scale_x_continuous(expand = c(0.01, 0)) +
theme_ridges(grid = FALSE)
```

Index