Package ‘ggfortify’

January 3, 2022

Type Package

Title Data Visualization Tools for Statistical Analysis Results

Version 0.4.14

Date 2022-01-02

Maintainer Yuan Tang <terrytangyuan@gmail.com>

URL https://github.com/sinhrks/ggfortify

BugReports https://github.com/sinhrks/ggfortify/issues

Encoding UTF-8

Description Unified plotting tools for statistics commonly used, such as GLM, time series, PCA families, clustering and survival analysis. The package offers a single plotting interface for these analysis results and plots in a unified style using 'ggplot2'.

License MIT + file LICENSE

VignetteBuilder knitr

Depends methods, ggplot2 (>= 2.0.0)

Imports dplyr (>= 0.3), tidyr, gridExtra, grid, scales, stringr, tibble

Suggests testthat, cluster, changepoint, dlm, fGarch, forecast, ggrepel, glmnet, grDevices, KFAS, knitr, lintr, mapdata, markdown, MASS, MSwM, nlme, raster, ROCR, sp, stats, strucchange, survival, timeSeries, tseries, utils, vars, xts, zoo, lfda

RoxygenNote 7.1.1

NeedsCompilation no

Author Masaaki Horikoshi [aut],
Yuan Tang [aut, cre] (<https://orcid.org/0000-0001-5243-233X>),
Austin Dickey [ctb],
Matthias Grenié [ctb],
Ryan Thompson [ctb],
Luciano Selzer [ctb],

Dario Strbenac [ctb],
Kirill Voronin [ctb],
Damir Pulatov [ctb]

Repository CRAN

Date/Publication 2022-01-03 15:30:02 UTC

R topics documented:

+,ggmultiplot,ANY-method *Generic add operator for* ggmultiplot

Description

Generic add operator for ggmultiplot

Usage

```
## S4 method for signature 'ggmultiplot,ANY'
e1 + e2
```

Arguments

e1	first argument
e2	second argument

Value

ggmultiplot

apply_facets *Apply facets to to* ggplot2::ggplot

Description

Apply facets to to ggplot2::ggplot

Usage

```
apply_facets(
  p,
  formula,
  facets = TRUE,
  nrow = NULL,
  ncol = 1,
  scales = "free_y",
  ...
)
```

Arguments

`p`	`ggplot2::ggplot` instance
`formula`	`stats::formula` instance
`facets`	Logical value to specify use facets
`nrow`	Number of facet/subplot rows
`ncol`	Number of facet/subplot columns
`scales`	Scale value passed to `ggplot2`
`...`	other arguments passed to methods

Value

ggplot

`apply_grid` *Apply grid to to* `ggplot2::ggplot`

Description

Apply grid to to `ggplot2::ggplot`

Usage

```
apply_grid(p, formula, scales = "free_y", ...)
```

Arguments

`p`	`ggplot2::ggplot` instance
`formula`	`stats::formula` instance
`scales`	Scale value passed to `ggplot2`
`...`	other arguments passed to methods

as_tibble.basis *Convert a spline basis to a tibble*

Description

Convert a spline basis to a tibble

Usage

```
## S3 method for class 'basis'
as_tibble(x, ...)
```

Arguments

x	object of class "basis"
...	Ignored.

Details

This function is needed because the default method for converting a matrix object with an additional class attribute to a tibble causes issues because each column of the resulting tibble has the attributes, including the matrix class, copied from the source. Having matrices as columns in a tibble causes dplyr to throw errors, so a special method is needed to avoid copying the class attribute.

Value

A tibble constructed from the underlying matrix of the basis object. Each column will possess all the attributes from the source object, except that the "class" attribute will be renamed to "basis.class" to avoid interfering with dplyr operations.

Examples

```
library(splines)
library(tibble)
x <- seq(0, 1, by=0.001)
spl <- bs(x, df=6)
as_tibble(spl)
```

autoplot.aareg *Autoplot* survival::aareg

Description

Autoplot survival::aareg

Usage

```
## S3 method for class 'aareg'
autoplot(
  object,
  maxtime = NULL,
  surv.connect = TRUE,
  facets = TRUE,
  ncol = NULL,
  xlab = "",
  ylab = "",
  ...
)
```

Arguments

object	survival::aareg instance
maxtime	truncate the input to the model at time "maxtime"
surv.connect	logical frag indicates whether connects survival curve to the origin
facets	Logical value to specify use facets
ncol	Number of facet/subplot columns
xlab	character vector or expression for x axis label
ylab	character vector or expression for y axis label
...	other arguments passed to autoplot.survfit

Value

ggplot

Examples

```
library(survival)
autoplot(aareg(Surv(time, status) ~ age + sex + ph.ecog, data = lung, nmin = 1))
```

`autoplot.acf` *Autoplot* `stats::acf`. *Note to pass 'plot = FALSE' to original function to suppress standard plot output*

Description

Autoplot `stats::acf`. Note to pass 'plot = FALSE' to original function to suppress standard plot output

Usage

```
## S3 method for class 'acf'
autoplot(
  object,
  colour = "#000000",
  linetype = "solid",
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "dashed",
  conf.int.fill = NULL,
  conf.int.alpha = 0.3,
  conf.int.value = 0.95,
  conf.int.type = "white",
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = "ACF",
  asp = NULL,
  ...
)
```

Arguments

`object`	`stats::acf` instance
`colour`	Line colour
`linetype`	Line type
`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals
`conf.int.fill`	fill colour for confidence intervals
`conf.int.alpha`	alpha for confidence intervals

conf.int.value	Coverage probability for confidence interval
conf.int.type	Type of confidence interval, 'white' for white noise or 'ma' MA(k-1) model
xlim	limits for x axis
ylim	limits for y axis
log	which variables to log transform ("x", "y", or "xy")
main	character vector or expression for plot title
xlab	character vector or expression for x axis label
ylab	character vector or expression for y axis label
asp	the y/x aspect ratio
...	other arguments passed to methods

Value

ggplot

Examples

```
autoplot(stats::acf(AirPassengers, plot = FALSE))
autoplot(stats::pacf(AirPassengers, plot = FALSE))
autoplot(stats::ccf(AirPassengers, AirPassengers, plot = FALSE))
```

autoplot.basis *Autoplot spline basis instances*

Description

Autoplot spline basis instances

Usage

```
## S3 method for class 'basis'
autoplot(object, data, n = 256, ...)
```

Arguments

object	spline basis object
data	x-values at which to evaluate the splines. Optional. By default, an evenly spaced sequence of 256 values covering the range of the splines will be used.
n	If data is not provided, instead use an evenly-spaced sequence of x-values of this length (plus one, since both endpoints are included). If data is provided, this argument is ignored.
...	Ignored.

Value

ggplot

Examples

```
library(splines)
x <- seq(0, 1, by=0.001)
spl <- bs(x, df=6)
autoplot(spl)
autoplot(spl, n=5)
```

`autoplot.breakpoints` *Autoplot* `strucchange::breakpoints`

Description

Autoplot `strucchange::breakpoints`

Usage

```
## S3 method for class 'breakpoints'
autoplot(
  object,
  data = NULL,
  cpt.colour = "#FF0000",
  cpt.linetype = "dashed",
  ...
)
```

Arguments

`object`	`strucchange::breakpoints` or `strucchange::breakpointsfull` instance.
`data`	Original time series. Mandatory for plotting `strucchange::breakpoints` instance.
`cpt.colour`	Line colour for changepoints
`cpt.linetype`	Line type for changepoints
`...`	other arguments passed to `autoplot.ts`

Value

ggplot

Examples

```
library(strucchange)
bp.nile <- breakpoints(Nile ~ 1)
autoplot(bp.nile)
autoplot(bp.nile, is.date = TRUE)
autoplot(breakpoints(bp.nile, breaks = 2), data = Nile)
```

`autoplot.cpt` *Autoplot* `changepoint::cpt`

Description

Autoplot `changepoint::cpt`

Usage

```
## S3 method for class 'cpt'
autoplot(
  object,
  is.date = NULL,
  cpt.colour = "#FF0000",
  cpt.linetype = "dashed",
  ...
)
```

Arguments

`object`	`changepoint::cpt` instance
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.
`cpt.colour`	Line colour for changepoints
`cpt.linetype`	Line type for changepoints
`...`	other arguments passed `autoplot.ts`

Value

ggplot

Examples

```
library(changepoint)
autoplot(cpt.mean(AirPassengers))
autoplot(cpt.meanvar(AirPassengers))
```

autoplot.cv.glmnet *Autoplot* glmnet::cv.glmnet

Description

Autoplot glmnet::cv.glmnet

Usage

```
## S3 method for class 'cv.glmnet'
autoplot(
  object,
  sign.lambda = 1,
  label.n = 12,
  label = TRUE,
  label.label = "nz",
  label.colour = NULL,
  label.alpha = NULL,
  label.size = NULL,
  label.angle = NULL,
  label.family = NULL,
  label.fontface = NULL,
  label.lineheight = NULL,
  label.hjust = NULL,
  label.vjust = NULL,
  label.repel = FALSE,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

object	glmnet::cv.glmnet instance
sign.lambda	Either plot against log(lambda) (default) or its negative if sign.lambda=-1.
label.n	Number of Df labels
label	Logical value whether to display labels
label.label	Column name used for label text
label.colour	Colour for text labels
label.alpha	Alpha for text labels

`label.size`	Size for text labels
`label.angle`	Angle for text labels
`label.family`	Font family for text labels
`label.fontface`	Fontface for text labels
`label.lineheight`	Lineheight for text labels
`label.hjust`	Horizontal adjustment for text labels
`label.vjust`	Vertical adjustment for text labels
`label.repel`	Logical flag indicating whether to use `ggrepel`, enabling this may take some time for plotting
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
autoplot(glmnet::cv.glmnet(data.matrix(Orange[-3]), data.matrix(Orange[3])))
```

`autoplot.density` *Autoplot* `stats::density`

Description

Autoplot `stats::density`

Usage

```
## S3 method for class 'density'
autoplot(
  object,
  p = NULL,
  colour = "#000000",
  linetype = NULL,
  fill = NULL,
  alpha = NULL,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

`object`	`stats::density` instance
`p`	`ggplot2::ggplot` instance to plot
`colour`	Line colour
`linetype`	Line type
`fill`	Fill colour
`alpha`	Alpha
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to PDC/CDF func

Value

ggplot

Examples

```
autoplot(stats::density(stats::rnorm(1:50)))
autoplot(stats::density(stats::rnorm(1:50)), fill = 'blue')
```

autoplot.forecast *Autoplot* forecast::forecast

Description

Autoplot forecast::forecast

Usage

```
## S3 method for class 'forecast'
autoplot(
  object,
  is.date = NULL,
  ts.connect = TRUE,
  predict.geom = "line",
  predict.colour = "#0000FF",
  predict.size = NULL,
  predict.linetype = NULL,
  predict.alpha = NULL,
  predict.fill = NULL,
  predict.shape = NULL,
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "none",
  conf.int.fill = "#000000",
  conf.int.alpha = 0.3,
  ...
)
```

Arguments

object	forecast::forecast instance
is.date	Logical frag indicates whether the stats::ts is date or not. If not provided, regard the input as date when the frequency is 4 or 12
ts.connect	Logical frag indicates whether connects original time-series and predicted values
predict.geom	geometric string for predicted time-series
predict.colour	line colour for predicted time-series
predict.size	point size for predicted time-series
predict.linetype	line type for predicted time-series
predict.alpha	alpha for predicted time-series
predict.fill	fill colour for predicted time-series
predict.shape	point shape for predicted time-series

`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals
`conf.int.fill`	fill colour for confidence intervals
`conf.int.alpha`	alpha for confidence intervals
`...`	other arguments passed to `autoplot.ts`

Value

ggplot

Examples

```
d.arima <- forecast::auto.arima(AirPassengers)
autoplot(forecast::forecast(d.arima, h = 10))
autoplot(forecast::forecast(d.arima, level = c(85), h = 10))
autoplot(forecast::forecast(d.arima, h = 5), conf.int = FALSE, is.date = FALSE)
autoplot(forecast::forecast(stats::HoltWinters(UKgas), h = 10))
## Not run:
autoplot(forecast::forecast(forecast::ets(UKgas), h = 5))

## End(Not run)
```

`autoplot.ggmultiplot` *Autoplot* `ggmultiplot` *instances. It returns the passed instance as it is.*

Description

Autoplot `ggmultiplot` instances. It returns the passed instance as it is.

Usage

```
## S3 method for class 'ggmultiplot'
autoplot(object, ...)
```

Arguments

`object`	ggmultiplot instance
`...`	Not used.

Value

ggmultiplot

autoplot.ggplot *Autoplot* `ggplot` *instances. It returns the passed instance as it is.*

Description

Autoplot `ggplot` instances. It returns the passed instance as it is.

Usage

```
## S3 method for class 'ggplot'
autoplot(object, ...)
```

Arguments

`object`	ggplot instance
`...`	Not used.

Value

ggplot

autoplot.glmnet *Autoplot* `glmnet::glmnet`

Description

Autoplot `glmnet::glmnet`

Usage

```
## S3 method for class 'glmnet'
autoplot(
  object,
  xvar = c("norm", "lambda", "dev"),
  label.n = 7,
  label = TRUE,
  label.label = "Df",
  label.colour = NULL,
  label.alpha = NULL,
  label.size = NULL,
  label.angle = NULL,
  label.family = NULL,
  label.fontface = NULL,
  label.lineheight = NULL,
  label.hjust = NULL,
  label.vjust = NULL,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = "Coefficients",
  asp = NULL,
  ...
)
```

Arguments

`object`	`glmnet::glmnet` instance
`xvar`	values to be dranw on the X axis. Either "norm" (L1-norm), "lambda" (log-lambda sequence) or "dev" (percent deviance)
`label.n`	Number of Df labels
`label`	Logical value whether to display labels
`label.label`	Column name used for label text
`label.colour`	Colour for text labels
`label.alpha`	Alpha for text labels
`label.size`	Size for text labels
`label.angle`	Angle for text labels
`label.family`	Font family for text labels
`label.fontface`	Fontface for text labels
`label.lineheight`	Lineheight for text labels
`label.hjust`	Horizontal adjustment for text labels
`label.vjust`	Vertical adjustment for text labels
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
autoplot(glmnet::glmnet(data.matrix(Orange[-3]), data.matrix(Orange[3])))
```

`autoplot.kmeans` *Autoplot cluster instances*

Description

Autoplot cluster instances

Usage

```
## S3 method for class 'kmeans'
autoplot(object, data = NULL, colour = ”cluster”, ...)
```

Arguments

`object`	Clustered instance
`data`	Original data used for clustering. Mandatory for `stats::kmeans`.
`colour`	line colour for points
`...`	other arguments passed to `autoplot::prcomp`

Value

ggplot

Examples

```
autoplot(stats::kmeans(iris[-5], 3), data = iris)
autoplot(cluster::clara(iris[-5], 3), label = TRUE)
autoplot(cluster::fanny(iris[-5], 3))
autoplot(cluster::fanny(iris[-5], 3), frame = TRUE)
autoplot(cluster::pam(iris[-5], 3), data = iris, colour = 'Species')
autoplot(cluster::pam(iris[-5], 3), data = iris, frame = TRUE, frame.type = 't')
```

`autoplot.list` *Autoplot list*

Description

Autoplot list

Usage

```
## S3 method for class 'list'
autoplot(object, data = NULL, nrow = NULL, ncol = NULL, scales = ”free_y”, ...)
```

Arguments

`object`	`list` instance
`data`	original dataset, if needed
`nrow`	Number of facet/subplot rows
`ncol`	Number of facet/subplot columns
`scales`	Scale value passed to `ggplot2`
`...`	other arguments passed to methods

Value

ggplot

`autoplot.lm` *Autoplot* `stats::lm` *and* `stats::glm`

Description

Autoplot `stats::lm` and `stats::glm`

Usage

```
## S3 method for class 'lm'
autoplot(
  object,
  which = c(1:3, 5),
  data = NULL,
  colour = "#444444",
  size = NULL,
  linetype = NULL,
  alpha = NULL,
  fill = NULL,
  shape = NULL,
  label = TRUE,
  label.label = ".label",
  label.colour = "#000000",
  label.alpha = NULL,
  label.size = NULL,
  label.angle = NULL,
  label.family = NULL,
  label.fontface = NULL,
  label.lineheight = NULL,
  label.hjust = NULL,
  label.vjust = NULL,
  label.repel = FALSE,
  label.n = 3,
  smooth.colour = "#0000FF",
  smooth.linetype = "solid",
  ad.colour = "#888888",
  ad.linetype = "dashed",
  ad.size = 0.2,
  nrow = NULL,
  ncol = NULL,
  ...
)
```

Arguments

`object`	`stats::lm` instance
`which`	If a subset of the plots is required, specify a subset of the numbers 1:6.
`data`	original dataset, if needed
`colour`	line colour
`size`	point size
`linetype`	line type
`alpha`	alpha
`fill`	fill colour
`shape`	point shape
`label`	Logical value whether to display labels
`label.label`	Column name used for label text
`label.colour`	Colour for text labels
`label.alpha`	Alpha for text labels
`label.size`	Size for text labels
`label.angle`	Angle for text labels
`label.family`	Font family for text labels
`label.fontface`	Fontface for text labels
`label.lineheight`	Lineheight for text labels
`label.hjust`	Horizontal adjustment for text labels
`label.vjust`	Vertical adjustment for text labels
`label.repel`	Logical flag indicating whether to use `ggrepel`, enabling this may take some time for plotting
`label.n`	Number of points to be laeled in each plot, starting with the most extreme
`smooth.colour`	Line colour for smoother lines
`smooth.linetype`	Line type for smoother lines
`ad.colour`	Line colour for additional lines
`ad.linetype`	Line type for additional lines

ad.size	Fill colour for additional lines
nrow	Number of facet/subplot rows
ncol	Number of facet/subplot columns
...	other arguments passed to methods

Value

ggplot

Examples

```
## Not run:
autoplot(lm(Petal.Width ~ Petal.Length, data = iris))
autoplot(glm(Petal.Width ~ Petal.Length, data = iris), which = 1:6)
autoplot(lm(Petal.Width~Petal.Length, data = iris), data = iris, colour = 'Species')

## End(Not run)
```

---

autoplot.map *Autoplot* maps::map

Description

Autoplot maps::map

Usage

```
## S3 method for class 'map'
autoplot(
  object,
  p = NULL,
  geom = "path",
  group = "group",
  colour = "black",
  size = NULL,
  linetype = NULL,
  alpha = NULL,
  fill = NULL,
  shape = NULL,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = "",
  ylab = "",
  asp = NULL,
  ...
)
```

Arguments

`object`	`maps::map` instance
`p`	`ggplot2::ggplot` instance
`geom`	geometric string for map. 'path', 'point' or 'polygon'
`group`	key for grouping geoms
`colour`	line colour
`size`	point size
`linetype`	line type
`alpha`	alpha
`fill`	fill colour
`shape`	point shape
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

`autoplot.matrix` *Plot* `base::matrix`

Description

Plot `base::matrix`

Usage

```
## S3 method for class 'matrix'
autoplot(
  object,
  original = NULL,
  geom = "tile",
  colour = NULL,
  size = NULL,
  alpha = NULL,
  fill = "#0000FF",
  shape = NULL,
  label = FALSE,
  label.label = "rownames",
  label.colour = colour,
  label.alpha = NULL,
  label.size = NULL,
  label.angle = NULL,
  label.family = NULL,
  label.fontface = NULL,
  label.lineheight = NULL,
  label.hjust = NULL,
  label.vjust = NULL,
  label.repel = FALSE,
  scale = NULL,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

`object`	`base::matrix` instance
`original`	Combined to data by column if provided. Intended to be used for stat functions which returns not containing original data.
`geom`	Geometric string for plotting. 'tile' or 'point'.
`colour`	colour for points ('point' only)
`size`	point size
`alpha`	alpha
`fill`	fill colour. Ignored if scale keyword is passed. ('tile' Only)
`shape`	point shape
`label`	Logical value whether to display labels
`label.label`	Column name used for label text
`label.colour`	Colour for text labels
`label.alpha`	Alpha for text labels
`label.size`	Size for text labels
`label.angle`	Angle for text labels
`label.family`	Font family for text labels
`label.fontface`	Fontface for text labels

`label.lineheight`	Lineheight for text labels
`label.hjust`	Horizontal adjustment for text labels
`label.vjust`	Vertical adjustment for text labels
`label.repel`	Logical flag indicating whether to use `ggrepel`, enabling this may take some time for plotting
`scale`	(Deprecated) `ggplot2::scale` instance to plot. ('tile' Only)
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
autoplot(matrix(rnorm(20), nc = 5))
autoplot(matrix(rnorm(20), nc = 5), fill = 'red')
autoplot(matrix(rnorm(20), nc = 2), geom = 'point')
```

---

autoplot.MSM.lm *Autoplot* MSwM::MSM.lm

Description

Autoplot `MSwM::MSM.lm`

Usage

```
## S3 method for class 'MSM.lm'
autoplot(object, prob.colour = "#FF0000", prob.linetype = "dashed", ...)
```

Arguments

`object`	`MSwM::MSM.lm` instance
`prob.colour`	Line colour for probabilities
`prob.linetype`	Line type for probabilities
`...`	other arguments passed to `autoplot.ts`

Value

ggplot

Examples

```
## Not run:
library(MSwM)
d <- data.frame(Data = c(rnorm(50, mean = -10), rnorm(50, mean = 10)),
                exog = cos(seq(-pi/2, pi/2, length.out = 100)))
d.mswm <- MSwM::msmFit(lm(Data ~.-1, data = d), k=2, sw=rep(TRUE, 2),
                       control = list(parallelization = FALSE))
autoplot(d.mswm)

## End(Not run)
```

---

autoplot.pca_common *Autoplot PCA-likes*

Description

Autoplot PCA-likes

Usage

```
## S3 method for class 'pca_common'
autoplot(
  object,
  data = NULL,
  scale = 1,
  x = 1,
  y = 2,
  variance_percentage = TRUE,
  ...
)
```

Arguments

object	PCA-like instance
data	Joined to fitting result if provided.
scale	scaling parameter, disabled by 0
x	principal component number used in x axis
y	principal component number used in y axis
variance_percentage	show the variance explained by the principal component?
...	other arguments passed to [ggbiplot()]

Examples

```
autoplot(stats::prcomp(iris[-5]))
autoplot(stats::prcomp(iris[-5]), data = iris)
autoplot(stats::prcomp(iris[-5]), data = iris, colour = 'Species')
autoplot(stats::prcomp(iris[-5]), label = TRUE, loadings = TRUE, loadings.label = TRUE)
autoplot(stats::prcomp(iris[-5]), frame = TRUE)
autoplot(stats::prcomp(iris[-5]), data = iris, frame = TRUE,
         frame.colour = 'Species')
autoplot(stats::prcomp(iris[-5]), data = iris, frame = TRUE,
         frame.type = 't', frame.colour = 'Species')

autoplot(stats::princomp(iris[-5]))
autoplot(stats::princomp(iris[-5]), data = iris)
autoplot(stats::princomp(iris[-5]), data = iris, colour = 'Species')
autoplot(stats::princomp(iris[-5]), label = TRUE, loadings = TRUE, loadings.label = TRUE)

#Plot PC 2 and 3
autoplot(stats::princomp(iris[-5]), x = 2, y = 3)

#Don't show the variance explained
autoplot(stats::princomp(iris[-5]), variance_percentage = FALSE)

d.factanal <- stats::factanal(state.x77, factors = 3, scores = 'regression')
autoplot(d.factanal)
autoplot(d.factanal, data = state.x77, colour = 'Income')
autoplot(d.factanal, label = TRUE, loadings = TRUE, loadings.label = TRUE)
```

---

autoplot.performance *Autoplot* ROCR::performance

Description

Autoplot `ROCR::performance`

Usage

```
## S3 method for class 'performance'
autoplot(object, p = NULL, bins = 5, ...)
```

Arguments

`object` `ROCR::performance` instance

`p` `ggplot2::ggplot` instances

`bins` If `object` represents a measure whose value is just a scalar (e.g. `performance(predObj,'auc')`), a histogram will be plotted of this scalar's values for different runs. `bins` is the number of bins for this histogram.

`...` other arguments passed to methods

Value

ggplot

autoplot.RasterCommon *Autoplot* raster::raster

Description

Only plot the first layer of the given raster

Usage

```
## S3 method for class 'RasterCommon'
autoplot(
  object,
  raster.layer = NULL,
  p = NULL,
  alpha = NULL,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = "",
  ylab = "",
  asp = NULL,
  ...
)
```

Arguments

object	raster::raster instance
raster.layer	name of the layer to plot
p	ggplot2::ggplot instance
alpha	alpha
xlim	limits for x axis
ylim	limits for y axis
log	which variables to log transform ("x", "y", or "xy")
main	character vector or expression for plot title
xlab	character vector or expression for x axis label
ylab	character vector or expression for y axis label
asp	the y/x aspect ratio
...	other arguments passed to methods

Value

ggplot

autoplot.silhouette *Autoplot silhouette instances*

Description

Autoplot silhouette instances

Usage

```
## S3 method for class 'silhouette'
autoplot(
  object,
  colour = "red",
  linetype = "dashed",
  size = 0.5,
  bar.width = 1,
  ...
)
```

Arguments

object	Silhouette instance
colour	reference line color
linetype	reference line type
size	reference line size
bar.width	bar width
...	other arguments passed to methods

Value

ggplot

Examples

```
model = cluster::pam(iris[-5], 3L)
sil = cluster::silhouette(model)
autoplot(sil)

autoplot(cluster::silhouette(cluster::clara(iris[-5], 3)))
autoplot(cluster::silhouette(cluster::fanny(iris[-5], 3)))

model = stats::kmeans(iris[-5], 3)
sil = cluster::silhouette(model$cluster, stats::dist(iris[-5]))
autoplot(sil)
```

autoplot.SpatialCommon *Autoplot* `maps::map`

Description

Autoplot `maps::map`

Usage

```
## S3 method for class 'SpatialCommon'
autoplot(
  object,
  p = NULL,
  group = NULL,
  colour = "black",
  size = NULL,
  linetype = NULL,
  alpha = NULL,
  fill = NULL,
  shape = NULL,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = "",
  ylab = "",
  asp = NULL,
  ...
)
```

Arguments

`object`	`maps::map` instance
`p`	`ggplot2::ggplot` instance
`group`	key for grouping geoms
`colour`	line colour
`size`	point size
`linetype`	line type
`alpha`	alpha
`fill`	fill colour
`shape`	point shape
`xlim`	limits for x axis
`ylim`	limits for y axis

`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

`autoplot.spec` *Autoplot* `stats::spec`

Description

Autoplot `stats::spec`

Usage

```
## S3 method for class 'spec'
autoplot(
  object,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "y",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

`object`	`stats::spec` instance
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
autoplot(stats::spec.ar(AirPassengers))
autoplot(stats::spec.pgram(AirPassengers))
```

autoplot.stepfun *Plot* stats::stepfun

Description

Plot stats::stepfun

Usage

```
## S3 method for class 'stepfun'
autoplot(
  object,
  colour = NULL,
  size = NULL,
  linetype = NULL,
  alpha = NULL,
  shape = 1,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

object	stats::stepfun instance
colour	colour
size	point size
linetype	line type
alpha	alpha
shape	point shape
xlim	limits for x axis
ylim	limits for y axis

`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
autoplot(stepfun(c(1, 2, 3), c(4, 5, 6, 7)))
autoplot(stepfun(c(1), c(4, 5)), shape = NULL)
autoplot(stepfun(c(1, 3, 4, 8), c(4, 5, 2, 3, 5)), linetype = 'dashed')
autoplot(stepfun(c(1, 2, 3, 4, 5, 6, 7, 8, 10), c(4, 5, 6, 7, 8, 9, 10, 11, 12, 9)), colour = 'red')
```

`autoplot.survfit` *Autoplot* `survival::survfit`

Description

Autoplot `survival::survfit`

Usage

```
## S3 method for class 'survfit'
autoplot(
  object,
  fun = NULL,
  surv.geom = "step",
  surv.colour = NULL,
  surv.size = NULL,
  surv.linetype = NULL,
  surv.alpha = NULL,
  surv.fill = NULL,
  surv.shape = NULL,
  surv.connect = TRUE,
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "none",
  conf.int.fill = "#000000",
  conf.int.alpha = 0.3,
  censor = TRUE,
  censor.colour = NULL,
  censor.size = 3,
  censor.alpha = NULL,
  censor.shape = "+",
  facets = FALSE,
  nrow = NULL,
  ncol = 1,
  grid = FALSE,
  strip_swap = FALSE,
  scales = "free_y",
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

`object`	`survival::survfit` instance
`fun`	an arbitrary function defining a transformation of the survival curve
`surv.geom`	geometric string for survival curve. 'step', 'line' or 'point'
`surv.colour`	line colour for survival curve
`surv.size`	point size for survival curve
`surv.linetype`	line type for survival curve
`surv.alpha`	alpha for survival curve
`surv.fill`	fill colour survival curve
`surv.shape`	point shape survival curve
`surv.connect`	logical frag indicates whether connects survival curve to the origin
`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals
`conf.int.fill`	fill colour for confidence intervals
`conf.int.alpha`	alpha for confidence intervals
`censor`	Logical flag indicating whether to plot censors
`censor.colour`	colour for censors
`censor.size`	size for censors
`censor.alpha`	alpha for censors
`censor.shape`	shape for censors

`facets`	Logical value to specify use facets
`nrow`	Number of facet/subplot rows
`ncol`	Number of facet/subplot columns
`grid`	Logical flag indicating whether to draw grid
`strip_swap`	swap facet or grid strips
`scales`	Scale value passed to `ggplot2`
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
library(survival)
autoplot(survfit(Surv(time, status) ~ sex, data = lung))
autoplot(survfit(Surv(time, status) ~ sex, data = lung), facets = TRUE)
autoplot(survfit(Surv(time, status) ~ 1, data = lung))
autoplot(survfit(Surv(time, status) ~ sex, data=lung), conf.int = FALSE, censor = FALSE)
autoplot(survfit(coxph(Surv(time, status) ~ sex, data = lung)))
```

`autoplot.ts` *Autoplot time-series-like*

Description

Autoplot time-series-like

Usage

```
## S3 method for class 'ts'
autoplot(
  object,
  columns = NULL,
  group = NULL,
  is.date = NULL,
  index.name = "Index",
  p = NULL,
  ts.scale = FALSE,
  stacked = FALSE,
  facets = TRUE,
  nrow = NULL,
  ncol = 1,
  scales = "free_y",
  ts.geom = "line",
  ts.colour = NULL,
  ts.size = NULL,
  ts.linetype = NULL,
  ts.alpha = NULL,
  ts.fill = NULL,
  ts.shape = NULL,
  geom = ts.geom,
  colour = ts.colour,
  size = ts.size,
  linetype = ts.linetype,
  alpha = ts.alpha,
  fill = ts.fill,
  shape = ts.shape,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = "",
  ylab = "",
  asp = NULL,
  ...
)
```

Arguments

`object`	time-series-like instance
`columns`	Character vector specifies target column name(s)
`group`	Character vector specifies grouping
`is.date`	Logical frag indicates whether the `stats::ts` is date or not If not provided, regard the input as date when the frequency is 4 or 12
`index.name`	Specify column name for time series index when passing `data.frame` via data.
`p`	`ggplot2::ggplot` instance
`ts.scale`	Logical flag indicating whether to perform scaling each timeseries
`stacked`	Logical flag indicating whether to stack multivariate timeseries
`facets`	Logical value to specify use facets
`nrow`	Number of facet/subplot rows
`ncol`	Number of facet/subplot columns
`scales`	Scale value passed to `ggplot2`

`ts.geom`	geometric string for time-series. ’line’, ’bar’, ’ribbon’, or ’point’
`ts.colour`	line colour for time-series
`ts.size`	point size for time-series
`ts.linetype`	line type for time-series
`ts.alpha`	alpha for time-series
`ts.fill`	fill colour for time-series
`ts.shape`	point shape for time-series
`geom`	same as ts.geom
`colour`	same as ts.colour
`size`	same as ts.size
`linetype`	same as ts.linetype
`alpha`	same as ts.alpha
`fill`	same as ts.fill
`shape`	same as ts.shape
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

Examples

```
## Not run:
data(Canada, package = 'vars')
autoplot(AirPassengers)
autoplot(UKgas, ts.geom = 'bar')
autoplot(Canada)
autoplot(Canada, facets = FALSE)

library(zoo)
autoplot(xts::as.xts(AirPassengers))
autoplot(timeSeries::as.timeSeries(AirPassengers))
its <- tseries::irts(cumsum(rexp(10, rate = 0.1)), matrix(rnorm(20), ncol=2))
autoplot(its)

autoplot(stats::stl(UKgas, s.window = 'periodic'))
autoplot(stats::decompose(UKgas))

## End(Not run)
```

autoplot.tsmodel *Autoplot time series models (like AR, ARIMA)*

Description

Autoplot time series models (like AR, ARIMA)

Usage

```
## S3 method for class 'tsmodel'
autoplot(
  object,
  data = NULL,
  predict = NULL,
  is.date = NULL,
  ts.connect = TRUE,
  fitted.geom = "line",
  fitted.colour = "#FF0000",
  fitted.size = NULL,
  fitted.linetype = NULL,
  fitted.alpha = NULL,
  fitted.fill = NULL,
  fitted.shape = NULL,
  predict.geom = "line",
  predict.colour = "#0000FF",
  predict.size = NULL,
  predict.linetype = NULL,
  predict.alpha = NULL,
  predict.fill = NULL,
  predict.shape = NULL,
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "none",
  conf.int.fill = "#000000",
  conf.int.alpha = 0.3,
  ...
)
```

Arguments

`object`	Time series model instance
`data`	original dataset, needed for `stats::ar`, `stats::Arima`
`predict`	Predicted `stats::ts` If not provided, try to retrieve from current environment using variable name.
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12

`ts.connect` Logical frag indicates whether connects original time-series and predicted values

`fitted.geom` geometric string for fitted time-series

`fitted.colour` line colour for fitted time-series

`fitted.size` point size for fitted time-series

`fitted.linetype` line type for fitted time-series

`fitted.alpha` alpha for fitted time-series

`fitted.fill` fill colour for fitted time-series

`fitted.shape` point shape for fitted time-series

`predict.geom` geometric string for predicted time-series

`predict.colour` line colour for predicted time-series

`predict.size` point size for predicted time-series

`predict.linetype` line type for predicted time-series

`predict.alpha` alpha for predicted time-series

`predict.fill` fill colour for predicted time-series

`predict.shape` point shape for predicted time-series

`conf.int` Logical flag indicating whether to plot confidence intervals

`conf.int.colour` line colour for confidence intervals

`conf.int.linetype` line type for confidence intervals

`conf.int.fill` fill colour for confidence intervals

`conf.int.alpha` alpha for confidence intervals

`...` Keywords passed to `autoplot.ts`

Value

ggplot

Examples

```
d.ar <- stats::ar(AirPassengers)
autoplot(d.ar)
autoplot(d.ar, predict = predict(d.ar, n.ahead = 5))
autoplot(stats::arima(UKgas), data = UKgas)
autoplot(forecast::arfima(AirPassengers))
autoplot(forecast::nnetar(UKgas), is.date = FALSE)

d.holt <- stats::HoltWinters(USAccDeaths)
autoplot(d.holt)
autoplot(d.holt, predict = predict(d.holt, n.ahead = 5))
autoplot(d.holt, predict = predict(d.holt, n.ahead = 5, prediction.interval = TRUE))
```

`autoplot.varprd` *Autoplot* `vars::varprd`

Description

Autoplot `vars::varprd`

Usage

```
## S3 method for class 'varprd'
autoplot(
  object,
  is.date = NULL,
  ts.connect = TRUE,
  scales = "free_y",
  predict.geom = "line",
  predict.colour = "#0000FF",
  predict.size = NULL,
  predict.linetype = NULL,
  predict.alpha = NULL,
  predict.fill = NULL,
  predict.shape = NULL,
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "none",
  conf.int.fill = "#000000",
  conf.int.alpha = 0.3,
  ...
)
```

Arguments

`object` `vars::varpred` instance

`is.date` Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.

`ts.connect` Logical frag indicates whether connects original time-series and predicted values

`scales` Scale value passed to `ggplot2`

`predict.geom` geometric string for predicted time-series

`predict.colour` line colour for predicted time-series

`predict.size` point size for predicted time-series

`predict.linetype` line type for predicted time-series

`predict.alpha` alpha for predicted time-series

`predict.fill`	fill colour for predicted time-series
`predict.shape`	point shape for predicted time-series
`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals
`conf.int.fill`	fill colour for confidence intervals
`conf.int.alpha`	alpha for confidence intervals
`...`	other arguments passed to `autoplot.ts`

Value

ggplot

Examples

```
data(Canada, package = 'vars')
d.var <- vars::VAR(Canada, p = 3, type = 'const')
autoplot(stats::predict(d.var, n.ahead = 50), is.date = TRUE)
autoplot(stats::predict(d.var, n.ahead = 50), conf.int = FALSE)
```

`cbind_wraps` *Wrapper for cbind*

Description

Wrapper for cbind

Usage

```
cbind_wraps(df1, df2)
```

Arguments

`df1`	1st data
`df2`	2nd data

Value

list

Examples

```
ggfortify:::cbind_wraps(iris[1:2], iris[3:5])
```

`check_names` *Check data names are equal with expected*

Description

Check data names are equal with expected

Usage

```
check_names(data, expected)
```

Arguments

`data`	`list` instance to be checked
`expected`	expected character vector

Value

logical

`confint.acf` *Calculate confidence interval for* `stats::acf`

Description

Calculate confidence interval for `stats::acf`

Usage

```
## S3 method for class 'acf'
confint(x, ci = 0.95, ci.type = "white")
```

Arguments

`x`	`stats::acf` instance
`ci`	Float value for confidence interval
`ci.type`	"white" or "ma"

Value

vector

Examples

```
air.acf <- acf(AirPassengers, plot = FALSE)
ggfortify:::confint.acf(air.acf)
ggfortify:::confint.acf(air.acf, ci.type = 'ma')
```

`deprecate.warning` *Show deprecate warning*

Description

Show deprecate warning

Usage

```
deprecate.warning(old.kw, new.kw)
```

Arguments

`old.kw`	Keyword being deprecated
`new.kw`	Keyword being replaced

Examples

```
ggfortify:::deprecate.warning('old', 'new')
```

`fitted.ar` *Calculate fitted values for* `stats::ar`

Description

Calculate fitted values for `stats::ar`

Usage

```
## S3 method for class 'ar'
fitted(object, ...)
```

Arguments

`object`	`stats::ar` instance
`...`	other keywords

Value

ts An time series of the one-step forecasts

Examples

```
fitted(ar(WWWusage))
```

`flatten` *Flatten dataframe contains matrix*

Description

tains list or matrix as column

Usage

```
flatten(df)
```

Arguments

`df`	`data.frame` to be flatten

`fortify.aareg` *Convert* `survival::aareg` *to* `data.frame`

Description

Convert `survival::aareg` to `data.frame`

Usage

```
## S3 method for class 'aareg'
fortify(
  model,
  data = NULL,
  maxtime = NULL,
  surv.connect = TRUE,
  melt = FALSE,
  ...
)
```

Arguments

`model`	`survival::aareg` instance
`data`	original dataset, if needed
`maxtime`	truncate the input to the model at time "maxtime"
`surv.connect`	logical frag indicates whether connects survival curve to the origin
`melt`	Logical flag indicating whether to melt each timeseries as variable
`...`	other arguments passed to methods

Value

data.frame

Examples

```
library(survival)
fortify(aareg(Surv(time, status) ~ age + sex + ph.ecog, data = lung, nmin = 1))
fortify(aareg(Surv(time, status) ~ age + sex + ph.ecog, data = lung, nmin = 1), melt = TRUE)
```

`fortify.acf` *Convert* `stats::acf` *to* `data.frame`

Description

Convert `stats::acf` to `data.frame`

Usage

```
## S3 method for class 'acf'
fortify(
  model,
  data = NULL,
  conf.int = TRUE,
  conf.int.value = 0.95,
  conf.int.type = "white",
  ...
)
```

Arguments

`model`	`stats::acf` instance
`data`	original dataset, if needed
`conf.int`	Logical flag indicating whether to attach confidence intervals
`conf.int.value`	Coverage probability for confidence interval
`conf.int.type`	Type of confidence interval, 'white' for white noise or 'ma' MA(k-1) model
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(stats::acf(AirPassengers))
fortify(stats::pacf(AirPassengers))
fortify(stats::ccf(AirPassengers, AirPassengers))

fortify(stats::acf(AirPassengers), conf.int = TRUE)
```

fortify.basis *Convert spline basis instances to* data.frame

Description

Convert spline basis instances to data.frame

Usage

```
## S3 method for class 'basis'
fortify(model, data, n = 256, ...)
```

Arguments

model	spline basis object
data	x-values at which to evaluate the splines. Optional. By default, an evenly spaced sequence of 256 values covering the range of the splines will be used.
n	If data is not provided, instead use an evenly-spaced sequence of x-values of this length (plus one, since both endpoints are included). If data is provided, this argument is ignored.
...	other arguments passed to methods

Value

data.frame with 3 columns: Spline (character), x (numeric), and y (numeric); giving the interpolated x and y values for each of the splines in the basis.

Examples

```
library(splines)
x <- seq(0, 1, by=0.001)
spl <- bs(x, df=6)
fortify(spl)
```

fortify.cpt *Convert* changepoint::cpt *and* strucchange::breakpoints *to* data.frame

Description

Convert changepoint::cpt and strucchange::breakpoints to data.frame

Usage

```
## S3 method for class 'cpt'
fortify(model, data = NULL, is.date = NULL, ...)
```

Arguments

`model`	`chantepoint::cpt` or `strucchange::breakpoints` instance
`data`	original dataset, if needed
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.
`...`	other arguments passed to methods

Value

data.frame

Examples

```
library(changepoint)
fortify(cpt.mean(AirPassengers))
fortify(cpt.var(AirPassengers))
fortify(cpt.meanvar(AirPassengers))

library(strucchange)
bp.nile <- breakpoints(Nile ~ 1)
fortify(bp.nile)
fortify(breakpoints(bp.nile, breaks = 2))
fortify(breakpoints(bp.nile, breaks = 2), data = Nile)
```

fortify.cv.glmnet *Convert* `glmnet::cv.glmnet` *to* `data.frame`

Description

Convert `glmnet::cv.glmnet` to `data.frame`

Usage

```
## S3 method for class 'cv.glmnet'
fortify(model, data = NULL, ...)
```

Arguments

`model`	`glmnet::cv.glmnet` instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(glmnet::cv.glmnet(data.matrix(Orange[-3]), data.matrix(Orange[3])))
```

fortify.density *Convert* stats::density *to* data.frame

Description

Convert stats::density to data.frame

Usage

```
## S3 method for class 'density'
fortify(model, data = NULL, ...)
```

Arguments

model	stats::density instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(stats::density(stats::rnorm(1:50)))
```

fortify.dist *Convert* stats::dist *to* data.frame

Description

Convert stats::dist to data.frame

Usage

```
## S3 method for class 'dist'
fortify(model, data = NULL, ...)
```

Arguments

model	stats::dist instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(eurodist)
```

---

fortify.ets *Convert* forecast::bats *and* forecast::ets *to* data.frame

---

Description

Convert forecast::bats and forecast::ets to data.frame

Usage

```
## S3 method for class 'ets'
fortify(model, data = NULL, ...)
```

Arguments

model	forecast::bats or forecast::ets instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
## Not run:
fortify(forecast::bats(UKgas))
fortify(forecast::ets(UKgas))

## End(Not run)
```

fortify.factanal *Convert* stats::factanal *to* data.frame

Description

Convert stats::factanal to data.frame

Usage

```
## S3 method for class 'factanal'
fortify(model, data = NULL, ...)
```

Arguments

model	stats::factanal instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
d.factanal <- stats::factanal(state.x77, factors = 3, scores = 'regression')
fortify(d.factanal)
fortify(d.factanal, data = state.x77)
```

fortify.forecast *Convert* forecast::forecast *to* data.frame

Description

Convert forecast::forecast to data.frame

Usage

```
## S3 method for class 'forecast'
fortify(model, data = NULL, is.date = NULL, ts.connect = FALSE, ...)
```

Arguments

`model`	`forecast::forecast` instance
`data`	original dataset, if needed
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.
`ts.connect`	Logical frag indicates whether connects original time-series and predicted values
`...`	other arguments passed to methods

Value

data.frame

Examples

```
d.arima <- forecast::auto.arima(AirPassengers)
d.forecast <- forecast::forecast(d.arima, level = c(95), h = 50)
fortify(d.forecast)
fortify(d.forecast, ts.connect = TRUE)
```

`fortify.glmnet` *Convert* `glmnet::glmnet` *to* `data.frame`

Description

Convert `glmnet::glmnet` to `data.frame`

Usage

```
## S3 method for class 'glmnet'
fortify(model, data = NULL, ...)
```

Arguments

`model`	`glmnet::glmnet` instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(glmnet::glmnet(data.matrix(Orange[-3]), data.matrix(Orange[3])))
```

fortify.kmeans *Convert cluster instances to* `data.frame`

Description

Convert cluster instances to `data.frame`

Usage

```
## S3 method for class 'kmeans'
fortify(model, data = NULL, ...)
```

Arguments

`model`	Clustered instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(stats::kmeans(iris[-5], 3))
fortify(stats::kmeans(iris[-5], 3), data = iris)
fortify(cluster::clara(iris[-5], 3))
fortify(cluster::fanny(iris[-5], 3))
fortify(cluster::pam(iris[-5], 3), data = iris)
```

fortify.lfda *Convert* `lfda::lfda` *or* `lfda::klfda` *or* `lfda::self` *to* `data.frame`

Description

Convert `lfda::lfda` or `lfda::klfda` or `lfda::self` to `data.frame`

Usage

```
## S3 method for class 'lfda'
fortify(model, data = NULL, ...)
```

Arguments

`model`	`lfda::lfda` or `lfda::klfda` or `lfda::self` instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

Examples

```
## Not run:
model <- lfda::lfda(iris[, -5], iris[, 5], 3, metric = "plain")
fortify(model)

## End(Not run)
```

`fortify.list` *Convert list to data.frame*

Description

Convert list to data.frame

Usage

```
## S3 method for class 'list'
fortify(model, data = NULL, ...)
```

Arguments

`model`	`list` instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

`fortify.matrix` *Convert* `base::matrix` *to* `data.frame`

Description

Different from `as.data.frame`

Usage

```
## S3 method for class 'matrix'
fortify(model, data = NULL, compat = FALSE, ...)
```

Arguments

`model`	`base::matrix` instance
`data`	original dataset, if needed
`compat`	Logical frag to specify the behaviour when converting matrix which has no column name. If `FALSE`, result has character columns like c('1', '2', ...). If `TRUE`, result has character columns like c('V1', 'V2', ...).
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(matrix(1:6, nrow=2, ncol=3))
```

`fortify.MSM.lm` *Convert* `MSwM::MSM.lm` *to* `data.frame`

Description

Convert `MSwM::MSM.lm` to `data.frame`

Usage

```
## S3 method for class 'MSM.lm'
fortify(model, data = NULL, melt = FALSE, ...)
```

Arguments

`model`	`MSwM::MSM.lm` instance
`data`	original dataset, if needed
`melt`	Logical flag indicating whether to melt each models
`...`	other arguments passed to methods

Value

data.frame

Examples

```
library(MSwM)
d <- data.frame(Data = c(rnorm(50, mean = -10), rnorm(50, mean = 10)),
                exog = cos(seq(-pi/2, pi/2, length.out = 100)))
d.mswm <- MSwM::msmFit(lm(Data ~.-1, data = d), k=2, sw=rep(TRUE, 2),
                       control = list(parallelization = FALSE))
fortify(d.mswm)
```

fortify.performance *Convert* ROCR::performance *objects to* data.frame

Description

Convert ROCR::performance objects to data.frame

Usage

```
## S3 method for class 'performance'
fortify(model, data = NULL, ...)
```

Arguments

model	performance instances
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

fortify.prcomp *Convert* stats::prcomp, stats::princomp *to* data.frame

Description

Convert stats::prcomp, stats::princomp to data.frame

Usage

```
## S3 method for class 'prcomp'
fortify(model, data = NULL, ...)
```

Arguments

model	stats::prcomp or stats::princomp instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(stats::prcomp(iris[-5]))
fortify(stats::prcomp(iris[-5]), data = iris)

fortify(stats::princomp(iris[-5]))
fortify(stats::princomp(iris[-5]), data = iris)
```

fortify.RasterCommon *Convert* `raster` *to* `data.frame`

Description

Convert `raster` to `data.frame`

Usage

```
## S3 method for class 'RasterCommon'
fortify(model, data = NULL, maxpixels = 1e+05, rename = TRUE, ...)
```

Arguments

`model`	`raster` instances
`data`	original dataset, if needed
`maxpixels`	number of pixels for resampling
`rename`	logical flag indicating whether to rename coordinates to long and lat
`...`	other arguments passed to methods

Value

data.frame

fortify.silhouette *Convert* `cluster::silhouette` *to* `data.frame`

Description

Convert `cluster::silhouette` to `data.frame`

Usage

```
## S3 method for class 'silhouette'
fortify(model, data = NULL, ...)
```

Arguments

`model`	Silhouette instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(cluster::silhouette(cluster::pam(iris[-5], 3)))
fortify(cluster::silhouette(cluster::clara(iris[-5], 3)))
fortify(cluster::silhouette(cluster::fanny(iris[-5], 3)))

mod = stats::kmeans(iris[-5], 3)
fortify(cluster::silhouette(mod$cluster, stats::dist(iris[-5])))
```

`fortify.SpatialCommon` *Convert* `sp` *instances to* `data.frame`.

Description

Convert `sp` instances to `data.frame`.

Usage

```
## S3 method for class 'SpatialCommon'
fortify(model, data = NULL, rename = TRUE, ...)
```

Arguments

`model`	`sp` instances
`data`	original dataset, if needed
`rename`	logical flag indicating whether to rename coordinates to long and lat
`...`	other arguments passed to methods

Value

data.frame

fortify.spec *Convert* stats::spec *to* data.frame

Description

Convert stats::spec to data.frame

Usage

```
## S3 method for class 'spec'
fortify(model, data = NULL, ...)
```

Arguments

model	stats::spec instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(spectrum(AirPassengers))
fortify(stats::spec.ar(AirPassengers))
fortify(stats::spec.pgram(AirPassengers))
```

fortify.stepfun *Convert* stats::stepfun *to* data.frame

Description

Convert stats::stepfun to data.frame

Usage

```
## S3 method for class 'stepfun'
fortify(model, data, ...)
```

Arguments

model	stats::stepfun instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(stepfun(c(1, 2, 3), c(4, 5, 6, 7)))
fortify(stepfun(c(1), c(4, 5)))
fortify(stepfun(c(1, 3, 4, 8), c(4, 5, 2, 3, 5)))
fortify(stepfun(c(1, 2, 3, 4, 5, 6, 7, 8, 10), c(4, 5, 6, 7, 8, 9, 10, 11, 12, 9)))
```

`fortify.survfit` *Convert* `survival::survfit` *to* `data.frame`

Description

Convert `survival::survfit` to `data.frame`

Usage

```
## S3 method for class 'survfit'
fortify(model, data = NULL, surv.connect = FALSE, fun = NULL, ...)
```

Arguments

`model` `survival::survfit` instance

`data` original dataset, if needed

`surv.connect` logical frag indicates whether connects survival curve to the origin

`fun` an arbitrary function defining a transformation of the survival curve

`...` other arguments passed to methods

Value

data.frame

Examples

```
library(survival)
fortify(survfit(Surv(time, status) ~ sex, data = lung))
fortify(survfit(Surv(time, status) ~ 1, data = lung))
fortify(survfit(coxph(Surv(time, status) ~ sex, data = lung)))
fortify(survfit(coxph(Surv(time, status) ~ 1, data = lung)))
```

fortify.table *Convert* base::table *to* data.frame

Description

Convert base::table to data.frame

Usage

```
## S3 method for class 'table'
fortify(model, data, ...)
```

Arguments

model	base::table instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(Titanic)
```

fortify.ts *Convert time-series-like to data.frame*

Description

Convert time-series-like to data.frame

Usage

```
## S3 method for class 'ts'
fortify(
  model,
  data = NULL,
  columns = NULL,
  is.date = NULL,
  index.name = "Index",
  data.name = "Data",
  scale = FALSE,
  melt = FALSE,
  ...
)
```

Arguments

`model`	time-series-like instance
`data`	original dataset, if needed
`columns`	character vector specifies target column name(s)
`is.date`	logical frag indicates whether the `stats::ts` is date or not If not provided, regard the input as date when the frequency is 4 or 12
`index.name`	specify column name for time series index
`data.name`	specify column name for univariate time series data. Ignored in multivariate time series.
`scale`	logical flag indicating whether to perform scaling each timeseries
`melt`	logical flag indicating whether to melt each timeseries as variable
`...`	other arguments passed to methods

Value

data.frame

Examples

```
fortify(AirPassengers)
fortify(timeSeries::as.timeSeries(AirPassengers))
fortify(tseries::irts(cumsum(rexp(10, rate = 0.1)), matrix(rnorm(20), ncol=2)))
fortify(stats::stl(UKgas, s.window = 'periodic'))
fortify(stats::decompose(UKgas))
```

`fortify.tsmodel` *Convert time series models (like AR, ARIMA) to* `data.frame`

Description

Convert time series models (like AR, ARIMA) to `data.frame`

Usage

```
## S3 method for class 'tsmodel'
fortify(
  model,
  data = NULL,
  predict = NULL,
  is.date = NULL,
  ts.connect = TRUE,
  ...
)
```

Arguments

model	Time series model instance
data	original dataset, needed for `stats::ar`, `stats::Arima`
predict	Predicted `stats::ts` If not provided, try to retrieve from current environment using variable name.
is.date	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.
ts.connect	Logical frag indicates whether connects original time-series and predicted values
...	other arguments passed to methods

Value

data.frame

Examples

```
fortify(stats::ar(AirPassengers))
fortify(stats::arima(UKgas))
fortify(stats::arima(UKgas), data = UKgas, is.date = TRUE)
fortify(forecast::auto.arima(austres))
fortify(forecast::arfima(AirPassengers))
fortify(forecast::nnetar(UKgas))
fortify(stats::HoltWinters(USAccDeaths))

data(LPP2005REC, package = 'timeSeries')
x = timeSeries::as.timeSeries(LPP2005REC)
d.Garch = fGarch::garchFit(LPP40 ~ garch(1, 1), data = 100 * x, trace = FALSE)
fortify(d.Garch)
```

---

fortify.varprd *Convert* `vars::varprd` *to* `data.frame`

Description

Convert `vars::varprd` to `data.frame`

Usage

```
## S3 method for class 'varprd'
fortify(
  model,
  data = NULL,
  is.date = NULL,
  ts.connect = FALSE,
  melt = FALSE,
  ...
)
```

Arguments

`model`	`vars::varprd` instance
`data`	original dataset, if needed
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.
`ts.connect`	Logical frag indicates whether connects original time-series and predicted values
`melt`	Logical flag indicating whether to melt each timeseries as variable
`...`	other arguments passed to methods

Value

data.frame

Examples

```
data(Canada, package = 'vars')
d.var <- vars::VAR(Canada, p = 3, type = 'const')
fortify(stats::predict(d.var, n.ahead = 50))
```

`fortify_base` *Convert* `base::table` *to* `data.frame`

Description

Convert `base::table` to `data.frame`

Usage

```
fortify_base(model, data, ...)
```

Arguments

`model`	`base::table` instance
`data`	original dataset, if needed
`...`	other arguments passed to methods

Value

data.frame

fortify_map *Convert maps::map to data.frame.*

Description

Convert maps::map to data.frame.

Usage

```
fortify_map(model, data = NULL, ...)
```

Arguments

model	maps::map instance
data	original dataset, if needed
...	other arguments passed to methods

Value

data.frame

geom_confint *Connect observations by stairs.*

Description

Connect observations by stairs.

Usage

```
geom_confint(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  ...
)
```

Arguments

`mapping`	the aesthetic mapping
`data`	a layer specific dataset
`stat`	the statistical transformation to use on the data for this layer
`position`	the position adjustment to use for overlapping points on this layer
`na.rm`	logical frag whether silently remove missing values
`...`	other arguments passed to methods

`geom_factory` *Factory function to control* `ggplot2::geom_xxx` *functions*

Description

Factory function to control `ggplot2::geom_xxx` functions

Usage

```
geom_factory(geomfunc, data = NULL, position = NULL, ...)
```

Arguments

`geomfunc`	`ggplot2::geom_xxx` function
`data`	plotting data
`position`	A position function or character
`...`	other arguments passed to methods

Value

proto

`get.dtindex` *Convert* `ts` *index to* `Date vector`

Description

Convert `ts` index to `Date vector`

Usage

```
get.dtindex(data, is.tsp = FALSE, is.date = NULL)
```

Arguments

`data`	`ts` instance
`is.tsp`	Logical frag whether data is `tsp` itself or not
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.

Value

vector

Examples

```
ggfortify:::get.dtindex(AirPassengers)
ggfortify:::get.dtindex(UKgas)
ggfortify:::get.dtindex(Nile, is.date = FALSE)
ggfortify:::get.dtindex(Nile, is.date = TRUE)
```

`get.dtindex.continuous`

Get `Date vector` *continue to* `ts` *index*

Description

Get `Date vector` continue to `ts` index

Usage

```
get.dtindex.continuous(data, length, is.tsp = FALSE, is.date = NULL)
```

Arguments

`data`	`ts` instance
`length`	A number to continue
`is.tsp`	Logical frag whether data is `tsp` itself or not
`is.date`	Logical frag indicates whether the `stats::ts` is date or not. If not provided, regard the input as date when the frequency is 4 or 12.

Value

vector

Examples

```
ggfortify:::get.dtindex.continuous(AirPassengers, length = 10)
ggfortify:::get.dtindex.continuous(UKgas, length = 10)
```

get.layout *Calcurate layout matrix for* `ggmultiplot`

Description

Calcurate layout matrix for `ggmultiplot`

Usage

```
get.layout(nplots, ncol, nrow)
```

Arguments

`nplots`	Number of plots
`ncol`	Number of grid columns
`nrow`	Number of grid rows

Value

matrix

Examples

```
ggfortify:::get.layout(3, 2, 2)
```

get_geom_function *Factory function to control* `ggplot2::geom_xxx` *functions*

Description

Factory function to control `ggplot2::geom_xxx` functions

Usage

```
get_geom_function(geom, allowed = c("line", "bar", "point"))
```

Arguments

`geom`	string representation of `ggplot2::geom_xxx` function
`allowed`	character vector contains allowed values

Value

function

Examples

```
ggfortify:::get_geom_function('point')
ggfortify:::get_geom_function('line', allowed = c('line'))
```

---

ggbiplot *Draw* biplot *using* ggplot2.

Description

Draw biplot using ggplot2.

Usage

```
ggbiplot(
  plot.data,
  loadings.data = NULL,
  colour = NULL,
  size = NULL,
  linetype = NULL,
  alpha = NULL,
  fill = NULL,
  shape = NULL,
  label = FALSE,
  label.label = "rownames",
  label.colour = colour,
  label.alpha = NULL,
  label.size = NULL,
  label.angle = NULL,
  label.family = NULL,
  label.fontface = NULL,
  label.lineheight = NULL,
  label.hjust = NULL,
  label.vjust = NULL,
  label.repel = FALSE,
  label.position = "identity",
  loadings = FALSE,
  loadings.arrow = grid::arrow(length = grid::unit(8, "points")),
  loadings.colour = "#FF0000",
  loadings.label = FALSE,
  loadings.label.label = "rownames",
  loadings.label.colour = "#FF0000",
  loadings.label.alpha = NULL,
  loadings.label.size = NULL,
  loadings.label.angle = NULL,
  loadings.label.family = NULL,
  loadings.label.fontface = NULL,
  loadings.label.lineheight = NULL,
  loadings.label.hjust = NULL,
  loadings.label.vjust = NULL,
  loadings.label.repel = FALSE,
  label.show.legend = NA,
  frame = FALSE,
  frame.type = NULL,
  frame.colour = colour,
  frame.level = 0.95,
  frame.alpha = 0.2,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL,
  ...
)
```

Arguments

`plot.data`	data.frame
`loadings.data`	data.frame
`colour`	colour
`size`	size
`linetype`	line type
`alpha`	alpha
`fill`	fill
`shape`	shape
`label`	Logical value whether to display data labels
`label.label`	Column name used for label text
`label.colour`	Colour for text labels
`label.alpha`	Alpha for text labels
`label.size`	Size for text labels
`label.angle`	Angle for text labels
`label.family`	Font family for text labels
`label.fontface`	Fontface for text labels
`label.lineheight`	Lineheight for text labels
`label.hjust`	Horizontal adjustment for text labels
`label.vjust`	Vertical adjustment for text labels
`label.repel`	Logical flag indicating whether to use `ggrepel`, enabling this may take some time for plotting

`label.position`	Character or a position function
`loadings`	Logical value whether to display loadings arrows
`loadings.arrow`	An arrow definition
`loadings.colour`	Point colour for data
`loadings.label`	Logical value whether to display loadings labels
`loadings.label.label`	Column name used for loadings text labels
`loadings.label.colour`	Colour for loadings text labels
`loadings.label.alpha`	Alpha for loadings text labels
`loadings.label.size`	Size for loadings text labels
`loadings.label.angle`	Angle for loadings text labels
`loadings.label.family`	Font family for loadings text labels
`loadings.label.fontface`	Fontface for loadings text labels
`loadings.label.lineheight`	Lineheight for loadings text labels
`loadings.label.hjust`	Horizontal adjustment for loadings text labels
`loadings.label.vjust`	Vertical adjustment for loadings text labels
`loadings.label.repel`	Logical flag indicating whether to use `ggrepel` automatically
`label.show.legend`	Logical value indicating whether to show the legend of text labels
`frame`	Logical value whether to draw outliner convex / ellipse
`frame.type`	Character specifying frame type. 'convex' or types supporeted by `ggplot2::stat_ellipse` can be used.
`frame.colour`	Colour for frame
`frame.level`	Passed for `ggplot2::stat_ellipse` 's level. Ignored in 'convex'.
`frame.alpha`	Alpha for frame
`xlim`	limits for x axis
`ylim`	limits for y axis
`log`	which variables to log transform ("x", "y", or "xy")
`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio
`...`	other arguments passed to methods

Value

ggplot

`ggcpgram` *Plots a cumulative periodogram*

Description

Plots a cumulative periodogram

Usage

```
ggcpgram(
  ts,
  taper = 0.1,
  colour = "#000000",
  linetype = "solid",
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "dashed",
  conf.int.fill = NULL,
  conf.int.alpha = 0.3
)
```

Arguments

`ts`	`stats::ts` instance
`taper`	Proportion tapered in forming the periodogram
`colour`	Line colour
`linetype`	Line type
`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals
`conf.int.fill`	fill colour for confidence intervals
`conf.int.alpha`	alpha for confidence intervals

Value

ggplot

Examples

```
ggcpgram(AirPassengers)
```

`ggdistribution` *Plot distribution*

Description

Plot distribution

Usage

```
ggdistribution(
  func,
  x,
  p = NULL,
  colour = "#000000",
  linetype = NULL,
  fill = NULL,
  alpha = NULL,
  ...
)
```

Arguments

`func`	PDF or CDF function
`x`	Numeric vector to be passed to func
`p`	`ggplot2::ggplot` instance to plot
`colour`	Line colour
`linetype`	Line type
`fill`	Fill colour
`alpha`	Alpha
`...`	Keywords passed to PDC/CDF func

Value

ggplot

Examples

```
ggdistribution(dnorm, seq(-3, 3, 0.1), mean = 0, sd = 1)
ggdistribution(ppois, seq(0, 30), lambda = 20)

p <- ggdistribution(pchisq, 0:20, df = 7, fill = 'blue')
ggdistribution(pchisq, 0:20, p = p, df = 9, fill = 'red')
```

ggfortify *ggfortify*

Description

Define Fortify and Autoplot to Allow 'ggplot2' to Draw Some Popular Packages

ggfreqplot *Plot seasonal subseries of time series, generalization of* `stats::monthplot`

Description

Plot seasonal subseries of time series, generalization of `stats::monthplot`

Usage

```
ggfreqplot(
  data,
  freq = NULL,
  nrow = NULL,
  ncol = NULL,
  conf.int = FALSE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "dashed",
  conf.int.fill = NULL,
  conf.int.alpha = 0.3,
  conf.int.value = 0.95,
  facet.labeller = NULL,
  ...
)
```

Arguments

`data`	`stats::ts` instance
`freq`	Length of frequency. If not provided, use time-series frequency
`nrow`	Number of plot rows
`ncol`	Number of plot columns
`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals

conf.int.fill	fill colour for confidence intervals
conf.int.alpha	alpha for confidence intervals
conf.int.value	Coverage probability for confidence interval
facet.labeller	A vector used as facet labels
...	Keywords passed to autoplot.ts

Value

ggplot

Examples

```
ggfreqplot(AirPassengers)
ggfreqplot(AirPassengers, freq = 4)
ggfreqplot(AirPassengers, conf.int = TRUE)
```

ggmultiplot-class *An S4 class to hold multiple* ggplot2::ggplot *instances*

Description

An S4 class to hold multiple ggplot2::ggplot instances

Usage

```
## S4 method for signature 'ggmultiplot'
length(x)

## S4 method for signature 'ggmultiplot,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'ggmultiplot'
x[[i, j, ..., drop]]

## S4 replacement method for signature 'ggmultiplot,ANY,ANY,ANY'
x[i, j, ...] <- value

## S4 replacement method for signature 'ggmultiplot'
x[[i, j, ...]] <- value
```

Arguments

x	ggmultiplot
i	elements to extract or replace
j	not used
...	not used
drop	not used
value	value to be set

Slots

`plots` List of `ggplot2::ggplot` instances

`ncol` Number of grid columns

`nrow` Number of grid rows

`ggtsdiag` *Plots time-series diagnostics*

Description

Plots time-series diagnostics

Usage

```
ggtsdiag(
  object,
  gof.lag = 10,
  conf.int = TRUE,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "dashed",
  conf.int.fill = NULL,
  conf.int.alpha = 0.3,
  ad.colour = "#888888",
  ad.linetype = "dashed",
  ad.size = 0.2,
  nrow = NULL,
  ncol = 1,
  ...
)
```

Arguments

`object`	A fitted time-series model
`gof.lag`	The maximum number of lags for a Portmanteau goodness-of-fit test
`conf.int`	Logical flag indicating whether to plot confidence intervals
`conf.int.colour`	line colour for confidence intervals
`conf.int.linetype`	line type for confidence intervals
`conf.int.fill`	fill colour for confidence intervals
`conf.int.alpha`	alpha for confidence intervals
`ad.colour`	Line colour for additional lines
`ad.linetype`	Line type for additional lines
`ad.size`	Fill colour for additional lines

`nrow`	Number of facet/subplot rows
`ncol`	Number of facet/subplot columns
`...`	other keywords

Value

ggplot

Examples

```
ggtsdiag(arima(AirPassengers))
```

`grid.draw.ggmultiplot` *The implemented grid.draw method for ggmultiplot, in order to work with ggsave() properly*

Description

The implemented grid.draw method for ggmultiplot, in order to work with ggsave() properly

Usage

```
## S3 method for class 'ggmultiplot'
grid.draw(plot)
```

Arguments

`plot`	`ggmultiplot`

`infer` *Infer class name*

Description

Infer class name

Usage

```
infer(data)
```

Arguments

`data`	list instance

Value

character

is.univariate *Check if Validates number of* `ts` *variates*

Description

Check if Validates number of `ts` variates

Usage

```
is.univariate(data, raise = TRUE)
```

Arguments

`data`	`ts` instance
`raise`	Logical flag whether raise an error

Value

logical

Examples

```
ggfortify:::is.univariate(AirPassengers)
```

is_derived_from *Check object is target class, or object is* `data.frame` *fortified from target.*

Description

Check object is target class, or object is `data.frame` fortified from target.

Usage

```
is_derived_from(object, target)
```

Arguments

`object`	instance to be checked. For data.frame, check whether it is fortified from target class
`target`	class name

Value

logical

Examples

```
ggfortify:::is_derived_from(prcomp(iris[-5]), 'prcomp')
```

plot_confint *Attach confidence interval to* ggplot2::ggplot

Description

Attach confidence interval to ggplot2::ggplot

Usage

```
plot_confint(
  p,
  data = NULL,
  lower = "lower",
  upper = "upper",
  conf.int = TRUE,
  conf.int.geom = "line",
  conf.int.group = NULL,
  conf.int.colour = "#0000FF",
  conf.int.linetype = "none",
  conf.int.fill = "#000000",
  conf.int.alpha = 0.3
)
```

Arguments

p	ggplot2::ggplot instance
data	data contains lower and upper confidence intervals
lower	column name for lower confidence interval
upper	column name for upper confidence interval
conf.int	Logical flag indicating whether to plot confidence intervals
conf.int.geom	geometric string for confidence interval. 'line' or 'step'
conf.int.group	name of grouping variable for confidence intervals
conf.int.colour	line colour for confidence intervals
conf.int.linetype	line type for confidence intervals
conf.int.fill	fill colour for confidence intervals
conf.int.alpha	alpha for confidence intervals

Value

ggplot

Examples

```
d <- fortify(stats::acf(AirPassengers, plot = FALSE))
p <- ggplot(data = d, mapping = aes(x = Lag))
ggfortify:::plot_confint(p, data = d)
```

---

plot_label *Attach label to* ggplot2::ggplot

Description

Attach label to ggplot2::ggplot

Usage

```
plot_label(
  p,
  data,
  x = NULL,
  y = NULL,
  label = TRUE,
  label.label = "rownames",
  label.colour = NULL,
  label.alpha = NULL,
  label.size = NULL,
  label.angle = NULL,
  label.family = NULL,
  label.fontface = NULL,
  label.lineheight = NULL,
  label.hjust = NULL,
  label.vjust = NULL,
  label.repel = FALSE,
  label.show.legend = NA,
  label.position = "identity"
)
```

Arguments

p	ggplot2::ggplot instance
data	Data contains text label
x	x coordinates for label
y	y coordinates for label
label	Logical value whether to display labels
label.label	Column name used for label text
label.colour	Colour for text labels
label.alpha	Alpha for text labels

`label.size` Size for text labels

`label.angle` Angle for text labels

`label.family` Font family for text labels

`label.fontface` Fontface for text labels

`label.lineheight` Lineheight for text labels

`label.hjust` Horizontal adjustment for text labels

`label.vjust` Vertical adjustment for text labels

`label.repel` Logical flag indicating whether to use `ggrepel`, enabling this may take some time for plotting

`label.show.legend` Logical value indicating whether to show the legend of the text labels

`label.position` Character or a position function

Value

ggplot

`post_autoplot` *Post process for fortify. Based on* `ggplot2::qplot`

Description

Post process for fortify. Based on `ggplot2::qplot`

Usage

```
post_autoplot(
  p,
  xlim = c(NA, NA),
  ylim = c(NA, NA),
  log = "",
  main = NULL,
  xlab = NULL,
  ylab = NULL,
  asp = NULL
)
```

Arguments

`p` `ggplot2::ggplot` instances

`xlim` limits for x axis

`ylim` limits for y axis

`log` which variables to log transform ("x", "y", or "xy")

`main`	character vector or expression for plot title
`xlab`	character vector or expression for x axis label
`ylab`	character vector or expression for y axis label
`asp`	the y/x aspect ratio

Value

data.frame

Examples

```
p <- qplot(Petal.Length, Petal.Width, data = iris)
ggfortify:::post_autoplot(p, xlim = c(1, 5), ylim = c(1, 5), log = 'xy', main = 'title',
                          xlab = 'x', ylab = 'y', asp = 1.5)
```

`post_fortify` *Post process for fortify.*

Description

Post process for fortify.

Usage

```
post_fortify(data, klass = NULL)
```

Arguments

`data`	data.frame
`klass`	instance to be added as base_class attr, should be original model before fortified

Value

data.frame

print,ggmultiplot-method   *Generic print function for* ggmultiplot

Description

Generic print function for ggmultiplot

Usage

```
## S4 method for signature 'ggmultiplot'
print(x)
```

Arguments

x	ggmultiplot

rbind_ts   *Rbind original and predicted time-series-like instances as fortified* data.frame

Description

Rbind original and predicted time-series-like instances as fortified data.frame

Usage

```
rbind_ts(
  data,
  original,
  ts.connect = TRUE,
  index.name = "Index",
  data.name = "Data"
)
```

Arguments

data	Predicted/forecasted ts instance
original	Original ts instance
ts.connect	Logical frag indicates whether connects original time-series and predicted values
index.name	Specify column name for time series index
data.name	Specify column name for univariate time series data. Ignored in multivariate time series.

Value

data.frame

Examples

```
predicted <- predict(stats::HoltWinters(UKgas), n.ahead = 5, prediction.interval = TRUE)
rbind_ts(predicted, UKgas, ts.connect = TRUE)
```

residuals.ar *Calculate residuals for* `stats::ar`

Description

Calculate residuals for `stats::ar`

Usage

```
## S3 method for class 'ar'
residuals(object, ...)
```

Arguments

`object`	`stats::ar` instance
`...`	other keywords

Value

ts Residuals extracted from the object object.

Examples

```
residuals(ar(WWWusage))
```

show,ggmultiplot-method *Generic show function for* `ggmultiplot`

Description

Generic show function for `ggmultiplot`

Usage

```
## S4 method for signature 'ggmultiplot'
show(object)
```

Arguments

`object`	ggmultiplot

`support_autoplot` *Check if passed object is supported by* `ggplot2::autoplot`

Description

Check if passed object is supported by `ggplot2::autoplot`

Usage

```
support_autoplot(obj)
```

Arguments

`obj`	object

Value

logical

`unscale` *Backtransform* `scale`*-ed object*

Description

Backtransform `scale`-ed object

Usage

```
unscale(data, center = NULL, scale = NULL)
```

Arguments

`data`	Scaled data
`center`	Centered vector
`scale`	Scale vector

Value

data.frame

Examples

```
df <- iris[-5]
ggfortify::unscale(base::scale(df))
```

Index