Package 'metaplot'

February 20, 2018

Type Package

Title Data-Driven Diagnostic Plots

Version 0.4.4

Author Tim Bergsma

Maintainer Tim Bergsma <bergsmat@gmail.com>

Description Designs plots in terms of core structure. See 'example(metaplot)'. Primary arguments are (unquoted) column names; order and type (numeric or not) dictate the resulting plot. Specify any y variables, x variable, any groups variable, and any conditioning variables to metaplot() to generate density plots, boxplots, mosaic plots, scatterplots, scatterplot matrices, or conditioned plots. Use multiplot() to arrange plots in grids. Wherever present, scalar column attributes 'label' and 'guide' are honored, producing fully annotated plots with minimal effort. Attribute 'guide' is typically units, but may be encoded() to provide interpretations of categorical values (see '?encode'). Utility unpack() transforms scalar column attributes to row values and pack() does the reverse, supporting tool-neutral storage of metadata along with primary data. The package supports customizable aesthetics such as such as reference lines, unity lines, smooths, log transformation, and linear fits. Compact syntax and integrated metadata promote workflow scalability.

Imports encode (>= 0.3.2), lattice, magrittr, dplyr (>= 0.7.1), tidyr, rlang, grid

Suggests csv, nlme

Depends R (>= 2.10)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-20 08:43:17 UTC

R topics documented:

`boxplot` *Boxplots*

Description

Boxplots

boxplot.data.frame *Boxplot Method for Data Frame*

Description

Boxplot for data.frame. Parses arguments and generates the call: fun(x, yvar, xvar, facets, ...).

Usage

```
## S3 method for class 'data.frame'
boxplot(x, ..., fun = getOption("metaplot_box",
  "boxplot_data_frame"))
```

Arguments

x	data.frame
...	passed to fun
fun	function that does the actual plotting

See Also

Other mixedvariate plots: boxplot_data_frame, boxplot_panel

Other boxplot: boxplot_data_frame

Other methods: axislabel.data.frame, categorical.data.frame, corsplom.data.frame, densplot.data.frame, metaplot.data.frame, pack.data.frame, scatter.data.frame, unpack.data.frame

Examples

```
library(dplyr)
library(magrittr)
Theoph %<>% mutate(site = ifelse(as.numeric(Subject) > 6, 'Site A','Site B'))
boxplot(Theoph,'Subject','conc')
boxplot(Theoph,Subject,conc)
boxplot(Theoph,conc,Subject)
boxplot(Theoph,conc,Subject,site)
attr(Theoph,'title') <- 'Theophylline'
boxplot(Theoph, Subject, conc, main = function(x,...)attr(x,'title'))
boxplot(Theoph, Subject, conc, sub= function(x,...)attr(x,'title'))
```

boxplot_data_frame *Boxplot Function for Data Frame*

Description

Boxplot for data.frame. Creates a boxplot using `boxplot_panel` by default.

Usage

```
boxplot_data_frame(x, yvar, xvar, facets = NULL,
  log = getOption("metaplot_log", FALSE), crit = getOption("metaplot_crit",
  1.3), horizontal = getOption("metaplot_horizontal", NULL),
  panel = getOption("metaplot_boxplot_panel", boxplot_panel),
  ref = getOption("metaplot_ref", metaplot_ref),
  ref.col = getOption("metaplot_ref.col", "grey"),
  ref.lty = getOption("metaplot_ref.lty", "solid"),
  ref.alpha = getOption("metaplot_ref.alpha", 1),
  nobs = getOption("metaplot_nobs", FALSE),
  na.rm = getOption("metaplot_na.rm", TRUE), xlab = NULL, ylab = NULL,
  numlab = getOption("metaplot_lab", axislabel),
  catlab = getOption("metaplot_lab", axislabel),
  aspect = getOption("metaplot_aspect", 1),
  main = getOption("metaplot_main", NULL), sub = getOption("metaplot_sub",
  NULL), par.settings = standard.theme("pdf", color = FALSE), ...)
```

Arguments

x	data.frame
yvar	y variable
xvar	x variable
facets	optional conditioning variables
log	whether to log transform numeric variable (auto-selected if NA)
crit	if log is NA, log-transform if mean/median ratio for non-missing values is greater than this value
horizontal	whether box/whisker axis should be horizontal (numeric x, categorical y); defaults TRUE if (var[[2]] is numeric
panel	panel function
ref	optional reference line(s) on numeric axis; can be function(x = x, var = con, ...)
ref.col	color for reference line(s)
ref.lty	line type for reference line(s)
ref.alpha	transparency for reference line(s)
nobs	whether to include the number of observations under the category label
na.rm	whether to remove data points with one or more missing coordinates

xlab	x axis label
ylab	y axis label
numlab	numeric axis label; can be function(x = x, var = numvar, log = ylog, ...)
catlab	categorical axis label; can be function(x = x, var = catvar, ...)
aspect	passed to bwplot
main	character, or a function of x, yvar, xvar, facets, and log
sub	character, or a function of x, yvar, xvar, facets, and log
par.settings	default parameter settings
...	passed arguments

See Also

Other mixedvariate plots: boxplot.data.frame, boxplot_panel

Other boxplot: boxplot.data.frame

Other metaplot: categorical_data_frame, corsplom_data_frame, densplot_data_frame, metaplot, scatter_data_frame

Examples

```
library(magrittr)
library(dplyr)
boxplot_data_frame(Theoph,'Subject','conc')
boxplot_data_frame(Theoph %>% filter(conc > 0),
'conc','Subject', log = TRUE, ref = c(2,5),horizontal = FALSE)
```

---

categorical *Categorical Plot*

Description

Categorical Plot. Generic, with method for 'data.frame'.

Usage

```
categorical(x, ...)
```

Arguments

x	object of dispatch
...	passed arugments

See Also

Other generic functions: axislabel, corsplom, densplot, metaplot, pack, scatter, unpack

Other categorical: categorical.data.frame, categorical_data_frame, categorical_panel, panel_tile

categorical.data.frame

Categorical Method for Data Frame

Description

Categorical method for 'data.frame'.

Usage

```
## S3 method for class 'data.frame'
categorical(x, ...,
  fun = getOption("metaplot_categorical", "categorical_data_frame"))
```

Arguments

x	data.frame
...	other arguments
fun	function to draw the plot

Value

character

See Also

Other categorical: categorical_data_frame, categorical_panel, categorical, panel_tile

Other methods: axislabel.data.frame, boxplot.data.frame, corsplom.data.frame, densplot.data.frame, metaplot.data.frame, pack.data.frame, scatter.data.frame, unpack.data.frame

categorical_data_frame

Categorical Function for Data Frame

Description

Categorical function for class 'data.frame'. Default panel function implements a simple mosaic plot.

Usage

```
categorical_data_frame(x, yvar = NULL, xvar, groups = NULL, facets = NULL,
  ylab = getOption("metaplot_lab", axislabel),
  xlab = getOption("metaplot_lab", axislabel),
  na.rm = getOption("metaplot_na.rm", TRUE),
  aspect = getOption("metaplot_aspect", 1),
  auto.key = getOption("metaplot_auto.key", NULL),
  keycols = getOption("metaplot_keycols", NULL),
  as.table = getOption("metaplot_categorical_as.table", TRUE),
  prepanel = getOption("metaplot_categorical_prepanel", function(...)
  list(xlim = 0:1, ylim = 0:1)),
  scales = getOption("metaplot_categorical_scales", NULL),
  panel = getOption("metaplot_categorical_panel", categorical_panel),
  colors = getOption("metaplot_colors", NULL),
  tiles = getOption("metaplot_tiles", 0.5),
  lines = getOption("metaplot_lines", TRUE),
  main = getOption("metaplot_main", NULL), sub = getOption("metaplot_sub",
  NULL), tex = getOption("metaplot_tex", 0.9),
  pch = getOption("metaplot_categorical_pch", 22),
  rot = getOption("metaplot_categorical_rot", c(90, 0)), subscripts = TRUE,
  par.settings = NULL, ...)
```

Arguments

x	data.frame
yvar	character: y variable (optional)
xvar	character: x variable
groups	optional grouping variable (can be missing)
facets	optional conditioning variables
ylab	y axis label; can be function(x = x, var = yvar, ..)
xlab	x axis label; can be function(x = x, var = xvar, ..)
na.rm	whether to remove data points with one or more missing coordinates
aspect	passed to xyplot
auto.key	passed to xyplot
keycols	number of auto.key columns
as.table	passed to xyplot
prepanel	passed to xyplot (guessed if NULL)
scales	passed to xyplot (guessed if NULL)
panel	name or definition of panel function
colors	replacements for default colors in group order
tiles	whether to fill rectangles for each group: logical, or alpha values between 0 and 1
lines	whether to plot borders for each group: logical, or alpha values between 0 and 1

main	character, or a function of x, yvar, xvar, groups, facets
sub	character, or a function of x, yvar, xvar, groups, facets
tex	tile expansion: scale factor for reducing each tile size relative to full size (<= 1)
pch	symbol character for legend
rot	rotation for axis labels; can be length 2 for y and x axes, respectively
subscripts	passed to xyplot
par.settings	passed to xyplot (calculated if null)
...	passed to region

See Also

categorical_panel

Other categorical: categorical.data.frame, categorical_panel, categorical, panel_tile

Other metaplot: boxplot_data_frame, corsplom_data_frame, densplot_data_frame, metaplot, scatter_data_frame

Examples

```
library(magrittr)
library(dplyr)
library(csv)
x <- as.csv(system.file(package = 'metaplot', 'extdata/theoph.csv'))
x %<>% pack
x %>% metaplot(site)
x %>% metaplot(arm, site)
x %>% metaplot(arm, site, cohort)
x %>% metaplot(arm, site, , cohort)
x %>% metaplot(arm, site, , cohort, rot = c(0,90))
x %>% metaplot(arm, site, , cohort, rot = c(45, 45))
x %>% metaplot(subject,cohort,arm, site, lines = F, rot = c(45,45))
```

---

categorical_panel *Panel Function for Metaplot Categorical Plot*

Description

Default panel function for categorical_data_frame. Implements a simple mosaic plot.

Usage

```
categorical_panel(x, y, groups, bivariate = TRUE,
  loc = getOption("metaplot_loc", 5), msg = getOption("metaplot_msg",
  "tilestats"), tex = getOption("metaplot_tex", 0.9),
  cex = getOption("metaplot_categorical_panel_cex", 1),
  pch = getOption("metaplot_categorical_pch", 22),
  rot = getOption("metaplot_categorical_rot", c(90, 0)), subscripts, ...)
```

Arguments

x	x values
y	y values
groups	optional grouping item
bivariate	whether to create y axis
loc	where to print statistics in a tile
msg	a function of x and y to print text in a tile
tex	tile expansion: scale factor for reducing each tile size relative to full size (<= 1)
cex	expansion for msg text
pch	symbol character for legend
rot	rotation for axis labels; can be length 2 for y and x axes, respectively
subscripts	subscripts of the original data for this panel
...	passed to link[lattice]{panel.superpose}

See Also

tilestats

categorical.data.frame

Other panel functions: boxplot_panel, corsplom_panel_correlation, corsplom_panel_diagonal, corsplom_panel_scatter, dens_panel, diag_label, diag_pin, iso_prepanel, metaplot_ref, panel_tile, scatter_panel_ref, scatter_panel

Other categorical: categorical.data.frame, categorical_data_frame, categorical, panel_tile

---

corsplom *Correlated Splom*

Description

Scatterplot matrix with correlations.

Usage

```
corsplom(x, ...)
```

Arguments

x	object
...	passed arguments

See Also

Other generic functions: axislabel, categorical, densplot, metaplot, pack, scatter, unpack

Other corsplom: corsplom.data.frame, corsplom_data_frame

corsplom.data.frame *Correlated Scatterplot Matrix Method for Data Frame*

Description

Creates a scatterplot matrix. Parses arguments and generates the call: fun(x, xvar, ...).

Usage

```
## S3 method for class 'data.frame'
corsplom(x, ..., fun = getOption("metaplot_corsplom",
  "corsplom_data_frame"))
```

Arguments

x	data.frame
...	passed to fun
fun	function to do the actual plotting

See Also

Other multivariate plots: corsplom_data_frame, metaplot.data.frame

Other corsplom: corsplom_data_frame, corsplom

Other methods: axislabel.data.frame, boxplot.data.frame, categorical.data.frame, densplot.data.frame, metaplot.data.frame, pack.data.frame, scatter.data.frame, unpack.data.frame

corsplom_data_frame *Correlated Scatterplot Matrix Function for Data Frame*

Description

Creates a scatterplot matrix with correlations in lower panel, by default.

Usage

```
corsplom_data_frame(x, xvar = names(x),
  upper.panel = getOption("metaplot_upper.panel", corsplom_panel_scatter),
  lower.panel = getOption("metaplot_lower.panel", corsplom_panel_correlation),
  diag.panel = getOption("metaplot_diag.panel", corsplom_panel_diagonal),
  pscales = getOption("metaplot_pscales", 0),
  xlab = getOption("metaplot_corsplom_xlab", NULL),
  varname.cex = getOption("metaplot_varname.cex", 1),
  main = getOption("metaplot_main", NULL), sub = getOption("metaplot_sub",
  NULL), ...)
```

Arguments

x	data.frame
xvar	variables to plot
upper.panel	passed to splom
lower.panel	passed to splom
diag.panel	passed to splom
pscales	passed to splom
xlab	passed to splom, can be function(x = x, var = xvar, ...)
varname.cex	passed to splom
main	character, or a function of x, xvar
sub	character, or a function of x, xvar
...	extra arguments passed to splom

See Also

Other multivariate plots: corsplom.data.frame, metaplot.data.frame

Other corsplom: corsplom.data.frame, corsplom

Other metaplot: boxplot_data_frame, categorical_data_frame, densplot_data_frame, metaplot, scatter_data_frame

---

densplot *Density Plot*

Description

Creates a density plot.

Usage

```
densplot(x, ...)
```

Arguments

x	object
...	passed arguments

See Also

Other generic functions: axislabel, categorical, corsplom, metaplot, pack, scatter, unpack

Other univariate plots: dens_panel, densplot.data.frame, densplot_data_frame, metaplot.data.frame

Other densplot: densplot.data.frame, densplot_data_frame

densplot.data.frame *Densplot Method for Data Frame*

Description

Plot density for object of class 'data.frame'. Parses arguments and generates the call: fun(x, xvar, groups, facets,...).

Usage

```
## S3 method for class 'data.frame'
densplot(x, ..., fun = getOption("metaplot_dens",
  "densplot_data_frame"))
```

Arguments

x	data.frame
...	passed to fun
fun	plotting function

See Also

Other univariate plots: dens_panel, densplot_data_frame, densplot, metaplot.data.frame

Other densplot: densplot_data_frame, densplot

Other methods: axislabel.data.frame, boxplot.data.frame, categorical.data.frame, corsplom.data.frame, metaplot.data.frame, pack.data.frame, scatter.data.frame, unpack.data.frame

Examples

```
densplot(Theoph, conc, grid = TRUE )
densplot(Theoph, conc, Subject )
densplot(Theoph, conc, , Subject )
attr(Theoph,'title') <- 'Theophylline'
densplot(Theoph, conc, main= function(x,...)attr(x,'title'))
densplot(Theoph, conc, sub= function(x,...)attr(x,'title'))
```

densplot_data_frame *Density Function for Data Frame*

Description

Plot density for object of class 'data.frame' using dens_panel by default.

Usage

```
densplot_data_frame(x, xvar, groups = NULL, facets = NULL,
  xlab = getOption("metaplot_lab", axislabel),
  ref = getOption("metaplot_ref", metaplot_ref),
  ref.col = getOption("metaplot_ref.col", "grey"),
  ref.lty = getOption("metaplot_ref.lty", "solid"),
  ref.alpha = getOption("metaplot_ref.alpha", 1),
  log = getOption("metaplot_log", FALSE), crit = getOption("metaplot_crit",
  1.3), aspect = getOption("metaplot_aspect", 1),
  scales = getOption("metaplot_dens.scales", NULL),
  panel = getOption("metaplot_dens.panel", dens_panel),
  auto.key = getOption("metaplot_auto.key", NULL),
  keycols = getOption("metaplot_keycols", NULL),
  main = getOption("metaplot_main", NULL), sub = getOption("metaplot_sub",
  NULL), ...)
```

Arguments

x	data.frame
xvar	variable to plot
groups	optional grouping variable
facets	optional conditioning variables
xlab	x axis label; can be function(x = x, var = xvar, log = log, ...)
ref	reference line; can be function(x = x, var = xvar, ...)
ref.col	color for reference line(s)
ref.lty	type for reference line(s)
ref.alpha	transparency for reference line(s)
log	whether to log-transform x axis (auto-selected if NA)
crit	if log is NA, log-transform if mean/median ratio for non-missing x is greater than this value (and no negative values)
aspect	passed to densityplot
scales	passed to densityplot
panel	passed to densityplot
auto.key	passed to densityplot
keycols	number of auto.key columns
main	character, or a function of x, xvar, groups, facets, and log
sub	character, or a function of x, xvar, groups, facets, and log
...	passed to densityplot

See Also

Other univariate plots: dens_panel, densplot.data.frame, densplot, metaplot.data.frame

Other densplot: densplot.data.frame, densplot

Other metaplot: boxplot_data_frame, categorical_data_frame, corsplom_data_frame, metaplot, scatter_data_frame

Examples

```
densplot_data_frame(Theoph, 'conc', grid = TRUE)
densplot_data_frame(Theoph, 'conc', 'Subject')
densplot_data_frame(Theoph, 'conc', , 'Subject')
```

`diag_label` *Format a Diagonal Label*

Description

Formats a diagonal label. Can return a simple column name, a column label (if attribute defined), a fractured column label (split on spaces), or a processed symbol (over-rides label).

Usage

```
diag_label(varname, .data,
  diag_label_simple = getOption("metaplot_diag_label_simple", FALSE),
  diag_label_split = getOption("metaplot_diag_label_split", TRUE),
  diag_symbol_format = getOption("metaplot_diag_symbol_format",
  "wikisym2plotmath"), ...)
```

Arguments

`varname`	character
`.data`	data.frame
`diag_label_simple`	logical: just return varname?
`diag_label_split`	whether to substitute line breaks for spaces
`diag_symbol_format`	function to process symbol attribute, if present
`...`	ignored

Value

character

See Also

Other panel functions: `boxplot_panel`, `categorical_panel`, `corsplom_panel_correlation`, `corsplom_panel_diagonal`, `corsplom_panel_scatter`, `dens_panel`, `diag_pin`, `iso_prepanel`, `metaplot_ref`, `panel_tile`, `scatter_panel_ref`, `scatter_panel`

Other formatters: `wikisym2plotmath_`, `wikisym2plotmath`

diag_pin *Calculate Pin Placement*

Description

Calculates pin placement in the density region, inside margin of diagonal panels.

Usage

```
diag_pin(x, varname, .data, ...)
```

Arguments

x	vector of data
varname	name of vector in .data
.data	original dataset, possibly with column attributes such as 'reference'
...	passed arguments

Value

numeric

See Also

Other panel functions: boxplot_panel, categorical_panel, corsplom_panel_correlation, corsplom_panel_diagonal, corsplom_panel_scatter, dens_panel, diag_label, iso_prepanel, metaplot_ref, panel_tile, scatter_panel_ref, scatter_panel

Other reference lines: metaplot_ref, scatter_panel_ref

metaplot *Metaplot*

Description

Metaplot creates univariate, bivariate, or multivariate plots depending on the number and types of variables represented by the anonymous arguments. Types are either numeric (NUM, e.g. real, integer) or categorical (CAT, e.g. factor, character). A variable stored as numeric that nonetheless has an encoded guide attribute will be treated as categorical. Mnemonic: `x %>% metaplot(yvars, xvar, groupvar, facets)` where arguments are unquoted column names, and only xvar is required. Column attributes label, guide, reference, and symbol modify the behavior of the default handlers.

Usage

```
metaplot(x, ...)
```

Arguments

x	object
...	passed arguments

Details

Design your plot by specifying y variables (optional), the x variable, the groups variable (optional) and the conditioning variables (i.e., facets, optional).

The single groups variable, if any, is the first categorical in the third position or later. An earlier categorical gives a "mixed" bivariate plot or mosaic plot, depending on the type of the remaining variable.

The x variable is the last variable before groups, if present.

The y variables are those before x. If none, the result is univariate. If one, the result is typically a boxplot or scatterplot, depending on x. Several numeric y followed by a numeric x are treated as multivariate (scatterplot matrix). But if all y have the same `guide` attribute and it is different from that for x, the result is bivariate (i.e, an `overlay` scatterplot).

A single categorical variable results in a simple mosaic plot (see `link[graphics]{mosaicplot}` and **vcd** for more sophisticated treatment). Mosaic plots support only a single y variable; thus, whenever the first two variables are categorical, a two-way mosaic plot results, with remaining variables understood as groups and facets.

Wherever a groups argument is meaningful, it may be missing. This allows specification of facets in the absence of groups, e.g., `(metaplot(y, x, , facet1, facet2))`. For multiple y (overlay), the sources of y are the implied groups: any trailing categorical arguments are treated as facets.

Template designs follow; substitute behaviors by setting global options (see argument list).

- NUM: univariate (densityplot)
- CAT: categorical (one-way mosaic plot)
- CAT, CAT: categorical (two-way mosaic plot)
- CAT, CAT, CAT:grouped mosaic
- CAT, CAT, CAT, CAT:grouped mosaic with one facet
- CAT, CAT, CAT,, CAT:non-grouped mosaic with one facet
- NUM, CAT: mixedvariate (vertical boxplot)
- CAT, NUM: mixedvariate (horizontal boxplot)
- CAT, NUM, CAT: mixedvariate with one facet
- NUM, NUM: bivariate (scatterplot)
- NUM, NUM, CAT: grouped bivariate (grouped scatterplot)
- NUM, NUM,, CAT: non-grouped bivariate with one facet
- NUM, NUM, CAT, CAT: grouped bivariate with one facet
- NUM, NUM, CAT, CAT, CAT: grouped bivariate with two facets
- NUM, NUM, NUM: multivariate, or grouped bivariate for `overlay`
- NUM, NUM, NUM, CAT multivariate, or faceted bivariate for `overlay`

- NUM, NUM, NUM, CAT, CAT multivariate, or bivariate with two facets for `overlay`

Variable attributes may be supplied by conventional means; `pack` and `unpack` support storing and retrieving scalar column attributes. The following scalar attributes are currently supported.

- label: A variable descriptor. If present, panel functions will use label to create informative axis labels. See `axislabel`.
- guide: Units for a numeric variable, or an encoding (scalar string giving codes and possibly decodes) for a categorical item. If present, units will be used to inform the corresponding axis label (`axislabel`). If present, codes will be used to impose sort order on categorical variables. If present, decodes will be used as substitutes for stored values when presenting categorical labels, legends, and facet names. For more on encodings, see `encode`.
- reference: Some variables have values to which they can be compared. For example, residual error is often expected to be centered at zero. Default panel functions plot corresponding reference lines if this attribute is present. See for example `dens_panel`.
- symbol: Variable names are useful for programming, and variable labels are useful as axis labels. A symbol can be more formal than a variable name and more compact than a label. For example, `diag_label` will use variable names as labels for the diagonal panels of a scatterplot matrix; but it will prefer labels, if available; and will prefer symbols most of all. Markup rules for symbols are given in `wikisym2plotmath_`.

See Also

Other generic functions: `axislabel`, `categorical`, `corsplom`, `densplot`, `pack`, `scatter`, `unpack`

Other metaplot: `boxplot_data_frame`, `categorical_data_frame`, `corsplom_data_frame`, `densplot_data_frame`, `scatter_data_frame`

Examples

```
library(magrittr)
library(dplyr)
library(csv)
x <- as.csv(system.file(package = 'metaplot', 'extdata/theoph.csv'))
x %<>% pack
# sample plots
x %>% metaplot(sres)
x %>% metaplot(site)
x %>% metaplot(conc, arm)
x %>% densplot(conc, arm)
x %>% metaplot(arm, conc)
x %>% metaplot(conc, arm, site)
x %>% metaplot(conc, site, arm)
x %>% metaplot(conc, time)
x %>% metaplot(conc, time, panel = panel.smoothScatter)
x %>% metaplot(arm, site)
x %>% metaplot(arm, site, cohort)
x %>% metaplot(arm, site, , cohort)
x %>% metaplot(conc, time, subject)
x %>% metaplot(conc, time, , subject)
```

```
x %>% metaplot(conc, time, subject, site)
x %>% metaplot(conc, time, subject, site, arm)
x %>% metaplot(lKe, lKa, lCl)
x %>% metaplot(
  lKe, lKa, lCl,
  col = 'black',loess.col = 'red', pin.col = 'red',
  dens.col='blue',dens.alpha = 0.1
)
x %>% metaplot(conc, pred, ipred, time)
x %>% metaplot(conc, pred, ipred, time, subject)
x %>% metaplot(conc, pred, ipred, time, subject,
colors = c('black','blue','magenta'),
points = c(0.9,0, 0.4),
lines = c(F,T,T))
x %>% metaplot(conc, ipred, time, site, arm)
x %>% metaplot(res, conc, yref = 0, ysmooth = T, conf = T, grid = T, loc = 1)
x %>% metaplot(res, conc, arm, ysmooth = T, conf = T )
x %>% metaplot(res, conc, arm, ysmooth = T, conf = T, global = T, ref.col = 'red')
x %>% metaplot(subject,conc)
```

`metaplot.data.frame` *Create Metaplot for Data Frame.*

Description

Creates a metaplot for class 'data.frame'. Implements a rule to decided whether to make a density plot, a boxplot, a scatter plot, or a scatterplot matrix, given the supplied column names.

Usage

```
## S3 method for class 'data.frame'
metaplot(x, ...,
  univariate = getOption("metaplot_univariate", "densplot"),
  mixedvariate = getOption("metaplot_mixedvariate", "boxplot"),
  bivariate = getOption("metaplot_bivariate", "scatter"),
  multivariate = getOption("metaplot_multivariate", "corsplom"),
  categorical = getOption("metaplot_categorical", "categorical"))
```

Arguments

`x`	object
`...`	passed arguments
`univariate`	function for univariate arguments
`mixedvariate`	function for bivariate combinations of numeric and categoral arguments
`bivariate`	function for arguments that resolve to two numerics (see rules)
`multivariate`	function for more than two numeric arguments
`categorical`	function for categorical arguments

See Also

Other methods: axislabel.data.frame, boxplot.data.frame, categorical.data.frame, corsplom.data.frame, densplot.data.frame, pack.data.frame, scatter.data.frame, unpack.data.frame

Other univariate plots: dens_panel, densplot.data.frame, densplot_data_frame, densplot

Other bivariate plots: iso_prepanel, scatter.data.frame, scatter_data_frame, scatter

Other multivariate plots: corsplom.data.frame, corsplom_data_frame

Examples

```
## Not run:
library(magrittr)
library(dplyr)
library(csv)
library(nlme)
x <- Theoph

# mixed effects model
m1 <- nlme(
  conc ~ SSfol(Dose, Time, lKe, lKa, lCl),
  data = x,
  fixed = lKe + lKa + lCl ~ 1,
  random = lKe + lKa + lCl ~ 1
)

# some numeric and categorical properties
names(x) <- tolower(names(x))
x %<>% mutate(arm = ifelse(as.numeric(as.character(subject)) %% 2 == 0, 1, 2))
x %<>% mutate(site = ifelse(as.numeric(as.character(subject)) < 6, 1, 2))
x %<>% mutate(cohort = ifelse(as.numeric(as.character(subject)) %in% c(1:2,6:8), 1,2))
x %<>% mutate(pred = predict(m1,level = 0) %>% signif(4))
x %<>% mutate(ipred = predict(m1) %>% signif(4))
x %<>% mutate(res = residuals(m1) %>% signif(4))
x %<>% mutate(sres = residuals(m1, type = 'pearson') %>% signif(4))
r <- ranef(m1) %>% signif(4)
r$subject <- rownames(r)
x %<>% left_join(r)
# metadata
attr(x$subject,'label') <- 'subject identifier'
attr(x$wt,'label') <- 'subject weight'
attr(x$dose,'label') <- 'theophylline dose'
attr(x$time,'label') <- 'time since dose administration'
attr(x$conc,'label') <- 'theophylline concentration'
attr(x$arm,'label') <- 'trial arm'
attr(x$site,'label') <- 'investigational site'
attr(x$cohort,'label') <- 'recruitment cohort'
attr(x$pred,'label') <- 'population-predicted concentration'
attr(x$ipred,'label') <- 'individual-predicted concentration'
attr(x$res,'label') <- 'residuals'
attr(x$sres,'label') <- 'standardized residuals'
attr(x$lKe,'label') <- 'natural log of elimination rate constant'
attr(x$lKa,'label') <- 'natural log of absorption rate constant'
```

```
attr(x$lCl,'label') <- 'natural log of clearance'
attr(x$subject,'guide') <- '....'
attr(x$wt,'guide') <- 'kg'
attr(x$dose,'guide') <- 'mg/kg'
attr(x$time,'guide') <- 'h'
attr(x$conc,'guide') <- 'mg/L'
attr(x$arm,'guide') <- '//1/Arm A//2/Arm B//'
attr(x$site,'guide') <- '//1/Site 1//2/Site 2//'
attr(x$cohort,'guide') <- '//1/Cohort 1//2/Cohort 2//'
attr(x$pred,'guide') <- 'mg/L'
attr(x$ipred,'guide') <- 'mg/L'

attr(x$lKe,'reference') <- 0
attr(x$lKa,'reference') <- 0
attr(x$lCl,'reference') <- 0
attr(x$res,'reference') <- 0
attr(x$sres,'reference') <- '//-1.96//1.96//'

attr(x$subject,'symbol') <- 'ID_i'
attr(x$wt,'symbol') <- 'W_i'
attr(x$dose,'symbol') <- 'A_i'
attr(x$time,'symbol') <- 't_i,j'
attr(x$conc,'symbol') <- 'C_i,j'
attr(x$arm,'symbol') <- 'Arm_i'
attr(x$site,'symbol') <- 'Site_i'
attr(x$cohort,'symbol') <- 'Cohort_i'
attr(x$pred,'symbol') <- 'C_pred_p'
attr(x$ipred,'symbol') <- 'C_pred_i'
attr(x$res,'symbol') <- '\\epsilon'
attr(x$sres,'symbol') <- '\\epsilon_st'
attr(x$lKe,'symbol') <- 'ln(K_e.)'
attr(x$lKa,'symbol') <- 'ln(K_a.)'
attr(x$lCl,'symbol') <- 'ln(Cl_c./F)'


x %>% unpack %>% as.csv('theoph.csv')

## End(Not run)
```

`metaplot_ref` *Calculate Reference Values*

Description

Calculates reference values for x and y axes. Coerces column attribute 'reference' to numeric: a single value or an encoding giving multiple numeric values (decodes are ignored).

Usage

```
metaplot_ref(x, var, ...)
```

Arguments

x	data.frame
var	name of vector in x
...	ignored

Value

numeric

See Also

Other panel functions: boxplot_panel, categorical_panel, corsplom_panel_correlation, corsplom_panel_diagonal, corsplom_panel_scatter, dens_panel, diag_label, diag_pin, iso_prepanel, panel_tile, scatter_panel_ref, scatter_panel

Other reference lines: diag_pin, scatter_panel_ref

---

metastats *Format GLM Statistics*

Description

Formats GLM statistics. Uses a gaussian family by default, or binomial family if all y are 0 or 1, to fit a general linear model. Formats number of observations, p-value, and Pearson correlation coefficient into a string for printing.

Usage

```
metastats(x, y, family = if (all(y %in% 0:1, na.rm = TRUE)) "binomial" else
  "gaussian", ...)
```

Arguments

x	x values
y	y values
family	regression family
...	other arguments

Value

character

See Also

scatter_panel

Other regression functions: model, region

`model` *Execute Linear Model*

Description

Executes a linear model, automatically choosing binomial family as necessary.

Usage

```
model(x, y, family = if (all(y %in% 0:1, na.rm = TRUE)) "binomial" else
  "gaussian", ...)
```

Arguments

x	x values
y	y values
family	gaussian by default, or binomial for all y either zero or 1
...	passed to glm

Value

glm

See Also

Other regression functions: metastats, region

`multiplot` *Arrange Multiple Plots in a Grid*

Description

Arranges mutiple trellis plots in a grid, automatically choosing number of rows and columns. By default, number of rows is one less than or equal to the number of columns.

Usage

```
multiplot(..., nrow = NULL, ncol = NULL)
```

Arguments

...	trellis objects
nrow	number of rows of plots
ncol	number of columns of plots

Examples

```
library(lattice)
a <- xyplot(
conc ~ Time,
xlab=NULL,
ylab = NULL,
Theoph,
aspect = 1,
scales=list(draw=FALSE)
)
multiplot(a,a,a,a,a,a)
multiplot(a,a,a,a,a,a,a)
multiplot(a,a,a,a,a,a,a,a)
multiplot(a,a,a,a,a,a,a,a,a)
multiplot(a,a,a,a,a,a,a,a,a,a)
multiplot(a,a,a,a,a,a,a,a,a, nrow = 2)
multiplot(a,a,a,a,a,a,a,a,a, ncol = 4)
multiplot(a,a,a,a,a,a,a,a,a, ncol = 2)
multiplot(a,a,a,a,a,a,a,a,a, ncol = 4, nrow = 3)
```

---

pack *Pack Something*

Description

Pack Something. Generic, with method for data.frame.

Usage

```
pack(x, ...)
```

Arguments

x	object
...	other arguments

See Also

Other generic functions: axislabel, categorical, corsplom, densplot, metaplot, scatter, unpack

Other pack: pack.data.frame, unpack.data.frame, unpack

pack.data.frame *Capture Scalar Column Metadata as Column Attributes*

Description

Captures scalar column metadata (row values) as column attributes. Excises rows with non-missing values of `meta`, converting column values to column attributes. Afterward, column classes are re-optimized using default behavior of `read.table`. It is an error if `meta` is not in `names(x)`.

Usage

```
## S3 method for class 'data.frame'
pack(x, meta = getOption("meta", "meta"), as.is = TRUE,
  attributes = TRUE, ...)
```

Arguments

`x`	data.frame
`meta`	column in x giving names of attributes
`as.is`	passed to `type.convert`
`attributes`	preserve non-standard attributes (ignores names, row.names, class)
`...`	ignored arguments

Value

data.frame

See Also

Other pack: `pack`, `unpack.data.frame`, `unpack`

Other methods: `axislabel.data.frame`, `boxplot.data.frame`, `categorical.data.frame`, `corsplom.data.frame`, `densplot.data.frame`, `metaplot.data.frame`, `scatter.data.frame`, `unpack.data.frame`

Examples

```
foo <- data.frame(head(Theoph))
attr(foo$Subject, 'label') <-  'subject identifier'
attr(foo$Wt, 'label') <-  'weight'
attr(foo$Dose, 'label') <-  'dose'
attr(foo$Time, 'label') <-  'time'
attr(foo$conc, 'label') <-  'concentration'
attr(foo$Subject, 'guide') <-  '////'
attr(foo$Wt, 'guide') <-  'kg'
attr(foo$Dose, 'guide') <-  'mg/kg'
attr(foo$Time, 'guide') <-  'h'
attr(foo$conc, 'guide') <-  'mg/L'
unpack(foo, pos = 1)
```

```
unpack(foo, pos = 2)
unpack(foo, pos = 3)
unpack(foo, pos = 4)
bar <- unpack(foo)
pack(bar)
attributes(pack(bar)$Subject)
```

`panel_tile` *Draw a Tile*

Description

Draws a tile in a mosaic.

Usage

```
panel_tile(x, y, subscripts, group.number, group.value, fill.alpha, line.alpha,
  col, col.line, loc, msg, .src, alpha, cex, tex, ...)
```

Arguments

`x`	x values
`y`	y values
`subscripts`	subscripts
`group.number`	group number
`group.value`	group value
`fill.alpha`	alpha transparency for fill
`line.alpha`	alpha transparency for line
`col`	fill color
`col.line`	border color
`loc`	location for output of msg
`msg`	ignored
`.src`	data source for which subscripts give x, y, msg, and tile limits
`alpha`	ignored
`cex`	expansion for msg text; passed to msg
`tex`	tile expansion: scale factor for reducing each tile size relative to full size (<= 1)
`...`	passed arguments

See Also

Other panel functions: `boxplot_panel`, `categorical_panel`, `corsplom_panel_correlation`, `corsplom_panel_diagonal`, `corsplom_panel_scatter`, `dens_panel`, `diag_label`, `diag_pin`, `iso_prepanel`, `metaplot_ref`, `scatter_panel_ref`, `scatter_panel`

Other categorical: `categorical.data.frame`, `categorical_data_frame`, `categorical_panel`, `categorical`

region *Calculate a Confidence Region*

Description

Calculates a confidence region. `se.fit` from `predict.glm` is multiplied by `z` and added or subtracted from fits to give `hi` and `lo` columns in return value. `z` is normal quantile for the one-tailed probablitity corresponding to `conf`, e.g. ~ 1.96 for `conf = 0.95`. If non-missing `y` is only 0 or 1, the model family is binomial and resulting confidence intervals are back-transformed using `plogis`.

Usage

```
region(x, y, family = if (all(y %in% 0:1, na.rm = TRUE)) "binomial" else
  "gaussian", length.out = 1000, conf = 0.95, ...)
```

Arguments

`x`	x values
`y`	y values
`family`	gaussian by default, or binomial for all y either zero or 1
`length.out`	number of prediction points
`conf`	width of confidence interval; logical TRUE defaults to 0.95
`...`	passed to `model`

Value

data.frame with x, y, hi, lo at 1000 points

See Also

https://stackoverflow.com/questions/14423325/confidence-intervals-for-predictions-from-logistic-reg

http://www.rnr.lsu.edu/bret/BretWebSiteDocs/GLMCI.pdf

https://stat.ethz.ch/pipermail/r-help/2010-September/254465.html

http://r.789695.n4.nabble.com/Confidence-Intervals-for-logistic-regression-td2315932.html

Other regression functions: `metastats`, `model`

scatter *Scatterplot*

Description

Scatterplot.

Usage

```
scatter(x, ...)
```

Arguments

x	object
...	passed arguments

See Also

Other generic functions: axislabel, categorical, corsplom, densplot, metaplot, pack, unpack

Other scatter: scatter.data.frame, scatter_data_frame, scatter_panel

Other bivariate plots: iso_prepanel, metaplot.data.frame, scatter.data.frame, scatter_data_frame

scatter.data.frame *Scatterplot Method for Data Frame*

Description

Scatterplot method for class 'data.frame'. Parses arguments and generates the call: fun(x, yvar, xvar, groups, facets, ...).

Usage

```
## S3 method for class 'data.frame'
scatter(x, ..., fun = getOption("metaplot_scatter",
  "scatter_data_frame"))
```

Arguments

x	data.frame
...	passed to fun
fun	function to draw the plot

See Also

scatter_data_frame

Other bivariate plots: iso_prepanel, metaplot.data.frame, scatter_data_frame, scatter

Other scatter: scatter_data_frame, scatter_panel, scatter

Other methods: axislabel.data.frame, boxplot.data.frame, categorical.data.frame, corsplom.data.frame, densplot.data.frame, metaplot.data.frame, pack.data.frame, unpack.data.frame

Examples

```
library(magrittr)
library(dplyr)
attr(Theoph$conc,'label') <- 'theophylline concentration'
attr(Theoph$conc,'guide') <- 'mg/L'
attr(Theoph$Time,'label') <- 'time'
attr(Theoph$Time,'guide') <- 'h'
attr(Theoph$Subject,'guide') <- '////'
scatter(Theoph,conc, Time)
scatter(Theoph, conc, Time, Subject) # Subject as groups
scatter(Theoph, conc, Time, , Subject) # Subject as facet
scatter(Theoph %>% filter(conc > 0), conc, Time, Subject, ylog = TRUE, yref = 5)
scatter(Theoph, conc, Time, Subject, ysmooth = TRUE)
scatter(Theoph, conc, Time, conf = TRUE, loc = 3, yref = 6)
scatter(Theoph, conc, Time, conf = TRUE, loc = 3, yref = 6, global = TRUE)
## Not run:
\dontshow{
attr(Theoph,'title') <- 'Theophylline'
scatter(Theoph, conc, Time, main = function(x,...)attr(x,'title'))
scatter(Theoph, conc, Time, sub= function(x,...)attr(x,'title'))
options(metaplot_main = function(x,...)attr(x,'title'))
scatter(Theoph, conc, Time)
}

## End(Not run)
```

---

scatter_data_frame *Scatterplot Function for Data Frame*

Description

Scatterplot function for class 'data.frame'.

Usage

```
scatter_data_frame(x, yvar, xvar, groups = NULL, facets = NULL,
  log = getOption("metaplot_log", FALSE), ylog = getOption("metaplot_ylog",
  log), xlog = getOption("metaplot_xlog", log),
  crit = getOption("metaplot_crit", 1.3), yref = getOption("metaplot_ref",
  metaplot_ref), xref = getOption("metaplot_ref", metaplot_ref),
```

```
  ylab = getOption("metaplot_lab", axislabel),
  xlab = getOption("metaplot_lab", axislabel),
  ysmooth = getOption("metaplot_ysmooth", FALSE),
  xsmooth = getOption("metaplot_xsmooth", FALSE),
  iso = getOption("metaplot_iso", FALSE),
  na.rm = getOption("metaplot_na.rm", TRUE),
  aspect = getOption("metaplot_aspect", 1),
  auto.key = getOption("metaplot_auto.key", NULL),
  keycols = getOption("metaplot_keycols", NULL),
  as.table = getOption("metaplot_scatter_as.table", TRUE),
  prepanel = getOption("metaplot_scatter_prepanel", NULL),
  scales = getOption("metaplot_scatter_scales", NULL),
  panel = getOption("metaplot_scatter_panel", scatter_panel),
  colors = getOption("metaplot_colors", NULL),
  symbols = getOption("metaplot_symbols", NULL),
  points = getOption("metaplot_points", TRUE),
  lines = getOption("metaplot_lines", FALSE),
  main = getOption("metaplot_main", NULL), sub = getOption("metaplot_sub",
  NULL), subscripts = TRUE, par.settings = NULL, ...)
```

Arguments

x	data.frame
yvar	character: y variable(s)
xvar	character: x variable
groups	optional grouping variable; ignored if more than one yvar
facets	optional conditioning variables
log	a default shared by ylog and xlog
ylog	log transform y axis (auto-selected if NA)
xlog	log transform x axis (auto-selected if NA)
crit	if ylog or xlog missing, log transform if mean/median ratio for non-missing values is greater than crit
yref	reference line from y axis; can be function(x = x, var = yvar, ...)
xref	reference line from x axis; can be function(x = x, var = xvar, ...)
ylab	y axis label; can be function(x = x, var = yvar, log = ylog, ..)
xlab	x axis label; can be function(x = x, var = xvar, log = xlog, ..)
ysmooth	supply loess smooth of y on x
xsmooth	supply loess smmoth of x on y
iso	plot line of unity (auto-selected if NA)
na.rm	whether to remove data points with one or more missing coordinates
aspect	passed to xyplot
auto.key	passed to xyplot
keycols	number of auto.key columns

as.table	passed to xyplot
prepanel	passed to xyplot (guessed if NULL)
scales	passed to xyplot (guessed if NULL)
panel	name or definition of panel function
colors	replacements for default colors in group order
symbols	replacements for default symbols in group order
points	whether to plot points for each group: logical, or alpha values between 0 and 1
lines	whether to plot lines for each group: logical, or alpha values between 0 and 1
main	character, or a function of x, yvar, xvar, groups, facets, and log
sub	character, or a function of x, yvar, xvar, groups, facets, and log
subscripts	passed to xyplot
par.settings	passed to xyplot (calculated if NULL)
...	passed to region

See Also

scatter_panel

Other bivariate plots: iso_prepanel, metaplot.data.frame, scatter.data.frame, scatter

Other metaplot: boxplot_data_frame, categorical_data_frame, corsplom_data_frame, densplot_data_frame, metaplot

Other scatter: scatter.data.frame, scatter_panel, scatter

Examples

```
library(magrittr)
library(dplyr)
attr(Theoph$conc,'label') <- 'theophylline concentration'
attr(Theoph$conc,'guide') <- 'mg/L'
attr(Theoph$Time,'label') <- 'time'
attr(Theoph$Time,'guide') <- 'h'
attr(Theoph$Subject,'guide') <- '////'
scatter_data_frame(Theoph, 'conc', 'Time')
scatter_data_frame(Theoph, 'conc','Time', 'Subject')
scatter_data_frame(Theoph, 'conc','Time', facets = 'Subject')
scatter_data_frame(Theoph %>% filter(conc > 0), 'conc','Time', 'Subject',ylog = TRUE, yref = 5)
scatter_data_frame(Theoph, 'conc','Time', 'Subject',ysmooth = TRUE)
scatter_data_frame(Theoph, 'conc','Time', 'Subject',ysmooth = TRUE,global = TRUE)
scatter_data_frame(Theoph, 'conc','Time', conf = TRUE, loc = 3, yref = 6)
scatter_data_frame(Theoph, 'conc','Time', conf = TRUE, loc = 3, yref = 6)
```

scatter_panel *Panel Function for Metaplot Scatterplot*

Description

Default panel function for scatter_data_frame. Calls panel.xyplot and optionally plots linear fit, confidence region, reference lines, and statistics.

Usage

```
scatter_panel(x, y, groups, xref = getOption("scatter_panel_ref",
  scatter_panel_ref), yref = getOption("scatter_panel_ref",
  scatter_panel_ref), ref.col = getOption("metaplot_ref.col", "grey"),
  ref.lty = getOption("metaplot_ref.lty", "solid"),
  ref.alpha = getOption("metaplot_ref.alpha", 1),
  ysmooth = getOption("metaplot_ysmooth", FALSE),
  xsmooth = getOption("metaplot_xsmooth", FALSE),
  smooth.lty = getOption("metaplot_smooth.lty", "dashed"),
  smooth.alpha = getOption("metaplot_smooth.alpha", 1),
  fit = getOption("metaplot_fit", conf),
  fit.lty = getOption("metaplot_fit.lty", "solid"),
  fit.alpha = getOption("metaplot_fit.alpha", 1),
  conf = getOption("metaplot_conf", FALSE),
  conf.alpha = getOption("metaplot_conf.alpha", 0.3),
  loc = getOption("metaplot_loc", 0), iso = getOption("metaplot_iso",
  FALSE), global = getOption("metaplot_global", FALSE),
  global.col = getOption("metaplot_global.col", "grey"),
  msg = getOption("metaplot_msg", "metastats"), type, ...)
```

Arguments

x	x values
y	y values
groups	optional grouping item
xref	reference line from x axis; can be function(x, y, ...)
yref	reference line from y axis; can be function(y, x, ...)
ref.col	reference line color
ref.lty	reference line type
ref.alpha	reference line alpha
ysmooth	supply loess smooth of y on x
xsmooth	supply loess smmoth of x on y
smooth.lty	smooth line type
smooth.alpha	smooth alpha

fit	draw a linear fit of y ~ x
fit.lty	fit line type
fit.alpha	fit alpha
conf	logical, or width for a confidence region around a linear fit; passed to region; TRUE defaults to 95 percent confidence interval; may not make sense if xlog is TRUE
conf.alpha	alpha transparency for confidence region
loc	where to print statistics on a panel; suppressed for grouped plots
iso	use isometric axes with line of unity (auto-selected if NA)
global	if TRUE, xsmooth, ysmooth, fit, and conf are applied to all data rather than groupwise
global.col	color for global aesthetics
msg	a function to print text on a panel: called with x values, y values, and ....
type	overridden by scatter_panel
...	passed to panel.superpose, panel.xyplot, panel.polygon, region, panel.text

See Also

metastats

scatter.data.frame

Other panel functions: boxplot_panel, categorical_panel, corsplom_panel_correlation, corsplom_panel_diagonal, corsplom_panel_scatter, dens_panel, diag_label, diag_pin, iso_prepanel, metaplot_ref, panel_tile, scatter_panel_ref

Other scatter: scatter.data.frame, scatter_data_frame, scatter

---

scatter_panel_ref *Calculate Panel Reference Values*

Description

Calculates reference values for x and y axes at the panel level.

Usage

```
scatter_panel_ref(a, b, ...)
```

Arguments

a	vector of interest
b	vector for other axis
...	ignored

Value

numeric

See Also

Other panel functions: boxplot_panel, categorical_panel, corsplom_panel_correlation, corsplom_panel_diagonal, corsplom_panel_scatter, dens_panel, diag_label, diag_pin, iso_prepanel, metaplot_ref, panel_tile, scatter_panel

Other reference lines: diag_pin, metaplot_ref

---

tilestats *Format Tile Statistics*

Description

Formats statistics for a mosaic tile.

Usage

```
tilestats(x, y, ...)
```

Arguments

x	x values
y	y values
...	other arguments

Value

character

See Also

categorical_panel

unpack *Unpack Something*

Description

Unpack Something. Generic, with method for data.frame.

Usage

```
unpack(x, ...)
```

Arguments

x	object
...	other arguments

See Also

Other pack: pack.data.frame, pack, unpack.data.frame

Other generic functions: axislabel, categorical, corsplom, densplot, metaplot, pack, scatter

unpack.data.frame *Express Scalar Column Attributes as Column Metadata*

Description

Expresses scalar column attributes as column metadata (row values). Column with name meta is created to hold names of attributes, if any. A transposed table (sorted by attribute name) of scalar column attribute values (coerced to character) is bound to the existing data.frame (the attributes themselves are removed from columns). Bind position is controlled by position such that the intersection of new rows and column occurs in the corresponding corner, numbered clockwise from top-left. Resulting column classes are character. It is an error if meta is already in names(x).

Usage

```
## S3 method for class 'data.frame'
unpack(x, meta = getOption("meta", "meta"),
  position = 1L, ignore = c("class", "levels"), ...)
```

Arguments

x	data.frame
meta	column in result giving names of attributes
position	1 (top-left), 2 (top-right), 3 (bottom-right), or 4 (bottom-left)
ignore	character: attributes to ignore
...	ignored arguments

Value

data.frame

data.frame with all columns of class character

See Also

Other pack: pack.data.frame, pack, unpack

Other methods: axislabel.data.frame, boxplot.data.frame, categorical.data.frame, corsplom.data.frame, densplot.data.frame, metaplot.data.frame, pack.data.frame, scatter.data.frame

---

wikisym2plotmath *Convert Wiki Symbol to Plotmath*

Description

Converts wiki symbol to plotmath. Vectorized version of wikisym2plotmath_.

Usage

```
wikisym2plotmath(x, ...)
```

Arguments

x	character
...	ignored

Value

expression

See Also

Other formatters: diag_label, wikisym2plotmath_

`wikisym2plotmath_` *Convert One Wiki Symbol to Plotmath*

Description

Converts one wiki symbol to plotmath. A Wiki symbol is simple text with arbitrarily nested subscript (`_`) and superscript (`^`) groupings. Use dot (`.`) to explicitly terminate a grouping, and use backslash-dot (`\.`) for a literal dot. Examples: `V_c./F`. Trailing dots need not be supplied. Leading/trailing whitespace is removed. Tab character not allowed.

Usage

```
wikisym2plotmath_(x, ...)
```

Arguments

`x`	character
`...`	ignored

Value

expression

See Also

Other formatters: `diag_label`, `wikisym2plotmath`

Examples

```
wikisym2plotmath_('V_c./F')
wikisym2plotmath_('AUC_ss')
wikisym2plotmath_('C_max_ss')
wikisym2plotmath_('var^eta_j')
```

Index