

# Package ‘ggfan’

October 13, 2022

**Type** Package

**Title** Summarise a Distribution Through Coloured Intervals

**Description** Implements the functionality of the 'fanplot' package as 'geoms' for 'ggplot2'. Designed for summarising MCMC samples from a posterior distribution, where a visualisation is desired for several values of a continuous covariate. Increasing posterior intervals of the sampled quantity are mapped to a continuous colour scale.

**Version** 0.1.3

**License** GPL-2 | file LICENSE

**LazyData** TRUE

**Depends** R (>= 3.1)

**Imports** ggplot2, colorspace, dplyr, stats, grid, rstan

**Suggests** testthat, knitr, rmarkdown, tidyr, magrittr, tibble

**URL** <https://github.com/jasonhilton/ggfan>

**BugReports** <https://github.com/jasonhilton/ggfan/issues>

**VignetteBuilder** knitr

**RoxygenNote** 6.1.1

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Jason Hilton [aut, cre]

**Maintainer** Jason Hilton <jason\_hilton@yahoo.com>

**Repository** CRAN

**Date/Publication** 2019-03-07 09:02:42 UTC

## R topics documented:

.in_numeric	2
calc_quantiles	3
fake_df	3

GeomIntervalPath . . . . .	4
GeomIntervalPoly . . . . .	4
geom_fan . . . . .	5
geom_interval . . . . .	6
ggfan . . . . .	8
gp_model_fit . . . . .	9
StatInterval . . . . .	10
StatIntervalFctr . . . . .	10
StatSample . . . . .	10
stat_interval . . . . .	10
stat_sample . . . . .	11
<b>Index</b>	<b>13</b>

---

<code>.in_numeric</code>	<i>Find elements of one numeric vector in another.</i>
--------------------------	--

---

## Description

Find which elements of 'a' are equal to at least one element in 'b', to within some tolerance tol.

## Usage

```
.in_numeric(a, b, tol = 1e-06)
```

## Arguments

a	The vector for which comparison are to be made
b	The vector to match against.
tol	The tolerance within which values are assumed to be the same.

## Value

A vector of logical values the same length as 'a' describing whether this vector element is close to at least one element in 'b'

---

calc_quantiles	<i>Calculate quantiles of a tidy dataframe</i>
----------------	--

---

**Description**

Calculate quantiles of a tidy dataframe

**Usage**

```
calc_quantiles(data, intervals, x_var = "x", y_var = "y", rename = T)
```

**Arguments**

data	A data frame with containing x and y columns, with several y values for every x
intervals	A list of intervals for which corresponding quantiles are desired.
x_var	A character string giving the name of the x variable
y_var	A character string giving the name of the y variable
rename	Logical. Indicates whether to retain existing variable name or use x and y.

**Value**

A data frame containing x, y, and quantile columns (possibly renamed)

**Examples**

```
head(fake_df)

fake_q <- calc_quantiles(fake_df, intervals=c(0,0.5,0.8))
head(fake_q)
```

---

fake_df	<i>Fake dataset intended to resemble a set of MCMC samples of a variable over one covariate (perhaps time),</i>
---------	---

---

**Description**

The code needed to recreate the dataset is included in the examples

**Usage**

```
fake_df
```

**Format**

A data frame with 50000 rows and 3 columns

**x** Values of the covariate

**y** Values of the modelled quantity

**Sim** Index referring to a MCMC posterior sample

**Examples**

```
# generate mean and variance for sequence of samples over time
library(magrittr)
library(tidyr)
set.seed(234)
N_time <- 50
N_sims <- 1000
time <- 1:N_time
mu <- time**2 * 0.03 + time * 0.3
sds <- exp(time**2 * -0.001 + time * 0.1)

# simulate 1000 samples from each time point
fake_data <- sapply(time, function(i) rnorm(N_sims, mu[i], sds[i]))

# gather into a long-form, tidy dataset
fake_df <- data.frame(x=time, t(fake_data)) %>%
tidyr::gather(key=Sim, value=y, -x)
# devtools::use_data(fake_df)
```

---

GeomIntervalPath      *See* [ggplot2-ggproto](#)

---

**Description**

See [ggplot2-ggproto](#)

---

GeomIntervalPoly      *See* [ggplot2-ggproto](#)

---

**Description**

See [ggplot2-ggproto](#)

geom\_fan

*Fan plot visualising intervals of a distribution***Description**

Fan Plots allow the distribution of a variable to be visualised by representing sets of central probability intervals through colour. For every value of  $x$ , `geom_fan` computes quantiles of  $y$  and uses these to plot intervals containing increasing proportions of the total density of  $y$ . Intervals are mapped to a continuous colour scale, so that changes in colour represent intervals covering an increasing proportion of total density. Quantiles can also be precomputed and mapped to the aesthetic quantile. This function is designed with the need to summarise MCMC posterior distributions in mind, and implements the functionality of the `fanplot` package in `ggplot2`. Note that there should be enough observations of  $y$  at each  $x$  to allow estimation of the specified quantiles.

**Usage**

```
geom_fan(mapping = NULL, data = NULL, stat = "interval",
         position = "identity", show.legend = NA, inherit.aes = TRUE,
         intervals = (2:98)/100, ...)
```

**Arguments**

<code>mapping</code>	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply <code>mapping</code> if there is no plot mapping.
<code>data</code>	The data to be displayed in this layer. There are three options: If <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data.
<code>stat</code>	Use to override the default use of <code>stat_interval</code>
<code>position</code>	Position adjustment, either as a string, or the result of a call to a position adjustment function.
<code>show.legend</code>	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
<code>inherit.aes</code>	If <code>FALSE</code> , 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. <code>borders()</code> .
<code>intervals</code>	specify the collection of intervals to be represented in the fan.
<code>...</code>	other arguments passed on to <code>layer</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>color = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired <code>geom/stat</code> .

**Aesthetics**

geom\_fan understands the following aesthetics (required aesthetics are in bold):

- **x**
- **y**
- alpha
- group
- quantile

**See Also**

stat\_summary Summarises y at each value of x

stat\_quantile Uses quantile regression to predict quantiles

geom\_interval Plot intervals boundaries as lines

**Examples**

```
# Basic use. The data frame must have multiple y values for each
# x
library(ggplot2)

ggplot(fake_df, aes(x=x,y=y)) +geom_fan()

# use precomputed quantiles - reducing storage requirements.
intervals = 1:19/20
fake_q <- calc_quantiles(fake_df, intervals=intervals)
# intervals in geom_fan must be the same as used to compute quantiles.
ggplot(fake_q, aes(x=x,y=y, quantile=quantile)) +
  geom_fan(intervals=intervals)

# change the colour scale
ggplot(fake_df, aes(x=x,y=y)) + geom_fan() + scale_fill_gradient(low="red", high="pink")
```

---

geom\_interval

*Line plot visualising intervals of a distribution*

---

**Description**

For every value of x, computes quantiles of y and uses these to plot intervals containing increasing proportions of the total density of y. Boundaries of intervals are mapped to linetype. Quantiles can also be precomputed and mapped to the aesthetic quantile. This function is designed with the need to summarise MCMC posterior distributions in mind.

**Usage**

```
geom_interval(mapping = NULL, data = NULL, stat = "interval_fctr",
  position = "identity", intervals = c(0, 50, 90)/100,
  lineend = "butt", linejoin = "round", linemitre = 1,
  arrow = NULL, na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE, ...)
```

**Arguments**

mapping	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (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 <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data.
stat	Use to override the default use of <code>stat_interval</code>
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
intervals	specify the collection of intervals to be represented in the fan.
lineend	Line end style (round, butt, square).
linejoin	Line join style (round, mitre, bevel).
linemitre	Line mitre limit (number greater than 1).
arrow	Arrow specification, as created by <code>grid::arrow()</code> .
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , 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. <code>borders()</code> .
...	Other arguments passed on to <code>layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired <code>geom/stat</code> .

**Aesthetics**

`geom_interval` understands the following aesthetics (required aesthetics are in bold):

- **x**

- `y`
- `quantile`
- `group`
- `colour`
- `size`

### See Also

`stat_summary` Summarises `y` at each value of `x` `stat_quantile` Uses quantile regression to predict quantiles `geom_fan` Plot intervals on a continuous colour scale

### Examples

```
library(ggplot2)
# Basic use. The data frame must have multiple y values for each
# x
ggplot(fake_df, aes(x=x,y=y)) +geom_interval()

# use precomputed quantiles - reducing storage requirements.
intervals = c(0,50,90)/100
fake_q <- calc_quantiles(fake_df, intervals=intervals)
# intervals in geom_fan must be the same as used to compute quantiles.
ggplot(fake_q, aes(x=x,y=y, quantile=quantile)) +
  geom_interval(intervals=intervals)
```

---

ggfan

*Fanplots for ggplot2*

---

### Description

Implements the functionality of the `fanplot` package as `ggplot` geoms. Designed for summarising MCMC samples from a posterior distribution, where a visualisation is desired for several values of a continuous covariate. Increasing posterior intervals derived from the quantiles of the sampled quantity are mapped to a continuous colour scale.

### Functions

The package contains three plotting functions.

`geom_fan` produces fan plots

`geom_interval` produces line plots, where pairs of lines represent intervals

`stat_sample` randomly plots a specified number of samples from the data

An additional function `calc_quantiles` computes relevant quantiles for a specified set of intervals.



---

gp_model_fit	<i>A stan_fit object used in the ggfan_stan vignette, containing posterior samples from a latent gaussian process model. This is provided as data to avoid having to conduct computationally expensive sampling when producing the vignettes.</i>
--------------	---

---

### Description

The code needed to recreate the object is included in the examples, as well as in the vignette code chunks.

### Usage

```
gp_model_fit
```

### Format

A ‘stan\_fit’ object containing samples of the following parameters.

**eta\_sq** Gaussian process variance parameter

**rho\_sq** Gaussian process roughness parameter

**z** Latent poisson rate

**y\_gen** Posterior predictive sample of counts ‘y’

See the help page for [stanfit-class](#) for more details.

### Examples

```
## Not run:
# generate mean and variance for sequence of samples over time
library(rstan)
library(dplyr)
library(magrittr)
library(tidyr)
library(tibble)

library(ggfan)
seed <- 34526
set.seed(seed)

# data
x <- seq(-5,5,0.1)
N <- length(x)
y <- cbind(rpois(N, exp(sin(x)+2)),rpois(N, exp(sin(x)+2)))

stan_data <- list(N=N, x=x, y=y)

compiled_model <- stan_model(file=file.path(path.package("ggfan"),
```

```

                                "stan", "latent_gp_pois.stan"))
gp_model_fit <- sampling(compiled_model, data=stan_data, iter=3000, thin=6)
#devtools::use_data(gp_model_fit, internal=FALSE)

## End(Not run)

```

---

StatInterval	See <a href="#">ggplot2-ggproto</a>
--------------	-------------------------------------

---

### Description

See [ggplot2-ggproto](#)

---

StatIntervalFctr	See <a href="#">ggplot2-ggproto</a>
------------------	-------------------------------------

---

### Description

See [ggplot2-ggproto](#)

---

StatSample	See <a href="#">ggplot2-ggproto</a>
------------	-------------------------------------

---

### Description

See [ggplot2-ggproto](#)

---

stat_interval	<i>Line plot visualising intervals of a distribution</i>
---------------	--

---

### Description

Very similar to [geom\\_interval](#), except uses [geom\\_line](#) to handle the plotting. This makes handling plotting of intervals for several groups difficult to achieve, so [geom\\_interval](#) is preferred.

### Usage

```

stat_interval(mapping = NULL, data = NULL, stat = "interval_fctr",
              position = "identity", na.rm = FALSE, show.legend = NA,
              inherit.aes = TRUE, intervals = c(0, 50, 90)/100, ...)

```

**Arguments**

mapping	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (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 <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data.
stat	Use to override the default use of <code>stat_interval</code>
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , 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. <code>borders()</code> .
intervals	specify the collection of intervals to be represented in the fan.
...	Other arguments passed on to <code>layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired geom/stat.

---

stat_sample	<i>Plots a randomly chosen sample of the specified groups using <code>geom_line</code></i>
-------------	--

---

**Description**

Plots a randomly chosen sample of the specified groups using `geom_line`

**Usage**

```
stat_sample(mapping = NULL, data = NULL, stat = "sample",
            position = "identity", na.rm = FALSE, show.legend = F,
            inherit.aes = TRUE, n_samples = 5, size = 0.2, alpha = 1, ...)
```

**Arguments**

mapping	Set of aesthetic mappings created by <code>aes()</code> or <code>aes_()</code> . If specified and <code>inherit.aes = TRUE</code> (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 <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <code>ggplot()</code> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <code>fortify()</code> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , 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. <code>borders()</code> .
n_samples	number of samples to plot
size	The width of the line in mm
alpha	The transparency of lines to be drawn. Must lie between 0 and 1.
...	Other arguments passed on to <code>layer()</code> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>colour = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired <code>geom/stat</code> .

# Index

## \* datasets

- fake\_df, 3
- GeomIntervalPath, 4
- GeomIntervalPoly, 4
- gp\_model\_fit, 9
- StatInterval, 10
- StatIntervalFctr, 10
- StatSample, 10
- .in\_numeric, 2

aes(), 5, 7, 11, 12

aes\_(), 5, 7, 11, 12

borders(), 5, 7, 11, 12

calc\_quantiles, 3, 8

fake\_df, 3

fortify(), 5, 7, 11, 12

geom\_fan, 5, 8

geom\_interval, 6, 8, 10

geom\_line, 10, 11

GeomIntervalPath, 4

GeomIntervalPoly, 4

ggfan, 8

ggfan-package (ggfan), 8

ggplot(), 5, 7, 11, 12

ggplot2-ggproto, 4, 10

gp\_model\_fit, 9

grid::arrow(), 7

layer(), 7, 11, 12

stat\_interval, 5, 7, 10, 11

stat\_sample, 8, 11

StatInterval, 10

StatIntervalFctr, 10

StatSample, 10