

# Package ‘preplot’

July 23, 2025

**Version** 1.0-2

**Date** 2025-04-16

**Title** Prepare Figure Region for Base Graphics

**Imports** graphics, grDevices, plotrix, shape

**Suggests** bookdown, knitr, rmarkdown, ggplot2, gridBase

**VignetteBuilder** knitr

**Description** A figure region is prepared, creating a plot region with suitable background color, grid lines or shadings, and providing axes and labeling if not suppressed. Subsequently, information carrying graphics elements can be added (points, lines, barplot with add=TRUE and so forth).

**License** GPL (>= 2)

**NeedsCompilation** no

**Author** Ulrike Groemping [aut, cre]

**Maintainer** Ulrike Groemping <ulrike.groemping@bht-berlin.de>

**Repository** CRAN

**Date/Publication** 2025-04-17 07:40:02 UTC

## Contents

preplot . . . . .	1
<b>Index</b>	<b>8</b>

---

preplot	<i>Functions to prepare a figure region for base graphics</i>
---------	---

---

### Description

Function preplot prepares a figure region according to individual preferences regarding background color, axes, stripes and/or gridlines; optionally, labelling can be done with the function. Function stripes draws the vertical or horizontal stripes for preplot and can also be used independently.

**Usage**

```

preplot(xlim, ylim,
        bg = par("bg"), xaxs=NULL, yaxs=NULL,
        lwd.axis = 0, col.axis = "grey20",
        border = ifelse(lwd.axis == 0, bg, col.axis),
        axes = TRUE, xaxt = par("xaxt"), yaxt = par("yaxt"),
        xticks = NULL, yticks = NULL,
        xticklabs = xticks, yticklabs = yticks,
        mgpx = par("mgp"), xaxside = 1,
        mgpy = mgpx, yaxside = 2,
        xlab = NULL, ylab = NULL,
        cex = par("cex"), las = 1, lasx = las, lasy = las,
        gridx = FALSE, gridy = FALSE,
        gridxminor = 0, gridyminor = 0,
        lty.grid = ifelse(max(gridxminor, gridyminor) > 0,
                          "solid", "dotted"),
        col.grid = "grey50",
        lwd = par("lwd"), lwd.grid = lwd,
        lty.grid.minor = "dotted",
        col.grid.minor = col.grid,
        lwd.grid.minor = 0.5 * lwd.grid,
        stripesx = FALSE, stripesy = FALSE,
        col.stripes = "grey90",
        axis.arrow = FALSE,
        arrow.length = 0.3, arrow.width = 0.2,
        arrow.code = 2, arrow.type = "triangle", ...)

stripes(stripesx = numeric(0), stripesy = numeric(0), col.stripes = "grey90",
        usr=NULL, ...)

```

**Arguments**

<code>xlim</code>	required; a numeric vector of length 2 or more, whose range will be used for the horizontal axis limits
<code>ylim</code>	required; a numeric vector of length 2 or more, whose range will be used for the vertical axis limits
<code>bg</code>	background color
<code>xaxs</code>	"r" for regular style (4% extended versus <code>range(xlim)</code> ), "i" for using unextended <code>range(xlim)</code> NULL for using <code>par("xaxs")</code> , which has default "r"
<code>yaxs</code>	analogous to <code>xaxs</code>
<code>lwd.axis</code>	axis line width; default 0 for no axis line (see below also for <code>lwd</code> and <code>lwd.grid</code> )
<code>col.axis</code>	axis color; also affects ticks, tickmark labels and axis labels
<code>border</code>	border color; default depends on whether or not there is an axis line
<code>axes</code>	logical that determines whether or not axes are to be drawn (only if <code>xaxt</code> and <code>yaxt</code> do not suppress their respective axes); default: TRUE; if FALSE, both tick

mark labels and axis labels are suppressed; even if TRUE, an axis line is only drawn, if `lwd.axis` is set to a positive value; if the default axis positioning of R itself is desired, choose `axes=FALSE`, and add an `axis` statement after the `preplot` call (see also the Examples section)

<code>xaxt</code>	default: <code>par("xaxt")</code> ; set to "n" for suppressing the x axis in spite of " <code>axes=TRUE</code> " (you can add a horizontal axis with the <code>axis</code> command later on, see examples section)
<code>yaxt</code>	default: <code>=par("yaxt")</code> ; set to "n" for suppressing the y axis in spite of " <code>axes=TRUE</code> " (you can add a vertical axis with the <code>axis</code> command later on, see examples section)
<code>xticks</code>	tick positions for the horizontal axis; default NULL (see Details section)
<code>yticks</code>	tick positions for the vertical axis; default NULL (see Details section)
<code>xticklabs</code>	tick labels for the horizontal axis; default <code>xticks</code> ; if explicitly provided, must have the same length as the <code>xticks</code> values derived from <code>xticks</code> , <code>gridx</code> or <code>stripesx</code> (see Details section)
<code>yticklabs</code>	tick labels for the vertical axis; default <code>yticks</code> ; if explicitly provided, must have the same length as the <code>yticks</code> values derived from <code>yticks</code> , <code>gridy</code> or <code>stripesy</code> (see Details section)
<code>mgpx</code>	position of the horizontal axis label, tick marks and line in terms of margin lines (default: <code>par("mgp")</code> ); see Details section
<code>xaxside</code>	side of the horizontal axis; 1 for bottom (default), 3 for top
<code>mgpy</code>	analogous to <code>mgpx</code> , for vertical axis; default: equal to <code>mgpx</code>
<code>yaxside</code>	side of the vertical axis; 2 for left (default), 4 for right
<code>xlab</code>	label for horizontal axis; default: empty
<code>ylab</code>	label for vertical axis; default: empty
<code>cex</code>	general annotation size; default <code>par("cex")</code> ; if specified, it also adapts defaults for <code>cex.main</code> , <code>cex.sub</code> , <code>cex.axis</code> and <code>cex.lab</code> , multiplying them with <code>cex/par("cex")</code> ; these can be overruled by explicit specification in ...
<code>las</code>	direction of axis labeling; default: parallel to horizontal axis; you may want to change this to 0 (parallel to axes) in case of longer y-axis labels
<code>lasx</code>	direction of tick labels for horizontal axis; default: equal to <code>las</code> ; note that <code>xlab</code> remains parallel to the axis, regardless of the choice for <code>las</code> (can only be changed by adding it later with <code>mtext</code> )
<code>lasy</code>	direction of tick labels for vertical axis; default: equal to <code>las</code> ; note that <code>ylab</code> remains parallel to the axis, regardless of the choice for <code>las</code> (can only be changed by adding it later with <code>mtext</code> )
<code>gridx</code>	logical or numeric; if TRUE, draws grid lines at the tick mark positions; if numeric, number of grid lines between the <code>xlim</code> values or numeric vector of grid line positions for horizontal axis; default: FALSE for no grid lines (positive <code>gridxminor</code> implies switch to <code>gridx=TRUE</code> )
<code>gridy</code>	logical or numeric; if TRUE, draws grid lines at the tick mark positions; if numeric, number of grid lines between the <code>ylim</code> values or numeric vector of grid line positions for vertical axis; default: FALSE for no grid lines (positive <code>gridyminor</code> implies switch to <code>gridy=TRUE</code> )

<code>gridxminor</code>	number of minor grid lines between pairs of major grid lines for horizontal axis; default: 0 for no minor grid lines
<code>gridyminor</code>	number of minor grid lines between pairs of major grid lines for vertical axis; default: 0 for no minor grid lines
<code>col.grid</code>	color for grid lines
<code>lty.grid</code>	line type for (major) grid lines; defaults to a dotted line with no minor grid lines and a solid line with minor grid lines (regardless of the direction of the minor grid lines)
<code>lwd</code>	line width (for grid lines), default is from the current <code>par</code> setting
<code>lwd.grid</code>	line width for grid lines, default is <code>lwd</code>
<code>col.grid.minor</code>	color for minor grid lines
<code>lty.grid.minor</code>	line type for minor grid lines; default: dotted line
<code>lwd.grid.minor</code>	line width for minor grid lines (default: $0.5 * \text{lwd.grid}$ )
<code>stripesx</code>	logical (default FALSE) or numeric vector of color change positions in striped background (vertical stripes with reference to horizontal axis); see Details section for further information; default: no stripes
<code>stripesy</code>	logical (default FALSE) or numeric vector of color change positions in striped background (horizontal stripes with reference to vertical axis); increasing size is assumed (or ascertained by sorting); see Details section for further information; default: no stripes
<code>col.stripes</code>	color of the stripes
<code>axis.arrow</code>	logical (default FALSE) or numeric vector of two positions at which to draw axis arrows for vertical and horizontal axis (first value: y position for horizontal axis, second value: x position for vertical axis); if set to TRUE, the positions are the values for <code>xaxpos</code> and <code>yaxpos</code> , respectively; for one axis arrow only, use NA for the other element
<code>arrow.length</code>	numeric arrow length for function <code>Arrows</code> ; default: 0.3
<code>arrow.width</code>	numeric arrow width for function <code>Arrows</code> ; default: 0.2
<code>arrow.type</code>	character arrow type for function <code>Arrows</code> ; default: "triangle"
<code>arrow.code</code>	default: 2 (y axis arrow upwards, x axis arrow to the right); for other choices see Details section of <code>Arrows</code>
<code>usr</code>	NULL (default), or a numeric vector with limits of the area within which to draw the stripes; the default amounts to <code>par("usr")</code>
<code>...</code>	further arguments given to functions <code>plot</code> , <code>mtext</code> , <code>stripes</code> , <code>axis</code> and <code>Arrows</code> ; these can e.g. include the option <code>tcl</code> for tick length.

## Details

`preplot` supports the preparation of a customized plot region to which the information carrying graphical elements can be added. It can be used with all functions that allow adding to existing base graphics plots (e.g. `points`, `lines`, `barplot`, `rect`, `symbols`, ...). Usage with other functions is also possible, but requires careful application after setting `par(new=TRUE)`.

If an axis is not suppressed (by `axes=FALSE` or `xaxt="n"` or `yaxt="n"`), axis ticks are placed at the positions specified in `xticks` or `yticks`; if these are `NULL`, numeric specifications for `gridx` or `gridy` determine the tick positions, and if these are also non-existent, numeric specifications for `stripesx` or `stripesy` determine the tick positions. If all these are unspecified, the tick positions are determined from the default axis behavior using function `axisTicks` (with option `log=FALSE`). Per default, axis lines (whether visible or width 0) meet at the position `(xlim[1], ylim[1])` (assuming 2-element limits, which are determined from longer `*lim` arguments with the `range` function, where needed); options `mgpx` or `mgpy` can be used for moving the axis line, tick labels and/or axis labels inwards or outwards; they default to the settings in `par("mgp")`, and it may be convenient to change that setting rather than using the option in several `preplot` calls (see also Example section). Instead of specifying the labelling with function `preplot`, it can also be handled by subsequent `title` and/or `axis` statements.

For grid lines, it is possible to provide minor grid lines (by specifying the number of minor grid lines between major ones); it is also possible to specify major grid lines in complete independence of the tick mark positions, e.g. for displaying information on regulatory limits, target values or specific events (on a time axis).

For stripes, specifying `TRUE` uses `xticks` or `yticks` values for the creation of stripes. Elements of stripes vectors that are outside their respective axis limits are silently moved to the nearest limit of the plot area (i.e. to the suitable element of `usr`). Note that the sorting of stripe entries should correspond to the sorting of axis limits (i.e., e.g., if `xlim[1]>xlim[2]`, sorting is decreasing instead of increasing); elements of `stripesx` are sorted to conform to this rule, and duplicates are removed. If the remaining vector `stripesx` has an even number of elements, `length(stripesx/2)` vertical stripes are drawn between pairs of neighbouring `stripesx` elements. Otherwise, the handling depends on the first and last (sorted) element of `stripesx`: if the first (sorted) element equals `xlim[1]`, a narrow stripe in the beginning is drawn, and the remaining even number of `stripesx` elements is treated in pairs as before; otherwise, if the last (sorted) element equals `xlim[2]`, a narrow stripe in the end is drawn; if neither the first nor the last element coincides with an element of `xlim`, the last element of `stripesx` is simply omitted. `stripesy` is treated analogously to `stripesx`. Like gridlines, stripes can be completely independent of tick marks (see e.g. the last example).

### Value

The function does not return anything; it is called for its side effects.

### Author(s)

Ulrike Groemping

### References

- Murrell, P. (2011). R graphics. CRC, Boca Raton. <https://www.stat.auckland.ac.nz/~paul/RG2e/>
- Rahlf, T. (2017). Data visualisation with R. Springer International Publishing AG, Cham.

### See Also

See also [Arrows](#) for arrows.

**Examples**

```

## default
preplot(0:10, -5:5)
preplot(0:10, -5:5, xaxs="i", yaxs="i")
## with stripes and grid based on default tick positions
preplot(0:10, -5:5, stripesy=TRUE, gridx=TRUE)

## with white background,
## axis lines and small ticks,
## major and minor grid for y,
## and plot area defined by axis limits
## instead of default usr coordinates
## (border is drawn because of lwd.axis)
## mgpx moves tick position labels closer to axes
preplot(0:10, -5:5, bg="white", xaxs="i", yaxs="i",
        lwd.axis = 1,
        gridy=c(-5,0,5), gridyminor=4,
        tcl=-0.2, mgpx=c(3,0.5,0))

## without axis lines but with default background
## looks better with bg.area="lim"
## unless actual data points extend to the limits
preplot(0:10, -5:5, yticks=seq(-5,5,5),
        gridy=-5:5, gridx=TRUE,
        xaxs="i", yaxs="i")

## with axis arrows
## narrower margins
## small tick marks
## tick annotations close to axis
par(mar=c(3,3,2,1), mgp=c(2,0.35,0))
preplot(0:10, -5:5, yticks=seq(-5,5,5),
        gridy=-5:5, gridx=TRUE, lwd.axis=1, tcl=-0.2,
        border="grey92", axis.arrow=TRUE)
dev.off() ## eliminates modified par settings

## xaxs and yaxs set in par
## labeling subsequently or in preplot
## mgp or mgpx used for moving labeling closer to axis
par(mfrow=c(1,2), xaxs="i", yaxs="i")
## adding labeling subsequently
par(mgp=c(2.25,0.75,0))
preplot(0:10, -5:5, yticks=seq(-5,5,5),
        gridy=-5:5, gridx=TRUE)
title(xlab="x axis label", ylab="y axis label",
      sub="Labeling added subsequently", main="mgp set in par")
## adding labeling subsequently
par(mgp=c(3,1,0)) # back to default
## adding labeling within the function
preplot(0:10, -5:5, yticks=seq(-5,5,5),
        gridy=-5:5, gridx=TRUE,
        xlab="x axis label", ylab="y axis label",

```

```
      mgpx=c(2.25,0.75,0),
      main="mgpx set in preplot",
      sub="Labeling added within preplot")
## the difference: sub reacts to mgp, not to mgpx
dev.off() ## eliminates modified par settings

## further examples in the pdf vignette
## access with vignette("preplotOverview")
```

# Index

\* **aplot**

preplot, 1

\* **hplot**

preplot, 1

Arrows, 4, 5

axisTicks, 5

mtext, 3

par, 4

preplot, 1

stripes (preplot), 1