Package 'ggfan'

June 14, 2018

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 0.1.2

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 6.0.1

NeedsCompilation no

Author Jason Hilton [aut, cre]

Maintainer Jason Hilton <jason_hilton@yahoo.com>

Repository CRAN

Date/Publication 2018-06-14 07:49:58 UTC

R topics documented:

`calc_quantiles` *Calculate quantiles of a tidy dataframe*

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 *Fake dataset intended to resemble a set of MCMC samples of a variable over one covariate (perhaps time),*

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

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.
stat	Use to override the default use of stat_interval
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.

show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes.
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.
intervals	specify the collection of intervals to be represented in the fan.
...	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.

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 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.
stat	Use to override the default use of stat_interval
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 arrow
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.

... — 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.

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 *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.*

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* ggplot2-ggproto

Description

See ggplot2-ggproto

`StatIntervalFctr` *See* ggplot2-ggproto

Description

See ggplot2-ggproto

StatSample *See* ggplot2-ggproto

Description

See ggplot2-ggproto

stat_interval *Line plot visualising intervals of a distribution*

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 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.
stat	Use to override the default use of stat_interval
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes.

inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. 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.
intervals	specify the collection of intervals to be represented in the fan.
...	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.

---

stat_sample *Plots a randomly chosen sample of the specified groups using geom_line*

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 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.
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. 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.

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 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.

Index