Package 'shape'

September 13, 2020

Version 1.4.5

Title Functions for Plotting Graphical Shapes, Colors

Author Karline Soetaert <karline.soetaert@nioz.nl>

Maintainer Karline Soetaert <karline.soetaert@nioz.nl>

Depends R (>= 2.01)

Imports stats, graphics, grDevices

Description Functions for plotting graphical shapes
such as ellipses, circles, cylinders, arrows, ...

License GPL (>= 3)

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2020-09-13 13:10:02 UTC

R topics documented:

---

shape-package *Functions for plotting graphical shapes, colors*

Description

Functions for plotting graphical shapes such as ellipses, circles, cylinders, arrows, ...

Support for the book "A practical guide to ecological modelling - using R as a simulation platform" by Karline Soetaert and Peter M.J. Herman (2009). Springer.

Details

Package:	shape
Type:	Package
Version:	1.3.4
Date:	2011-07-30
License:	GNU Public License 3 or above

This package is used in R-package ecolMod, which includes many more examples.

See also R-package diagram.

Changes in version 1.3.4: more consistent drawing of ellipse and circle segments, (functions getellipse, getcircle), added textflag. (both suggested by Tom Wilson)

Author(s)

Karline Soetaert (Maintainer)

See Also

A4, writelabel, emptyplot, drapecol, femmecol, intpalette, shadepalette, colorlegend, greycol, rotatexy, Arrowhead, Arrows, cylindersegment, filledcylinder, filledcircle, filledellipse, filledmultigonal, filledrectangle, filledshape, getellipse, plotcircle, plotellipse, roundrect, textflag.

Examples

```
## Not run:
## show examples (see respective help pages for details)
example(rotatexy)
example(filledshape)

## run demos
demo("colorshapes") # creating colored shapes

## open the directory with source code of demos
browseURL(paste(system.file(package="shape"), "/demo", sep=""))

## show package vignette
vignette("shape")
edit(vignette("shape"))
browseURL(paste(system.file(package="shape"), "/doc", sep=""))

## End(Not run)
```

A4 *opens A4-sized window*

Description

opens a graphics window, 8.5 inches wide, 11 inches high

Usage

```
A4 (...)
```

Arguments

...	arguments passed to R-function X11.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

`Arrowhead` *adds arrowheads to a plot*

Description

adds one or more arrowheads to a plot; shape is either curved, a triangle, a circle or ellipse.

Usage

```
Arrowhead(x0, y0, angle = 0, arr.length = 0.4,
          arr.width = arr.length/2, arr.adj = 0.5,
          arr.type = "curved", lcol = "black", lty = 1,
          arr.col = lcol, arr.lwd = 2, npoint = 5, ...)
```

Arguments

`x0`	x-coordinates of points at which to draw arrowhead; either one value or a vector.
`y0`	y-coordinates of points at which to draw arrowhead; either one value or a vector.
`angle`	angle of arrowhead (anti-clockwise, relative to x-axis), in degrees [0,360]; either one value or a vector.
`arr.length`	approximate length of arrowhead, in cm; either one value or a vector.
`arr.width`	approximate width of arrowhead, in cm; either one value or a vector.
`arr.adj`	0,0.5,1 specifying the adjustment of the arrowhead.
`arr.type`	type of arrowhead to draw, one of "curved", "triangle", "circle", "ellipse".
`lcol`	line color specifications; either one value or a vector.
`lty`	line type specifications; either one value or a vector.
`arr.col`	color of arrowhead; either one value or a vector.
`arr.lwd`	line width of arrowhead.
`npoint`	only if `arr.type` = "curved": number of points to draw the curve; increase for smoother arrowheads
`...`	arguments passed to the `polygon` function.

Details

`x0`, `y0`, `angle`, `arr.length`, `arr.width`, `lcol`, `lty` and `arr.col` can be a vector, of the same length.

- if `arr.adj` = 0.5, then the centre of the arrowhead is at the point at which it is drawn.
- `arr.adj` = 1 causes the tip of the arrowhead to touch the point.
- `arr.adj` = 0 causes the base of the arrowhead to touch the point.

The type of the arrowhead is set with `arr.type` which can take the values:

- "triangle": uses filled triangle
- "curved" : draws arrowhead with curved edges
- "circle" : draws circular head (where arr.width=arr.length)
- "ellipse" : draws ellipsoid head

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

Arrows

Examples

```
emptyplot(main = "Arrowhead")
Arrowhead(x0 = runif(10), y0 = runif(10), angle = runif(10)*360,
          arr.length = 0.3, arr.type = "circle", arr.col = "green")
Arrowhead(x0 = runif(10), y0 = runif(10), angle = runif(10)*360,
          arr.length = 0.4, arr.type = "curved", arr.col = "red")
Arrowhead(x0 = runif(10), y0 = runif(10), angle = runif(10)*360,
          arr.length = runif(10), arr.type = "triangle",
          arr.col = rainbow(10))
```

Arrows *adds arrows with improved arrowhead to a plot*

Description

adds one or more arrows to a plot; arrowhead shape is either curved, a triangle, a circle or simple

Usage

```
Arrows(x0, y0, x1, y1, code = 2, arr.length = 0.4,
  arr.width = arr.length/2, arr.adj = 0.5, arr.type = "curved",
  segment = TRUE, col = "black", lcol = col, lty = 1, arr.col = lcol,
  lwd = 1, arr.lwd = lwd, ...)
```

Arguments

x0	x-coordinates of points *from* which to draw arrows; either one value or a vector.
y0	y-coordinates of points *from* which to draw arrows; either one value or a vector.
x1	x-coordinates of points *to* which to draw arrows; either one value or a vector.
y1	y-coordinates of points *to* which to draw arrows; either one value or a vector.
code	integer code determining kind of arrows to draw.
arr.length	approximate length of arrowhead, in cm; either one value or a vector.
arr.width	approximate width of arrowhead, in cm; either one value or a vector.
arr.adj	0,0.5,1 specifying the adjustment of the arrowhead.

arr.type	type of arrowhead to draw, one of "none", "simple", "curved", "triangle", "circle", "ellipse" or "T".
segment	logical specifying whether or not to draw line segments.
col	general line color specification; one value or a vector.
lcol	line color specifications; either one value or a vector. ignored when arr.type = "simple" or "T" - use "col"
lty	line type specifications; either one value or a vector.
arr.col	color of arrowhead; either one value or a vector.
lwd	general line width specification. The default value changed to 1 from version 1.4 (was 2)
arr.lwd	line width of arrowhead.
...	arguments passed to lines, segments or Arrowhead function.

Details

x0, y0, x1, y1, arr.length, arr.width, arr.adj, lcol, lty and arr.col can be a vector, of the same length.

For each 'i', an arrow is drawn between the point '(x0[i], y0[i])' and the point '(x1[i],y1[i])'.

- If code=1 an arrowhead is drawn at '(x0[i],y0[i])'
- if code=2 an arrowhead is drawn at '(x1[i],y1[i])'.
- If code=3 an arrowhead is drawn at both ends of the arrow
- unless arr.length = 0, when no head is drawn.

- If arr.adj = 0.5 then the centre of the arrowhead is at the point at which it is drawn.
- arr.adj = 1 causes the tip of the arrowhead to touch the point.
- arr.adj = 2 causes the base of the arrowhead to touch the point.

The type of the arrowhead is set with arr.type which can take the values:

- "simple" : uses comparable R function arrows
- "triangle": uses filled triangle
- "curved" : draws arrowhead with curved edges
- "circle" : draws circular head
- "ellipse" : draws ellepsoid head
- "T" : draws T-shaped (blunt) head

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

arrows the comparable R function

Arrowhead

Examples

```
xlim <- c(-5 , 5)
ylim <- c(-10, 10)
plot(0, type = "n", xlim = xlim, ylim = ylim,
     main = "Arrows,  type = 'curved'")
x0 <- runif(100, xlim[1], xlim[2])
y0 <- runif(100, ylim[1], ylim[2])
x1 <- x0+runif(100, -1, 1)
y1 <- y0+runif(100, -1, 1)
Arrows(x0, y0, x1, y1, arr.length = runif(100), code = 2,
       arr.type = "curved", arr.col = 1:100, lcol = 1:100)

plot(0, type = "n", xlim = xlim, ylim = ylim,
     main = "Arrows,  type = 'circle'")
x0 <- runif(100, xlim[1], xlim[2])
y0 <- runif(100, ylim[1], ylim[2])
x1 <- x0 + runif(100, -1, 1)
y1 <- y0 + runif(100, -1, 1)
Arrows(x0, y0, x1, y1, arr.length = 0.2, code = 3,
       arr.type = "circle", arr.col = "grey")

plot(0, type = "n", xlim = xlim, ylim = ylim,
     main = "Arrows,  type = 'ellipse'")
Arrows(x0, y0, x1, y1, arr.length = 0.2, arr.width = 0.5,
       code = 3, arr.type = "ellipse",  arr.col = "grey")

curve(expr = sin(x), 0, 2*pi+0.25, main = "Arrows")
x  <- seq(0, 2*pi, length.out = 10)
xd <- x + 0.025
Arrows(x, sin(x), xd, sin(xd), type = "triangle",
       arr.length = 0.5, segment = FALSE)

xx <- seq(0, 10*pi, length.out = 1000)
plot(sin(xx)*xx, cos(xx)*xx, type = "l", axes = FALSE,
     xlab = "", ylab = "", main = "Arrows,  type = 'curved'")
x  <- seq(0, 10*pi, length.out = 20)
x1 <- sin(x)*x
y1 <- cos(x)*x
xd <- x+0.01
x2 <- sin(xd)*xd
y2 <- cos(xd)*xd
Arrows(x1, y1, x2, y2, arr.type = "curved", arr.length = 0.4,
       segment = FALSE, code = 1, arr.adj = 0.5 )

plot(sin(xx)*xx, cos(xx)*xx, type = "l", axes = FALSE,
     xlab = "", ylab = "", main = "Arrows,  type = 'T'")
Arrows(x1, y1, x2, y2, arr.type = "T", arr.length = 0.4,
       code = 1, arr.lwd = 2)


# arguments passed to polygon:
```

```
xlim <- c(-5 , 5)
ylim <- c(-10, 10)
plot(0, type = "n", xlim = xlim, ylim = ylim,
     main = "Arrows,  type = 'curved'")
x0 <- runif(100, xlim[1]-1, xlim[2]+0.5)  # exceeds the x-range
y0 <- runif(100, ylim[1], ylim[2])
x1 <- x0+runif(100, -1, 1)
y1 <- y0+runif(100, -1, 1)
Arrows(x0, y0, x1, y1, arr.length = runif(100), code = 2,
       arr.type = "curved", arr.col = 1:100, lcol = 1:100, xpd = TRUE)
```

`colorlegend` *adds a color legend to a plot.*

Description

Adds a color legend to a plot.

Usage

```
colorlegend(col = femmecol(100), zlim, zlevels = 5, dz = NULL,
            zval = NULL, log = FALSE, posx = c(0.9, 0.93),
            posy = c(0.05, 0.9), main = NULL, main.cex = 1.0,
            main.col = "black", lab.col = "black",
            digit = 0, left = FALSE, ...)
```

Arguments

`col`	color palette to be used; also allowed are two extremes or one value.
`zlim`	two-valued vector, the minimum and maximum z values.
`zlevels`	number of z-levels, one value, ignored if `dz` or `zval` not equal to `NULL`.
`dz`	increment in legend values, one value; ignored if `zval` not equal to `NULL`.
`zval`	a vector of z-values to label legend.
`log`	logical indicating whether to log transform or not.
`posx`	relative position of left and right edge of color bar on first axis, [0,1].
`posy`	relative position on lower and upper edge of colar bar on second axis, [0,1].
`main`	main title, written above the color bar.
`main.cex`	relative size of main title.
`main.col`	color of main title.
`lab.col`	color of labels.
`digit`	number of significant digits in labels.
`left`	logical indicating whether to put the labels on the right (`TRUE`) or on the left (`FALSE`).
`...`	arguments passed to R-function text when writing labels.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

Examples

```
emptyplot(main = "colorlegend")
colorlegend(zlim = c(0, 10))
colorlegend(posx = c(0.8, 0.83), col = greycol(100),
            zlim = c(0, 1), digit = 1)
colorlegend(posx = c(0.7, 0.73), left = TRUE, col = rainbow(100),
            zlim = c(0, 10), digit = 1, dz = 2.5)
colorlegend(posx = c(0.5, 0.53),
            col = intpalette(c("red", "yellow", "black"), 100),
            zlim = c(0, 20), zval = c(1, 3, 7, 15))
colorlegend(posy = c(0.0, 0.15), posx = c(0.2, 0.3),
            col = rainbow(100), zlim = c(0, 1),
            zlevels = NULL, main = "rainbow")
colorlegend(posy = c(0.25, 0.4), posx = c(0.2, 0.3),
            zlim = c(0, 1), zlevels = NULL, main = "femmecol")
colorlegend(posy = c(0.5, 0.65), posx = c(0.2, 0.3),
            col = terrain.colors(100), zlim = c(0, 1),
            zlevels = NULL, main = "terrain.colors")
colorlegend(posy = c(0.75, 0.9), posx = c(0.2, 0.3),
            col = heat.colors(100), zlim = c(0, 1),
            zlevels = NULL, main = "heat.colors")
```

---

cylindersegment *adds part of a cylinder to a plot*

Description

adds a segment of a cylinder to a plot

Usage

```
cylindersegment(rx = 1, ry = rx, from = pi, to = 3*pi/2, len = 1,
                mid = c(0,0), angle = 0, dr = 0.01, col = "black",
                delt = 1.0, ...)
```

Arguments

rx	horizontal radius of full cylinder.
ry	vertical radius of full cylinder.
from	start radius of segment, radians.
to	end radius of segment, radians.
len	cylinder length.
mid	midpoint of cylinder.

angle	rotation angle, degrees.
dr	size of segments, in radians, to draw top/bottom ellipse (decrease for smoother).
col	color of slice.
delt	increase factor, from left to right.
...	arguments passed to polygon function.

Details

When angle = 0 (the default), the cylindersegment is parallel to the x-axis.

rx and ry are the horizontal and vertical radiusses of the bordering ellipses. Here "horizontal" and "vertical" denote the position BEFORE rotation

if delt > 1, the width of the cylinder will increase from left to right.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

filledcylinder

Examples

```
emptyplot(main = "cylindersegment")
cylindersegment(mid = c(0.1, 0.5), rx = 0.1, ry = 0.1,
                from = pi, to = 3*pi/2, col = "blue",
                len = 0.5, delt = 1.1, lwd = 2, angle = 90)
cylindersegment(mid = c(0.8, 0.5), rx = 0.1, ry = 0.1,
                from = 0, to = pi/2, col = "red", len = 0.5,
                delt = 1.0, lwd = 2, angle = 45)
cylindersegment(mid = c(0.5, 0.5), rx = 0.1, ry = 0.1,
                from = pi/2, to = pi, col = "lightblue",
                len = 0.2, delt = 1.5, lwd = 2)
for (i in seq(0.1, 0.9, 0.1))
  cylindersegment(mid = c(i, 0.9), rx = 0.035, ry = 0.05,
                  from = pi/2, to = 3*pi/2, col = "darkblue",
                  len = 0.1, angle = 90)
```

---

drapecol *draping colors over a persp plot*

Description

generates color(s) that will appear on the surface facets of a "persp" plot.

Usage

```
drapecol(A, col = femmecol(100), NAcol = "white", lim = NULL)
```

Arguments

A	matrix with input grid.
col	color palette.
NAcol	color of NA elements.
lim	The limits of the data; if NULL, the data range will be chosen.

Value

a vector of character strings giving the colors in hexadecimal format, one for each surface facet.

Note

This function is inspired by a similar function in package fields, unfortunately made unavailable in most recent version of fields

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

persp

Examples

```
persp(volcano, theta = 135, phi = 30, col = drapecol(volcano),
      main = "drapecol")
persp(volcano, theta = 135, phi = 30, col = drapecol(volcano),
      border = NA, main = "drapecol")
```

---

emptyplot *open a plot without axes, labels,...*

Description

Creates a plotting region, bounded by xlim and ylim; without axes, labels, titles, useful for plotting shapes.

Usage

```
emptyplot(xlim = c(0, 1), ylim = xlim, asp = 1, frame.plot = FALSE,
          col = NULL, ...)
```

Arguments

xlim	the x limits (min,max) of the plot.
ylim	the y limits (min,max) of the plot.
asp	the y/x aspect ratio.
frame.plot	to toggle off drawing of a bounding box.
col	the background color.
...	arguments passed to R-function plot.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

plot, plot.default

---

femmecol *red-green-blue color palette*

Description

Creates a vector of (n) contiguous colors (darkblue-blue-cyan-yellow-red-darkred).

Usage

```
femmecol(n = 100)
```

Arguments

n	number of colors.

Value

a vector of character strings giving the colors in hexadecimal format

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

rainbow, heat.colors, topo.colors, the comparable R-functions.

intpalette, shadepalette

Examples

```
filled.contour(volcano, color = femmecol, asp = 1, main = "femmecol")
femmecol(10)
image(matrix(nrow = 1, ncol = 100, data = 1:100),
      col = femmecol(100), main = "femmecol")
```

`filledcircle` *adds colored circle to a plot*

Description

plots (part of) outer and inner circle and colors inbetween; color can be a palette.

Usage

```
filledcircle(r1 = 1, r2 = 0, mid = c(0,0), dr = 0.01, from = -pi, to = pi,
  col = femmecol(100), values = NULL, zlim = NULL, lwd = 2, lcol = NA, ...)
```

Arguments

`r1`	radius of outer circle.
`r2`	radius of inner circle.
`mid`	midpoint of circle.
`dr`	size of segments, in radians, to draw circle (decrease for smoother).
`from`	starting angle for circle segment, radians.
`to`	final angle for circle segment, radians. The segment is drawn counterclockwise. The default is to draw a full circle.
`col`	color palette to be used; also allowed are two extremes or one value.
`values`	if not `NULL`, a matrix providing (radius,z-values) couples, used for coloring. .
`zlim`	Only if `values` is not `NULL`: the minimum and maximum z values for which colors should be plotted, defaulting to the range of the finite values of the second column of `values`.
`lwd`	width of external line.
`lcol`	line color.
`...`	arguments passed to R-function polygon.

Details

see `filledellipse` for details

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

filledshape, filledcylinder, filledellipse

Examples

```
color <-graycol(n = 50)
dr    <- 0.05
emptyplot(xlim = c(-2, 2), col = color[length(color)],
          main = "filledcircle")
filledcircle(r1 = 1, mid = c(1, 1), dr = dr,
          col = shadepalette(endcol = "darkblue"))
filledcircle(r1 = 1, mid = c(-1, -1), dr = dr,
          col = shadepalette(endcol = "darkred"))
filledcircle(r1 = 1, r2 = 0.5, mid = c(0, 0), dr = dr,
          col = c(rev(color), color))
filledcircle(r1 = 1, mid = c(1, -1), dr = dr,
          col = intpalette(c("red", "blue", "orange"), 100))
filledcircle(mid = c(-1, 1))

emptyplot(main = "filledcircle")

for (i in seq(0, 0.45, 0.05))
   filledcircle(r1 = i+0.05, r2 = i,
                mid = c(0.5, 0.5), col = i*20)
```

---

filledcylinder *adds a colored and rotated cylinder to a plot*

Description

adds a rotated and colored cylinder to a plot; color can be a palette

Usage

```
filledcylinder(rx = 1, ry = rx, len = 1, col = femmecol(100),
               lcol = NA, lwd = 2, lcolint = NULL, ltyint = 1,
               lwdint = lwd, mid = c(0,0), angle = 0, delt = 1,
               dr = 0.01, topcol = NULL, botcol = NULL, ...)
```

Arguments

rx	horizontal radius.
ry	vertical radius.
len	length.
col	color palette to be used; also allowed are two extremes or one value.
lcol	line color on external surface.
lwd	only if lcol!=NA, width of external line.

lcolint	only if lcol!=NA, line color on internal (hidden) surface.
ltyint	only if lcol!=NA, line type on internal (hidden) surface.
lwdint	only if dlcol!=NA, line width on internal (hidden) surface.
mid	midpoint of cylinder.
angle	rotation angle, degrees.
delt	increase factor, from left to right.
dr	size of segments, in radians, to draw top/bottom ellipse (decrease for smoother).
topcol	color (palette) of top (right) surface.
botcol	color (palette) of bottom (left) surface.
...	arguments passed to function filledellipse.

Details

When angle = 0 (the default), the cylinder is parallel to the x-axis

rx and ry are the horizontal and vertical radiusses of the bordering ellipses. Here "horizontal" and "vertical" denote the position BEFORE rotation

if delt > 1, the width of the cylinder will increase from left to right.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

filledellipse, filledshape

Examples

```
emptyplot(c(-1.2, 1.2), c(-1, 1), main = "filledcylinder")
col  <- c(rev(greycol(n = 50)), greycol(n = 50))
col2 <- shadepalette("red", "blue", n = 50)
col3 <- shadepalette("yellow", "black", n = 50)
filledcylinder(rx = 0., ry = 0.2, len = 0.25, angle = 0, col = col,
          mid = c(-1, 0), topcol = col[25])
filledcylinder(rx = 0., ry = 0.2, angle = 90, col = col,
          mid = c(-0.5, 0), topcol = col[25])
filledcylinder(rx = 0.1, ry = 0.2, angle = 90, col = c(col2, rev(col2)),
          mid = c(0.45, 0), topcol = col2[25])
filledcylinder(rx = 0.05, ry = 0.2, angle = 90, col = c(col3, rev(col3)),
          mid = c(0.9, 0), topcol = col3[25])
filledcylinder(rx = 0.1, ry = 0.2, angle = 90, col = "white",
          lcol = "black", lcolint = "grey")

emptyplot(c(-1, 1), c(-1, 1), main = "filledcylinder")
col  <- shadepalette("blue", "black", n = 50)
col2 <- shadepalette("red", "black", n = 50)
col3 <- shadepalette("yellow", "black", n = 50)
filledcylinder(rx = 0.025, ry = 0.2, angle = 90, col = c(col2, rev(col2)),
```

```
          mid = c(-0.8, 0), topcol = col2[25], delt = -1, lcol = "black")
filledcylinder(rx = 0.1, ry = 0.2, angle = 00, col = c(col, rev(col)),
          mid = c(0.0, 0.0), topcol = col, delt = -1.2, lcol = "black")
filledcylinder(rx = 0.075, ry = 0.2, angle = 90, col = c(col3, rev(col3)),
          mid = c(0.8, 0), topcol = col3[25], delt = 0.0, lcol = "black")
```

`filledellipse` *adds a colored and rotated ellipse to a plot*

Description

plots (part of) outer and inner ellipses and colors inbetween; color can be a palette

Usage

```
filledellipse(rx1 = 1, rx2 = 0, ry1 = rx1, ry2 = NULL, mid = c(0,0),
  dr = 0.01, angle = 0, from = -pi, to = pi, col = femmecol(100),
  values = NULL, zlim = NULL, lwd = 2, lcol = NA, ...)
```

Arguments

`rx1`	horizontal radius of outer ellipse.
`rx2`	horizontal radius of inner ellipse.
`ry1`	vertical radius of outer ellipse.
`ry2`	vertical radius of inner ellipse.
`mid`	midpoint of ellipse.
`dr`	size of segments, in radians, to draw ellipse (decrease for smoother).
`angle`	rotation angle, degrees.
`from`	starting angle for ellipse segment, radians.
`to`	final angle for ellipse segment, radians. The segment is drawn counterclockwise. The default is draw a full ellipse.
`col`	color palette to be used; also allowed are two extremes or one value.
`values`	if not `NULL`, a matrix providing (radius,z-values) couples, used for coloring. .
`zlim`	Only if `values` is not `NULL`: the minimum and maximum z values for which colors should be plotted, defaulting to the range of the finite values of the second column of `values`.
`lwd`	width of external line.
`lcol`	line color.
`...`	arguments passed to R-function polygon.

Details

draws (part of) an outer and inner ellipse, as specified by inner and outer radiusses:

`rx1`,`ry1`: horizontal and vertical radiusses of outer ellipse; `rx2`,`ry2`: same for inner ellipse. Here "horizontal" and "vertical" denote the position BEFORE rotation

Fills with a palette of colors inbetween

`values`: if not `NULL`, a matrix providing (radius,z-values) couples, used for coloring. Here radius are positive values denoting the relative distance between the shapes centre and edge. The radiusses are rescaled to be in [0,1] if needed. z-values (2nd column of `values`) together with `zlim` and `col` denote the coloration level.

Colors in `col` will be interpolated to the z-values and used to color an interval as given by the input radiusses.

If `rx2`, the radius of the inner ellipse is 0, the ellipse is full.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

`filledshape`, `filledcylinder`

Examples

```
color <- greycol(50)
dr    <- 0.05
emptyplot(xlim = c(-2, 2), ylim = c(-2, 2), col = color[length(color)],
          main = "filledellipse")
filledellipse(rx1 = 1, mid = c(1, 1) , dr = dr,
              col = shadepalette(endcol = "darkblue"))
filledellipse(rx1 = 1, ry1 = 0.5, mid = c(-1, -1), dr = dr, angle = 90,
              col = shadepalette(endcol = "darkred"))
filledellipse(rx1 = 1, ry1 = 0.5, rx2 = 0.5, dr = dr, mid = c(0, 0),
              col = c(rev(color), color))
filledellipse(rx1 = 0.5, mid = c(1, -1), dr = dr, from = pi,  to = 1.5*pi,
              col = rev(shadepalette(endcol = "black")))
filledellipse(mid = c(-1, 1))

emptyplot(xlim = c(-2, 2), ylim = c(-2, 2), main = "filledellipse")
filledellipse(rx1 = 0.75, mid = c(-1, 1), col = greycol(100) , dr = dr,
              values = cbind (1:100, (1:100)^0.5))
filledellipse(rx1 = 0.75, mid = c(1, 1), col = greycol(100)  , dr = dr,
              values = cbind (1:100, (1:100)))
filledellipse(rx1 = 0.75, mid = c(-1, -1), col = greycol(100), dr = dr,
              values = cbind (1:100, (1:100)^2))
filledellipse(rx1 = 0.75, mid = c(1, -1), col = greycol(100) , dr = dr,
              values = cbind (1:100, (1:100)^5))
```

`filledmultigonal` *adds a colored and rotated multigonal shape to a plot*

Description

draws and colors a rotated shape with equal-sized vertices ; color can be a palette.

Usage

```
filledmultigonal(mid = c(0, 0), rx = 1, ry = rx, nr = 4,
                 col = femmecol(100), values = NULL,
                 zlim = NULL, lwd = 2, lcol = NA, angle = 0, ...)
```

Arguments

`mid`	midpoint of multigonal.
`rx`	horizontal radius.
`ry`	vertical radius.
`nr`	number of sides.
`col`	color palette to be used; also allowed are two extremes or one value.
`values`	if not `NULL`, a matrix providing (radius,z-values) couples, used for coloring.
`zlim`	Only if `values` is not `NULL`: the minimum and maximum z values for which colors should be plotted, defaulting to the range of the finite values of the second column of `values`.
`lwd`	width of external line.
`lcol`	line color.
`angle`	angle of rotation, in degrees.
`...`	arguments passed to R-function polygon.

Details

Coloration proceeds from midpoint to external edge

`rx`,`ry`: horizontal and vertical radiusses of the shape. Here "horizontal" and "vertical" denote the position BEFORE rotation

`values`: if not `NULL`, a matrix providing (radius,z-values) couples, used for coloring. Here radius are positive values denoting the relative distance between the shapes centre and edge. The radiusses are rescaled to be in [0,1] if needed. z-values (2nd column of `values`) together with `zlim` and `col` denote the coloration level.

Colors in `col` will be interpolated to the z-values and used to color an interval as given by the input radiusses.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

filledrectangle, filledshape, filledcylinder, filledellipse

Examples

```
emptyplot(c(-1, 1), main = "filledmultigonal")

filledmultigonal(rx = 0.25, ry = 0.125, nr = 3, mid = c(-0.75, 0.75),
            angle = 45, col = shadepalette("red", "blue", n = 50))
filledmultigonal(rx = 0.125, ry = 0.25, nr = 3, mid = c(-0.25, 0.75),
            col = shadepalette("red", "yellow", n = 50))
filledmultigonal(rx = 0.25, ry = 0.25 , nr = 3, mid = c(0.25, 0.75),
            col = c("red", "orange"))
filledmultigonal(rx = 0.25, ry = 0.25 , nr = 3, mid = c(0.75, 0.75),
            angle = 90,  col = "red")

filledmultigonal(rx = 0.25, ry = 0.25, nr = 4, mid = c(-0.75, 0.25),
            angle = 0, col = shadepalette("red", "blue", n = 50))
filledmultigonal(rx = 0.25, ry = 0.25, nr = 4, mid = c(-0.25, 0.25),
            angle = 45, col = shadepalette("red", "blue", n = 50))
filledmultigonal(rx = 0.25, ry = 0.125, nr = 4, mid = c(0.25, 0.25),
            angle = 0, col = shadepalette("red", "blue", n = 50))
filledmultigonal(rx = 0.25, ry = 0.125, nr = 4, mid = c(0.75, 0.25),
            angle = 45, col = shadepalette("red", "blue", n = 50))

filledmultigonal(rx = 0.25, ry = 0.25, nr = 5, mid = c(-0.75, -0.25),
            angle = 0, col = shadepalette("darkgreen", "lightgreen", n = 50))
filledmultigonal(rx = 0.25, angle = 0, nr = 5, mid = c(-0.25, -0.25),
            col = rainbow(50))
filledmultigonal(rx = 0.25, angle = 30, nr = 6, mid = c(0.25, -0.25),
            col = femmecol(50))
filledmultigonal(rx = 0.25, ry = 0.125, angle = 30, nr = 6, mid = c(0.75, -0.25),
            col = "black")

filledmultigonal(rx = 0.25, col = "darkblue", nr = 7, mid = c(-0.75, -0.75))
filledmultigonal(rx = 0.25, col = "darkblue", nr = 9, mid = c(-0.25, -0.75))
filledmultigonal(rx = 0.25, col = "darkblue", nr = 3.7, mid = c(0.25, -0.75))
filledmultigonal(rx = 0.25, col = "darkblue", nr = 4.5, mid = c(0.75, -0.75))
```

---

filledrectangle *adds a colored and rotated rectangle to a plot*

Description

plots and colors a rotated rectangle; color can be a palette

Usage

```
filledrectangle(mid = c(0, 0), wx = 1, wy = wx, col = femmecol(100),
                values = NULL, zlim = NULL, lwd = 2, lcol = NA,
                angle = 0, ...)
```

Arguments

mid	midpoint of rectangle.
wx	horizontal width.
wy	vertical width.
col	color palette to be used; also allowed are two extremes or one value.
values	if not NULL, a matrix providing (radius,z-values) couples, used for coloring.
zlim	Only if values is not NULL: the minimum and maximum z values for which colors should be plotted, defaulting to the range of the finite values of the second column of values.
lwd	width of external line.
lcol	line color.
angle	angle of rotation, in degrees.
...	arguments passed to R-function polygon.

Details

If angle=0, coloration starts from top to bottom. This is different from filledmultigonal, where coloration proceeds from middle to external

wx,wy: horizontal and vertical width of the shape Here "horizontal" and "vertical" denote the position BEFORE rotation

values: if not NULL, a matrix providing (radius,z-values) couples, used for coloring. Here radius are positive values denoting the relative distance between the shapes centre and edge. The radiusses are rescaled to be in [0,1] if needed. z-values (2nd column of values) together with zlim and col denote the coloration level.

Colors in col will be interpolated to the z-values and used to color an interval as given by the input radiusses.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

filledmultigonal, filledshape, filledcylinder, filledellipse

polygon, rect for corresponding R-functions.

Examples

```
color <- shadepalette(grey(0.3), "lightblue", n = 50)
emptyplot(main = "filledrectangle")
filledrectangle(wx = 0.5, wy = 0.5, col = color,
                mid = c(0.5, 0.5), angle = 0)
filledrectangle(wx = 0.25, wy = 0.25, col = "darkblue",
                mid = c(0.5, 0.5), angle = 45)
filledrectangle(wx = 0.125, wy = 0.125, col = c("lightblue","blue"),
```

```
                mid = c(0.5, 0.5), angle = 90)

color <- shadepalette(grey(0.3), "blue", n = 50)
emptyplot(c(-1, 1), main = "filledrectangle")
filledrectangle(wx = 0.5, wy = 0.5, col = color,
                mid = c(0, 0), angle = 0)
filledrectangle(wx = 0.5, wy = 0.5, col = color,
                mid = c(0.5, 0.5), angle = 90)
filledrectangle(wx = 0.5, wy = 0.5, col = color,
                mid = c(-0.5, -0.5), angle = -90)
filledrectangle(wx = 0.5, wy = 0.5, col = color,
                mid = c(0.5, -0.5), angle = 180)
filledrectangle(wx = 0.5, wy = 0.5, col = color,
                mid = c(-0.5, 0.5), angle = 270)
```

`filledshape` *adds a colored shape to a plot*

Description

plots outer and inner shape and colors inbetween; color can be a palette

Usage

```
filledshape(xyouter, xyinner = colMeans(xyouter),
            col = femmecol(100), values = NULL,
            zlim = NULL, lcol = NA, lwd = 2, ...)
```

Arguments

`xyouter`	2-column matrix with x,y values of outer shape.
`xyinner`	2-column matrix of 2-valued vector with x,y values of inner shape; default is centroid of `xyouter`.
`col`	color palette to be used; also allowed are two extremes.
`values`	if not `NULL`, a matrix providing (radius,z-values) couples, used for coloring.
`zlim`	Only if `values` is not `NULL`: the minimum and maximum z values for which colors should be plotted, defaulting to the range of the finite values of the second column of *values*.
`lcol`	line color.
`lwd`	width of external line, only if `lcol` != `NA`.
`...`	arguments passed to R-function polygon

Details

draws and outer and inner shape, as specified in `xyouter`, and `xyinner` and fills with a palette of colors inbetween;

`values`: if not null, a matrix providing (radius,z-values) couples, used for coloring. Here radius are positive values denoting the relative distance between the shapes centre and edge. The radiusses are rescaled to be in [0,1] if needed. z-values (2nd column of `values`) together with `zlim` and `col` denote the coloration level.

Colors in `col` will be interpolated to the z-values and used to color an interval as given by the input radiusses.

If `xyinner` is a point, the shape is full.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

`filledellipse`, `filledcylinder`

Examples

```
#an egg
color <-greycol(100)
emptyplot(c(-3.2, 3.2), col = color[length(color)], main = "filledshape")
b <- 4
a <- 9
x      <- seq(-sqrt(a), sqrt(a), by = 0.01)
g      <- b-b/a*x^2 - 0.2*b*x + 0.2*b/a*x^3
g[g<0] <- 0
x1     <- c(x, rev(x))
g1     <- c(sqrt(g), rev(-sqrt(g)))
xouter <- cbind(x1, g1)
xouter <- rbind(xouter, xouter[1,])
filledshape(xouter, xyinner = c(-1, 0), col = color)

# a mill
color <- shadepalette(grey(0.3), "yellow", n = 50)
emptyplot(c(-3.3, 3.3), col = color[length(color)], main = "filledshape")
x <- seq(0, 0.8*pi, pi/100)
y <- sin(x)
xouter <- cbind(x, y)

for (i in seq(0, 360, 60))
  xouter <- rbind(xouter, rotatexy(cbind(x, y), mid = c(0, 0), angle = i))
filledshape(xouter, c(0, 0), col = color)

# abstract art
emptyplot(col = "darkgrey", main = "filledshape")
filledshape(matrix(ncol = 2, runif(100)), col = "darkblue")
```

getellipse *x-y coordinates of ellipse*

Description

calculates x-y values for (part of) an ellipse; the ellipse can be rotated

Usage

```
getellipse(rx = 1, ry = rx, mid = c(0, 0), dr = 0.01,
           angle = 0, from = -pi, to = pi)
```

Arguments

rx	long radius of ellipse.
ry	short radius of ellipse.
mid	midpoint of ellipse.
dr	size of segments, in radians, to specify ellipse (decrease for smoother).
angle	rotation angle, degrees.
from	starting angle for ellipse segment, radians.
to	final angle for ellipse segment, radians. The segment is generated counterclockwise. The default is draw a full ellipse.

Details

rx and ry are the horizontal and vertical radiusses of the ellipses.

points from and to are joined counterclockwise. (this has changed since version 1.3.4).

Value

a 2-column matrix with x-y values of the ellipse

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

plotellipse, filledellipse

Examples

```
plot(getellipse(1, from = 0, to = pi/2), type = "l", col = "red",
     lwd = 2, main = "getellipse")
lines(getellipse(0.5, 0.25, mid = c(0.5, 0.5)), type = "l",
      col = "blue", lwd = 2)
lines(getellipse(0.5, 0.25, mid = c(0.5, 0.5), angle = 45),
      type = "l", col = "green", lwd = 2)

lines(getellipse(0.2, 0.2, mid = c(0.5, 0.5), from = 0, to = pi/2),
      type = "l", col = "orange", lwd = 2)
lines(getellipse(0.2, 0.2, mid = c(0.5, 0.5), from = pi/2, to = 0),
      type = "l", col = "black", lwd = 2)
lines(getellipse(0.1, 0.1, mid = c(0.75, 0.5), from = -pi/2, to = pi/2),
      type = "l", col = "black", lwd = 2)

emptyplot(main = "getellipse")
col <- femmecol(90)
for (i in seq(0, 180, by = 2))
      lines(getellipse(0.5, 0.25, mid = c(0.5, 0.5), angle = i),
            type = "l", col = col[(i/2)+1], lwd = 2)
```

---

greycol *white-black color palette*

Description

Creates a vector of (n) contiguous colors from white/grey to black

Usage

```
greycol(n = 100, interval = c(0.0, 0.7))
```

Arguments

n	number of colors.
interval	interval *to* where to interpolate.

Details

greycol is an alias of graycol

Value

a vector of character strings giving the colors in hexadecimal format.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

rainbow, heat.colors, topo.colors, femmecol

Examples

```
filled.contour(volcano, color = graycol, asp = 1, main = "greycol,graycol")
graycol(10)
image(matrix(nrow = 1, ncol = 100, data = 1:100),
      col = graycol(100), main = "greycol,graycol")
```

---

intpalette *color palettes*

Description

Returns color(s) that are a linear interpolation of a given set of colors.

Usage

```
intpalette(inputcol, numcol = length(x.to), x.from = NULL, x.to = NULL)
```

Arguments

inputcol	initial colors, *from* where to interpolate.
numcol	number of colors to interpolate *to*.
x.from	x-values *from* where to interpolate.
x.to	x-values where to interpolate *to*.

Details

Return value is a vector of *colors* in hexadecimal format.

This is different from colorRamp(R function), that returns a *function*

Value

a vector of character strings giving the interpolated colors in hexadecimal format

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

greycol, femmecol, shadepalette, colorRamp for comparable R function

Examples

```
intpalette(c("white", "black"), n = 10)
grey(seq(1, 0, length.out = 10))
image(matrix(nrow = 1,ncol = 100, data = 1:100),
      col = intpalette(c("red", "blue"), numcol = 100),
      main = "intpalette")
image(matrix(nrow = 1, ncol = 100, data = 1:100),
      col = intpalette(c("red", "blue", "yellow"), numcol = 100),
      main = "intpalette")
```

---

plotcircle *adds part of a colored circle to a plot*

Description

adds (part of) a colored circle to a plot; an arrow can be drawn at a specified position

Usage

```
plotcircle(r = 1, ...)
```

Arguments

r	radius of circle.
...	arguments passed to function plotellipse.

Details

plotcircle calls plotellipse, making sure that the figure drawn effectively looks like a circle. For graphs that have both axes of equal size, the circle will be equal to the ellipse with equal rx and ry. See second example

see plotellipse for details

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

plotellipse to draw ellipses

Examples

```
# symmetrical axes
emptyplot(c(0, 1))
plotcircle(mid = c(0.5, 0.5), r = 0.25, from = 0, to = 3*pi/2,
           arrow = TRUE, arr.pos = 0.5, col = "red")
# symmetrical
plotellipse(mid = c(0.5, 0.5), rx = 0.2, ry = 0.2,
           arrow = TRUE, arr.pos = 0.5, col = "blue")


#non-symmetrical axes
emptyplot(c(0, 1), c(0, 2), main = "plotcircle", asp = FALSE)
plotcircle(mid = c(0.5, 0.5), r = 0.25, from = 0, to = 3*pi/2,
           arrow = TRUE, arr.pos = 0.5, col = "red")
plotellipse(mid = c(0.5, 0.5), rx = 0.25, ry = 0.25,
           arrow = TRUE, arr.pos = 0.5, col = "blue")
```

---

plotellipse *adds part of a colored and rotated ellipse to a plot*

Description

adds (part of) a colored, and rotated ellipse to a plot; an arrow can be drawn at a specified position.

Usage

```
plotellipse(rx = 1, ry = 0.2, mid = c(0,0), dr = 0.01,
            angle = 0, from = -pi, to = pi, type = "l", lwd = 2,
            lcol = "black", col = NULL, arrow = FALSE,
            arr.length = 0.4, arr.width = arr.length*0.5,
            arr.type = "curved", arr.pos = 1, arr.code = 2,
            arr.adj = 0.5, arr.col = "black", ...)
```

Arguments

rx	long radius of ellipse.
ry	short radius of ellipse.
mid	midpoint of ellipse.
dr	size of segments, in radians, to draw ellipse (decrease for smoother).
angle	rotation angle, degrees.
from	starting angle for ellipse segment, radians.
to	final angle for ellipse segment, radians.
type	external line or points; "n" if no line.
lwd	width of external line.
lcol	line color.

col	fill color.
arrow	drawing arrowhead yes/no.
arr.length	length of arrowhead.
arr.width	width of arrowhead.
arr.type	type of arrow.
arr.pos	position of arrow, 0=start,1=end.
arr.code	integer code determining kind of arrows to draw.
arr.adj	adjustment of arrow.
arr.col	color of arrow head.
...	arguments passed to R-function lines.

Details

rx and ry are the horizontal and vertical radiusses of the ellipses.

The ellipse is drawn from the point defined by from to the point defined as to which are joined anti-clockwise.

if arrow is TRUE, an arrow is drawn along the path of the ellipse.

arr.length and arr.width set the size of the arrow.

The type of the arrowhead is set with arr.type which can take the values:

- "simple" : uses comparable R function arrows.
- "triangle": uses filled triangle.
- "curved" : draws arrowhead with curved edges.
- "circle" : draws circular head.

arr.pos, a real value between 0 and 1 gives the position (0=start,1=end).

arr.col specifies the color, arr.code specifies where the angle points to.

arr.adj specifies the position adjustment - see Arrows for details.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

getellipse, filledellipse, plotcircle.

Examples

```
emptyplot(c(-1, 1), main = "plotellipse")
plotellipse(rx = 0.8, ry = 0.3, angle = 60, col = "blue")
plotellipse(rx = 1.0, ry = 0.6, angle = 0, from = pi, to = 2*pi,
            arrow = TRUE, arr.pos = seq(0.1, 0.5, by = 0.1),
            arr.col = rainbow(5))
plotellipse(rx = 1.0, ry = 0.6, angle = 30, from = pi, to = 1.2*pi,
```

```
            col = "red")
plotellipse(rx = 0.1, ry = 0.6, from = 1.5*pi, to = pi,
            lcol = "orange", mid = c(0.2,0.2))
plotellipse(rx = 0.1, ry = 0.6, angle = 30, from = 1.5*pi, to = pi,
            lcol = "orange", mid = c(0.2,0.2))
```

`rotatexy` *rotates 2-column matrix around a midpoint*

Description

rotates xy values around a midpoint; xy is either a 2-columned matrix or a 2-valued vector

Usage

```
rotatexy(xy, angle, mid = colMeans(xy), asp = FALSE)
```

Arguments

`xy`	matrix with 2 columns, or a 2-valued vector to be rotated.
`angle`	angle of rotation, in degrees.
`mid`	rotation point, default=centroid.
`asp`	if true: aspect ratio is kept.

Value

a 2-column matrix with rotated values

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

Examples

```
x <- seq(0, 2*pi, pi/100)
y <- sin(x)
cols <- intpalette(c("blue", "green", "yellow", "red"), n = 500)
cols <- c(cols,rev(cols))
plot(x, y, type = "l", ylim = c(-3, 3), main = "rotatexy",
     col = cols[1], lwd = 2)
for (i in 2:1000)
  lines(rotatexy( cbind(x, y), angle = 0.18*i),
        col = cols[i], lwd = 2)

cols <- femmecol(1000)
plot(x, y, xlim = c(-1, 1), ylim = c(-1, 1), main = "rotatexy",
     col = cols[1], type = "n")
for (i in 2:1000) {
```

```
  xy <- rotatexy(c(0, 1), angle = 0.36*i, mid = c(0,0))
  points(xy[1], xy[2], col = cols[i], pch = ".", cex = 2)
}
```

`roundrect` *adds a rounded rectangular box to a plot*

Description

adds a rectangular box with rounded left and right edges to a plot

Usage

```
roundrect(mid, radx, rady, rx = rady, dr = 0.01,
          col = "white", lcol = "black", lwd = 2, angle = 0, ...)
```

Arguments

`mid`	midpoint (x,y) of the box.
`radx`	horizontal radius of the box.
`rady`	vertical radius of the box.
`rx`	radius of rounded part.
`dr`	size of segments, in radians, to draw the rounded line (decrease for smoother).
`col`	fill color of the box.
`lcol`	line color surrounding box.
`lwd`	line width of line surrounding the box.
`angle`	rotation angle, degrees.
`...`	arguments passed to function filledshape.

Details

`radx` and `rady` are the horizontal and vertical radiusses of the box; `rx` is the horizontal radius of the rounded part.

Here `horizontal` and `vertical` denote the position BEFORE rotation.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

Examples

```
emptyplot(c(-0.1, 1.1), main = "roundrect")
for (i in 1:10)
  roundrect(mid = runif(2), col = i, radx = 0.1, rady = 0.05)
for (i in 1:5)
  roundrect(mid = runif(2), col = greycol(20), radx = 0.05,
            rady = 0.05, angle = runif(1)*360)
```

shadepalette *color palette inbetween two extremes*

Description

Returns color(s) that are a linear interpolation between two colors

these colors are suitable for shading shapes

Usage

```
shadepalette(n = 100, endcol = "red", inicol = "white",
             interval = c(0.0, 1.0))
```

Arguments

n	number of colors.
endcol	final color.
inicol	initial color.
interval	interval *to* where to interpolate.

Value

a vector of character strings giving the interpolated colors in hexadecimal format

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

See Also

intpalette, grey, femmecol colorRamp for comparable R functions.

Examples

```
shadepalette(n = 10, "white", "black")
image(matrix(nrow = 1, ncol = 100, data = 1:100),
      col = shadepalette(100, "red", "blue"), main = "shadepalette")
```

`textflag` *adds a filled rounded rectangular box with a text to a plot*

Description

adds a rectangular box with rounded left and right edges to a plot

Usage

```
textflag(mid, radx, rady, rx = rady, dr = 0.01,
         col = femmecol(100), lcol = "white",
         bcol = lcol, lwd = 2, angle = 0, lab = NULL,
         leftright = TRUE, tcol = NULL, ...)
```

Arguments

`mid`	midpoint (x,y) of the box.
`radx`	horizontal radius of the box.
`rady`	vertical radius of the box.
`rx`	radius of rounded part.
`dr`	size of segments, in radians, to draw the rounded line (decrease for smoother).
`col`	fill color of the box; the box will be filled from left to right.
`lcol`	line color surrounding box.
`bcol`	line color to remove the ellipse from the rectangular box.
`tcol`	text color.
`lwd`	line width of line surrounding the box.
`angle`	rotation angle, degrees.
`lab`	one label or a vector string of labels to be added in box.
`leftright`	if `TRUE` then coloring is from left to right else the coloring is from bottom to top box (for `angle` = 0).
`...`	other arguments passed to function text.

Details

`radx` and `rady` are the horizontal and vertical radiusses of the box; `rx` is the horizontal radius of the rounded part.

Here `horizontal` and `vertical` denote the position BEFORE rotation.

This function is similar to function roundrect, except that coloring is from left to right.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

Examples

```
emptyplot()
textflag(mid = c(0.5, 0.5), radx = 0.5, rady = 0.1,
         lcol = "white",  lab = "hello", cex = 5, font = 2:3)

textflag(mid = c(0.5, 0.15), radx = 0.5, rady = 0.1,
         rx = 0.3,lcol = "black", lab = "hello 2", cex = 4,
         font = 2, angle = 20, tcol = "darkblue")

textflag(mid = c(0.5, 0.85), radx = 0.5, rady = 0.1, rx = 0.03,
         lcol = "white", lab = "hello 3", cex = 4, font = 2,
         leftright = FALSE)
```

---

writelabel *adds a label next to a plot*

Description

adds one-character label on left-upper margin, next to a plot

Usage

```
writelabel(text = NULL, nr = 1, at = -0.1, line = 1, cex = 1.5, ...)
```

Arguments

text	text to write.
nr	integer; if text = NULL: nr is converted to uppercase letter.
at	relative distance of label position, from left margin of plot region.
line	line above the plot region of label position.
cex	relative size of label.
...	arguments passed to R-function mtext.

Author(s)

Karline Soetaert <karline.soetaert@nioz.nl>

Examples

```
plot(runif(2), main = "writelabel")
writelabel("A")
writelabel("B", at = 0)
writelabel("C", at = 1)
```

Index