Package ‘vioplot’

November 29, 2019

Title    Violin Plot
Version  0.3.4
Date     2019-11-30
Description A violin plot is a combination of a box plot and a kernel density plot. This package allows extensive customisation of violin plots.
Depends  sm, zoo
License  BSD_3_clause + file LICENSE
URL      https://github.com/TomKellyGenetics/vioplot
BugReports https://github.com/TomKellyGenetics/vioplot/issues
LazyData  true
RoxygenNote 7.0.1
Suggests ggplot2, RColorBrewer, knitr, rmarkdown, testthat
Language en-GB
VignetteBuilder knitr
Encoding UTF-8
NeedsCompilation no
Author  Daniel Adler [aut, cph],
        S. Thomas Kelly [aut, cre],
        Tom M. Elliott [ctb]
Maintainer S. Thomas Kelly <tom.kelly@riken.jp>
Repository CRAN
Date/Publication 2019-11-29 18:00:02 UTC

R topics documented:

vioplot ...................................................... 2

Index  9
vioplot  

violin plot

Description

Produce violin plot(s) of the given (grouped) values with enhanced annotation and colour per group. Includes customisation of colours for each aspect of the violin, boxplot, and separate violins. This supports input of data as a list or formula, being backwards compatible with vioplot (0.2) and taking input in a formula as used for boxplot.

Usage

```r
## S3 method for class 'formula'
vioplot(
  formula,
  data = NULL,
  ..., 
  subset, 
  na.action = NULL,
  add = FALSE, 
  ann = !add,
  horizontal = FALSE, 
  xlab = mklab(y_var = horizontal), 
  ylab = mklab(y_var = !horizontal), 
  names = NULL, 
  drop = FALSE, 
  sep = ".", 
  lex.order = FALSE
)

## Default S3 method:
vioplot(
  x,
  ..., 
  data = NULL, 
  range = 1.5, 
  h = NULL, 
  ylim = NULL, 
  names = NULL, 
  horizontal = FALSE, 
  col = "grey50", 
  border = par()$fg, 
  lty = 1, 
  lwd = 1, 
  rectCol = par()$fg, 
  lineCol = par()$fg, 
  pchMed = 19,
```
colMed = "white",
colMed2 = "grey 75",
at,
add = FALSE,
wex = 1,
drawRect = TRUE,
areaEqual = FALSE,
axes = TRUE,
frame.plot = axes,
panel.first = NULL,
panel.last = NULL,
asp = NA,
main = "",
sub = "",
xlab = NA,
ylab = NA,
line = NA,
outer = FALSE,
xlog = NA,
ylog = NA,
adj = NA,
ann = NA,
ask = NA,
bg = NA,
bty = NA,
cex = NA,
cex.axis = NA,
cex.lab = NA,
cex.main = NA,
cex.names = NULL,
cex.sub = NA,
cin = NA,
col.axis = NA,
col.lab = NA,
col.main = NA,
col.sub = NA,
cra = NA,
crt = NA,
csi = NA,
cxy = NA,
din = NA,
err = NA,
family = NA,
fg = NA,
fig = NA,
fin = NA,
font = NA,
font.axis = NA,
vioplot

font.lab = NA,
font.main = NA,
font.sub = NA,
lab = NA,
las = NA,
lend = NA,
lheight = NA,
ljoin = NA,
lmitre = NA,
mai = NA,
mar = NA,
mex = NA,
mfcol = NA,
mfg = NA,
mfrow = NA,
mgp = NA,
mkh = NA,
new = NA,
oma = NA,
omd = NA,
omi = NA,
page = NA,
pch = NA,
pin = NA,
plt = NA,
ps = NA,
pty = NA,
smo = NA,
srt = NA,
tck = NA,
tcl = NA,
usr = NA,
xaxp = NA,
xaxs = NA,
xaxt = NA,
xpd = NA,
yaxp = NA,
yaxs = NA,
yaxt = NA,
ylbias = NA,
log = "",
logLab = c(1, 2, 5),
na.action = NULL,
na.rm = T,
side = "both",
plotCentre = "point"
)}
Arguments

x for specifying data from which the boxplots are to be produced. Either a numeric vector, or a single list containing such vectors. Additional unnamed arguments specify further data as separate vectors (each corresponding to a component boxplot). NAs are allowed in the data.

... additional data vectors or formula parameters. For the formula method, named arguments to be passed to the default method.

formula a formula, such as y ~ grp, where y is a numeric vector of data values to be split into groups according to the grouping variable grp (usually a factor).

data a data.frame (or list) from which the variables in formula should be taken.

subset an optional vector specifying a subset of observations to be used for plotting.

na.action a function which indicates what should happen when the data contain NAs. The default is to ignore missing values in either the response or the group.

add logical. if FALSE (default) a new plot is created

horizontal logical. horizontal or vertical violins

names one label, or a vector of labels for the data must match the number of data given
drop, sep, lex.order defines groups to plot from formula, passed to split.default, see there.

range a factor to calculate the upper/lower adjacent values

h the height for the density estimator, if omit as explained in sm.density, h will be set to an optimum

ylim y limits

col Graphical parameter for fill colour of the violin(s) polygon. NA for no fill colour. If col is a vector, it specifies the colour per violin, and colours are reused if necessary.

border Graphical parameters for the colour of the violin border passed to lines. NA for no border. If border is a vector, it specifies the colour per violin, and colours are reused if necessary.

lty, lwd Graphical parameters for the violin passed to lines and polygon

rectCol Graphical parameters to control fill colour of the box. NA for no fill colour. If col is a vector, it specifies the colour per violin, and colours are reused if necessary.

lineCol Graphical parameters to control colour of the box outline and whiskers. NA for no border. If lineCol is a vector, it specifies the colour per violin, and colours are reused if necessary.

pchMed Graphical parameters to control shape of the median point. If pchMed is a vector, it specifies the shape per violin.

colMed, colMed2 Graphical parameters to control colour of the median point. If colMed is a vector, it specifies the colour per violin. colMed specifies the fill colour in all cases unless pchMed is 21:25 in which case colMed is the border colour and colMed2 is the fill colour.
at position of each violin. Default to 1:n
wex relative expansion of the violin. If wex is a vector, it specifies the area/width
size per violin and sizes are reused if necessary.
drawRect logical. The box is drawn if TRUE.
areaEqual logical. Density plots checked for equal area if TRUE. wex must be scalar,
relative widths of violins depend on area.
axes, frame.plot, panel.first, panel.last, asp, line, outer, adj, ann, ask, bg, bty, cin, col.axis, col.lab,
Arguments to be passed to methods, such as graphical parameters (see par)).
main, sub, xlab, ylab
graphical parameters passed to plot.
ylog A logical value (see log in plot.default). If TRUE, a logarithmic scale is in
use (e.g., after plot(*, log = "y"). For a new device, it defaults to FALSE, i.e.,
linear scale.
cex A numerical value giving the amount by which plotting text should be magnified
relative to the default.
cex.axis The magnification to be used for y axis annotation relative to the current setting
of cex.
cex.lab The magnification to be used for x and y labels relative to the current setting of
cex.
cex.main The magnification to be used for main titles relative to the current setting of cex.
cex.names The magnification to be used for x axis annotation relative to the current setting
of cex. Takes the value of cex.axis if not given.
cex.sub The magnification to be used for sub-titles relative to the current setting of cex.
yaxt A character which specifies the y axis type. Specifying "n" suppresses plotting.
log Logarithmic scale if log = "y" or TRUE. Invokes ylog = TRUE.
logLab Increments for labelling y-axis on log-scale, defaults to numbers starting with 1,
2, 5, and 10.
na.rm logical value indicating whether NA values should be stripped before the com-
putation proceeds. Defaults to TRUE.
side defaults to "both". Assigning "left" or "right" enables one sided plotting of vio-
lins. May be applied as a scalar across all groups.
plotCentre defaults to "points", plotting a central point at the median. If "line" is given a
median line is plotted (subject to side) alternatively.

Examples

# box- vs violin-plot
par(mfrow=c(2,1))
muc<-2
si<-0.6
bimodal<-c(rnorm(1000,-mu,si),rnorm(1000,mu,si))
uniform<-runif(2000,-4,4)
normal<-rnorm(2000,0,3)
vioplot(bimodal, uniform, normal)
boxplot(bimodal, uniform, normal)

# add to an existing plot
x <- rnorm(100)
y <- rnorm(100)
plot(x, y, xlim=c(-5,5), ylim=c(-5,5))
vioplot(x, col="tomato", horizontal=TRUE, at=-4, add=TRUE, lty=2, rectCol="gray")
vioplot(y, col="cyan", horizontal=FALSE, at=-4, add=TRUE, lty=2)

# formula input
data("iris")
vioplot(Sepal.Length~Species, data = iris, main = "Sepal Length",
col=c("lightgreen", "lightblue", "palevioletred"))
legend("topleft", legend=c("setosa", "versicolor", "virginica"),
fill=c("lightgreen", "lightblue", "palevioletred"), cex = 0.5)
data("diamonds", package = "ggplot2")
palette <- RColorBrewer::brewer.pal(9, "Pastel1")
par(mfrow=c(3, 1))
vioplot(price ~ cut, data = diamonds, las = 1, col = palette)
vioplot(price ~ clarity, data = diamonds, las = 2, col = palette)
vioplot(price ~ color, data = diamonds, las = 2, col = palette)
par(mfrow=c(3, 1))

# generate example data
data_one <- rnorm(100)
data_two <- rnorm(50, 1, 2)

# generate violin plot with similar functionality to vioplot
vioplot(data_one, data_two, col="magenta")

# note vioplox defaults to a greyscale plot
vioplot(data_one, data_two)

# colours can be customised separately, with axis labels, legends, and titles
vioplot(data_one, data_two, col=c("red", "blue"), names=c("data one", "data two"),
main="data violin", xlab="data class", ylab="data read")
legend("topleft", fill=c("red", "blue"), legend=c("data one", "data two"))

# colours can be customised for the violin fill and border separately
vioplot(data_one, data_two, col="grey85", border="purple", names=c("data one", "data two"),
main="data violin", xlab="data class", ylab="data read")

# colours can also be customised for the boxplot rectangle and lines (border and whiskers)
vioplot(data_one, data_two, col="grey85", rectCol="lightblue", lineCol="blue",
border="purple", names=c("data one", "data two"),
main="data violin", xlab="data class", ylab="data read")

# these colours can also be customised separately for each violin
vioplot(data_one, data_two, col=c("skyblue", "plum"), rectCol=c("lightblue", "palevioletred"),
lineCol="blue", border=c("royalblue", "purple"), names=c("data one", "data two"),
main="data violin", xlab="data class", ylab="data read"
# this applies to any number of violins, given that colours are provided for each
vioplot(data_one, data_two, rnorm(200, 3, 0.5), rpois(200, 2.5), rbinom(100, 10, 0.4),
col=c("red", "orange", "green", "blue", "violet"),
rectCol=c("palevioletred", "peachpuff", "lightgreen", "lightblue", "plum"),
lineCol=c("red4", "orangered", "forestgreen", "royalblue", "mediumorchid"),
border=c("red4", "orangered", "forestgreen", "royalblue", "mediumorchid"),
names=c("data one", "data two", "data three", "data four", "data five"),
main="data violin", xlab="data class", ylab="data read")

# The areaEqual parameter scales with width of violins
# Violins will have equal density area (including missing tails) rather than equal maximum width
vioplot(data_one, data_two, areaEqual=TRUE)
vioplot(data_one, data_two, areaEqual=TRUE, col=c("skyblue", "plum"), rectCol=c("lightblue", "palevioletred"),
lineCol="blue", border=c("royalblue", "purple"), names=c("data one", "data two"),
main="data violin", xlab="data class", ylab="data read")

vioplot(data_one, data_two, rnorm(200, 3, 0.5), rpois(200, 2.5), rbinom(100, 10, 0.4),
areaEqual=TRUE, col=c("red", "orange", "green", "blue", "violet"),
rectCol=c("palevioletred", "peachpuff", "lightgreen", "lightblue", "plum"),
lineCol=c("red4", "orangered", "forestgreen", "royalblue", "mediumorchid"),
border=c("red4", "orangered", "forestgreen", "royalblue", "mediumorchid"),
names=c("data one", "data two", "data three", "data four", "data five"),
main="data violin", xlab="data class", ylab="data read")
Index

*Topic **graphics**
   vioplot, 2

*Topic **plot**
   vioplot, 2

*Topic **violin**
   vioplot, 2

boxplot, 2

par, 6
plot.default, 6

split.default, 5

vioplot, 2, 2