

Package: fanplot (via r-universe)

October 11, 2026

Type Package

Title Visualisation of Sequential Probability Distributions Using Fan Charts

Version 4.0.1

Maintainer Guy J. Abel <g.j.abel@gmail.com>

Description Visualise sequential distributions using a range of plotting styles. Sequential distribution data can be input as either simulations or values corresponding to percentiles over time. Plots are added to existing graphic devices using the fan function. Users can choose from four different styles, including fan chart type plots, where a set of coloured polygon, with shadings corresponding to the percentile values are layered to represent different uncertainty levels. Full details in R Journal article; Abel (2015) <doi:10.32614/RJ-2015-002>.

License GPL-3

Encoding UTF-8

URL <http://guyabel.github.io/fanplot/>

BugReports <https://github.com/guyabel/fanplot/issues/>

Depends R (>= 3.5)

Imports methods

Suggests shiny, RColorBrewer, colorspace, zoo, forecast

LazyData true

RoxygenNote 7.3.3

Repository <https://guyabel.r-universe.dev>

Date/Publication 2026-01-26 10:08:53 UTC

RemoteUrl <https://github.com/guyabel/fanplot>

RemoteRef HEAD

RemoteSha c51f878479e169d50d4f3391a6f550c9d85c4679

Contents

fanplot-package	2
boe	3
cpi	4
dsplitnorm	5
fan	6
ips	11
svpdx	12
th.mcmc	13
Index	15

fanplot-package	<i>fanplot: Visualisation of Sequential Probability Distributions Using Fan Charts</i>
-----------------	--

Description

Visualise sequential distributions using a range of plotting styles. Sequential distribution data can be input as either simulations or values corresponding to percentiles over time. Plots are added to existing graphic devices using the `fan` function. Users can choose from four different styles, including fan chart type plots, where a set of coloured polygons, with shadings corresponding to the percentile values, are layered to represent different uncertainty levels.

Details

License: GPL-2

Author(s)

Guy J. Abel

References

Abel, G. J. (2015). fanplot: An R Package for visualising sequential distributions. *The R Journal*, 7(2), 15–23.

See Also

Useful links:

- <http://guyabel.github.io/fanplot/>
- Report bugs at <https://github.com/guyabel/fanplot/issues/>

boe	<i>Parameters for MPC CPI Inflation Projections from Q1 2004 to Q4 2013</i>
-----	---

Description

Numerical parameters for inflation report of the Bank of England used to specify the probability distributions for forecast charts of CPI inflation. Data formatted from the November 2013 Bank of England Inflation Report.

Usage

```
data(boe)
```

Format

A data frame with 512 observations on the following 5 variables:

time0 Publication time of parameters

time Future time of projected parameter

mode Central location parameter of split-normal distribution

uncertainty Uncertainty parameter of split-normal distribution

skew Skew parameter of split-normal distribution

Details

mode, uncertainty, and skew parameters relate to those given in [dsplitnorm](#), where uncertainty is the standard deviation.

Source

Bank of England Inflation Report November 2013. Retrieved from "Parameters for MPC CPI Inflation Projections from February 2004" spreadsheet. Original spreadsheet no longer available on the Bank of England website, but a copy exists at <https://github.com/guyabel/fanplot/tree/master/data-raw/>

Examples

```
## Q1 2013
y0 <- 2013
boe0 <- subset(boe, time0 == y0)
k <- nrow(boe0)

# guess work to set percentiles the boe are plotting
p <- seq(0.05, 0.95, 0.05)
p <- c(0.01, p, 0.99)

# estimate percentiles for future time period
```

```

pp <- matrix(NA, nrow = length(p), ncol = k)
for (i in 1:k)
  pp[, i] <- qsplitnorm(p, mode = boe0$mode[i],
                        sd = boe0$uncertainty[i],
                        skew = boe0$skew[i])

pp

## Q4 2013 (coarser fan)
y0 <- 2013.75
boe0 <- subset(boe, time0 == y0)
k <- nrow(boe0)

p <- seq(0.2, 0.8, 0.2)
p <- c(0.05, p, 0.95)
pp <- matrix(NA, nrow = length(p), ncol = k)
for (i in 1:k)
  pp[, i] <- qsplitnorm(p, mode = boe0$mode[i],
                        sd = boe0$uncertainty[i],
                        skew = boe0$skew[i])

pp

```

cpi	<i>Percentage Change (over 12 months) of United Kingdom Consumer Price Inflation</i>
-----	--

Description

Time series of quarterly UK CPI from Q1 1997 to Q3 2013. Data formatted from the October 2013 release of the CPI data by the Office of National Statistics. Q1 values are taken from February, Q2 from May, Q3 from August, and Q4 from November.

Usage

```
data(cpi)
```

Format

Time-Series [1:67] from 1997 to 2014: 88.8, 89.6, 90, 90.4, 90.3, 91.5, 91.2, 91.7, 91.5, 92.7, ...

Source

October 2013 CPI data by the Office of National Statistics. Retrieved from "Consumer Price Inflation Reference Tables, October 2013".

Original spreadsheet no longer available on the ONS website, but a copy exists at <https://github.com/guyabel/fanplot/tree/master/data-raw/>

Examples

```
data(cpi)
```

dsplitnorm

The Split Normal Distribution (or Two-Piece Normal Distribution)

Description

Density, distribution function, quantile function and random generation for the split normal distribution.

Usage

```
dsplitnorm(x, mode = 0, sd = 1, skew = 0, sd1 = NULL, sd2 = NULL)
```

```
psplitnorm(x, mode = 0, sd = 1, skew = 0, sd1 = NULL, sd2 = NULL)
```

```
qsplitnorm(p, mode = 0, sd = 1, skew = 0, sd1 = NULL, sd2 = NULL)
```

```
rsplitnorm(n, mode = 0, sd = 1, skew = 0, sd1 = NULL, sd2 = NULL)
```

Arguments

x	Vector of quantiles (for dsplitnorm, psplitnorm).
mode	Vector of modes.
sd	Vector of uncertainty indicators.
skew	Vector of inverse skewness indicators. Must range between -1 and 1.
sd1	Vector of standard deviations for left-hand side. NULL by default.
sd2	Vector of standard deviations for right-hand side. NULL by default.
p	Vector of probabilities (for qsplitnorm).
n	Number of observations required (for rsplitnorm).

Details

If mode, sd or skew are not specified they assume the default values of 0, 1 and 0, respectively. This results in identical values as those obtained from a normal distribution.

The probability density function is:

$$f(x; \mu, \sigma_1, \sigma_2) = \frac{\sqrt{2}}{\sqrt{\pi}(\sigma_1 + \sigma_2)} e^{-\frac{1}{2\sigma_1^2}(x-\mu)^2}$$

for $-\infty < x < \mu$, and

$$f(x; \mu, \sigma_1, \sigma_2) = \frac{\sqrt{2}}{\sqrt{\pi}(\sigma_1 + \sigma_2)} e^{-\frac{1}{2\sigma_2^2}(x-\mu)^2}$$

for $\mu < x < \infty$.

If not specified (via `sd1` and `sd2`), σ_1 and σ_2 are derived as:

$$\sigma_1 = \sigma / \sqrt{1 - \gamma}$$

$$\sigma_2 = \sigma / \sqrt{1 + \gamma}$$

where σ is the overall uncertainty indicator `sd` and γ is the inverse skewness indicator `skew`.

Value

`dsplitnorm` gives the density, `psplitnorm` gives the distribution function, `qsplitnorm` gives the quantile function, and `rsplitnorm` generates random deviates.

Examples

```
x <- seq(-5, 5, length = 110)
plot(x, dsplitnorm(x), type = "l")

# compare to normal density
lines(x, dnorm(x), lty = 2, col = "red", lwd = 5)

# add positive skew
lines(x, dsplitnorm(x, mode = 0, sd = 1, skew = 0.8))

# add negative skew
lines(x, dsplitnorm(x, mode = 0, sd = 1, skew = -0.5))

# add left and right hand sd
lines(x, dsplitnorm(x, mode = 0, sd1 = 1, sd2 = 2), col = "blue")

# psplitnorm
x <- seq(-5, 5, length = 100)
plot(x, pnorm(x), type = "l")
lines(x, psplitnorm(x, skew = -0.9), col = "red")

# qsplitnorm
x <- seq(0, 1, length = 100)
plot(qnorm(x), type = "l", x)
lines(qsplitnorm(x), x, lty = 2, col = "blue")
lines(qsplitnorm(x, skew = -0.3), x, col = "red")

# rsplitnorm
hist(rsplitnorm(n = 10000, mode = 1, sd = 1, skew = 0.9), 100)
```

Description

Visualise sequential distributions using a range of plotting styles.

Usage

```

fan(
  data = NULL,
  data.type = "simulations",
  style = "fan",
  type = "percentile",
  probs = if (type == "percentile") seq(0.01, 0.99, 0.01) else c(0.5, 0.8, 0.95),
  start = 1,
  frequency = 1,
  anchor = NULL,
  anchor.time = NULL,
  fan.col = grDevices::heat.colors,
  alpha = if (style == "spaghetti") 0.5 else 1,
  n.fan = NULL,
  ln = if (length(probs) < 10) probs else probs[round(probs, 2) %in% round(seq(0.1,
    0.9, 0.1), 2)],
  ln.col = if (style == "spaghetti") "gray" else NULL,
  med.ln = if (type == "interval") TRUE else FALSE,
  med.col = "orange",
  rlab = ln,
  rpos = 4,
  roffset = 0.1,
  rcex = 0.8,
  rcol = NULL,
  llab = FALSE,
  lpos = 2,
  loffset = roffset,
  lcex = rcex,
  lcol = rcol,
  upplab = "U",
  lowlab = "L",
  medlab = if (type == "interval") "M" else NULL,
  n.spag = 30,
  space = if (style == "boxplot") 1/frequency else 0.9/frequency,
  add = FALSE,
  ylim = range(data) * 0.8,
  ...
)

```

Arguments

data	Set of sequential simulation data, where rows represent simulation number and columns represent some form of time index. If <code>data.type = "values"</code> , data must instead be a set of quantile values by rows for a set of probabilities (which need to be provided in <code>probs</code>) and by column for some form of time index. Data can take multiple classes, where the contents are converted to a matrix. If the input is a <code>mts</code> or <code>zoo</code> , the time series properties will be inherited (and <code>start</code> and <code>frequency</code> arguments will be ignored).
------	---

<code>data.type</code>	Indicates if data are sets of pre-calculated values for defined probabilities ("values") or simulated data ("simulations"). Default is "simulations".
<code>style</code>	Plot style, choose from "fan" (default), "spaghetti", "boxplot" or "boxfan".
<code>type</code>	Type of percentiles to plot in fan or boxfan. Choose from "percentile" (default) or "interval".
<code>probs</code>	Probabilities related to percentiles or prediction intervals to be plotted (dependent on the <code>type</code> argument). Must be between 0 and 100 (inclusive) or 0 and 1. Percentiles greater than 50 (or 0.5), if not given, are automatically calculated as $100 - p$, to ensure symmetric fan. Defaults to single percentile values when <code>type = "percentile"</code> and the 50th, 80th and 95th prediction interval when <code>type = "interval"</code> .
<code>start</code>	The time of the first distribution in sims. Similar to use in ts .
<code>frequency</code>	The number of distributions in sims per unit of time. Similar to use in ts .
<code>anchor</code>	Optional data value to anchor a forecast fan on. Typically the last observation of the observed series.
<code>anchor.time</code>	Optional time value for the anchor. Useful for irregular time series.
<code>fan.col</code>	Palette of colours used in the fan or boxfan.
<code>alpha</code>	Factor modifying the opacity alpha; typically in [0,1].
<code>n.fan</code>	Number of colours to use in the fan.
<code>ln</code>	Vector of numbers to plot contour lines on top of fan or boxfan. Must correspond to calculated percentiles in <code>probs</code> .
<code>ln.col</code>	Line colour imposed on top of the fan. Defaults to darkest colour from <code>fan.col</code> , unless <code>style = "spaghetti"</code> .
<code>med.ln</code>	Logical; add a median line to fan. Useful if <code>type = "interval"</code> .
<code>med.col</code>	Median line colour. Defaults to first colour in <code>fan.col</code> .
<code>rlab</code>	Vector of labels at the end (right) of corresponding percentiles or prediction intervals.
<code>rpos</code>	Position of right labels. See text .
<code>roffset</code>	Offset of right labels. See text .
<code>rcex</code>	Text size of right labels. See text .
<code>rcol</code>	Colour of text for right labels. See text .
<code>llab</code>	Either logical (TRUE/FALSE) to plot labels at the start (left) of percentiles, or a vector of percentiles. Only works for fan or boxfan.
<code>lpos</code>	Position of left labels. See text .
<code>loffset</code>	Offset of left labels. Defaults to <code>roffset</code> .
<code>lcex</code>	Text size of left labels. Defaults to <code>rcex</code> .
<code>lcol</code>	Colour of text for left labels. Defaults to <code>rcol</code> .
<code>upplab</code>	Prefix string for upper labels when <code>type = "interval"</code> .
<code>lowlab</code>	Prefix string for lower labels when <code>type = "interval"</code> .
<code>medlab</code>	Character string for median label.

n.spag	Number of simulations to plot in the spaghetti style.
space	Space between boxes in the boxfan plot.
add	Logical; add to active plot. Defaults to FALSE for fan, TRUE for fan0.
ylim	Passed to plot when add = TRUE.
...	Additional arguments passed to boxplot for fan and to plot for fan0.

Details

Sequential distribution data can be input as either simulations or pre-computed values over time (columns). For the latter, declare input data as percentiles by setting `data.type = "values"`. Users can choose from four styles:

- fan, boxfan: shaded distributions with optional contour lines and labels.
- spaghetti: random draws plotted along the sequence of distributions.
- boxplot: box plots for simulated data at appropriate locations.

Value

See details.

Author(s)

Guy J. Abel

References

Abel, G. J. (2015). fanplot: An R Package for visualising sequential distributions. *The R Journal*, 7(2), 15–23.

Examples

```
## Basic Fan: fan0()

##
## 20 or so examples of fan charts and
## spaghetti plots based on the th.mcmc object
##
## Make sure you have zoo, tsbugs, RColorBrewer and
## colorspace packages installed
##
## Not run:
demo("sv_fan", "fanplot")

## End(Not run)

##
## Fans for forecasted values
##
## Not run:
#create time series
```

```

net <- ts(ips$net, start=1975)

# fit model
library("forecast")
m <- auto.arima(net)

# plot in forecast package (limited customisation possible)
plot(forecast(m, h=5))

# another plot in forecast (with some customisation, no
# labels or anchoring possible at the moment)
plot(forecast(m, h=5, level=c(50,80,95)),
      shadecols=rev(heat.colors(3)))

# simulate future values
mm <- matrix(NA, nrow=1000, ncol=5)
for(i in 1:1000)
  mm[i,] <- simulate(m, nsim=5)

# interval fan chart
plot(net, xlim=c(1975,2020), ylim=c(-100,300))
fan(mm, type="interval", start=2013)

# anchor fan chart
plot(net, xlim=c(1975,2020), ylim=c(-100,300))
fan(mm, type="interval", start=2013,
     anchor=net[time(net)==2012])

# anchor spaghetti plot with underlying fan chart
plot(net, xlim=c(1975,2020), ylim=c(-100,300))
fan(mm, type="interval", start=2013,
     anchor=net[time(net)==2012], alpha=0, ln.col="orange")
fan(mm, type="interval", start=2013,
     anchor=net[time(net)==2012], alpha=0.5, style="spaghetti")

## End(Not run)

##
## Box Plots
##
# sample every 21st day of theta_t
th.mcmc21 <- th.mcmc[, seq(1, 945, 21)]
plot(NULL, xlim = c(1, 945), ylim = range(th.mcmc21))
fan(th.mcmc21, style = "boxplot", frequency = 1/21)

# additional arguments for boxplot
plot(NULL, xlim = c(1, 945), ylim = range(th.mcmc21))
fan(th.mcmc21, style = "boxplot", frequency = 1/21,
     outline = FALSE, col = "red", notch = TRUE)

##
## Fan Boxes
##

```

```

plot(NULL, xlim = c(1, 945), ylim = range(th.mcmc21))
fan(th.mcmc21, style = "boxfan", type = "interval", frequency = 1/21)

# more space between boxes
plot(NULL, xlim = c(1, 945), ylim = range(th.mcmc21))
fan(th.mcmc21, style = "boxfan", type = "interval",
    frequency = 1/21, space = 10)

# overlay spaghetti
fan(th.mcmc21, style = "spaghetti",
    frequency = 1/21, n.spag = 50, ln.col = "red", alpha=0.2)

```

ips

ONS International Passenger Survey Long-Term International Migration 1975–2012

Description

Immigration, emigration and net migration flow counts (and their confidence intervals) for the UK from the International Passenger Survey (IPS) conducted by the Office of National Statistics. Data formatted from the 2012 release of the Long-Term International Migration Statistics.

Usage

```
data(ips)
```

Format

A data frame with 38 observations on the following 7 variables:

year Numeric vector; year of observation.

imm Numeric vector; immigration flow counts.

imm.ci Numeric vector; confidence intervals for immigration.

emi Numeric vector; emigration flow counts.

emi.ci Numeric vector; confidence intervals for emigration.

net Numeric vector; net migration flow counts.

net.ci Numeric vector; confidence intervals for net migration.

Details

Data differ slightly from the final adjusted migration estimates published by the ONS, which take account of certain types of migration that the IPS does not capture, such as asylum seekers, people migrating for longer or shorter than they thought they would, and migration over land to and from Northern Ireland.

Source

Annual statistics on flows of international migrants to and from the UK and England and Wales by the Office of National Statistics. Retrieved from "1.02 IPS Margins of Error, 1975–2012" spreadsheet.

Original spreadsheet no longer available on the ONS website, but a copy exists at <https://github.com/guyabel/fanplot/tree/master/data-raw/>

Examples

```
# Standard plot
net <- ts(ips$net, start = 1975)
plot(net, ylim = range(net - ips$net.ci, net + ips$net.ci))
lines(net + ips$net.ci, lty = 2, col = "red")
lines(net - ips$net.ci, lty = 2, col = "red")

# Simulate values
ips.sim <- matrix(NA, nrow = 10000, ncol = length(net))
for (i in 1:length(net)) {
  ips.sim[, i] <- rnorm(10000, mean = ips$net[i], sd = ips$net.ci[i] / 1.96)
}

# Spaghetti plot
plot(net, ylim = range(net - ips$net.ci, net + ips$net.ci), type = "n")
fan(ips.sim, style = "spaghetti", start = tsp(net)[1], n.spag = 50)

# Box plot
plot(net, ylim = range(net - ips$net.ci, net + ips$net.ci), type = "n")
fan(ips.sim, style = "boxplot", start = tsp(net)[1], llab = TRUE, outline = FALSE)

# Box fan
plot(net, ylim = range(net - ips$net.ci, net + ips$net.ci), type = "n")
fan(ips.sim, style = "boxfan", type = "interval", start = tsp(net)[1])
```

svpdx

Pound–Dollar Exchange Rate Data

Description

Pound–Dollar exchange rate data from 2 October 1981 to 28 June 1985.

Usage

```
data(svpdx)
```

Format

A data frame with 945 observations on the following 2 variables:

date Date of observation.

pdx Logarithm of returns for Pound–Dollar exchange.

Details

Raw data on log returns.

Source

<http://www.econ.vu.nl/koopman/sv/svpdx.dat>

References

Meyer, R. and J. Yu (2002). BUGS for a Bayesian analysis of stochastic volatility models. *Econometrics Journal*, 3(2), 198–215.

Examples

```
data(svpdx)

# plot
plot(svpdx$pdx, type = "l", xaxt = "n",
     xlab = "Time", ylab = "Return")

# add x-axis
svpdx$rdate <- format(svpdx$date, format = "%b %Y")
mth <- unique(svpdx$rdate)
qtr <- mth[seq(1, length(mth), 3)]
axis(1, at = match(qtr, svpdx$rdate), labels = qtr, cex.axis = 0.75)
axis(1, at = match(mth, svpdx$rdate), labels = FALSE, tcl = -0.2)
```

th.mcmc

*1000 MCMC Simulations of Estimated Volatility from Pound–Dollar
Exchange Rate Data*

Description

MCMC simulations of volatility obtained from the bugs function in the R2OpenBUGS package. Estimates are based on the stochastic volatility model for Pound–Dollar exchange rate data presented in the appendix of Meyer and Yu (2002). The MCMC was run for 1100 simulations, thinning to keep every 10th iteration, and treating the first 100 simulations as burn-in.

Usage

```
data(th.mcmc)
```

Format

An object of class `matrix` (inherits from `array`) with 1000 rows and 945 columns.

Details

Larger simulations (without thinning) can be obtained using the data (`svpdx`) and the `my1.txt` BUGS file contained in this package; see the example below.

See Meyer and Yu (2010) for model specification.

References

Meyer, R. and J. Yu (2002). BUGS for a Bayesian analysis of stochastic volatility models. *Econometrics Journal*, 3(2), 198–215.

Sturtz, S., U. Ligges, and A. Gelman (2005). R2WinBUGS: a package for running WinBUGS from R. *Journal of Statistical Software*, 12(3), 1–16.

Examples

```
# empty plot
plot(NULL, type = "n", xlim = c(1, 945),
      ylim = range(th.mcmc), ylab = "Theta")

# add fan
fan(th.mcmc)

## Create your own (longer) MCMC sample:
## Not run:
# library(tsbugs)
# library(R2OpenBUGS)
#
# # write model file:
# my1.bug <- dget(system.file("model", "my1.R", package = "fanplot"))
# write.model(my1.bug, "my1.txt")
#
# # take a look:
# file.show("my1.txt")
#
# # run OpenBUGS (include theta as a parameter or nothing will be plotted)
# my1.mcmc <- bugs(
#   data = list(n = length(svpdx$spdx), y = svpdx$spdx),
#   inits = list(list(phistar = 0.975, mu = 0, itau2 = 50)),
#   param = c("mu", "phi", "tau", "theta"),
#   model = "my1.txt",
#   n.iter = 11000, n.burnin = 1000, n.chains = 1
# )
#
# th.mcmc <- my1.mcmc$sims.list$theta

## End(Not run)
```

Index

* datasets

boe, [3](#)

cpi, [4](#)

ips, [11](#)

svpdx, [12](#)

th.mcmc, [13](#)

boe, [3](#)

boxplot, [9](#)

cpi, [4](#)

dsplitnorm, [3](#), [5](#)

fan, [2](#), [6](#)

fan0 (fan), [6](#)

fanplot (fanplot-package), [2](#)

fanplot-package, [2](#)

ips, [11](#)

plot, [9](#)

psplitnorm (dsplitnorm), [5](#)

qsplitnorm (dsplitnorm), [5](#)

rsplitnorm (dsplitnorm), [5](#)

svpdx, [12](#)

text, [8](#)

th.mcmc, [13](#)

ts, [8](#)