

Package ‘beanplot’

January 21, 2012

Type Package

Title Visualization via Beanplots (like Boxplot/Stripchart/Violin Plot)

Version 1.1

Date 2008-11-06

Author Peter Kampstra

Maintainer Peter Kampstra <pkampst=beanplot@cs.vu.nl>

Description Plots univariate comparison graphs, an alternative to boxplot/stripchart/violin plot.

Suggests vioplot, lattice

License GPL-2

Repository CRAN

Date/Publication 2008-11-06 22:33:45

R topics documented:

beanplot-package	1
beanplot	2
Index	7

beanplot-package	<i>Visualization via Beanplots (like Boxplot/Stripchart/Violin Plot)</i>
------------------	--

Description

Plots univariate comparison graphs, alternative to boxplot/stripchart/violin plot

Details

Package: beanplot
 Type: Package
 Version: 1.1
 Date: 2008-11-06
 License: GPL-2

The function `beanplot` does all the work.

Author(s)

Peter Kampstra <pkampst=beanplot@cs.vu.nl>

References

Kampstra, P. (2008) Beanplot: A Boxplot Alternative for Visual Comparison of Distributions. *Journal of Statistical Software, Code Snippets*, **28**(1), 1-9. URL <http://www.jstatsoft.org/v28/c01/>

See Also

`graphics` `vioplot`

Examples

```
beanplot(rnorm(100), runif(100))
vignette("beanplot", package = "beanplot")
```

beanplot

Beanplot: A Boxplot/Stripchart/Vioplot Alternative

Description

Plots beans to compare the distributions of different groups; it draws one bean per group of data. A bean consists of a one-dimensional scatter plot, its distribution as a density shape and an average line for the distribution. Next to that, an overall average for the whole plot is drawn per default.

Usage

```
beanplot(..., bw = "SJ-dpi", kernel = "gaussian", cut = 3, cutmin = -Inf,
  cutmax = Inf, grownage = 10, what = c(TRUE, TRUE, TRUE, TRUE),
  add = FALSE, col, axes = TRUE, log = "auto", handlelog = NA,
  ll = 0.16, wd = NA, maxwidth = 0.8, maxstripline = 0.96,
  method = "stack", names, overallline = "mean", beanlines = overallline,
  horizontal = FALSE, side = "no", jitter = NULL, beanlinewd = 2,
  frame.plot = axes, border = NULL, innerborder = NA, at = NULL,
  boxwex = 1, ylim = NULL, xlim = NULL, show.names = NA)
```

Arguments

...	data which to perform the beanplot on. This data can consist of dataframes, vectors and/or formulas. For each formula, a dataset can be specified with <code>data=[dataset]</code> , and a subset can be specified with <code>subset=[subset]</code> . If <code>subset/data</code> arguments are passed, but there are not enough <code>subset/data</code> arguments, they are reused. Additionally, <code>na.action</code> , <code>drop.unused.levels</code> and <code>xlev</code> can be passed to <code>model.frame</code> in the same way. Also, parameters for <code>axis</code> and <code>title</code> can be passed.
<code>bw</code>	the bandwidth (method) being used, used by <code>density</code> . In case of a method, the average computed bandwidth is used.
<code>kernel</code>	see <code>density</code> .
<code>cut</code>	the beans are cut beyond <code>cut*bw</code>
<code>cutmin</code>	the low-ends of the beans are cut below <code>mincut*bw</code> . Defaults to <code>cut</code> .
<code>cutmax</code>	the high-ends of the beans are cut beyond <code>maxcut*bw</code> . Defaults to <code>cut</code> .
<code>grownage</code>	the width of a bean grows linearly with the count of points, until <code>grownage</code> is reached.
<code>what</code>	a vector of four booleans describing what to plot. In the following order, these booleans stand for the total average line, the beans, the bean average, and the beanlines. For example, <code>what=c(0, 0, 0, 1)</code> produces a <code>stripchart</code>
<code>add</code>	if true, do not start a new plot
<code>col</code>	the colors to be used. A vector of up to four colors can be used. In the following order, these colors stand for the area of the beans (without the border, use <code>border</code> for that color), the lines inside the bean, the lines outside the bean, and the average line per bean. Transparent colors are supported. <code>col</code> can be a list of color vectors, the vectors are then used per bean, and reused if necessary.
<code>axes</code>	if false, no axes are drawn.
<code>log</code>	use <code>log="y"</code> or <code>log=""</code> to force a log-axis. In case of <code>log="auto"</code> , a log-transformation is used if appropriate
<code>handlelog</code>	if <code>handlelog</code> then all the calculations are done using a log-scale. By default this is determined using the <code>log</code> parameter.
<code>ll</code>	the length of the beanline per point found.
<code>wd</code>	the linear transformation that determines the width of the beans. By default determined using <code>maxwidth</code> , and returned.
<code>maxwidth</code>	the maximum width of a bean.
<code>maxstripline</code>	the maximum length of a beanline.
<code>method</code>	the method used when two points on a bean are the same. <code>"stack"</code> , <code>"overplot"</code> and <code>"jitter"</code> are supported.
<code>names</code>	a vector of names for the groups.
<code>overallline</code>	the method used for determining the overall line. Defaults to <code>"mean"</code> , <code>"median"</code> is also supported.
<code>beanlines</code>	the method used for determining the average bean line(s). Defaults to <code>"mean"</code> , <code>"median"</code> and <code>"quantiles"</code> are also supported.

horizontal	if true, the beanplot is horizontal
side	the side on which the beans are plot. Default is "no", for symmetric beans. "first", "second" and "both" are also supported.
jitter	passed to <code>jitter</code> as amount in case of <code>method="jitter"</code> .
beanlinewidth	the width used for the average bean line
frame.plot	if true, plots a frame.
border	the color for the border around a bean. NULL for <code>par("fg")</code> , NA for no border. If border is a vector, it specifies the color per bean, and colors are reused if necessary.
innerborder	a color (vector) for the border inside the bean(s). Especially useful if <code>side="both"</code> . Use NA for no inner border. Colors are reused if necessary. The inner border is drawn as the last step in the drawing process.
at	the positions at which a bean should be drawn.
boxwex	a scale factor applied to all beans. Compatible with <code>boxplot</code> .
ylim	the range to plot.
xlim	the range to plot the beans at.
show.names	if true, plots the names as axis labels

Details

Most parameters are compatible with `boxplot` and `stripchart`. For compatibility, arguments with the name "formula" or "x" are used as data. However, data or formulas do not need to be named "x" or "formula". The function handles (combinations of) dataframes, vectors and/or formulas.

Value

bw	The bandwidth (bw) used.
wd	The bean width (wd) used.

Note

In case of more than 5000 values per bean, the autodetection of log fails. In such cases, use `log=""` or `log="y"`.

Author(s)

Peter Kampstra <pkampst=beanplot@cs.vu.nl>

References

Kampstra, P. (2008) Beanplot: A Boxplot Alternative for Visual Comparison of Distributions. *Journal of Statistical Software, Code Snippets*, **28**(1), 1-9. URL <http://www.jstatsoft.org/v28/c01/>

See Also

`boxplot`, `stripchart`, `density`, `rug`, `vioplot` in package `vioplot`

Examples

```

beanplot(rnorm(22),rnorm(22),rnorm(22),main="Test!",rnorm(3))

#mostly examples taken from boxplot:
par(mfrow = c(1,2))
boxplot(count ~ spray, data = InsectSprays, col = "lightgray")
beanplot(count ~ spray, data = InsectSprays, col = "lightgray", border = "grey", cutmin = 0)

boxplot(count ~ spray, data = InsectSprays, col = "lightgray")
beanplot(count ~ spray, data = InsectSprays, col = "lightgray", border = "grey", overallline = "median")

boxplot(decrease ~ treatment, data = OrchardSprays,
        log = "y", col = "bisque", ylim = c(1,200))
beanplot(decrease ~ treatment, data = OrchardSprays,
        col = "bisque", ylim = c(1,200))

par(mfrow = c(2,1))
mat <- cbind(Uni05 = (1:100)/21, Norm = rnorm(100),
            T5 = rt(100, df = 5), Gam2 = rgamma(100, shape = 2))
par(las=1)# all axis labels horizontal
boxplot(data.frame(mat), main = "boxplot(*, horizontal = TRUE)",
        horizontal = TRUE, ylim = c(-5,8))
beanplot(data.frame(mat), main = "beanplot(*, horizontal = TRUE)",
        horizontal = TRUE, ylim = c(-5,8))

par(mfrow = c(1,2))
boxplot(len ~ dose, data = ToothGrowth,
        boxwex = 0.25, at = 1:3 - 0.2,
        subset = supp == "VC", col = "yellow",
        main = "Guinea Pigs' Tooth Growth",
        xlab = "Vitamin C dose mg",
        ylab = "tooth length", ylim = c(-1, 40), yaxs = "i")
boxplot(len ~ dose, data = ToothGrowth, add = TRUE,
        boxwex = 0.25, at = 1:3 + 0.2,
        subset = supp == "OJ", col = "orange")
legend("bottomright", bty="n", c("Ascorbic acid", "Orange juice"),
        fill = c("yellow", "orange"))
allplot <- beanplot(len ~ dose+supp, data = ToothGrowth,
        what=c(TRUE,FALSE,FALSE,FALSE),show.names=FALSE,ylim=c(-1,40), yaxs = "i")
beanplot(len ~ dose, data = ToothGrowth, add=TRUE,
        boxwex = 0.6, at = 1:3*2 - 0.9,
        subset = supp == "VC", col = "yellow",border="yellow2",
        main = "Guinea Pigs' Tooth Growth",
        xlab = "Vitamin C dose mg",
        ylab = "tooth length", ylim = c(3, 40), yaxs = "i",
        bw = allplot$bw, wd = allplot$wd, what = c(FALSE,TRUE,TRUE,TRUE))
beanplot(len ~ dose, data = ToothGrowth, add = TRUE,
        boxwex = 0.6, at = 1:3*2-0.1,
        subset = supp == "OJ", col = "orange",border="darkorange",
        bw = allplot$bw, wd = allplot$wd, what = c(FALSE,TRUE,TRUE,TRUE))
legend("bottomright", bty="n", c("Ascorbic acid", "Orange juice"),
        fill = c("yellow", "orange"))

```

```

par(mfrow = c(1,2))
boxplot(len ~ dose, data = ToothGrowth,
        boxwex = 0.25, at = 1:3 - 0.2,
        subset = supp == "VC", col = "yellow",
        main = "Guinea Pigs' Tooth Growth",
        xlab = "Vitamin C dose mg",
        ylab = "tooth length", ylim = c(-1, 40), yaxs = "i")
boxplot(len ~ dose, data = ToothGrowth, add = TRUE,
        boxwex = 0.25, at = 1:3 + 0.2,
        subset = supp == "OJ", col = "orange")
legend("bottomright", bty="n",c("Ascorbic acid", "Orange juice"),
      fill = c("yellow", "orange"))
beanplot(len ~ reorder(supp, len, mean) * dose, ToothGrowth,
        side = "b", col = list("yellow", "orange"), border = c("yellow2",
        "darkorange"), main = "Guinea Pigs' Tooth Growth",
        xlab = "Vitamin C dose mg", ylab = "tooth length", ylim = c(-1,
        40), yaxs = "i")
legend("bottomright", bty="n",c("Ascorbic acid", "Orange juice"),
      fill = c("yellow", "orange"))

#Example with multiple vectors and/or formulas
par(mfrow = c(2,1))
beanplot(list(all = ToothGrowth$len), len ~ supp, ToothGrowth, len ~ dose)
title("Tooth growth length (beanplot)")
#Trick using internal functions to do this with other functions:
mboxplot <- function(...){
  graphics::boxplot(beanplot:::getgroupsfromarguments(), ...)
}
mstripchart <- function(..., method = "overplot", jitter = 0.1, offset = 1/3,
  vertical = TRUE, group.names, add = FALSE,
  at = NULL, xlim = NULL, ylim = NULL,
  ylab = NULL, xlab=NULL, dlab = "", glab = "",
  log = "", pch = 0, col = par("fg"), cex = par("cex"),
  axes = TRUE, frame.plot = axes) {
  graphics::stripchart(beanplot:::getgroupsfromarguments(),
    method, jitter, offset, vertical, group.names, add,
    at, xlim, ylim, ylab, xlab, dlab, glab, log, pch, col, cex,
    axes, frame.plot)
}
mstripchart(list(all = ToothGrowth$len), len ~ supp, ToothGrowth, len ~ dose,
  xlim = c(0.5,6.5))
title("Tooth growth length (stripchart)")

```

Index

*Topic **hplot**

beanplot, 2

*Topic **package**

beanplot-package, 1

axis, 3

beanplot, 2, 2

beanplot-package, 1

boxplot, 4

density, 3, 4

graphics, 2

jitter, 4

model.frame, 3

rug, 4

stripchart, 3, 4

title, 3

vioplot, 2, 4