Package 'ggformula'

July 24, 2017

Title Formula Interface to the Grammar of Graphics

Description Provides a formula interface to 'ggplot2' graphics.

Type Package

Version 0.5

Date 2017-07-23

License MIT + file LICENSE

LazyData TRUE

LazyLoad TRUE

Depends R (>= 3.1), ggplot2

Imports mosaicCore, rlang, tidyr, magrittr, tibble, stringr, glue, grid

Suggests mosaic, dplyr, testthat, mosaicData, knitr, statisticalModeling, rmarkdown, weatherData, lubridate

VignetteBuilder knitr

RoxygenNote 6.0.1.9000

NeedsCompilation no

Author Daniel Kaplan [aut], Randall Pruim [aut, cre]

Maintainer Randall Pruim <rpruim@calvin.edu>

Repository CRAN

Date/Publication 2017-07-24 21:23:54 UTC

R topics documented:

gf_abline *Reference lines – horizontal, vertical, and diagonal.*

Description

These functions create layers that display lines described i various ways. Unlike most of the plotting functions in ggformula, these functions do not take a formula as input for describing positional attributes of the plot.

Usage

```
gf_abline(object = NULL, gformula = NULL, data = NA, geom = "abline",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = FALSE, ...)

gf_hline(object = NULL, gformula = NULL, data = NA, geom = "hline",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = FALSE, ...)

gf_vline(object = NULL, gformula = NULL, data = NA, geom = "vline",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = FALSE, ...)

gf_coefline(object = NULL, coef = NULL, model = NULL, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	Must be NULL.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include slope, intercept

coef	A numeric vector of length at least 2, treated as intercept and slope. Additional components, if any, are ignored (with a warning).
model	An object with a method for coef() that returns a numeric vector, the first two elements of which are intercept and slope. This is equivalent to coef = coef(model).

Value

a gg object

See Also

geom_abline(), geom_vline(), geom_hline()

Examples

```
mtcars2 <- df_stats( wt ~ cyl, data = mtcars)
gf_point(wt ~ hp, size = ~wt, color = ~cyl, data = mtcars) %>%
  gf_abline(slope = 0, intercept = ~median, color = ~cyl, data = mtcars2)
gf_point(wt ~ hp, size = ~wt, color = ~cyl, data = mtcars) %>%
  gf_hline(slope = 0, yintercept = ~median, color = ~cyl, data = mtcars2)

gf_point(mpg ~ hp, color = ~cyl, size = ~wt, data = mtcars) %>%
  gf_abline(color="red", slope = -0.10, intercept = 35)
gf_point(mpg ~ hp, color = ~cyl, size = ~wt, data = mtcars) %>%
  gf_abline(color = "red", slope = ~slope, intercept = ~intercept,
  data = data.frame(slope = -0.10, intercept = 33:35))
gf_point(mpg ~ hp, color = ~cyl, size = ~wt, data = mtcars) %>%
  gf_abline(intercept = ~ c(10, 20, 30), slope = ~c(1, 0, -1)/100,
    color = c("red", "green", "blue"))

# We can set the color of the guidelines while mapping color in other
# layers
gf_point(mpg ~ hp, color = ~cyl, size = ~wt, data = mtcars) %>%
  gf_hline(color = "navy", yintercept = ~c(20, 25)) %>%
  gf_vline(color = "brown", xintercept = ~c(200, 300))

# If we want to map the color of the guidelines, it must work with the
# scale of the other colors in the plot.
gf_point(mpg ~ hp, size = ~wt, data = mtcars, alpha = 0.3) %>%
  gf_hline(color = ~"horizontal", yintercept = ~c(20, 25)) %>%
  gf_vline(color = ~"vertical", xintercept = ~c(100, 200, 300), data = NA)
gf_point(mpg ~ hp, size = ~wt, color = ~ factor(cyl), data = mtcars, alpha = 0.3) %>%
  gf_hline(color = "orange", yintercept = 20, data = NA) %>%
  gf_vline(color = ~c("4", "6", "8"), xintercept = c(80, 120, 250), data = NA) %>%
# reversing the layers requires using inherit = FALSE
gf_hline(color = "orange", yintercept = 20, data = NA) %>%
  gf_vline(color = ~c("4", "6", "8"), xintercept = c(80, 120, 250), data = NA) %>%
  gf_point(mpg ~ hp, size = ~wt, color = ~ factor(cyl), data = mtcars, alpha = 0.3,
    inherit = FALSE)
```

gf_area *Formula interface to geom_area()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_area(object = NULL, gformula = NULL, data = NULL, geom = "area",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include alpha, color, fill, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form `+ attribute::value` map attribute to value. Additional terms of the form `+ attribute:value` will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, B will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_area()`

Examples

```
if (require(weatherData) && require(dplyr)) {
  Temps <- NewYork2013 %>%
    mutate(date = lubridate::date(Time),
         month = lubridate::month(Time)) %>%
    filter(month <= 4) %>%
    group_by(date) %>%
    summarise(
      hi = max(Temperature, na.rm = TRUE),
      lo = min(Temperature, na.rm = TRUE)
    )
  gf_linerange(lo + hi  ~ date, color = ~hi, data = Temps)
  gf_ribbon(lo + hi ~ date, data = Temps, color = "navy", alpha = 0.3)
  gf_area(hi ~ date, data = Temps, color = "navy", alpha = 0.3)

  Temps2 <- NewYork2013 %>% mutate(city = "NYC") %>%
    bind_rows(Mumbai2013 %>% mutate(city = "Mumbai")) %>%
    bind_rows(London2013 %>% mutate(city = "London")) %>%
    mutate(date = lubridate::date(Time),
           month = lubridate::month(Time)) %>%
    group_by(city, date) %>%
    summarise(
      hi = max(Temperature, na.rm = TRUE),
      lo = min(Temperature, na.rm = TRUE),
      mid = (hi + lo)/2
    )
  gf_ribbon(lo + hi ~ date, data = Temps2, alpha = 0.3) %>%
    gf_facet_grid(city ~ .)

  gf_linerange(lo + hi ~ date, color = ~ mid, data = Temps2) %>%
    gf_facet_grid(city ~ .) %>%
    gf_refine(scale_colour_gradientn(colors = rev(rainbow(5))))
}
```

gf_ash *Average Shifted Histograms*

Description

An ASH plot is the average over all histograms of a fixed bin width. `geom_ash()` and `gf_ash()` provide ways to create ASH plots using **ggplot2** or **ggformula**.

Usage

```
gf_ash(object = NULL, gformula = NULL, data = NULL, geom = "line",
  stat = "ash", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)

stat_ash(mapping = NULL, data = NULL, geom = "line",
  position = "identity", na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE, binwidth = NULL, adjust = 1, ...)

geom_ash(mapping = NULL, data = NULL, stat = "ash",
  position = "identity", na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE, binwidth = NULL, adjust = 1, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `~x` or `y ~ x`. `y` may be `..density..` or `..count..` or `..ndensity..` or `..ncount...` Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.
`position`	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
`show.legend`	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
`show.help`	If `TRUE`, display some minimal help.
`inherit`	A logical indicating whether default attributes are inherited.
`...`	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `fill`, `group`, `linetype`, `size`
`mapping`	set of aesthetic mappings created by `aes()` or `aes_()`.

na.rm	If FALSE (the default), removes missing values with a warning. If TRUE silently removes missing values.
inherit.aes	A logical indicating whether default aesthetics are inherited.
binwidth	the width of the histogram bins. If NULL (the default) the binwidth will be chosen so that approximately 10 bins cover the data. adjust can be used to to increase or decrease binwidth.
adjust	a numeric adjustment to binwidth. Primarily useful when binwidth is not specified. Increasing adjust makes the plot smoother.

Value

a gg object

See Also

geom_histogram(), link{gf_histogram}().

Examples

```
gf_ash(~Sepal.Length, color = ~ Species, data = iris)
gf_ash(~Sepal.Length, color = ~ Species, data = iris, binwidth = 0.3)
gf_ash(~Sepal.Length, color = ~ Species, data = iris, adjust = 2)
ggplot(faithful, aes(x = eruptions)) +
  geom_histogram(aes(y = ..density..),
    fill = "lightskyblue", colour = "gray50", alpha = 0.2) +
  geom_ash(colour = "red") +
  geom_ash(colour = "forestgreen", adjust = 2) +
  geom_ash(colour = "navy", adjust = 1/2) +
  theme_minimal()
```

---

gf_bar *Formula interface to geom_bar()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_bar(object = NULL, gformula = NULL, data = NULL, geom = "bar",
  stat = "count", position = "stack", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)

gf_counts(object = NULL, gformula = NULL, data = NULL, geom = "bar",
  stat = "count", position = "stack", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size, width, binwidth

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_bar()

Examples

```
if (require(mosaicData)) {
  gf_bar( ~ substance, data = HELPrct)
  gf_bar( ~ substance, data = HELPrct, fill = ~sex)
  gf_bar( ~ substance, data = HELPrct, fill = ~sex, position = position_dodge())
  # gf_counts() is another name for gf_bar()
  gf_counts( ~ substance, data = HELPrct, fill = ~sex, position = position_dodge())
}
```

---

gf_boxplot *Formula interface to geom_boxplot()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_boxplot(object = NULL, gformula = NULL, data = NULL,
  geom = "boxplot", stat = "boxplot", position = "dodge",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `fill`, `group`, `linetype`, `shape`, `size`, `weight`, `coef`, `outlier.color`, `outlier.fill`, `outlier.shape`, `outlier.size`, `outlier.stroke`, `outlier.alpha`, `notch`, `notchwidth`, `varwidth`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Value

a gg object Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

See Also

`geom_boxplot()`

Examples

```
if (require(mosaicData)) {
  gf_boxplot(age ~ substance, data = HELPrct)
  gf_boxplot(age ~ substance, data = HELPrct, varwidth = TRUE)
  gf_boxplot(age ~ substance, data = HELPrct, color = ~sex)
  gf_boxplot(age ~ substance, data = HELPrct, color = ~sex, outlier.color = "gray50")
  # longer whiskers
  gf_boxplot(age ~ substance, data = HELPrct, color = ~sex, coef = 2)
  gf_boxplot(age ~ substance, data = HELPrct, color = ~sex, position = position_dodge(width = 0.9))
}
```

---

gf_col *Formula interface to geom_col()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_col(object = NULL, gformula = NULL, data = NULL, geom = "col",
  stat = "identity", position = "stack", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape y ~ x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_col()

Examples

```
D <- data.frame(
  group = LETTERS[1:3],
  count = c(20, 25, 18)
)
gf_col(count ~ group, data = D)
```

---

gf_contour *Formula interface to geom_contour()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_contour(object = NULL, gformula = NULL, data = NULL,
  geom = "contour", stat = "contour", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `z ~ x + y`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_contour()`

Examples

```
gf_density_2d(eruptions ~ waiting, data = faithful, alpha = 0.5, color = "navy") %>%
  gf_contour(density ~ waiting + eruptions, data = faithfuld, bins = 10, color = "red")
```

`gf_count` *Formula interface to geom_count()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_count(object = NULL, gformula = NULL, data = NULL, geom = "point",
  stat = "sum", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.

data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, shape, size, stroke

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_count()

Examples

```
# Best used in conjunction with scale_size_area which ensures that
# counts of zero would be given size 0. Doesn't make much difference
# here because the smallest count is already close to 0.

gf_count(hwy ~ cty, data = mpg, alpha = 0.5) %>%
  gf_refine(scale_size_area())
```

gf_crossbar *Formula interface to geom_crossbar()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_crossbar(object = NULL, gformula = NULL, data = NULL,
  geom = "crossbar", stat = "identity", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y + ymin + ymax ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `fatten`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_crossbar()`

Examples

```
if (require(mosaicData) && require(dplyr)) {
HELP2 <- HELPrct %>%
  group_by(substance, sex) %>%
  summarise(
    mean.age = mean(age),
    median.age = median(age),
    max.age = max(age),
    min.age = min(age),
    sd.age = sd(age),
    lo = mean.age - sd.age,
    hi = mean.age + sd.age
    )

  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_pointrange( mean.age + lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_errorbar( lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_boxplot( age ~ substance,  data = HELPrct, color = "red") %>%
    gf_crossbar( mean.age + lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
}
```

`gf_curve` *Formula interface to geom_curve()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_curve(object = NULL, gformula = NULL, data = NULL, geom = "curve",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y + yend ~ x + xend`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `curvature`, `angle`, `ncp`, `arrow`, `lineend`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_curve()

Examples

```
D <- data.frame(x1 = 2.62, x2 = 3.57, y1 = 21.0, y2 = 15.0)
gf_point(mpg ~ wt, data = mtcars) %>%
  gf_curve(y1 + y2 ~ x1 + x2, data = D, color = "navy") %>%
  gf_segment(y1 + y2 ~ x1 + x2, data = D, color = "red")
```

---

gf_dens *Formula interface to geom_line() and stat_density()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_dens(object = NULL, gformula = NULL, data = NULL, geom = "line",
  stat = "density", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include stat, alpha, color, fill, group, linetype, size, weight

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_line()`

Examples

```
gf_dens()
gf_density(~ Sepal.Length,  color = ~Species, data = iris)
gf_dens(~ Sepal.Length, color = ~Species, data = iris)
gf_freqpoly(~ Sepal.Length, color = ~Species, data = iris)
# Chaining in the data
iris %>% gf_dens(~ Sepal.Length, color = ~Species)
```

---

gf_density *Formula interface to geom_density()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_density(object = NULL, gformula = NULL, data = NULL, geom = "area",
  stat = "density", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size, weight

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_density()

Examples

```
gf_dens()
gf_density(~ Sepal.Length,  color = ~Species, data = iris)
gf_dens(~ Sepal.Length, color = ~Species, data = iris)
gf_freqpoly(~ Sepal.Length, color = ~Species, data = iris)
# Chaining in the data
iris %>% gf_dens(~ Sepal.Length, color = ~Species)
```

---

gf_density2d *Formula interface to geom_density2d()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_density2d(object = NULL, gformula = NULL, data = NULL,
  geom = "density2d", stat = "density2d", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `contour`, `n`, `h`, `lineend`, `linejoin`, `linemitre`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_density2d()`

Examples

```
if (require(mosaicData)) {
  gf_jitter(i1 ~ age, alpha = 0.2, data = HELPrct, width = 0.4, height = 0.4) %>%
  gf_density2d(i1 ~ age, data = HELPrct)
}
```

---

gf_density_2d *Formula interface to geom_density_2d()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_density_2d(object = NULL, gformula = NULL, data = NULL,
  geom = "density_2d", stat = "density_2d", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `contour`, `n`, `h`, `lineend`, `linejoin`, `linemitre`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_density_2d()`

Examples

```
if (require(mosaicData)) {
  gf_jitter(i1 ~ age, alpha = 0.2, data = HELPrct, width = 0.4, height = 0.4) %>%
  gf_density_2d(i1 ~ age, data = HELPrct)
}
```

`gf_dist` *Plot distributions*

Description

Create a layer displaying a probability distribution.

Usage

```
gf_dist(object = geom_blank(), dist, ..., xlim = NULL, kind = c("density",
  "cdf", "qq", "qqstep", "histogram"), resolution = 5000L, params = NULL)
```

Arguments

`object`	a gg object.
`dist`	A character string providing the name of a distribution. Any distribution for which the functions with names formed by prepending "d", "p", or "q" to `dist` exist can be used.
`...`	additional arguments passed both to the distribution functions and to the layer. Note: avoid possible ambiguities using `params`.
`xlim`	A numeric vector of length 2 providing lower and upper bounds for the portion of the distribution that will be displayed. The default is to attempt to determine reasonable bounds using quantiles of the distribution.
`kind`	One of ‘"density"‘, ‘"cdf"‘, ‘"qq"‘, ‘"qqstep"‘, or ‘"histogram"‘ describing what kind of plot to create.
`resolution`	An integer specifying the number of points to use for creating the plot.
`params`	a list of parameters for the distribution.

Examples

```
gf_histogram( ..density.. ~ rnorm(100), bins = 20) %>%
  gf_dist("norm", color = "red")

gf_dist(dist = "norm", color = "red")

gf_dist("norm", color = "red")
gf_dist("norm", color = "red", kind = "cdf")
gf_dist("norm", fill = "red", kind = "histogram")
gf_dist("norm", color = "red", kind = "qqstep", resolution = 25) %>%
gf_dist("norm", color = "black", kind = "qq", resolution = 25)
```

```
# This doesn't work well because size has two meanings
gf_dist("binom", size = 20, prob = 0.25)
# This is better
gf_dist("binom", params = list(size = 20, prob = 0.25))
```

---

gf_dotplot *Formula interface to geom_dotplot()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_dotplot(object = NULL, gformula = NULL, data = NULL,
  geom = "dotplot", stat = "bindot", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, binwidth, binaxis, method, binpositions, stackdir, stackratio, dotsize, stackgroups, origin, right, width, drop

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_dotplot()`

Examples

```
gf_dotplot(~ Sepal.Length, fill = ~Species, data = iris)
```

---

gf_errorbar *Formula interface to geom_errorbar()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_errorbar(object = NULL, gformula = NULL, data = NULL,
  geom = "errorbar", stat = "identity", position = "identity",
  show.legend = NA, show.help = NULL, inherit = FALSE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `ymin + ymax ~ x`. Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.

geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_errorbar()

Examples

```
if (require(mosaicData) && require(dplyr)) {
HELP2 <- HELPrct %>%
  group_by(substance, sex) %>%
  summarise(
    mean.age = mean(age),
    median.age = median(age),
    max.age = max(age),
    min.age = min(age),
    sd.age = sd(age),
    lo = mean.age - sd.age,
```

```
    hi = mean.age + sd.age
    )

  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_pointrange( mean.age + lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_errorbar( lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_boxplot( age ~ substance,  data = HELPrct, color = "red") %>%
    gf_crossbar( mean.age + lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
}
```

`gf_errorbarh` *Formula interface to geom_errorbarh()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_errorbarh(object = NULL, gformula = NULL, data = NULL,
  geom = "errorbarh", stat = "identity", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `y ~ x + xmin + xmax`. Faceting can be achieved by including `|` in the formula. Note: The odd shape for this is due to a quirk in **ggplot2** which has been changed on github, but not yet on CRAN.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.
`position`	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
`show.legend`	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.

show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_errorbarh()

Examples

```
if (require(mosaicData) && require(dplyr)) {
HELP2 <- HELPrct %>%
  group_by(substance, sex) %>%
  summarise(
    mean.age = mean(age),
    median.age = median(age),
    max.age = max(age),
    min.age = min(age),
    sd.age = sd(age),
    lo = mean.age - sd.age,
    hi = mean.age + sd.age
    )

  gf_jitter(substance ~ age, data = HELPrct,
      alpha = 0.5, height = 0.2, width = 0, color = "skyblue") %>%
    gf_errorbarh( substance ~ mean.age + lo + hi,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
```

```
    gf_errorbar( lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
}
```

---

gf_frame *Formula interface to geom_blank()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_frame(object = NULL, gformula = NULL, data = NULL, geom = "blank",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)

gf_blank(object = NULL, gformula = NULL, data = NULL, geom = "blank",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_blank()`

Examples

```
gf_point((c(0,1)) ~ (c(0,5)))
gf_frame((c(0,1)) ~ (c(0,5)))
gf_blank((c(0,1)) ~ (c(0,5)))
```

---

gf_freqpoly *Formula interface to geom_freqpoly()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_freqpoly(object = NULL, gformula = NULL, data = NULL, geom = "path",
  stat = "bin", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, group, linetype, size, binwidth, bins, center, boundary,

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_freqpoly()

Examples

```
gf_histogram(~ Sepal.Length | Species, alpha = 0.2, data = iris, bins = 20) %>%
  gf_freqpoly(~ Sepal.Length, data = iris, color = ~Species, bins = 20)
gf_freqpoly(~ Sepal.Length, color = ~Species, data = iris, bins = 20)
gf_dens(~ Sepal.Length, data = iris, color = "navy") %>%
gf_freqpoly(~ Sepal.Length, y = ~..density.., data = iris, color = "red", bins = 20)
```

---

gf_function *Layers displaying graphs of functions*

Description

These functions provide two different interfaces for creating a layer that contains the graph of a function.

Usage

```
gf_function(object = NULL, fun, xlim, ..., inherit = FALSE)

gf_fun(object = NULL, formula, xlim, ..., inherit = FALSE)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
fun	A function.
xlim	A numeric vector providing the extent of the x-axis when creating the first layer in a plot. Ignored when creating a subsequent layer.
...	Other arguments such as position="dodge".
inherit	A logical indicating whether attributes should be inherited.
formula	A formula describing a function. See examples and makeFun().

Examples

```
gf_function(fun = sqrt, xlim = c(0, 10))
if (require(mosaicData)) {
  gf_histogram(..density.. ~ age, data = HELPrct, binwidth = 3, alpha = 0.6) %>%
    gf_function(fun = dnorm,
      args = list(mean = mean(HELPrct$age), sd = sd(HELPrct$age)),
      color = "red")
}
gf_fun(5 + 3 * cos(10 * x) ~ x, xlim = c(0,2))
# Utility bill is quadratic in month?
f <- makeFun(lm(totalbill ~ poly(month, 2), data = Utilities))
gf_point(totalbill ~ month, data = Utilities, alpha = 0.6) %>%
  gf_fun(f(m) ~ m, color = "red")
```

gf_hex *Formula interface to geom_hex()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_hex(object = NULL, gformula = NULL, data = NULL, geom = "hex",
  stat = "binhex", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include bins, binwidth, alpha, color, fill, group, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form `+ attribute::value` map attribute to value. Additional terms of the form `+ attribute:value` will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_hex()`

Examples

```
if (require(mosaicData)) {
  gf_hex(i1 ~ age, data = HELPrct, bins = 15) %>%
  gf_density2d(i1 ~ age, data = HELPrct, color = "red", alpha = 0.5)
}
```

---

gf_histogram *Formula interface to geom_histogram()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_histogram(object = NULL, gformula = NULL, data = NULL, geom = "bar",
  stat = "bin", position = "stack", show.legend = NA, show.help = NULL,
  inherit = TRUE, ...)

gf_dhistogram(object = NULL, gformula = NULL, data = NULL, geom = "bar",
  stat = "bin", position = "stack", show.legend = NA, show.help = NULL,
  inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `~x` or `y ~ x`. `y` may be `..density..` or `..count..` or `..ndensity..` or `..ncount...` Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.

position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_histogram()

Examples

```
x <- rnorm(1000)
gf_histogram(  ~ x, bins = 30)
gf_histogram( ..density.. ~ x, bins = 30)
gf_histogram(~ Sepal.Length | Species, data = iris, binwidth = 0.25)
if (require(mosaicData)) {
  gf_histogram(~age, data = HELPrct, binwidth = 5, fill = "skyblue", color = "black")
  # bins can be adjusted left/right using center or boundary
  gf_histogram(~age, data = HELPrct, binwidth = 5, fill = "skyblue", color = "black", center = 42.5)
  gf_histogram(~age, data = HELPrct, binwidth = 5, fill = "skyblue", color = "black", boundary = 40)
}
```

gf_jitter *Formula interface to geom_jitter()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_jitter(object = NULL, gformula = NULL, data = NULL, geom = "point",
  stat = "identity", position = "jitter", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `size`, `shape`, `fill`, `group`, `stroke`, `width`, `height`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form

facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_jitter()

Examples

```
gf_jitter()
if (require(mosaicData)) {
  # without jitter
  gf_point(age ~ sex, alpha = 0.25, data = HELPrct)
  # jitter only horizontally
  gf_jitter(age ~ sex, alpha = 0.25, data = HELPrct, width = 0.2, height = 0)
  # alternative way to get jitter
  gf_point(age ~ sex, alpha = 0.25, data = HELPrct,
    position = "jitter", width = 0.2, height = 0)
}
```

---

gf_label *Formula interface to geom_label()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_label(object = NULL, gformula = NULL, data = NULL, geom = "label",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape y ~ x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.

geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include label, alpha, angle, color, family, fontface, group, hjust, lineheight, size, vjust, parse, nudge_x, nudge_y, lparse, nudge_x, nudge_y, label.padding, label.r, label.size, check_overlap

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_label()

Examples

```
if (require(dplyr)) {
  iris_means <-
    iris %>%
    group_by(Species) %>%
    summarise(Sepal.Length = mean(Sepal.Length), Sepal.Width = mean(Sepal.Width))
  gf_point(Sepal.Length ~ Sepal.Width, data = iris, color = ~ Species) %>%
  gf_label(Sepal.Length ~ Sepal.Width, data = iris_means,
    label = ~Species, color = ~Species, size = 2, alpha = 0.7)
}
```

gf_labs *Non-layer functions for gf plots*

Description

These functions modify things like labels, limits, scales, etc. for plots ggplot2 plots. They are wrappers around functions in ggplot2 that allow for chaining syntax.

Usage

```
gf_labs(object, ...)

gf_lims(object, ...)

gf_facet_wrap(object, ...)

gf_facet_grid(object, ...)

gf_refine(object, ...)
```

Arguments

object	a gg object
...	additional arguments passed through to the similarly named function in **ggplot2**.

Details

`gf_refine()` provides a mechanism to replace + with the chaining operator from **magrittr**. Each of its `...` arguments is added in turn to the base plot in `object`. The other functions are thin wrappers around specific `ggplot2` refinement functions and pass their `...` arguments through to the similarly named `ggplot2` functions.

Value

a modified gg object

Examples

```
if (require(mosaicData)) {
gf_dens( ~ cesd, color = ~ substance, size = 1.5, data = HELPrct) %>%
gf_labs(
  title = "Center for Epidemiologic Studies Depression measure",
  subtitle = "(at baseline)",
  color = "Abused substance: ",
  x = "CESD score",
  y = "",
  caption = "Source: HELPrct"
) %>%
```

```
  gf_theme(theme_classic()) %>%
  gf_theme(
    axis.text.y = element_blank(),
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, color = "navy"),
    plot.subtitle = element_text(hjust = 0.5, color = "navy", size = 12))
}
gf_point(eruptions ~ waiting, data = faithful, alpha = 0.5)
gf_point(eruptions ~ waiting, data = faithful, alpha = 0.5) %>%
  gf_lims(x = c(65, NA), y = c(3, NA))

# modify scales using gf_refine()
gf_jitter(Sepal.Length ~ Sepal.Width, color = ~ Species, data = iris) %>%
  gf_refine(scale_color_brewer(type = "qual", palette = 3)) %>%
  gf_theme(theme_bw())

gf_jitter(Sepal.Length ~ Sepal.Width, color = ~ Species, data = iris) %>%
  gf_refine(scale_color_manual(values = c("red", "navy", "limegreen"))) %>%
  gf_theme(theme_bw())
```

`gf_line` *Formula interface to geom_line()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_line(object = NULL, gformula = NULL, data = NULL, geom = "line",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.

show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size, lineend, linejoin, linemitre, arrow

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_line()

Examples

```
gf_line()
if (require(mosaicData)) {
  gf_point(age ~ sex, alpha = 0.25, data = HELPrct)
  gf_point(births ~ date, color = ~wday, data = Births78)
  # lines make the exceptions stand out more prominently
  gf_line(births ~ date, color = ~wday, data = Births78)
  }
```

gf_linerange *Formula interface to geom_linerange()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_linerange(object = NULL, gformula = NULL, data = NULL,
  geom = "linerange", stat = "identity", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `ymin + ymax ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include alpha, color, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form `+ attribute::value` map attribute to value. Additional terms of the form `+ attribute:value` will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_linerange()`

Examples

```
gf_linerange()
if (require(weatherData) & require(dplyr)) {
Temps <- NewYork2013 %>% mutate(city = "NYC") %>%
bind_rows(Mumbai2013 %>% mutate(city = "Mumbai")) %>%
bind_rows(London2013 %>% mutate(city = "London")) %>%
  mutate(date = lubridate::date(Time),
         month = lubridate::month(Time)) %>%
  group_by(city, date) %>%
  summarise(
    hi = max(Temperature, na.rm = TRUE),
    lo = min(Temperature, na.rm = TRUE),
    mid = (hi + lo)/2
  )

gf_ribbon(lo + hi ~ date, data = Temps, fill = ~city, alpha = 0.4) %>%
   gf_theme(theme = theme_minimal())
gf_linerange(lo + hi ~ date | city ~ ., color = ~mid, data = Temps) %>%
  gf_refine(scale_colour_gradientn(colors = rev(rainbow(5))))
gf_ribbon(lo + hi ~ date | city ~ ., data = Temps)
# Chaining in the data
Temps %>% gf_ribbon(lo + hi ~ date, alpha = 0.4) %>%
  gf_facet_grid(city ~ .)
}
```

---

gf_path *Formula interface to geom_path()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_path(object = NULL, gformula = NULL, data = NULL, geom = "path",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `lineend`, `linejoin`, `linemitre`, `arrow`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_path()

Examples

```
gf_path()
if (require(dplyr)) {
  data.frame(t = seq(1, 10 * pi, length.out = 400)) %>%
  mutate( x = t * cos(t), y = t * sin(t)) %>%
  gf_path(y ~ x, color = ~t)
  }
```

---

gf_point *Formula interface to geom_point()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_point(object = NULL, gformula = NULL, data = NULL, geom = "point",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape y ~ x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, (c) attributes of the layer as a whole, which are set with attribute = value, or (d) arguments for the geom, stat, or position function. Available attributes include alpha, color, size, shape, fill, group, stroke

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_point()`

Examples

```
gf_point()
gf_point(mpg ~ hp, color = ~ cyl, size = ~wt, data = mtcars)
# faceting -- two ways
gf_point(mpg ~ hp, data = mtcars) %>%
  gf_facet_wrap(~ am)
gf_point(mpg ~ hp | am, group = ~ cyl, data = mtcars)
gf_point(mpg ~ hp | ~ am, group = ~ cyl, data = mtcars)
gf_point(mpg ~ hp | am ~ ., group = ~ cyl,  data = mtcars)

# Chaining in the data
mtcars %>% gf_point(mpg ~ wt)
```

---

gf_pointrange *Formula interface to geom_pointrange()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_pointrange(object = NULL, gformula = NULL, data = NULL,
  geom = "pointrange", stat = "identity", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y + ymin + ymax ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `fatten`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_pointrange()`

Examples

```
if (require(mosaicData) && require(dplyr)) {
HELP2 <- HELPrct %>%
  group_by(substance, sex) %>%
  summarise(
    mean.age = mean(age),
    median.age = median(age),
    max.age = max(age),
    min.age = min(age),
    sd.age = sd(age),
    lo = mean.age - sd.age,
    hi = mean.age + sd.age
    )

  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_pointrange( mean.age + lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
  gf_jitter(age ~ substance, data = HELPrct,
      alpha = 0.5, width = 0.2, height = 0, color = "skyblue") %>%
    gf_errorbar( lo + hi ~ substance,  data = HELP2) %>%
    gf_facet_grid( ~ sex)
}
```

---

gf_qq *Formula interface to geom_qq()*

Description

`gf_qq()` an `gf_qqstep()` both create quantile-quantile plots. They differ in how they display the qq-plot. `gf_qq()` uses points and `gf_qqstep()` plots a step function through these points.

Usage

```
gf_qq(object = NULL, gformula = NULL, data = NULL, geom = "point",
  stat = "qq", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)

gf_qqline(object = NULL, gformula = NULL, data = NULL, geom = "line",
  stat = "qqline", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)

gf_qqstep(object = NULL, gformula = NULL, data = NULL, geom = "step",
  stat = "qq", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~sample. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include group, x, y, distribution, dparams

Details

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_qq()

Examples

```
gf_qq(~rnorm(100))
gf_qq(~Sepal.Length | Species, data = iris) %>% gf_qqline()
gf_qq(~Sepal.Length | Species, data = iris) %>% gf_qqline(tail = 0.10)
gf_qq(~Sepal.Length, color = ~Species, data = iris) %>%
gf_qqstep(~Sepal.Length, color = ~Species, data = iris)
```

`gf_quantile` *Formula interface to geom_quantile()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_quantile(object = NULL, gformula = NULL, data = NULL,
  geom = "quantile", stat = "quantile", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.
`position`	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
`show.legend`	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
`show.help`	If `TRUE`, display some minimal help.
`inherit`	A logical indicating whether default attributes are inherited.
`...`	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `weight`, `lineend`, `linejoin`, `linemitre`, `quantiles`, `formula`, `method`, `method.args`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_quantile()`

Examples

```
gf_point((1/hwy) ~ displ, data = mpg) %>%
  gf_quantile((1/hwy) ~ displ)
```

---

gf_raster *Formula interface to geom_raster()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_raster(object = NULL, gformula = NULL, data = NULL, geom = "raster",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.

data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size, hjust, vjust, interpolate

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_raster()

Examples

```
# Justification controls where the cells are anchored
D <- expand.grid(x = 0:5, y = 0:5)
D$z <- runif(nrow(D))
# centered squares
gf_raster(z ~ x + y, data = D)
gf_raster(y ~ x, fill = ~ z, data = D)
# zero padding
gf_raster(z ~ x + y, data = D, hjust = 0, vjust = 0)
```

gf_rect *Formula interface to geom_rect()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_rect(object = NULL, gformula = NULL, data = NULL, geom = "rect",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ymin + ymax ~ xmin + xmax. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_rect()`

Examples

```
gf_rect( 1 + 2 ~ 3 + 4, alpha = 0.3, color = "red")
```

---

gf_ribbon *Formula interface to geom_ribbon()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_ribbon(object = NULL, gformula = NULL, data = NULL, geom = "ribbon",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `ymin + ymax ~ x`. Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.
`position`	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
`show.legend`	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
`show.help`	If `TRUE`, display some minimal help.

inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_ribbon()

Examples

```
gf_ribbon()
if (require(weatherData) & require(dplyr)) {
Temps <- NewYork2013 %>% mutate(city = "NYC") %>%
bind_rows(Mumbai2013 %>% mutate(city = "Mumbai")) %>%
bind_rows(London2013 %>% mutate(city = "London")) %>%
  mutate(date = lubridate::date(Time),
         month = lubridate::month(Time)) %>%
  group_by(city, date) %>%
  summarise(
    hi = max(Temperature, na.rm = TRUE),
    lo = min(Temperature, na.rm = TRUE),
    mid = (hi + lo)/2
  )

gf_ribbon(lo + hi ~ date, data = Temps, fill = ~city, alpha = 0.4) %>%
   gf_theme(theme = theme_minimal())
gf_linerange(lo + hi ~ date | city ~ ., color = ~mid, data = Temps) %>%
  gf_refine(scale_colour_gradientn(colors = rev(rainbow(5))))
gf_ribbon(lo + hi ~ date | city ~ ., data = Temps)
# Chaining in the data
```

```
Temps %>% gf_ribbon(lo + hi ~ date, alpha = 0.4) %>%
  gf_facet_grid(city ~ .)
}
```

---

gf_rug *Formula interface to geom_rug()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_rug(object = NULL, gformula = NULL, data = NULL, geom = "rug",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape ~x or y ~ x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include sides, alpha, color, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_rug()`

Examples

```
gf_histogram(~eruptions, data = faithful) %>%
gf_rug(~eruptions, data = faithful, color = "red", sides = "bl") %>%
gf_rug(~eruptions, data = faithful, color = "navy", sides = "tr")
gf_point(Sepal.Length ~ Sepal.Width, data = iris) %>%
gf_rug(Sepal.Length ~ Sepal.Width)
gf_point(Sepal.Length ~ Sepal.Width, data = iris) %>%
gf_rug( x = ~ Sepal.Width, data = iris, color = "navy") %>%
gf_rug( y = ~ Sepal.Length, data = iris, color = "red")
```

---

gf_segment *Formula interface to geom_segment()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_segment(object = NULL, gformula = NULL, data = NULL,
  geom = "segment", stat = "identity", position = "identity",
  show.legend = NA, show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape y + yend ~ x + xend. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, group, linetype, size, arrow, lineend

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_segment()

Examples

```
D <- data.frame(x1 = 2.62, x2 = 3.57, y1 = 21.0, y2 = 15.0)
gf_point(mpg ~ wt, data = mtcars) %>%
  gf_curve(y1 + y2 ~ x1 + x2, data = D, color = "navy") %>%
  gf_segment(y1 + y2 ~ x1 + x2, data = D, color = "red")
```

`gf_smooth` *Formula interface to geom_smooth()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_smooth(object = NULL, gformula = NULL, data = NULL, geom = "smooth",
  stat = "smooth", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)

gf_lm(object = NULL, gformula = NULL, data = NULL, geom = "lm",
  stat = "lm", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.
`position`	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
`show.legend`	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
`show.help`	If `TRUE`, display some minimal help.
`inherit`	A logical indicating whether default attributes are inherited.
`...`	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `method`, `formula`, `se`, `method.args`, `n`, `span`, `fullrange`, `level`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_smooth()`

Examples

```
gf_smooth()
gf_lm()
if (require(mosaicData)) {
  gf_smooth(births ~ date, color = ~wday, data = Births78)
  gf_smooth(births ~ date, color = ~wday, data = Births78, fullrange = TRUE)
  gf_smooth(births ~ date, color = ~wday, data = Births78, show.legend = FALSE, se = FALSE)
  gf_lm(length ~ width, data = KidsFeet, color = ~biggerfoot, alpha = 0.2) %>%
    gf_point()
  gf_lm(length ~ width, data = KidsFeet, color = ~biggerfoot, fullrange = FALSE, alpha = 0.2)
    gf_point()
  gf_lm(length ~ width, color = ~ sex, data = KidsFeet,
        formula = y ~ poly(x,2), linetype = "dashed") %>%
    gf_point()
  gf_lm(length ~ width, color = ~ sex, data = KidsFeet,
        formula = log(y) ~ x, backtrans = exp) %>%
    gf_point()
}
gf_lm(hwy ~ displ, data = mpg,
      formula = log(y) ~ poly(x,3), backtrans = exp,
     interval = "prediction", fill = "skyblue") %>%
  gf_lm(
     formula = log(y) ~ poly(x,3), backtrans = exp,
     interval = "confidence", color = "red") %>%
  gf_point()
```

gf_spline *Formula interface to geom_spline()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_spline(object = NULL, gformula = NULL, data = NULL, geom = "line",
  stat = "spline", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `weight`, `df`, `spar`, `tol`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form

facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_spline()

Examples

```
if (require(mosaicData)) {
  gf_spline(births ~ date, color = ~wday, data = Births78)
  gf_spline(births ~ date, color = ~wday, data = Births78, df = 20)
  gf_spline(births ~ date, color = ~wday, data = Births78, df = 4)
}
```

---

gf_spoke *Formula interface to geom_spoke()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_spoke(object = NULL, gformula = NULL, data = NULL, geom = "spoke",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape y ~ x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.

show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include angle, radius, alpha, color, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

Note angle and radius must be set or mapped.

NA

See Also

geom_spoke()

Examples

```
D <- expand.grid(x = 1:10, y=1:10)
D$angle <- runif(100, 0, 2*pi)
D$speed <- runif(100, 0, sqrt(0.1 * D$x))

gf_point(y ~ x, data = D) %>%
  gf_spoke(y ~ x, angle = ~angle, radius = 0.5)

gf_point(y ~ x, data = D) %>%
  gf_spoke(y ~ x, angle = ~angle, radius = ~speed)
```

gf_step *Formula interface to geom_step()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_step(object = NULL, gformula = NULL, data = NULL, geom = "step",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `group`, `linetype`, `size`, `direction`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_step()`

Examples

```
if (require(mosaicData)) {
  gf_step( births ~ date, data = Births78, color = ~wday)
}
```

`gf_text` *Formula interface to geom_text()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_text(object = NULL, gformula = NULL, data = NULL, geom = "text",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

`object`	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
`gformula`	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
`data`	A data frame with the variables to be plotted.
`geom`	A character string naming the geom used to make the layer.
`stat`	A character string naming the stat used to make the layer.
`position`	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
`show.legend`	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.

show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include label, alpha, angle, color, family, fontface, group, hjust, lineheight, size, vjust, parse, nudge_x, nudge_y, check_overlap

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Additional formula terms of the form + attribute::value map attribute to value. Additional terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value. Alternatively (and preferably) attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

geom_text()

Examples

```
gf_text(Sepal.Length ~ Sepal.Width, data = iris,
  label = ~Species, color = ~Species, size = 2, angle = 30)
```

---

gf_theme *Themes for ggformula*

Description

Themes for ggformula

Usage

```
gf_theme(object, theme, ...)
```

Arguments

object	a gg object
theme	a ggplot2 theme function like theme_minimal.
...	If theme is missing, then these additional arguments are theme elements of the sort handled by theme().

Value

a modified gg object

---

gf_tile *Formula interface to geom_tile()*

Description

ggformula functions provide a formula interface to ggplot2 layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_tile(object = NULL, gformula = NULL, data = NULL, geom = "tile",
  stat = "identity", position = "identity", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape y ~ x. Faceting can be achieved by including | in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. NA, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If TRUE, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with attribute = value, (b) ggplot2 aesthetics to be mapped with attribute = ~expression, or (c) attributes of the layer as a whole, which are set with attribute = value. Available attributes include alpha, color, fill, group, linetype, size

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_tile()`

Examples

```
D <- expand.grid(x = 0:5, y = 0:5)
D$z <- runif(nrow(D))
gf_tile(y ~ x, fill = ~ z, data = D)
gf_tile(z ~ x + y, data = D)
```

---

gf_violin *Formula interface to geom_violin()*

Description

ggformula functions provide a formula interface to `ggplot2` layer functions. For plots with just one layer, the formula interface is more compact and is consistent with modeling and **mosaic** notation.

Usage

```
gf_violin(object = NULL, gformula = NULL, data = NULL, geom = "violin",
  stat = "ydensity", position = "dodge", show.legend = NA,
  show.help = NULL, inherit = TRUE, ...)
```

Arguments

object	When chaining, this holds an object produced in the earlier portions of the chain. Most users can safely ignore this argument. See details and examples.
gformula	A formula with shape `y ~ x`. Faceting can be achieved by including `|` in the formula.
data	A data frame with the variables to be plotted.
geom	A character string naming the geom used to make the layer.
stat	A character string naming the stat used to make the layer.
position	Either a character string naming the position function used for the layer or a position object returned from a call to a position function.
show.legend	A logical indicating whether this layer should be included in the legends. `NA`, the default, includes layer in the legends if any of the attributes of the layer are mapped.
show.help	If `TRUE`, display some minimal help.
inherit	A logical indicating whether default attributes are inherited.
...	Additional arguments. Typically these are (a) ggplot2 aesthetics to be set with `attribute = value`, (b) ggplot2 aesthetics to be mapped with `attribute = ~expression`, or (c) attributes of the layer as a whole, which are set with `attribute = value`. Available attributes include `alpha`, `color`, `fill`, `group`, `linetype`, `size`, , `draw_quatiles`, `trim`, `scale`, `bw`, `adjust`, `kernel`

Details

Positional aesthetics are typically specified using a formula (see the `gformula` argument). Additional formula terms of the form `+ attribute::value` map `attribute` to `value`. Additional terms of the form `+ attribute:value` will map `attribute` to `value` if `value` is the name of a variable in `data`, else `attribute` will be set to the constant `value`. Alternatively (and preferably) attributes can be set can be set using arguments of the form `attribute = value` or mapped using arguments of the form `attribute = ~ expression`. In formulas of the form `A | B`, `B` will be used to form facets using `facet_wrap()` or `facet_grid()`. This provides an alternative to `gf_facet_wrap()` and `gf_facet_grid()` that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of `gformula`. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Value

a gg object

See Also

`geom_violin()`

Examples

```
if (require(mosaicData)) {
  gf_violin(age ~ substance, data = HELPrct)
  gf_violin(age ~ substance, data = HELPrct, fill = ~sex)
}
```

---

ggformula *Formula interface to ggplot2*

Description

The functions in **ggformula** provide a formula interface to ggplot2 layer functions and a system for working with pipes to create multi-layer plots and to refine plots. For plots with just one layer, the formula interface is more compact than native **ggplot2** code and is consistent with modeling functions like lm() that use a formula interface and with the numerical summary functions in the **mosaic** package.

Details

Positional aesthetics are typically specified using a formula (see the gformula argument). Setting and mapping of additional attributes can be done within the formula or through the use of additional arguments. The latter is considered preferable. Attributes can be set can be set using arguments of the form attribute = value or mapped using arguments of the form attribute = ~ expression. Additional formula terms of the form + attribute::value map attribute to value; terms of the form + attribute:value will map attribute to value if value is the name of a variable in data, else attribute will be set to the constant value.

In formulas of the form A | B, B will be used to form facets using facet_wrap() or facet_grid(). This provides an alternative to gf_facet_wrap() and gf_facet_grid() that is terser and may feel more familiar to users of **lattice**.

Evaluation of the **ggplot2** code occurs in the environment of gformula. This will typically do the right thing when formulas are created on the fly, but might not be the right thing if formulas created in one environment are used to create plots in another.

Examples

```
apropos("gf_")
gf_point()
```

StatAsh *ggproto classes for ggplot2*

Description

These are typically accessed through their associated `geom_*`, `stat_*` or `gf_*` functions.

Usage

```
StatAsh

StatSpline

StatQqline

StatLm

GeomLm
```

See Also

`stat_ash()`
`gf_ash()`
`stat_spline()`
`gf_spline()`
`stat_qq()`
`gf_qq()`
`stat_lm()`
`gf_lm()`
`geom_lm()`
`gf_lm()`

stat_lm *Linear Model Displays*

Description

Adds linear model fits to plots. `geom_lm()` and `stat_lm()` are essentially equivalent. Use `geom_lm()` unless you want a non-standard geom.

Usage

```
stat_lm(mapping = NULL, data = NULL, geom = "lm", position = "identity",
  interval = c("none", "prediction", "confidence"), level = 0.95,
  formula = y ~ x, lm.args = list(), backtrans = identity, ...,
  na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)

geom_lm(mapping = NULL, data = NULL, stat = "lm", position = "identity",
  interval = c("none", "prediction", "confidence"), level = 0.95,
  formula = y ~ x, lm.args = list(), backtrans = identity, ...,
  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.
geom, stat	Use to override the default connection between geom_lm and stat_lm.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
interval	One of "none", "confidence" or "prediction".
level	The level used for confidence or prediction intervals
formula	a formula describing the model in terms of y (response) and x (predictor).
lm.args	A list of arguments supplied to [lm()] when performing the fit.
backtrans	a function that transforms the response back to the original scale when the formula includes a transformtion on y.
...	other arguments passed on to layer. These are often aesthetics, used to set an aesthetic to a fixed value, like color = "red" or size = 3. They may also be parameters to the paired geom/stat.
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. 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.

Details

Stat calculation is performed by the (currently undocumented) `predictdf`. Pointwise confidence or prediction bands are calculated using the [predict()] method.

See Also

[lm()] for details on linear model fitting.

Examples

```
if (require(mosaicData)) {
  ggplot(data = KidsFeet, aes(y = length, x = width, color = sex)) +
    geom_lm() +
    geom_point()
  ggplot(data = KidsFeet, aes(y = length, x = width, color = sex)) +
    geom_lm(interval = "prediction", color = "skyblue") +
    geom_lm(interval = "confidence") +
    geom_point() +
    facet_wrap(~sex)
 # non-standard display
  ggplot(data = KidsFeet, aes(y = length, x = width, color = sex)) +
    stat_lm(aes(fill = sex), color = NA, interval = "confidence", geom = "ribbon",
    alpha = 0.2) +
    geom_point() +
    facet_wrap(~sex)
ggplot(mpg, aes(displ, hwy)) +
   geom_lm(formula = log(y) ~ poly(x,3), backtrans = exp,
     interval = "prediction", fill = "skyblue") +
   geom_lm(formula = log(y) ~ poly(x,3), backtrans = exp, interval = "confidence",
     color = "red") +
   geom_point()
}
```

`stat_qqline` *A Stat for Adding Reference Lines to QQ-Plots*

Description

This stat computes quantiles of the sample and theoretical distribution for the purpose of providing reference lines for QQ-plots.

Usage

```
stat_qqline(mapping = NULL, data = NULL, geom = "line",
  position = "identity", ..., distribution = stats::qnorm,
  dparams = list(), na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
```

Arguments

mapping	An aesthetic mapping produced with aes() or aes_string().
data	A data frame.
geom	A geom.
position	A position object.
...	Additional arguments
distribution	A quantile function.
dparams	A list of arguments for distribution.
na.rm	A logical indicating whether a warning should be issued when missing values are removed before plotting.
show.legend	A logical indicating whether legends should be included for this layer. If NA, legends will be include for each aesthetic that is mapped.
inherit.aes	A logical indicating whether aesthetics should be inherited. When FALSE, the supplied mapping will be the only aesthetics used.

Examples

```
ggplot(data = iris, aes(sample = Sepal.Length)) +
  geom_qq() +
  stat_qqline( alpha = 0.7, color = "red", linetype = "dashed") +
  facet_wrap(~Species)
```

---

stat_spline *Geoms and stats for spline smoothing*

Description

Similar to [geom_smooth], this adds spline fits to plots.

Usage

```
stat_spline(mapping = NULL, data = NULL, geom = "line",
  position = "identity", na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE, weight = NULL, df = NULL, spar = NULL,
  cv = FALSE, all.knots = FALSE, nknots = stats::.nknots.smspl,
  df.offset = 0, penalty = 1, control.spar = list(), tol = NULL, ...)

geom_spline(mapping = NULL, data = NULL, stat = "spline",
  position = "identity", na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE, weight = NULL, df = NULL, spar = NULL,
  cv = FALSE, all.knots = FALSE, nknots = stats::.nknots.smspl,
  df.offset = 0, penalty = 1, control.spar = list(), tol = NULL, ...)
```

Arguments

mapping	An aesthetic mapping produced with aes() or aes_string().
data	A data frame.
geom	A geom.
position	A position object.
na.rm	A logical indicating whether a warning should be issued when missing values are removed before plotting.
show.legend	A logical indicating whether legends should be included for this layer. If NA, legends will be inclued for each aesthetic that is mapped.
inherit.aes	A logical indicating whether aesthetics should be inherited. When FALSE, the supplied mapping will be the only aesthetics used.
weight	An optional vector of weights. See smooth.spline().
df	desired equivalent degrees of freedom. See smooth.spline() for details.
spar	A smoothing parameter, typically in (0,1]. See smooth.spline() for details.
cv	A logical. See smooth.spline() for details.
all.knots	A logical. See smooth.spline() for details.
nknots	An integer or function giving the number of knots to use when all.knots = FALSE. See smooth.spline() for details.
df.offset	A numerical value used to increase the degrees of freedom when using GVC. See smooth.spline() for details.
penalty	the coefficient of the penalty for degrees of freedom in the GVC criterion. See smooth.spline() for details.
control.spar	An optional list used to control root finding when the parameter spar is computed. See smooth.spline() for details.
tol	A tolerance for sameness or uniqueness of the x values. The values are binned into bins of size tol and values which fall into the same bin are regarded as the same. Must be strictly positive (and finite). When NULL, IQR(x) * 10e-6 is used.
...	Additional arguments
stat	A stat.

Examples

```
if (require(mosaicData)) {
  ggplot(Births) + geom_spline(aes(x = date, y=births, colour = wday))
  ggplot(Births) + geom_spline(aes(x = date, y=births, colour = wday), nknots = 10)
}
```

Index