

Package ‘broom’

February 13, 2017

Type Package

Title Convert Statistical Analysis Objects into Tidy Data Frames

Version 0.4.2

Date 2016-02-13

Maintainer David Robinson <admiral.david@gmail.com>

Description Convert statistical analysis objects from R into tidy data frames, so that they can more easily be combined, reshaped and otherwise processed with tools like 'dplyr', 'tidyr' and 'ggplot2'. The package provides three S3 generics: tidy, which summarizes a model's statistical findings such as coefficients of a regression; augment, which adds columns to the original data such as predictions, residuals and cluster assignments; and glance, which provides a one-row summary of model-level statistics.

Imports plyr, dplyr, tidyr, psych, stringr, reshape2, nlme, methods

Suggests knitr, boot, survival, gam, glmnet, lfe, Lahman, MASS, sp, maps, maptools, multcomp, testthat, lme4, zoo, lmtest, plm, biglm, ggplot2, nnet, geepack, AUC, ergm, network, statnet.common, xergm, btergm, binGroup, Hmisc, bbmle, gamlss, rstan, rstanarm, brms, coda, gmm, Matrix, ks, purrr, orcutt, mgcv, lmodel2, polCA, mclust, covr, lsmeans, betareg, robust, akima

URL <http://github.com/tidyverse/broom>

BugReports <http://github.com/tidyverse/broom/issues>

VignetteBuilder knitr

License MIT + file LICENSE

RoxygenNote 5.0.1

NeedsCompilation no

Author David Robinson [aut, cre],
Matthieu Gomez [ctb],
Boris Demeshev [ctb],
Dieter Menne [ctb],
Benjamin Nutter [ctb],

Luke Johnston [ctb],
 Ben Bolker [ctb],
 Francois Briatte [ctb],
 Jeffrey Arnold [ctb],
 Jonah Gabry [ctb],
 Luciano Selzer [ctb],
 Gavin Simpson [ctb],
 Jens Preussner [ctb],
 Jay Hesselberth [ctb],
 Hadley Wickham [ctb]

Repository CRAN

Date/Publication 2017-02-13 23:34:38

R topics documented:

aareg_tidiers	4
acf_tidiers	5
anova_tidiers	6
Arima_tidiers	8
auc_tidiers	9
augment	10
augment_columns	11
betareg_tidiers	11
biglm_tidiers	13
binDesign_tidiers	15
binWidth_tidiers	16
bootstrap	17
boot_tidiers	18
brms_tidiers	19
broom	21
btergm_tidiers	21
cch_tidiers	23
compact	25
confint.geeglm	25
confint_tidy	26
coxph_tidiers	26
cv.glmnet_tidiers	28
data.frame_tidiers	30
ergm_tidiers	32
felm_tidiers	34
finish_glance	36
fitdistr_tidiers	37
fix_data_frame	38
gamlss_tidiers	39
gam_tidiers	40
geeglm_tidiers	41
glance	42

glmnet_tidiers	43
glm_tidiers	44
gmm_tidiers	45
htest_tidiers	48
inflate	49
insert_NAs	50
kappa_tidiers	50
kde_tidiers	51
kmeans_tidiers	52
list_tidiers	54
lme4_tidiers	55
lmodel2_tidiers	57
lm_tidiers	59
loess_tidiers	62
lsmeans_tidiers	64
matrix_tidiers	66
mclust_tidiers	67
mcmc_tidiers	68
mle2_tidiers	70
multcomp_tidiers	71
multinom_tidiers	72
nlme_tidiers	74
nls_tidiers	76
optim_tidiers	78
orcutt_tidiers	79
plm_tidiers	80
poLCA_tidiers	82
prcomp_tidiers	84
process_ergm	86
process_geeglm	87
process_lm	87
process_rq	88
pyears_tidiers	88
rcorr_tidiers	90
ridgelm_tidiers	91
rlm_tidiers	92
robust_tidiers	93
rowwise_df_tidiers	95
rq_tidiers	96
rstanarm_tidiers	99
sexpfit_tidiers	101
smooth.spline_tidiers	102
sparse_tidiers	103
sp_tidiers	104
summary_tidiers	105
survfit_tidiers	106
survreg_tidiers	108
svd_tidiers	110

tidy	112
tidy.coefstest	112
tidy.default	113
tidy.density	114
tidy.dist	114
tidy.ftable	115
tidy.manova	116
tidy.map	117
tidy.NULL	118
tidy.numeric	118
tidy.pairwise.htest	119
tidy.power.htest	120
tidy.spec	121
tidy.table	122
tidy.ts	122
tidy.TukeyHSD	123
unrowname	124
xyz_tidiers	124
zoo_tidiers	125

Index**127**

aareg_tidiers	<i>Tidiers for aareg objects</i>
---------------	----------------------------------

Description

These tidy the coefficients of Aalen additive regression objects.

Usage

```
## S3 method for class 'aareg'
tidy(x, ...)

## S3 method for class 'aareg'
glance(x, ...)
```

Arguments

x	an "aareg" object
...	extra arguments (not used)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy.aareg returns one row for each coefficient, with the columns

term	name of coefficient
estimate	estimate of the slope
statistic	test statistic for coefficient
std.error	standard error of statistic
robust.se	robust version of standard error estimate
z	z score
p.value	p-value

glance returns a one-row data frame containing

statistic	chi-squared statistic
p.value	p-value based on chi-squared statistic
df	degrees of freedom used by coefficients

Examples

```
if (require("survival", quietly = TRUE)) {
  afit <- aareg(Surv(time, status) ~ age + sex + ph.ecog, data=lung,
               dfbeta=TRUE)
  summary(afit)
  tidy(afit)
}
```

 acf_tidiers

Tidying method for the acf function

Description

Tidy an "acf" object, which is the output of acf and the related pcf and ccf functions.

Usage

```
## S3 method for class 'acf'
tidy(x, ...)
```

Arguments

x	acf object
...	(not used)

Value

data.frame with columns

lag	lag values
acf	calculated correlation

Examples

```
# acf
result <- acf(lh, plot=FALSE)
tidy(result)

# ccf
result <- ccf(mdeaths, fdeaths, plot=FALSE)
tidy(result)

# pcf
result <- pacf(lh, plot=FALSE)
tidy(result)

# lag plot
library(ggplot2)
result <- tidy(acf(lh, plot=FALSE))
p <- ggplot(result, aes(x=lag, y=acf)) +
  geom_bar(stat='identity', width=0.1) +
  theme_bw()

p

# with confidence intervals
conf.level <- 0.95
# from \code{plot.acf} method
len.data <- length(lh) # same as acf$n.used
conf.int <- qnorm((1 + conf.level) / 2) / sqrt(len.data)
p + geom_hline(yintercept = c(-conf.int, conf.int),
  color='blue', linetype='dashed')
```

Description

Tidies the result of an analysis of variance into an ANOVA table. Only a tidy method is provided, not an augment or glance method.

Usage

```
## S3 method for class 'anova'  
tidy(x, ...)  
  
## S3 method for class 'aov'  
tidy(x, ...)  
  
## S3 method for class 'aovlist'  
tidy(x, ...)
```

Arguments

x	An object of class "anova", "aov", or "aovlist"
...	extra arguments (not used)

Details

Note that the "term" column of an ANOVA table can come with leading or trailing whitespace, which this tidying method trims.

Value

A data.frame with columns

term	Term within the model, or "Residuals"
df	Degrees of freedom used by this term in the model
sumsq	Sum of squares explained by this term
meansq	Mean of sum of squares among degrees of freedom
statistic	F statistic
p.value	P-value from F test

In the case of an "aovlist" object, there is also a stratum column describing the error stratum

Examples

```
a <- anova(lm(mpg ~ wt + qsec + disp, mtcars))  
tidy(a)  
  
a <- aov(mpg ~ wt + qsec + disp, mtcars)  
tidy(a)  
  
a1 <- aov(mpg ~ wt + qsec + Error(disp / am), mtcars)  
tidy(a1)
```

Description

These methods tidy the coefficients of ARIMA models of univariate time series.

Usage

```
## S3 method for class 'Arima'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)

## S3 method for class 'Arima'
glance(x, ...)
```

Arguments

x	An object of class "Arima"
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
...	extra arguments (not used)

Details

augment is not currently implemented, as it is not clear whether ARIMA predictions can or should be merged with the original data frame.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy returns one row for each coefficient in the model, with five columns:

term	The term in the nonlinear model being estimated and tested
estimate	The estimated coefficient
std.error	The standard error from the linear model

If conf.int = TRUE, also returns

conf.low	low end of confidence interval
conf.high	high end of confidence interval

glance returns one row with the columns

sigma	the square root of the estimated residual variance
logLik	the data's log-likelihood under the model
AIC	the Akaike Information Criterion
BIC	the Bayesian Information Criterion

See Also[arima](#)**Examples**

```
fit <- arima(lh, order = c(1, 0, 0))
tidy(fit)
glance(fit)
```

`auc_tidiers`*Tidiers for objects from the AUC package*

Description

Tidy "roc" objects from the "auc" package. This can be used to, for example, draw ROC curves in `ggplot2`.

Usage

```
## S3 method for class 'roc'
tidy(x, ...)
```

Arguments

<code>x</code>	an "roc" object
<code>...</code>	Additional arguments, not used

Value

A data frame with three columns:

<code>cutoff</code>	The cutoff of the prediction scores used for classification
<code>tpr</code>	The resulting true positive rate at that cutoff
<code>fpr</code>	The resulting false positive rate at that cutoff

If the labels had names, those are added as an "instance" column.

Examples

```
if (require("AUC", quietly = TRUE)) {
  data(churn)
  r <- roc(churn$predictions, churn$labels)

  td <- tidy(r)
  head(td)
```

```
library(ggplot2)
ggplot(td, aes(fpr, tpr)) +
  geom_line()

# compare the ROC curves for two prediction algorithms
library(dplyr)
library(tidyr)

rocs <- churn %>%
  tidyr::gather(algorithm, value, -labels) %>%
  group_by(algorithm) %>%
  do(tidy(roc(.$value, .$labels)))

ggplot(rocs, aes(fpr, tpr, color = algorithm)) +
  geom_line()
}
```

augment

Augment data according to a tidied model

Description

Given an R statistical model or other non-tidy object, add columns to the original dataset such as predictions, residuals and cluster assignments.

Usage

```
augment(x, ...)
```

Arguments

x	model or other R object to convert to data frame
...	other arguments passed to methods

Details

Note that by convention the first argument is almost always data, which specifies the original data object. This is not part of the S3 signature, partly because it prevents [rowwise_df_tidiers](#) from taking a column name as the first argument.

This generic originated in the `ggplot2` package, where it was called "fortify."

See Also

[augment.lm](#)

augment_columns	<i>add fitted values, residuals, and other common outputs to an augment call</i>
-----------------	--

Description

Add fitted values, residuals, and other common outputs to the value returned from `augment`.

Usage

```
augment_columns(x, data, newdata, type, type.predict = type,
               type.residuals = type, se.fit = TRUE, ...)
```

Arguments

<code>x</code>	a model
<code>data</code>	original data onto which columns should be added
<code>newdata</code>	new data to predict on, optional
<code>type</code>	Type of prediction and residuals to compute
<code>type.predict</code>	Type of prediction to compute; by default same as <code>type</code>
<code>type.residuals</code>	Type of residuals to compute; by default same as <code>type</code>
<code>se.fit</code>	Value to pass to <code>predict</code> 's <code>se.fit</code> , or <code>NULL</code> for no value
<code>...</code>	extra arguments (not used)

Details

In the case that a residuals or influence generic is not implemented for the model, fail quietly.

betareg_tidiers	<i>Tidy betareg objects from the betareg package</i>
-----------------	--

Description

Tidy beta regression objects into summarized coefficients, add their fitted values and residuals, or find their model parameters.

Usage

```
## S3 method for class 'betareg'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)

## S3 method for class 'betareg'
augment(x, data = stats::model.frame(x), newdata,
        type.predict, type.residuals, ...)

## S3 method for class 'betareg'
glance(x, ...)
```

Arguments

x	A "betareg" object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
...	Extra arguments, not used
data	Original data frame the regression was fit on
newdata	New data frame to use for prediction
type.predict	Type of predictions to calculate
type.residuals	Type of residuals to calculate

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy returns a data.frame with one row for each term used to predict the mean, along with at least one term used to predict phi (the inverse of the variance). It starts with the column component containing either "mean" or "precision" to describe which is being modeled, then has the same columns as tidied linear models or glm's (see [lm_tidiers](#)).

augment returns the original data, along with new columns describing each observation:

.fitted	Fitted values of model
.resid	Residuals
.cooks	Cooks distance, cooks.distance

glance returns a one-row data.frame with the columns

pseudo.r.squared	the deviance of the null model
logLik	the data's log-likelihood under the model
AIC	the Akaike Information Criterion
BIC	the Bayesian Information Criterion
df.residual	residual degrees of freedom
df.null	degrees of freedom under the null

Examples

```

if (require("betareg", quietly = TRUE)) {
  data("GasolineYield", package = "betareg")

  mod <- betareg(yield ~ batch + temp, data = GasolineYield)

  mod
  tidy(mod)
  tidy(mod, conf.int = TRUE)
  tidy(mod, conf.int = TRUE, conf.level = .99)

  head(augment(mod))

  glance(mod)
}

```

biglm_tidiers

Tidiers for biglm and bigglm object

Description

Tidiers for biglm object from the "biglm" package, which contains a linear model object that is limited in memory usage. Generally the behavior is as similar to the [lm_tidiers](#) as is possible. Currently no augment is defined.

Usage

```

## S3 method for class 'biglm'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = FALSE, quick = FALSE, ...)

## S3 method for class 'biglm'
glance(x, ...)

```

Arguments

x	a "biglm" object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals (typical for logistic regression)
quick	whether to compute a smaller and faster version, containing only the term and estimate columns.
...	extra arguments (not used)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy.biglm` returns one row for each coefficient, with columns

<code>term</code>	The term in the linear model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the linear model
<code>p.value</code>	two-sided p-value

If `conf.int=TRUE`, it also includes columns for `conf.low` and `conf.high`, computed with `confint`.

`glance.biglm` returns a one-row data frame, with columns

<code>r.squared</code>	The percent of variance explained by the model
<code>AIC</code>	the Akaike Information Criterion
<code>deviance</code>	deviance
<code>df.residual</code>	residual degrees of freedom

Examples

```
if (require("biglm", quietly = TRUE)) {
  bfit <- biglm(mpg ~ wt + disp, mtcars)
  tidy(bfit)
  tidy(bfit, conf.int = TRUE)
  tidy(bfit, conf.int = TRUE, conf.level = .9)

  glance(bfit)

  # bigglm: logistic regression
  bgfit <- bigglm(am ~ mpg, mtcars, family = binomial())
  tidy(bgfit)
  tidy(bgfit, exponentiate = TRUE)
  tidy(bgfit, conf.int = TRUE)
  tidy(bgfit, conf.int = TRUE, conf.level = .9)
  tidy(bgfit, conf.int = TRUE, conf.level = .9, exponentiate = TRUE)

  glance(bgfit)
}
```

binDesign_tidiers *Tidy a binDesign object*

Description

Tidy a binDesign object from the "binGroup" package, which determines the sample size needed for a particular power.

Usage

```
## S3 method for class 'binDesign'
tidy(x, ...)
```

```
## S3 method for class 'binDesign'
glance(x, ...)
```

Arguments

x	A "binDesign" object
...	Extra arguments (not used)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

The tidy method returns a data.frame with one row for each iteration that was performed, with columns

n	Number of trials in this iteration
power	The power achieved for this n

The glance method returns a one-row data.frame with columns

power	The power achieved by the analysis
n	The sample size used to achieve this power
power.reached	Whether the desired power was reached
maxit	Number of iterations performed

Examples

```
if (require("binGroup", quietly = TRUE)) {
  des <- binDesign(nmax = 300, delta = 0.06,
                  p.hyp = 0.1, power = .8)

  glance(des)
  head(tidy(des))
}
```

```

# the ggplot2 equivalent of plot(des)
library(ggplot2)
ggplot(tidy(des), aes(n, power)) +
  geom_line()
}

```

binWidth_tidiers	<i>Tidy a binWidth object</i>
------------------	-------------------------------

Description

Tidy a binWidth object from the "binGroup" package, which calculates the expected width of a confidence interval from a binomial test.

Usage

```

## S3 method for class 'binWidth'
tidy(x, ...)

```

Arguments

x	A "binWidth" object
...	Extra arguments (not used)

Value

A one-row data.frame with columns:

ci.width	Expected width of confidence interval
alternative	Alternative hypothesis
p	True proportion
n	Total sample size

Examples

```

if (require("binGroup", quietly = TRUE)) {
  bw <- binWidth(100, .1)
  bw
  tidy(bw)

  library(dplyr)
  d <- expand.grid(n = seq(100, 800, 100),
                 p = .5,
                 method = c("CP", "Blaker", "Score", "Wald"),
                 stringsAsFactors = FALSE) %>%
    group_by(n, p, method) %>%

```



```
do(tidy(binWidth(.$n, .$p, method = .$method)))

library(ggplot2)
ggplot(d, aes(n, ci.width, color = method)) +
  geom_line() +
  xlab("Total Observations") +
  ylab("Expected CI Width")
}
```

bootstrap*Set up bootstrap replicates of a dplyr operation*

Description

Set up bootstrap replicates of a dplyr operation

Usage

```
bootstrap(df, m, by_group = FALSE)
```

Arguments

df	a data frame
m	number of bootstrap replicates to perform
by_group	If TRUE, then bootstrap within each group if df is a grouped tbl.

Details

This code originates from Hadley Wickham (with a few small corrections) here:

<https://github.com/hadley/dplyr/issues/269>

Some examples can be found at

<https://github.com/dgrtwo/broom/blob/master/vignettes/bootstrapping.Rmd>

Examples

```
library(dplyr)
mtcars %>% bootstrap(10) %>% do(tidy(lm(mpg ~ wt, .)))
```

boot_tidiers

*Tidying methods for bootstrap computations***Description**

Tidying methods for "boot" objects from the "boot" package.

Usage

```
## S3 method for class 'boot'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     conf.method = "perc", ...)
```

Arguments

x	boot object
conf.int	whether to include a confidence interval
conf.level	confidence level for CI
conf.method	method for computing confidence intervals (see boot.ci)
...	extra arguments (not used)

Value

The tidy method returns a data frame with one row per bootstrapped statistic that was calculated, and the following columns:

term	Name of the computed statistic, if present
statistic	The original values of the statistic
bias	The bias of the original statistic value
std.error	Standard error of the statistic

If weights were provided to the boot function, an estimate column is included showing the weighted bootstrap estimate, and the standard error is of that estimate.

If there are no original statistics in the "boot" object, such as with a call to `tsboot` with `orig.t = FALSE`, the original and statistic columns are omitted, and only estimate and std.error columns shown.

Examples

```
if (require("boot")) {
  clotting <- data.frame(
    u = c(5, 10, 15, 20, 30, 40, 60, 80, 100),
    lot1 = c(118, 58, 42, 35, 27, 25, 21, 19, 18),
    lot2 = c(69, 35, 26, 21, 18, 16, 13, 12, 12))

  g1 <- glm(lot2 ~ log(u), data = clotting, family = Gamma)
```

```

bootfun <- function(d, i) {
  coef(update(g1, data= d[i,]))
}
bootres <- boot(clotting, bootfun, R = 999)
tidy(g1, conf.int=TRUE)
tidy(bootres, conf.int=TRUE)
}

```

brms_tidiers

*Tidying methods for a brms model***Description**

These methods tidy the estimates from [brmsfit-objects](#) (fitted model objects from the **brms** package) into a summary.

Usage

```

## S3 method for class 'brmsfit'
tidy(x, parameters = NA, par_type = c("all",
  "non-varying", "varying", "hierarchical"), robust = FALSE,
  intervals = TRUE, prob = 0.9, ...)

```

Arguments

x	Fitted model object from the brms package. See brmsfit-class .
parameters	Names of parameters for which a summary should be returned, as given by a character vector or regular expressions. If NA (the default) summarized parameters are specified by the par_type argument.
par_type	One of "all", "non-varying", "varying", or "hierarchical" (can be abbreviated). See the Value section for details.
robust	Whether to use median and median absolute deviation rather than mean and standard deviation.
intervals	If TRUE columns for the lower and upper bounds of posterior uncertainty intervals are included.
prob	Defines the range of the posterior uncertainty intervals, such that $100 * prob\%$ of the parameter's posterior distribution lies within the corresponding interval. Only used if intervals = TRUE.
...	Extra arguments, not used

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

When `parameters = NA`, the `par_type` argument is used to determine which parameters to summarize.

Generally, `tidy.brmsfit` returns one row for each coefficient, with at least three columns:

<code>term</code>	The name of the model parameter.
<code>estimate</code>	A point estimate of the coefficient (mean or median).
<code>std.error</code>	A standard error for the point estimate (sd or mad).

When `par_type = "non-varying"`, only population-level effects are returned.

When `par_type = "varying"`, only group-level effects are returned. In this case, two additional columns are added:

<code>group</code>	The name of the grouping factor.
<code>level</code>	The name of the level of the grouping factor.

Specifying `par_type = "hierarchical"` selects the standard deviations and correlations of the group-level parameters.

If `intervals = TRUE`, columns for the lower and upper bounds of the posterior intervals computed.

See Also

[brms](#), [brmsfit-class](#)

Examples

```
## Not run:
library(brms)
fit <- brm(mpg ~ wt + (1|cyl) + (1+wt|gear), data = mtcars,
          iter = 500, chains = 2)
tidy(fit)
tidy(fit, parameters = "^sd_", intervals = FALSE)
tidy(fit, par_type = "non-varying")
tidy(fit, par_type = "varying")
tidy(fit, par_type = "hierarchical", robust = TRUE)

## End(Not run)
```

 broom

Convert Statistical Analysis Objects into Tidy Data Frames

Description

Convert statistical analysis objects from R into tidy data frames, so that they can more easily be combined, reshaped and otherwise processed with tools like dplyr, tidyr and ggplot2. The package provides three S3 generics: tidy, which summarizes a model's statistical findings such as coefficients of a regression; augment, which adds columns to the original data such as predictions, residuals and cluster assignments; and glance, which provides a one-row summary of model-level statistics.

 btergm_tidiers

Tidying method for a bootstrapped temporal exponential random graph model

Description

This method tidies the coefficients of a bootstrapped temporal exponential random graph model estimated with the **xergm**. It simply returns the coefficients and their confidence intervals.

Usage

```
## S3 method for class 'btergm'
tidy(x, conf.level = 0.95, exponentiate = FALSE,
     quick = FALSE, ...)
```

Arguments

x	a btergm object
conf.level	confidence level of the bootstrapped interval
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals
quick	whether to compute a smaller and faster version, containing only the term and estimate columns.
...	extra arguments (currently not used)

Details

There is no augment or glance method for **ergm** objects.

Value

A data.frame without rownames.

tidy.btergm returns one row for each coefficient, with four columns:

term	The term in the model being estimated and tested
estimate	The estimated coefficient
conf.low	The lower bound of the confidence interval
conf.high	The lower bound of the confidence interval

See Also

[btergm](#)

Examples

```
if (require("xergm")) {
  # Using the same simulated example as the xergm package
  # Create 10 random networks with 10 actors
  networks <- list()
  for(i in 1:10){
    mat <- matrix(rbinom(100, 1, .25), nrow = 10, ncol = 10)
    diag(mat) <- 0
    nw <- network::network(mat)
    networks[[i]] <- nw
  }
  # Create 10 matrices as covariates
  covariates <- list()
  for (i in 1:10) {
    mat <- matrix(rnorm(100), nrow = 10, ncol = 10)
    covariates[[i]] <- mat
  }
  # Fit a model where the propensity to form ties depends
  # on the edge covariates, controlling for the number of
  # in-stars
  btfit <- btergm(networks ~ edges + istar(2) +
    edgecov(covariates), R = 100)

  # Show terms, coefficient estimates and errors
  tidy(btfit)

  # Show coefficients as odds ratios with a 99% CI
  tidy(btfit, exponentiate = TRUE, conf.level = 0.99)
}
```

cch_tidiers	<i>tidiers for case-cohort data</i>
-------------	-------------------------------------

Description

Tidiers for case-cohort analyses: summarize each estimated coefficient, or test the overall model.

Usage

```
## S3 method for class 'cch'
tidy(x, conf.level = 0.95, ...)

## S3 method for class 'cch'
glance(x, ...)
```

Arguments

x	a "cch" object
conf.level	confidence level for CI
...	extra arguments (not used)

Details

It is not clear what an augment method would look like, so none is provided. Nor is there currently any way to extract the covariance or the residuals.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy` returns a data.frame with one row for each term

term	name of term
estimate	estimate of coefficient
stderror	standard error
statistic	Z statistic
p.value	p-value
conf.low	low end of confidence interval
conf.high	high end of confidence interval

`glance` returns a one-row data.frame with the following columns:

score	score
rscore	rscore
p.value	p-value from Wald test

iter	number of iterations
n	number of predictions
nevent	number of events

See Also

[cch](#)

Examples

```

if (require("survival", quietly = TRUE)) {
  # examples come from cch documentation
  subcoh <- nwtco$in.subcohort
  selccoh <- with(nwtco, rel==1|subcoh==1)
  ccoh.data <- nwtco[selccoh,]
  ccoh.data$subcohort <- subcoh[selccoh]
  ## central-lab histology
  ccoh.data$histol <- factor(ccoh.data$histol,labels=c("FH","UH"))
  ## tumour stage
  ccoh.data$stage <- factor(ccoh.data$stage,labels=c("I","II","III","IV"))
  ccoh.data$age <- ccoh.data$age/12 # Age in years

  fit.ccP <- cch(Surv(edrel, rel) ~ stage + histol + age, data = ccoh.data,
                subcoh = ~subcohort, id= ~seqno, cohort.size = 4028)

  tidy(fit.ccP)

  # coefficient plot
  library(ggplot2)
  ggplot(tidy(fit.ccP), aes(x = estimate, y = term)) + geom_point() +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
    geom_vline(xintercept = 0)

  # compare between methods
  library(dplyr)
  fits <- data_frame(method = c("Prentice", "SelfPrentice", "LinYing")) %>%
    group_by(method) %>%
    do(tidy(cch(Surv(edrel, rel) ~ stage + histol + age, data = ccoh.data,
               subcoh = ~subcohort, id= ~seqno, cohort.size = 4028,
               method = .$method)))

  # coefficient plots comparing methods
  ggplot(fits, aes(x = estimate, y = term, color = method)) + geom_point() +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
    geom_vline(xintercept = 0)
}

```

compact	<i>Remove NULL items in a vector or list</i>
---------	--

Description

Remove NULL items in a vector or list

Usage

```
compact(x)
```

Arguments

x	a vector or list
---	------------------

confint.geeglm	<i>Confidence interval for geeglm objects</i>
----------------	---

Description

Generate confidence intervals for GEE analyses

Usage

```
## S3 method for class 'geeglm'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	The 'geeglm' object
parm	The parameter to calculate the confidence interval for. If not specified, the default is to calculate a confidence interval on all parameters (all variables in the model).
level	confidence level of the interval, used only if conf.int=TRUE
...	Additional parameters

Details

This function was taken from <http://stackoverflow.com/a/21221995/2632184>.

Value

Returns the upper and lower confidence intervals

confint_tidy	<i>Calculate confidence interval as a tidy data frame</i>
--------------	---

Description

Return a confidence interval as a tidy data frame. This directly wraps the `confint` function, but ensures it follows broom conventions: column names of `conf.low` and `conf.high`, and no row names

Usage

```
confint_tidy(x, conf.level = 0.95, func = stats::confint, ...)
```

Arguments

<code>x</code>	a model object for which <code>confint</code> can be calculated
<code>conf.level</code>	confidence level
<code>func</code>	Function to use for computing <code>confint</code>
<code>...</code>	extra arguments passed on to <code>confint</code>

Value

A data frame with two columns: `conf.low` and `conf.high`.

See Also

[confint](#)

coxph_tidiers	<i>Tidiers for coxph object</i>
---------------	---------------------------------

Description

Tidy the coefficients of a Cox proportional hazards regression model, construct predictions, or summarize the entire model into a single row.

Usage

```
## S3 method for class 'coxph'
tidy(x, exponentiate = FALSE, conf.int = 0.95, ...)

## S3 method for class 'coxph'
augment(x, data = stats::model.frame(x), newdata,
        type.predict = "lp", type.residuals = "martingale", ...)

## S3 method for class 'coxph'
glance(x, ...)
```

Arguments

x	"coxph" object
exponentiate	whether to report the estimate and confidence intervals on an exponential scale
conf.int	confidence level to be used for CI
...	Extra arguments, not used
data	original data for augment
newdata	new data on which to do predictions
type.predict	type of predicted value (see predict.coxph)
type.residuals	type of residuals (see residuals.coxph)

Details

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to `augment` and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Value

`tidy` returns a data.frame with one row for each term, with columns

estimate	estimate of slope
std.error	standard error of estimate
statistic	test statistic
p.value	p-value

`augment` returns the original data.frame with additional columns added:

.fitted	predicted values
.se.fit	standard errors
.resid	residuals (not present if newdata is provided)

`glance` returns a one-row data.frame with statistics calculated on the cox regression.

See Also

[na.action](#)

Examples

```

if (require("survival", quietly = TRUE)) {
  cfit <- coxph(Surv(time, status) ~ age + sex, lung)

  tidy(cfit)
  tidy(cfit, exponentiate = TRUE)

  lp <- augment(cfit, lung)
  risks <- augment(cfit, lung, type.predict = "risk")
  expected <- augment(cfit, lung, type.predict = "expected")

  glance(cfit)

  # also works on clogit models
  resp <- levels(logan$occupation)
  n <- nrow(logan)
  indx <- rep(1:n, length(resp))
  logan2 <- data.frame(logan[indx,],
                      id = indx,
                      tocc = factor(rep(resp, each=n)))
  logan2$case <- (logan2$occupation == logan2$tocc)

  cl <- clogit(case ~ tocc + tocc:education + strata(id), logan2)
  tidy(cl)
  glance(cl)

  library(ggplot2)
  ggplot(lp, aes(age, .fitted, color = sex)) + geom_point()
  ggplot(risks, aes(age, .fitted, color = sex)) + geom_point()
  ggplot(expected, aes(time, .fitted, color = sex)) + geom_point()
}

```

cv.glmnet_tidiers

Tidiers for glmnet cross-validation objects

Description

Tidying methods for cross-validation performed by `glmnet.cv`, summarizing the mean-squared-error across choices of the penalty parameter λ .

Usage

```
## S3 method for class 'cv.glmnet'
tidy(x, ...)
```

```
## S3 method for class 'cv.glmnet'
glance(x, ...)
```

Arguments

x a "cv.glmnet" object
 ... extra arguments (not used)

Details

No augment method exists for this class.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy produces a data.frame with one row per choice of lambda, with columns

lambda	penalty parameter lambda
estimate	estimate (median) of mean-squared error or other criterion
std.error	standard error of criterion
conf.high	high end of confidence interval on criterion
conf.low	low end of confidence interval on criterion
nzero	number of parameters that are zero at this choice of lambda

glance returns a one-row data.frame with the values

nulldev	null deviance
npasses	total passes over the data across all lambda values

Examples

```
if (require("glmnet", quietly = TRUE)) {
  set.seed(2014)

  nobs <- 100
  nvar <- 50
  real <- 5

  x <- matrix(rnorm(nobs * nvar), nobs, nvar)
  beta <- c(rnorm(real, 0, 1), rep(0, nvar - real))
  y <- c(t(beta) %*% t(x)) + rnorm(nvar, sd = 3)

  cvfit1 <- cv.glmnet(x,y)

  head(tidy(cvfit1))
  glance(cvfit1)

  library(ggplot2)
  tidied_cv <- tidy(cvfit1)
  glance_cv <- glance(cvfit1)
```

```

# plot of MSE as a function of lambda
g <- ggplot(tidied_cv, aes(lambda, estimate)) + geom_line() + scale_x_log10()
g

# plot of MSE as a function of lambda with confidence ribbon
g <- g + geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)
g

# plot of MSE as a function of lambda with confidence ribbon and choices
# of minimum lambda marked
g <- g + geom_vline(xintercept = glance_cv$lambda.min) +
  geom_vline(xintercept = glance_cv$lambda.1se, lty = 2)
g

# plot of number of zeros for each choice of lambda
ggplot(tidied_cv, aes(lambda, nzero)) + geom_line() + scale_x_log10()

# coefficient plot with min lambda shown
tidied <- tidy(cvfit1$glmnet.fit)
ggplot(tidied, aes(lambda, estimate, group = term)) + scale_x_log10() +
  geom_line() +
  geom_vline(xintercept = glance_cv$lambda.min) +
  geom_vline(xintercept = glance_cv$lambda.1se, lty = 2)
}

```

data.frame_tidiers *Tidiers for data.frame objects*

Description

These perform tidy summaries of data.frame objects. tidy produces summary statistics about each column, while glance simply reports the number of rows and columns. Note that augment.data.frame will throw an error.

Usage

```

## S3 method for class 'data.frame'
tidy(x, ...)

## S3 method for class 'data.frame'
augment(x, data, ...)

## S3 method for class 'data.frame'
glance(x, ...)

```

Arguments

x	A data.frame
...	extra arguments: for tidy, these are passed on to describe from psych package
data	data, not used

Details

The tidy method calls the psych method [describe](#) directly to produce its per-columns summary statistics.

Value

tidy.data.frame produces a data frame with one row per original column, containing summary statistics of each:

column	name of original column
n	Number of valid (non-NA) values
mean	mean
sd	standard deviation
median	median
trimmed	trimmed mean, with trim defaulting to .1
mad	median absolute deviation (from the median)
min	minimum value
max	maximum value
range	range
skew	skew
kurtosis	kurtosis
se	standard error

glance returns a one-row data.frame with

nrow	number of rows
ncol	number of columns
complete.obs	number of rows that have no missing values
na.fraction	fraction of values across all rows and columns that are missing

See Also

[describe](#)

Examples

```
td <- tidy(mtcars)
td

glance(mtcars)

library(ggplot2)
# compare mean and standard deviation
ggplot(td, aes(mean, sd)) + geom_point() +
  geom_text(aes(label = column), hjust = 1, vjust = 1) +
  scale_x_log10() + scale_y_log10() + geom_abline()
```

 ergm_tidiers

Tidying methods for an exponential random graph model

Description

These methods tidy the coefficients of an exponential random graph model estimated with the **ergm** package into a summary, and construct a one-row glance of the model's statistics. The methods should work with any model that conforms to the **ergm** class, such as those produced from weighted networks by the **ergm.count** package.

Usage

```
## S3 method for class 'ergm'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = FALSE, quick = FALSE, ...)

## S3 method for class 'ergm'
glance(x, deviance = FALSE, mcmc = FALSE, ...)
```

Arguments

x	an ergm object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals
quick	whether to compute a smaller and faster version, containing only the term and estimate columns.
...	extra arguments passed to summary.ergm
deviance	whether to report null and residual deviance for the model, along with degrees of freedom; defaults to FALSE
mcmc	whether to report MCMC interval, burn-in and sample size used to estimate the model; defaults to FALSE

Details

There is no `augment` method for **ergm** objects.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy.ergm` returns one row for each coefficient, with five columns:

<code>term</code>	The term in the model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error
<code>mcmc.error</code>	The MCMC error
<code>p.value</code>	The two-sided p-value

If `conf.int=TRUE`, it also includes columns for `conf.low` and `conf.high`.

`glance.ergm` returns a one-row `data.frame` with the columns

<code>independence</code>	Whether the model assumed dyadic independence
<code>iterations</code>	The number of iterations performed before convergence
<code>logLik</code>	If applicable, the log-likelihood associated with the model
<code>AIC</code>	The Akaike Information Criterion
<code>BIC</code>	The Bayesian Information Criterion

If `deviance=TRUE`, and if the model supports it, the data frame will also contain the columns

<code>null.deviance</code>	The null deviance of the model
<code>df.null</code>	The degrees of freedom of the null deviance
<code>residual.deviance</code>	The residual deviance of the model
<code>df.residual</code>	The degrees of freedom of the residual deviance

Last, if `mcmc=TRUE`, the data frame will also contain the columns

<code>MCMC.interval</code>	The interval used during MCMC estimation
<code>MCMC.burnin</code>	The burn-in period of the MCMC estimation
<code>MCMC.samplesize</code>	The sample size used during MCMC estimation

References

Hunter DR, Handcock MS, Butts CT, Goodreau SM, Morris M (2008b). **ergm**: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks. *Journal of Statistical Software*, 24(3). <http://www.jstatsoft.org/v24/i03/>.

See Also

[ergm](#), [control.ergm](#), [summary.ergm](#)

Examples

```
if (require("ergm")) {
  # Using the same example as the ergm package
  # Load the Florentine marriage network data
  data(florentine)

  # Fit a model where the propensity to form ties between
  # families depends on the absolute difference in wealth
  gest <- ergm(florentine ~ edges + absdiff("wealth"))

  # Show terms, coefficient estimates and errors
  tidy(gest)

  # Show coefficients as odds ratios with a 99% CI
  tidy(gest, exponentiate = TRUE, conf.int = TRUE, conf.level = 0.99)

  # Take a look at likelihood measures and other
  # control parameters used during MCMC estimation
  glance(gest)
  glance(gest, deviance = TRUE)
  glance(gest, mcmc = TRUE)
}
```

felm_tidiers

Tidying methods for models with multiple group fixed effects

Description

These methods tidy the coefficients of a linear model with multiple group fixed effects

Usage

```
## S3 method for class 'felm'
tidy(x, conf.int = FALSE, conf.level = 0.95, fe = FALSE,
     fe.error = fe, ...)

## S3 method for class 'felm'
augment(x, data = NULL, ...)

## S3 method for class 'felm'
glance(x, ...)
```

Arguments

x	felm object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
fe	whether to include estimates of fixed effects
fe.error	whether to include standard error of fixed effects
...	extra arguments (not used)
data	Original data, defaults to extracting it from the model

Details

If conf.int=TRUE, the confidence interval is computed

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy.felm returns one row for each coefficient. If fe=TRUE, it also includes rows for fixed effects estimates. There are five columns:

term	The term in the linear model being estimated and tested
estimate	The estimated coefficient
std.error	The standard error from the linear model
statistic	t-statistic
p.value	two-sided p-value

If cont.int=TRUE, it also includes columns for conf.low and conf.high, computed with [confint](#).

augment.felm returns one row for each observation, with multiple columns added to the original data:

.fitted	Fitted values of model
.resid	Residuals

If fixed effect are present,

.comp	Connected component
.fe_	Fixed effects (as many columns as factors)

glance.lm returns a one-row data.frame with the columns

r.squared	The percent of variance explained by the model
adj.r.squared	r.squared adjusted based on the degrees of freedom
sigma	The square root of the estimated residual variance
statistic	F-statistic
p.value	p-value from the F test
df	Degrees of freedom used by the coefficients
df.residual	residual degrees of freedom

Examples

```

if (require("lfe", quietly = TRUE)) {
  N=1e2
  DT <- data.frame(
    id = sample(5, N, TRUE),
    v1 = sample(5, N, TRUE),
    v2 = sample(1e6, N, TRUE),
    v3 = sample(round(runif(100,max=100),4), N, TRUE),
    v4 = sample(round(runif(100,max=100),4), N, TRUE)
  )

  result_felm <- felm(v2~v3, DT)
  tidy(result_felm)
  augment(result_felm)
  result_felm <- felm(v2~v3|id+v1, DT)
  tidy(result_felm, fe = TRUE)
  augment(result_felm)
  v1<-DT$v1
  v2 <- DT$v2
  v3 <- DT$v3
  id <- DT$id
  result_felm <- felm(v2~v3|id+v1)
  tidy(result_felm)
  augment(result_felm)
  glance(result_felm)
}

```

finish_glance	<i>Add logLik, AIC, BIC, and other common measurements to a glance of a prediction</i>
---------------	--

Description

A helper function for several functions in the glance generic. Methods such as logLik, AIC, and BIC are defined for many prediction objects, such as lm, glm, and nls. This is a helper function that adds them to a glance data.frame can be performed. If any of them cannot be computed, it fails quietly.

Usage

```
finish_glance(ret, x)
```

Arguments

ret	a one-row data frame (a partially complete glance)
x	the prediction model

Details

In one special case, deviance for objects of the `lmerMod` class from `lme4` is computed with `deviance(x, REML=FALSE)`.

Value

a one-row data frame with additional columns added, such as

<code>logLik</code>	log likelihoods
<code>AIC</code>	Akaike Information Criterion
<code>BIC</code>	Bayesian Information Criterion
<code>deviance</code>	deviance
<code>df.residual</code>	residual degrees of freedom

Each of these are produced by the corresponding generics

`fitdistr_tidiers` *Tidying methods for fitdistr objects from the MASS package*

Description

These methods tidies the parameter estimates resulting from an estimation of a univariate distribution's parameters.

Usage

```
## S3 method for class 'fitdistr'
tidy(x, ...)
```

```
## S3 method for class 'fitdistr'
glance(x, ...)
```

Arguments

<code>x</code>	An object of class "fitdistr"
<code>...</code>	extra arguments (not used)

Value

All tidying methods return a `data.frame` without rownames, whose structure depends on the method chosen.

`tidy.fitdistr` returns one row for each parameter that was estimated, with columns:

<code>term</code>	The term that was estimated
<code>estimate</code>	Estimated value
<code>std.error</code>	Standard error of estimate

glance.fitdistr returns a one-row data.frame with the columns

n	Number of observations used in estimation
logLik	log-likelihood of estimated data
AIC	Akaike Information Criterion
BIC	Bayesian Information Criterion

Examples

```
set.seed(2015)
x <- rnorm(100, 5, 2)

library(MASS)
fit <- fitdistr(x, dnorm, list(mean = 3, sd = 1))

tidy(fit)
glance(fit)
```

fix_data_frame	<i>Ensure an object is a data frame, with rownames moved into a column</i>
----------------	--

Description

Ensure an object is a data frame, with rownames moved into a column

Usage

```
fix_data_frame(x, newnames = NULL, newcol = "term")
```

Arguments

x	a data.frame or matrix
newnames	new column names, not including the rownames
newcol	the name of the new rownames column

Value

a data.frame, with rownames moved into a column and new column names assigned

gamlss_tidiers	<i>Tidying methods for gamlss objects</i>
----------------	---

Description

Tidying methods for "gamlss" objects from the gamlss package.

Usage

```
## S3 method for class 'gamlss'
tidy(x, quick = FALSE, ...)
```

Arguments

x	A "gamlss" object
quick	Whether to perform a fast version, and return only the coefficients
...	Extra arguments (not used)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

A data.frame with one row for each coefficient, containing columns

parameter	Type of coefficient being estimated: mu, sigma, nu, or tau
term	The term in the model being estimated and tested
estimate	The estimated coefficient
std.error	The standard error from the linear model
statistic	t-statistic
p.value	two-sided p-value

```
if (requireNamespace("gamlss", quietly = TRUE)) data(abdom) mod<-gamlss(y~pb(x),sigma.fo=~pb(x),family=BCT,
data=abdom, method=mixed(1,20))
```

```
tidy(mod)
```

Description

These methods tidy the coefficients of a "gam" object (generalized additive model) into a summary, augment the original data with information on the fitted values and residuals, and construct a one-row glance of the model's statistics.

Usage

```
## S3 method for class 'gam'
tidy(x, ...)
```

```
## S3 method for class 'gam'
glance(x, ...)
```

Arguments

x	gam object
...	extra arguments (not used)

Details

The "augment" method is handled by [lm_tidiers](#).

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy.gam` returns the tidied output of the parametric ANOVA for gam package with one row for each term in the formula. The columns match those in [anova_tidiers](#).

`glance.gam` returns a one-row data.frame with the columns

df	Degrees of freedom used by the coefficients
logLik	the data's log-likelihood under the model
AIC	the Akaike Information Criterion
BIC	the Bayesian Information Criterion
deviance	deviance
df.residual	residual degrees of freedom

See Also

[lm_tidiers](#), [anova_tidiers](#)

Examples

```

if (require("gam", quietly = TRUE)) {
  data(kyphosis)
  g <- gam(Kyphosis ~ s(Age,4) + Number, family = binomial, data = kyphosis)
  tidy(g)
  augment(g)
  glance(g)
}

```

geeglm_tidiere

*Tidying methods for generalized estimating equations models***Description**

These methods tidy the coefficients of generalized estimating equations models of the `geeglm` class from functions of the `geepack` package.

Usage

```

## S3 method for class 'geeglm'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = FALSE, quick = FALSE, ...)

```

Arguments

<code>x</code>	An object of class <code>geeglm</code> , such as from <code>geeglm</code>
<code>conf.int</code>	whether to include a confidence interval
<code>conf.level</code>	confidence level of the interval, used only if <code>conf.int=TRUE</code>
<code>exponentiate</code>	whether to exponentiate the coefficient estimates and confidence intervals (typical for log distributions)
<code>quick</code>	whether to compute a smaller and faster version, containing only the term and estimate columns.
<code>...</code>	Additional arguments to be passed to other methods. Currently not used.

Details

If `conf.int=TRUE`, the confidence interval is computed with the `confint.geeglm` function.

While `tidy` is supported for "geeglm" objects, `augment` and `glance` are not.

If you have missing values in your model data, you may need to refit the model with `na.action = na.exclude` or deal with the missingness in the data beforehand.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy.geeglm` returns one row for each coefficient, with five columns:

<code>term</code>	The term in the linear model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the GEE model
<code>statistic</code>	Wald statistic
<code>p.value</code>	two-sided p-value

If `conf.int=TRUE`, it also includes columns for `conf.low` and `conf.high`, computed with `confint.geeglm` (included as part of `broom`).

Examples

```
if (require('geepack')) {
  data(state)
  ds <- data.frame(state.region, state.x77)

  geefit <- geeglm(Income ~ Frost + Murder, id = state.region,
                  data = ds, family = gaussian,
                  constr = 'exchangeable')

  tidy(geefit)
  tidy(geefit, quick = TRUE)
  tidy(geefit, conf.int = TRUE)
}
```

<code>glance</code>	<i>Construct a single row summary "glance" of a model, fit, or other object</i>
---------------------	---

Description

`glance` methods always return either a one-row data frame, or `NULL`

Usage

```
glance(x, ...)
```

Arguments

<code>x</code>	model or other R object to convert to single-row data frame
<code>...</code>	other arguments passed to methods

glmnet_tidiers *Tidiers for LASSO or elasticnet regularized fits*

Description

Tidying methods for regularized fits produced by glmnet, summarizing the estimates across values of the penalty parameter lambda.

Usage

```
## S3 method for class 'glmnet'
tidy(x, ...)

## S3 method for class 'glmnet'
glance(x, ...)
```

Arguments

x a "glmnet" object
 ... extra arguments (not used)

Details

Note that while this representation of GLMs is much easier to plot and combine than the default structure, it is also much more memory-intensive. Do not use for extremely large, sparse matrices.

No augment method is yet provided even though the model produces predictions, because the input data is not tidy (it is a matrix that may be very wide) and therefore combining predictions with it is not logical. Furthermore, predictions make sense only with a specific choice of lambda.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy produces a data.frame with one row per combination of coefficient (including the intercept) and value of lambda for which the estimate is nonzero, with the columns:

term	coefficient name (V1...VN by default, along with "(Intercept)")
step	which step of lambda choices was used
estimate	estimate of coefficient
lambda	value of penalty parameter lambda
dev.ratio	fraction of null deviance explained at each value of lambda

glance returns a one-row data.frame with the values

nulldev	null deviance
npasses	total passes over the data across all lambda values

Examples

```

if (require("glmnet", quietly = TRUE)) {
  set.seed(2014)
  x <- matrix(rnorm(100*20),100,20)
  y <- rnorm(100)
  fit1 <- glmnet(x,y)

  head(tidy(fit1))
  glance(fit1)

  library(dplyr)
  library(ggplot2)

  tidied <- tidy(fit1) %>% filter(term != "(Intercept)")

  ggplot(tidied, aes(step, estimate, group = term)) + geom_line()
  ggplot(tidied, aes(lambda, estimate, group = term)) +
    geom_line() + scale_x_log10()

  ggplot(tidied, aes(lambda, dev.ratio)) + geom_line()

  # works for other types of regressions as well, such as logistic
  g2 <- sample(1:2, 100, replace=TRUE)
  fit2 <- glmnet(x, g2, family="binomial")
  head(tidy(fit2))
}

```

glm_tidiers

Tidying methods for a glm object

Description

Tidy a glm object. The tidy and augment methods are handled by [lm_tidiers](#).

Usage

```

## S3 method for class 'glm'
glance(x, ...)

```

Arguments

x	glm object
...	extra arguments, not used

Value

`tidy` and `augment` return the same values as do `tidy.lm` and `augment.lm`.

`glance` returns a one-row data.frame with the columns

<code>null.deviance</code>	the deviance of the null model
<code>df.null</code>	the residual degrees of freedom for the null model
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>deviance</code>	deviance
<code>df.residual</code>	residual degrees of freedom

See Also

`tidy.lm` and `augment.lm`. Also `glm`, which computes the values reported by the `glance` method.

Examples

```
g <- glm(am ~ mpg, mtcars, family = "binomial")
glance(g)
```

gmm_tidiers

Tidying methods for generalized method of moments "gmm" objects

Description

These methods tidy the coefficients of "gmm" objects from the `gmm` package, or glance at the model-wide statistics (especially the J-test).

Usage

```
## S3 method for class 'gmm'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = FALSE, quick = FALSE, ...)
```

```
## S3 method for class 'gmm'
glance(x, ...)
```

Arguments

<code>x</code>	gmm object
<code>conf.int</code>	whether to include a confidence interval
<code>conf.level</code>	confidence level of the interval, used only if <code>conf.int=TRUE</code>
<code>exponentiate</code>	whether to exponentiate the coefficient estimates and confidence intervals (typical for logistic regression)
<code>quick</code>	whether to compute a smaller and faster version, containing only the term and estimate columns (and confidence interval if requested, which may be slower)
<code>...</code>	extra arguments (not used)

Details

If `conf.int=TRUE`, the confidence interval is computed with the `confint` function.

Note that though the "gmm" object contains residuals and fitted values, there is not yet an `augment` method implemented. This is because the input to `gmm` is not tidy (it's a "wide" matrix), so it is not immediately clear what the augmented results should look like.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy.gmm` returns one row for each coefficient, with six columns:

<code>term</code>	The term in the model being estimated
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the linear model
<code>statistic</code>	t-statistic
<code>p.value</code>	two-sided p-value

If all the terms have `_` in them (e.g. `WМК_(Intercept)`), they are split into `variable` and `term`.

If `conf.int=TRUE`, it also includes columns for `conf.low` and `conf.high`, computed with `confint`.

`glance.gmm` returns a one-row `data.frame` with the columns

<code>df</code>	Degrees of freedom
<code>statistic</code>	Statistic from J-test for $E(g)=0$
<code>p.value</code>	P-value from J-test
<code>df.residual</code>	Residual degrees of freedom, if included in "gmm" object

Examples

```

if (require("gmm", quietly = TRUE)) {
  # examples come from the "gmm" package
  ## CAPM test with GMM
  data(Finance)
  r <- Finance[1:300, 1:10]
  rm <- Finance[1:300, "rm"]
  rf <- Finance[1:300, "rf"]

  z <- as.matrix(r-rf)
  t <- nrow(z)
  zm <- rm-rf
  h <- matrix(zm, t, 1)
  res <- gmm(z ~ zm, x = h)

  # tidy result
  tidy(res)
  tidy(res, conf.int = TRUE)
  tidy(res, conf.int = TRUE, conf.level = .99)

  # coefficient plot
  library(ggplot2)
  library(dplyr)
  tidy(res, conf.int = TRUE) %>%
    mutate(variable = reorder(variable, estimate)) %>%
    ggplot(aes(estimate, variable)) +
    geom_point() +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
    facet_wrap(~ term) +
    geom_vline(xintercept = 0, color = "red", lty = 2)

  # from a function instead of a matrix
  g <- function(theta, x) {
    e <- x[,2:11] - theta[1] - (x[,1] - theta[1]) %*% matrix(theta[2:11], 1, 10)
    gmat <- cbind(e, e*c(x[,1]))
    return(gmat) }

  x <- as.matrix(cbind(rm, r))
  res_black <- gmm(g, x = x, t0 = rep(0, 11))

  tidy(res_black)
  tidy(res_black, conf.int = TRUE)

  ## APT test with Fama-French factors and GMM

  f1 <- zm
  f2 <- Finance[1:300, "hml"] - rf
  f3 <- Finance[1:300, "smb"] - rf
  h <- cbind(f1, f2, f3)
  res2 <- gmm(z ~ f1 + f2 + f3, x = h)

```

```

td2 <- tidy(res2, conf.int = TRUE)
td2

# coefficient plot
td2 %>%
  mutate(variable = reorder(variable, estimate)) %>%
  ggplot(aes(estimate, variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
  facet_wrap(~ term) +
  geom_vline(xintercept = 0, color = "red", lty = 2)
}

```

htest_tidiers

Tidying methods for an htest object

Description

Tidies hypothesis test objects, such as those from `cor.test`, `t.test`, and `wilcox.test`, into a one-row data frame.

Usage

```

## S3 method for class 'htest'
tidy(x, ...)

## S3 method for class 'htest'
glance(x, ...)

```

Arguments

<code>x</code>	An object of class "htest"
<code>...</code>	extra arguments (not used)

Details

No augment method is provided for "htest", since there is no sense in which a hypothesis test generates one value for each observation.

Value

Both `tidy` and `glance` return the same output, a one-row data frame with one or more of the following columns:

<code>estimate</code>	Estimate of the effect size
<code>statistic</code>	Test statistic used to compute the p-value
<code>p.value</code>	P-value

parameter	Parameter field in the htest, typically degrees of freedom
conf.low	Lower bound on a confidence interval
conf.high	Upper bound on a confidence interval
estimate1	Sometimes two estimates are computed, such as in a two-sample t-test
estimate2	Sometimes two estimates are computed, such as in a two-sample t-test
method	Method used to compute the statistic as a string
alternative	Alternative hypothesis as a string

Which columns are included depends on the hypothesis test used.

Examples

```
tt <- t.test(rnorm(10))
tidy(tt)
glance(tt) # same output for all htests

tt <- t.test(mpg ~ am, data = mtcars)
tidy(tt)

wt <- wilcox.test(mpg ~ am, data = mtcars)
tidy(wt)

ct <- cor.test(mtcars$wt, mtcars$mpg)
tidy(ct)
```

inflation	<i>Expand a dataset to include all factorial combinations of one or more variables</i>
-----------	--

Description

Expand a dataset to include all factorial combinations of one or more variables

Usage

```
inflation(.data, ..., stringsAsFactors = FALSE)
```

Arguments

.data	a tbl
...	arguments
stringsAsFactors	logical specifying if character vectors are converted to factors.

Value

A tbl, grouped by the arguments in ...

insert_NAs	<i>insert a row of NAs into a data frame wherever another data frame has NAs</i>
------------	--

Description

insert a row of NAs into a data frame wherever another data frame has NAs

Usage

```
insert_NAs(x, original)
```

Arguments

x	data frame that has one row for each non-NA row in original
original	data frame with NAs

kappa_tidiers	<i>Tidy a kappa object from a Cohen's kappa calculation</i>
---------------	---

Description

Tidy a "kappa" object, from the [cohen.kappa](#) function in the psych package. This represents the agreement of two raters when using nominal scores.

Usage

```
## S3 method for class 'kappa'
tidy(x, ...)
```

Arguments

x	An object of class "kappa"
...	extra arguments (not used)

Details

Note that the alpha of the confidence interval is determined when the `cohen.kappa` function is originally run.

Value

A data.frame with columns

type	Either "weighted" or "unweighted"
estimate	The estimated value of kappa with this method
conf.low	Lower bound of confidence interval
conf.high	Upper bound of confidence interval

See Also

[cohen.kappa](#)

Examples

```
library(psych)

rater1 = 1:9
rater2 = c(1, 3, 1, 6, 1, 5, 5, 6, 7)
ck <- cohen.kappa(cbind(rater1, rater2))

tidy(ck)

# graph the confidence intervals
library(ggplot2)
ggplot(tidy(ck), aes(estimate, type)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high))
```

kde_tidiers

Tidy a kernel density estimate object from the ks package

Description

Tidy a kernel density estimate object, into a table with one row for each point in the estimated grid, and one column for each dimension (along with an estimate column with the estimated density).

Usage

```
## S3 method for class 'kde'
tidy(x, ...)
```

Arguments

x	A "ks" object from the kde package
...	Extra arguments, not used

Value

A data frame with one row for each point in the estimated grid. The result contains one column (named `x1`, `x2`, etc) for each dimension, and an `estimate` column containing the estimated density.

Examples

```
if (require("ks", quietly = TRUE)) {
  dat <- replicate(2, rnorm(100))
  k <- kde(dat)

  td <- tidy(k)
  head(td)

  library(ggplot2)
  ggplot(td, aes(x1, x2, fill = estimate)) +
    geom_tile() +
    theme_void()

  # also works with 3 dimensions
  dat3 <- replicate(3, rnorm(100))
  k3 <- kde(dat3)

  td3 <- tidy(k3)
  head(td3)
}
```

kmeans_tidiers

Tidying methods for kmeans objects

Description

These methods summarize the results of k-means clustering into three tidy forms. `tidy` describes the center and size of each cluster, `augment` adds the cluster assignments to the original data, and `glance` summarizes the total within and between sum of squares of the clustering.

Usage

```
## S3 method for class 'kmeans'
tidy(x, col.names = paste0("x", 1:ncol(x$centers)), ...)

## S3 method for class 'kmeans'
augment(x, data, ...)

## S3 method for class 'kmeans'
glance(x, ...)
```

Arguments

x	kmeans object
col.names	The names to call each dimension of the data in tidy. Defaults to x1, x2...
...	extra arguments, not used
data	Original data (required for augment)

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy` returns one row per cluster, with one column for each dimension in the data describing the center, followed by

size	The size of each cluster
withinss	The within-cluster sum of squares
cluster	A factor describing the cluster from 1:k

`augment` returns the original data with one extra column:

.cluster	The cluster assigned by the k-means algorithm
----------	---

`glance` returns a one-row `data.frame` with the columns

totss	The total sum of squares
tot.withinss	The total within-cluster sum of squares
betweenss	The total between-cluster sum of squares
iter	The number of (outer) iterations

See Also

[kmeans](#)

Examples

```
library(dplyr)
library(ggplot2)

set.seed(2014)
centers <- data.frame(cluster=factor(1:3), size=c(100, 150, 50),
  x1=c(5, 0, -3), x2=c(-1, 1, -2))
points <- centers %>% group_by(cluster) %>%
  do(data.frame(x1=rnorm(. $size[1], .$x1[1]),
    x2=rnorm(. $size[1], .$x2[1])))

k <- kmeans(points %>% dplyr::select(x1, x2), 3)
tidy(k)
head(augment(k, points))
glance(k)
```

```
ggplot(augment(k, points), aes(x1, x2)) +  
  geom_point(aes(color = .cluster)) +  
  geom_text(aes(label = cluster), data = tidy(k), size = 10)
```

list_tidiers*Tidiers for return values from functions that aren't S3 objects*

Description

This method handles the return values of functions that return lists rather than S3 objects, such as `optim`, `svd`, or [interp](#), and therefore cannot be handled by S3 dispatch.

Usage

```
## S3 method for class 'list'  
tidy(x, ...)  
  
## S3 method for class 'list'  
glance(x, ...)
```

Arguments

x	list object
...	extra arguments, passed to the tidying function

Details

Those tidiers themselves are implemented as functions of the form `tidy_<function>` or `glance_<function>` that are not exported.

See Also

[optim_tidiers](#), [xyz_tidiers](#), [svd_tidiers](#), [orcutt_tidiers](#)

Description

These methods tidy the coefficients of mixed effects models, particularly responses of the `merMod` class

Usage

```
## S3 method for class 'merMod'
tidy(x, effects = c("ran_pars", "fixed"), scales = NULL,
     ran_prefix = NULL, conf.int = FALSE, conf.level = 0.95,
     conf.method = "Wald", ...)

## S3 method for class 'merMod'
augment(x, data = stats::model.frame(x), newdata, ...)

## S3 method for class 'merMod'
glance(x, ...)
```

Arguments

<code>x</code>	An object of class <code>merMod</code> , such as those from <code>lmer</code> , <code>glmer</code> , or <code>nlmer</code>
<code>effects</code>	A character vector including one or more of "fixed" (fixed-effect parameters), "ran_pars" (variances and covariances or standard deviations and correlations of random effect terms) or "ran_modes" (conditional modes/BLUPs/latent variable estimates)
<code>scales</code>	scales on which to report the variables: for random effects, the choices are "sd-cor" (standard deviations and correlations: the default if <code>scales</code> is <code>NULL</code>) or "vcov" (variances and covariances). <code>NA</code> means no transformation, appropriate e.g. for fixed effects; inverse-link transformations (exponentiation or logistic) are not yet implemented, but may be in the future.
<code>ran_prefix</code>	a length-2 character vector specifying the strings to use as prefixes for self- (variance/standard deviation) and cross- (covariance/correlation) random effects terms
<code>conf.int</code>	whether to include a confidence interval
<code>conf.level</code>	confidence level for CI
<code>conf.method</code>	method for computing confidence intervals (see <code>lme4::confint.merMod</code>)
<code>...</code>	extra arguments (not used)
<code>data</code>	original data this was fitted on; if not given this will attempt to be reconstructed
<code>newdata</code>	new data to be used for prediction; optional

Details

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to augment and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy` returns one row for each estimated effect, either with groups depending on the effects parameter. It contains the columns

<code>group</code>	the group within which the random effect is being estimated: "fixed" for fixed effects
<code>level</code>	level within group (NA except for modes)
<code>term</code>	term being estimated
<code>estimate</code>	estimated coefficient
<code>std.error</code>	standard error
<code>statistic</code>	t- or Z-statistic (NA for modes)
<code>p.value</code>	P-value computed from t-statistic (may be missing/NA)

`augment` returns one row for each original observation, with columns (each prepended by a `.`) added. Included are the columns

<code>.fitted</code>	predicted values
<code>.resid</code>	residuals
<code>.fixed</code>	predicted values with no random effects

Also added for "merMod" objects, but not for "mer" objects, are values from the response object within the model (of type `lmResp`, `glmResp`, `nlsResp`, etc). These include `".mu"`, `".offset"`, `".sqrtXwt"`, `".sqrtrwt"`,

`glance` returns one row with the columns

<code>sigma</code>	the square root of the estimated residual variance
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>deviance</code>	deviance

See Also

[na.action](#)

Examples

```

if (require("lme4")) {
  # example regressions are from lme4 documentation
  lmm1 <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
  tidy(lmm1)
  tidy(lmm1, effects = "fixed")
  tidy(lmm1, effects = "fixed", conf.int=TRUE)
  tidy(lmm1, effects = "fixed", conf.int=TRUE, conf.method="profile")
  tidy(lmm1, effects = "ran_modes", conf.int=TRUE)
  head(augment(lmm1, sleepstudy))
  glance(lmm1)

  glmm1 <- glmer(cbind(incidence, size - incidence) ~ period + (1 | herd),
    data = cbpp, family = binomial)
  tidy(glmm1)
  tidy(glmm1, effects = "fixed")
  head(augment(glmm1, cbpp))
  glance(glmm1)

  startvec <- c(Asym = 200, xmid = 725, scal = 350)
  nm1 <- nlmer(circumference ~ SSlogis(age, Asym, xmid, scal) ~ Asym|Tree,
    Orange, start = startvec)
  tidy(nm1)
  tidy(nm1, effects = "fixed")
  head(augment(nm1, Orange))
  glance(nm1)
}

```

lmodel2_tidiers

*Tidiers for linear model II objects from the lmodel2 package***Description**

Tidy or glance an lmodel2 object. An lmodel2 represents model II simple linear regression, where both variables in the regression equation are random.

Usage

```

## S3 method for class 'lmodel2'
tidy(x, ...)

## S3 method for class 'lmodel2'
glance(x, ...)

```

Arguments

x	lmodel2 object
...	Extra arguments, not used

Details

Note that unlike linear regression, there are always only two terms in an `lmodel2`: Intercept and Slope. Furthermore, these are computed by four methods: OLS (ordinary least squares), MA (major axis), SMA (standard major axis), and RMA (ranged major axis). See the `lmodel2` documentation for more.

Note that there is no `augment` method for `lmodel2` objects because `lmodel2` does not provide a `predict` or `residuals` method (and since when both observations are random, fitted values and residuals have a less clear meaning).

Value

All tidying methods return a `data.frame` without `rownames`, whose structure depends on the method chosen.

`tidy` returns a data frame with one row for each combination of method (OLS/MA/SMA/RMA) and term (always Intercept/Slope). Its columns are:

method Either OLS/MA/SMA/RMA

term Either "Intercept" or "Slope"

estimate Estimated coefficient

conf.low Lower bound of 95% confidence interval

conf.high Upper bound of 95% confidence interval

`glance` returns a one-row data frame with columns

r.squared OLS R-squared

p.value OLS parametric p-value

theta Angle between OLS lines $\text{lm}(y \sim x)$ and $\text{lm}(x \sim y)$

H H statistic for computing confidence interval of major axis slope

Examples

```
if (require("lmodel2", quietly = TRUE)) {
  data(mod2ex2)
  Ex2.res <- lmodel2(Prey ~ Predators, data=mod2ex2, "relative", "relative", 99)
  Ex2.res

  tidy(Ex2.res)
  glance(Ex2.res)

  # this allows coefficient plots with ggplot2
  library(ggplot2)
  ggplot(tidy(Ex2.res), aes(estimate, term, color = method)) +
    geom_point() +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high))
}
```

Description

These methods tidy the coefficients of a linear model into a summary, augment the original data with information on the fitted values and residuals, and construct a one-row glance of the model's statistics.

Usage

```
## S3 method for class 'lm'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = FALSE, quick = FALSE, ...)

## S3 method for class 'summary.lm'
tidy(x, ...)

## S3 method for class 'lm'
augment(x, data = stats::model.frame(x), newdata, type.predict,
        type.residuals, ...)

## S3 method for class 'lm'
glance(x, ...)

## S3 method for class 'summary.lm'
glance(x, ...)
```

Arguments

x	lm object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals (typical for logistic regression)
quick	whether to compute a smaller and faster version, containing only the term and estimate columns.
...	extra arguments (not used)
data	Original data, defaults to the extracting it from the model
newdata	If provided, performs predictions on the new data
type.predict	Type of prediction to compute for a GLM; passed on to predict.glm
type.residuals	Type of residuals to compute for a GLM; passed on to residuals.glm

Details

If you have missing values in your model data, you may need to refit the model with `na.action = na.exclude`. If `conf.int=TRUE`, the confidence interval is computed with the `confint` function.

While `tidy` is supported for "mlm" objects, `augment` and `glance` are not.

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to `augment` and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Code and documentation for `augment.lm` originated in the `ggplot2` package, where it was called `fortify.lm`

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy.lm` returns one row for each coefficient, with five columns:

<code>term</code>	The term in the linear model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the linear model
<code>statistic</code>	t-statistic
<code>p.value</code>	two-sided p-value

If the linear model is an "mlm" object (multiple linear model), there is an additional column:

<code>response</code>	Which response column the coefficients correspond to (typically Y1, Y2, etc)
-----------------------	--

If `conf.int=TRUE`, it also includes columns for `conf.low` and `conf.high`, computed with `confint`.

When `newdata` is not supplied `augment.lm` returns one row for each observation, with seven columns added to the original data:

<code>.hat</code>	Diagonal of the hat matrix
<code>.sigma</code>	Estimate of residual standard deviation when corresponding observation is dropped from model
<code>.cooks</code>	Cooks distance, <code>cooks.distance</code>
<code>.fitted</code>	Fitted values of model
<code>.se.fit</code>	Standard errors of fitted values
<code>.resid</code>	Residuals
<code>.std.resid</code>	Standardised residuals

(Some unusual "lm" objects, such as "rlm" from MASS, may omit `.cooks` and `.std.resid`. "gam" from `mgcv` omits `.sigma`)

When `newdata` is supplied, `augment.lm` returns one row for each observation, with three columns added to the new data:

<code>.fitted</code>	Fitted values of model
<code>.se.fit</code>	Standard errors of fitted values
<code>.resid</code>	Residuals of fitted values on the new data

`glance.lm` returns a one-row data.frame with the columns

<code>r.squared</code>	The percent of variance explained by the model
<code>adj.r.squared</code>	<code>r.squared</code> adjusted based on the degrees of freedom
<code>sigma</code>	The square root of the estimated residual variance
<code>statistic</code>	F-statistic
<code>p.value</code>	p-value from the F test, describing whether the full regression is significant
<code>df</code>	Degrees of freedom used by the coefficients
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>deviance</code>	deviance
<code>df.residual</code>	residual degrees of freedom

See Also[summary.lm](#)[na.action](#)**Examples**

```
library(ggplot2)
library(dplyr)

mod <- lm(mpg ~ wt + qsec, data = mtcars)

tidy(mod)
glance(mod)

# coefficient plot
d <- tidy(mod) %>% mutate(low = estimate - std.error,
                          high = estimate + std.error)
ggplot(d, aes(estimate, term, xmin = low, xmax = high, height = 0)) +
  geom_point() +
  geom_vline(xintercept = 0) +
  geom_errorbarh()

head(augment(mod))
head(augment(mod, mtcars))

# predict on new data
newdata <- mtcars %>% head(6) %>% mutate(wt = wt + 1)
```

```

augment(mod, newdata = newdata)

au <- augment(mod, data = mtcars)

plot(mod, which = 1)
qplot(.fitted, .resid, data = au) +
  geom_hline(yintercept = 0) +
  geom_smooth(se = FALSE)
qplot(.fitted, .std.resid, data = au) +
  geom_hline(yintercept = 0) +
  geom_smooth(se = FALSE)
qplot(.fitted, .std.resid, data = au,
  colour = factor(cyl))
qplot(mpg, .std.resid, data = au, colour = factor(cyl))

plot(mod, which = 2)
qplot(sample = .std.resid, data = au, stat = "qq") +
  geom_abline()

plot(mod, which = 3)
qplot(.fitted, sqrt(abs(.std.resid)), data = au) + geom_smooth(se = FALSE)

plot(mod, which = 4)
qplot(seq_along(.cooksd), .cooksd, data = au)

plot(mod, which = 5)
qplot(.hat, .std.resid, data = au) + geom_smooth(se = FALSE)
ggplot(au, aes(.hat, .std.resid)) +
  geom_vline(size = 2, colour = "white", xintercept = 0) +
  geom_hline(size = 2, colour = "white", yintercept = 0) +
  geom_point() + geom_smooth(se = FALSE)

qplot(.hat, .std.resid, data = au, size = .cooksd) +
  geom_smooth(se = FALSE, size = 0.5)

plot(mod, which = 6)
ggplot(au, aes(.hat, .cooksd)) +
  geom_vline(xintercept = 0, colour = NA) +
  geom_abline(slope = seq(0, 3, by = 0.5), colour = "white") +
  geom_smooth(se = FALSE) +
  geom_point()
qplot(.hat, .cooksd, size = .cooksd / .hat, data = au) + scale_size_area()

# column-wise models
a <- matrix(rnorm(20), nrow = 10)
b <- a + rnorm(length(a))
result <- lm(b ~ a)
tidy(result)

```

Description

This method augments the original data with information on the fitted values and residuals, and optionally the standard errors.

Usage

```
## S3 method for class 'loess'  
augment(x, data = stats::model.frame(x), newdata, ...)
```

Arguments

x	A "loess" object
data	Original data, defaults to the extracting it from the model
newdata	If provided, performs predictions on the new data
...	extra arguments

Details

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to `augment` and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Value

When `newdata` is not supplied `augment.loess` returns one row for each observation with three columns added to the original data:

<code>.fitted</code>	Fitted values of model
<code>.se.fit</code>	Standard errors of the fitted values
<code>.resid</code>	Residuals of the fitted values

When `newdata` is supplied `augment.loess` returns one row for each observation with one additional column:

<code>.fitted</code>	Fitted values of model
<code>.se.fit</code>	Standard errors of the fitted values

See Also

[na.action](#)

Examples

```
lo <- loess(mpg ~ wt, mtcars)
augment(lo)

# with all columns of original data
augment(lo, mtcars)

# with a new dataset
augment(lo, newdata = head(mtcars))
```

lsmeans_tidiers

Tidy least-squares means objects from the lsmeans packages

Description

Tidiers for least-squares means objects, which report the predicted means for factors or factor combinations in a linear model. This covers two classes: `lsmobj` and `ref.grid`. (The two objects have slightly different purposes within the package but have similar output).

Usage

```
## S3 method for class 'lsmobj'
tidy(x, conf.level = 0.95, ...)

## S3 method for class 'ref.grid'
tidy(x, ...)
```

Arguments

<code>x</code>	"lsmobj" or "ref.grid" object
<code>conf.level</code>	Level of confidence interval, used only for <code>lsmobj</code> objects
<code>...</code>	Extra arguments, passed on to summary.ref.grid

Details

There are a large number of arguments that can be passed on to [summary.ref.grid](#). By broom convention, we use `conf.level` to pass the `level` argument.

Value

A data frame with one observation for each estimated mean, and one column for each combination of factors, along with the following variables:

<code>estimate</code>	Estimated least-squares mean
<code>std.error</code>	Standard error of estimate

df	Degrees of freedom
conf.low	Lower bound of confidence interval
conf.high	Upper bound of confidence interval

When the input is a contrast, each row will contain one estimated contrast, along with some of the following columns:

level1	One level of the factor being contrasted
level2	Second level
contrast	In cases where the contrast is not made up of two levels, describes each
statistic	T-ratio statistic
p.value	P-value

Examples

```
if (require("lsmeans", quietly = TRUE)) {
  # linear model for sales of oranges per day
  oranges_lm1 <- lm(sales1 ~ price1 + price2 + day + store, data = oranges)

  # reference grid (see lsmeans vignette)
  oranges_rg1 <- ref.grid(oranges_lm1)
  td <- tidy(oranges_rg1)
  head(td)

  # marginal averages
  marginal <- lsmeans(oranges_rg1, "day")
  tidy(marginal)

  # contrasts
  tidy(contrast(marginal))
  tidy(contrast(marginal, method = "pairwise"))

  # plot confidence intervals
  library(ggplot2)
  ggplot(tidy(marginal), aes(day, estimate)) +
    geom_point() +
    geom_errorbar(aes(ymin = conf.low, ymax = conf.high))

  # by multiple prices
  by_price <- lsmeans(oranges_lm1, "day", by = "price2",
                    at = list(price1 = 50, price2 = c(40, 60, 80),
                              day = c("2", "3", "4"))) )
  by_price
  tidy(by_price)

  ggplot(tidy(by_price), aes(price2, estimate, color = day)) +
    geom_line() +
    geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
}
```

`matrix_tidiers`*Tidiers for matrix objects*

Description

These perform tidying operations on matrix objects. `tidy` turns the matrix into a `data.frame` while bringing rownames, if they exist, in as a column called `.rownames` (since results of tidying operations never contain rownames). `glance` simply reports the number of rows and columns. Note that no `augment` method exists for matrices.

Usage

```
## S3 method for class 'matrix'
tidy(x, ...)

## S3 method for class 'matrix'
glance(x, ...)
```

Arguments

<code>x</code>	A matrix
<code>...</code>	extra arguments, not used

Value

`tidy.matrix` returns the original matrix converted into a `data.frame`, except that it incorporates rownames (if they exist) into a column called `.rownames`.

`glance` returns a one-row `data.frame` with

<code>nrow</code>	number of rows
<code>ncol</code>	number of columns
<code>complete.obs</code>	number of rows that have no missing values
<code>na.fraction</code>	fraction of values across all rows and columns that are missing

Examples

```
mat <- as.matrix(mtcars)
tidy(mat)
glance(mat)
```

mclust_tidiers	<i>Tidying methods for Mclust objects</i>
----------------	---

Description

These methods summarize the results of Mclust clustering into three tidy forms. `tidy` describes the size, mixing probability, mean and variability of each class, `augment` adds the class assignments and their probabilities to the original data, and `glance` summarizes the model parameters of the clustering.

Usage

```
## S3 method for class 'Mclust'
tidy(x, ...)

## S3 method for class 'Mclust'
augment(x, data, ...)

## S3 method for class 'Mclust'
glance(x, ...)
```

Arguments

<code>x</code>	Mclust object
<code>...</code>	extra arguments, not used
<code>data</code>	Original data (required for <code>augment</code>)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy` returns one row per component, with

<code>component</code>	A factor describing the cluster from 1:k (or 0:k in presence of a noise term in <code>x</code>)
<code>size</code>	The size of each component
<code>proportion</code>	The mixing proportion of each component
<code>variance</code>	In case of one-dimensional and spherical models, the variance for each component, omitted otherwise. NA for noise component
<code>mean</code>	The mean for each component. In case of two- or more dimensional models, a column with the mean is added for each dimension. NA for noise component

`augment` returns the original data with two extra columns:

<code>.class</code>	The class assigned by the Mclust algorithm
<code>.uncertainty</code>	The uncertainty associated with the classification

glance returns a one-row data.frame with the columns

model	A character string denoting the model at which the optimal BIC occurs
n	The number of observations in the data
G	The optimal number of mixture components
BIC	The optimal BIC value
logLik	The log-likelihood corresponding to the optimal BIC
df	The number of estimated parameters
hypvol	The hypervolume parameter for the noise component if required, otherwise set to NA

See Also

[Mclust](#)

Examples

```
library(dplyr)
library(ggplot2)
library(mclust)

set.seed(2016)
centers <- data.frame(cluster=factor(1:3), size=c(100, 150, 50),
                      x1=c(5, 0, -3), x2=c(-1, 1, -2))
points <- centers %>% group_by(cluster) %>%
  do(data.frame(x1=rnorm(. $size[1], .$x1[1]),
                x2=rnorm(. $size[1], .$x2[1]))) %>%
  ungroup()

m = Mclust(points %>% dplyr::select(x1, x2))

tidy(m)
head(augment(m, points))
glance(m)
```

Description

Tidying methods for MCMC (Stan, JAGS, etc.) fits

Usage

```
tidyMCMC(x, pars, estimate.method = "mean", conf.int = FALSE,
  conf.level = 0.95, conf.method = "quantile", droppars = "lp__",
  rhat = FALSE, ess = FALSE, ...)

## S3 method for class 'rjags'
tidy(x, pars, estimate.method = "mean", conf.int = FALSE,
  conf.level = 0.95, conf.method = "quantile", ...)

## S3 method for class 'stanfit'
tidy(x, pars, estimate.method = "mean", conf.int = FALSE,
  conf.level = 0.95, conf.method = "quantile", droppars = "lp__",
  rhat = FALSE, ess = FALSE, ...)
```

Arguments

x	an object of class "stanfit"
pars	(character) specification of which parameters to include
estimate.method	method for computing point estimate ("mean" or "median")
conf.int	(logical) include confidence interval?
conf.level	probability level for CI
conf.method	method for computing confidence intervals ("quantile" or "HPDinterval")
droppars	Parameters not to include in the output (such as log-probability information)
rhat, ess	(logical) include Rhat and/or effective sample size estimates?
...	unused

Examples

```
## Not run:

# Using example from "RStan Getting Started"
# https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started

model_file <- system.file("extdata", "8schools.stan", package = "broom")

schools_dat <- list(J = 8,
  y = c(28, 8, -3, 7, -1, 1, 18, 12),
  sigma = c(15, 10, 16, 11, 9, 11, 10, 18))

if (requireNamespace("rstan", quietly = TRUE)) {
  set.seed(2015)
  rstan_example <- stan(file = model_file, data = schools_dat,
    iter = 100, chains = 2)
}
```

```
## End(Not run)

if (requireNamespace("rstan", quietly = TRUE)) {
  # the object from the above code was saved as rstan_example.rda
  infile <- system.file("extdata", "rstan_example.rda", package = "broom")
  load(infile)

  tidy(rstan_example)
  tidy(rstan_example, conf.int = TRUE, pars = "theta")

  td_mean <- tidy(rstan_example, conf.int = TRUE)
  td_median <- tidy(rstan_example, conf.int = TRUE, estimate.method = "median")

  library(dplyr)
  library(ggplot2)
  tds <- rbind(mutate(td_mean, method = "mean"),
              mutate(td_median, method = "median"))

  ggplot(tds, aes(estimate, term)) +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
    geom_point(aes(color = method))
}
```

mle2_tidiers

Tidy mle2 maximum likelihood objects

Description

Tidy mle2 objects from the bbmle package.

Usage

```
## S3 method for class 'mle2'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	An "mle2" object
conf.int	Whether to add conf.low and conf.high columns
conf.level	Confidence level to use for interval
...	Extra arguments, not used

Examples

```

if (require("bbmle", quietly = TRUE)) {
  x <- 0:10
  y <- c(26, 17, 13, 12, 20, 5, 9, 8, 5, 4, 8)
  d <- data.frame(x,y)

  fit <- mle2(y ~ dpois(lambda = ymean),
             start = list(ymean = mean(y)), data = d)

  tidy(fit)
}

```

multcomp_tidiers

*tidying methods for objects produced by **multcomp***

Description

These methods originated in ggplot2, as "fortify." In broom, they were renamed "tidy" because they summarize terms and tests, rather than adding columns to a dataset.

Usage

```

## S3 method for class 'glht'
tidy(x, ...)

## S3 method for class 'confint.glht'
tidy(x, ...)

## S3 method for class 'summary.glht'
tidy(x, ...)

## S3 method for class 'cld'
tidy(x, ...)

```

Arguments

```

x          an object of class glht, confint.glht, summary.glht or cld
...        extra arguments (not used)

```

Examples

```

if (require("multcomp") && require("ggplot2")) {
  amod <- aov(breaks ~ wool + tension, data = warpbreaks)
  wht <- glht(amod, linfct = mcp(tension = "Tukey"))
}

```

```

tidy(wht)
ggplot(wht, aes(lhs, estimate)) + geom_point()

CI <- confint(wht)
tidy(CI)
ggplot(CI, aes(lhs, estimate, ymin = lwr, ymax = upr)) +
  geom_pointrange()

tidy(summary(wht))
ggplot(mapping = aes(lhs, estimate)) +
  geom_linerange(aes(ymin = lwr, ymax = upr), data = CI) +
  geom_point(aes(size = p), data = summary(wht)) +
  scale_size(trans = "reverse")

cld <- cld(wht)
tidy(cld)
}

```

multinom_tidiers

Tidying methods for multinomial logistic regression models

Description

These methods tidy the coefficients of multinomial logistic regression models generated by `multinom` of the `nnet` package.

Usage

```

## S3 method for class 'multinom'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = TRUE, ...)

## S3 method for class 'multinom'
glance(x, ...)

```

Arguments

<code>x</code>	A model object of class <code>multinom</code>
<code>conf.int</code>	whether to include a confidence interval
<code>conf.level</code>	confidence level of the interval, used only if <code>conf.int=TRUE</code>
<code>exponentiate</code>	whether to exponentiate the coefficient estimates and confidence intervals (typical for multinomial logistic regression)
<code>...</code>	extra arguments, not used

Details

If `conf.int=TRUE`, the confidence interval is computed with the `confint` function.

While `tidy` and `glance` are supported for "multinom" objects, `augment` is not.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy.multinom` returns one row for each coefficient at each level of the response variable, with six columns:

<code>y.value</code>	The response level
<code>term</code>	The term in the model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the linear model
<code>statistic</code>	Wald z-statistic
<code>p.value</code>	two-sided p-value

If `conf.int=TRUE`, it also includes columns for `conf.low` and `conf.high`, computed with `confint`.

`glance.multinom` returns a

`glance.multinom` returns a one-row `data.frame` with the columns

<code>edf</code>	The effective degrees of freedom
<code>deviance</code>	deviance
<code>AIC</code>	the Akaike Information Criterion

Examples

```
if (require(nnet) & require(MASS)){
  example(birthwt)
  bwt.mu <- multinom(low ~ ., bwt)
  tidy(bwt.mu)
  glance(bwt.mu)

  /* This model is a truly terrible model
  /* but it should show you what the output looks
  /* like in a multinomial logistic regression

  fit.gear <- multinom(gear ~ mpg + factor(am), data=mtcars)
  tidy(fit.gear)
  glance(fit.gear)
}
```

Description

These methods tidy the coefficients of mixed effects models of the `lme` class from functions of the `nlme` package.

Usage

```
## S3 method for class 'lme'
tidy(x, effects = "random", ...)

## S3 method for class 'lme'
augment(x, data = x$data, newdata, ...)

## S3 method for class 'lme'
glance(x, ...)
```

Arguments

<code>x</code>	An object of class <code>lme</code> , such as those from <code>lme</code> or <code>nlme</code>
<code>effects</code>	Either "random" (default) or "fixed"
<code>...</code>	extra arguments (not used)
<code>data</code>	original data this was fitted on; if not given this will attempt to be reconstructed
<code>newdata</code>	new data to be used for prediction; optional

Details

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to `augment` and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy` returns one row for each estimated effect, either random or fixed depending on the `effects` parameter. If `effects = "random"`, it contains the columns

<code>group</code>	the group within which the random effect is being estimated
<code>level</code>	level within group
<code>term</code>	term being estimated

estimate estimated coefficient

If effects="fixed", tidy returns the columns

term fixed term being estimated

estimate estimate of fixed effect

std.error standard error

statistic t-statistic

p.value P-value computed from t-statistic

augment returns one row for each original observation, with columns (each prepended by a .) added. Included are the columns

.fitted predicted values

.resid residuals

.fixed predicted values with no random effects

glance returns one row with the columns

sigma the square root of the estimated residual variance

logLik the data's log-likelihood under the model

AIC the Akaike Information Criterion

BIC the Bayesian Information Criterion

deviance returned as NA. To quote Brian Ripley on R-help: McCullagh & Nelder (1989) would be the authoritative reference, but the 1982 first edition manages to use 'deviance' in three separate senses on one page.

See Also

[na.action](#)

Examples

```
if (require("nlme") & require("lme4")) {
  # example regressions are from lme4 documentation, but used for nlme
  lmm1 <- lme(Reaction ~ Days, random=~ Days|Subject, sleepstudy)
  tidy(lmm1)
  tidy(lmm1, effects = "fixed")
  head(augment(lmm1, sleepstudy))
  glance(lmm1)
```

```
startvec <- c(Asym = 200, xmid = 725, scal = 350)
nm1 <- nlme(circumference ~ SSlogis(age, Asym, xmid, scal),
  data = Orange,
  fixed = Asym + xmid + scal ~1,
  random = Asym ~1,
  start = startvec)
```

```

  tidy(nm1)
  tidy(nm1, effects = "fixed")
  head(augment(nm1, Orange))
  glance(nm1)
}

```

nls_tidiers

Tidying methods for a nonlinear model

Description

These methods tidy the coefficients of a nonlinear model into a summary, augment the original data with information on the fitted values and residuals, and construct a one-row glance of the model's statistics.

Usage

```

## S3 method for class 'nls'
tidy(x, conf.int = FALSE, conf.level = 0.95, quick = FALSE,
     ...)

```

```

## S3 method for class 'nls'
augment(x, data = NULL, newdata = NULL, ...)

```

```

## S3 method for class 'nls'
glance(x, ...)

```

Arguments

x	An object of class "nls"
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
quick	whether to compute a smaller and faster version, containing only the term and estimate columns.
...	extra arguments (not used)
data	original data this was fitted on; if not given this will attempt to be reconstructed from nls (may not be successful)
newdata	new data frame to use for predictions

Details

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to `augment` and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

`tidy` returns one row for each coefficient in the model, with five columns:

<code>term</code>	The term in the nonlinear model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the linear model
<code>statistic</code>	t-statistic
<code>p.value</code>	two-sided p-value

`augment` returns one row for each original observation, with two columns added:

<code>.fitted</code>	Fitted values of model
<code>.resid</code>	Residuals

If `newdata` is provided, these are computed on based on predictions of the new data.

`glance` returns one row with the columns

<code>sigma</code>	the square root of the estimated residual variance
<code>isConv</code>	whether the fit successfully converged
<code>finTol</code>	the achieved convergence tolerance
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>deviance</code>	deviance
<code>df.residual</code>	residual degrees of freedom

See Also

[na.action](#)

[nls](#) and [summary.nls](#)

Examples

```
n <- nls(mpg ~ k * e ^ wt, data = mtcars, start = list(k = 1, e = 2))

tidy(n)
augment(n)
glance(n)

library(ggplot2)
ggplot(augment(n), aes(wt, mpg)) + geom_point() + geom_line(aes(y = .fitted))

# augment on new data
```

```
newdata <- head(mtcars)
newdata$wt <- newdata$wt + 1
augment(n, newdata = newdata)
```

 optim_tidiers

Tidiers for lists returned from optim

Description

Tidies objects returned by the `optim` function for general-purpose minimization and maximization.

Usage

```
tidy_optim(x, ...)
glance_optim(x, ...)
```

Arguments

<code>x</code>	list returned from <code>optim</code>
<code>...</code>	extra arguments

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy` returns a data frame with one row per parameter that was estimated, with columns

<code>parameter</code>	name of the parameter, or parameter1, parameter2... if the input vector is not named
------------------------	--

<code>value</code>	parameter value that minimizes or maximizes the output
--------------------	--

`glance` returns a one-row data frame with the columns

<code>value</code>	minimized or maximized output value
--------------------	-------------------------------------

<code>function.count</code>	number of calls to <code>fn</code>
-----------------------------	------------------------------------

<code>gradient.count</code>	number of calls to <code>gr</code>
-----------------------------	------------------------------------

<code>convergence</code>	convergence code representing the error state
--------------------------	---

See Also

[optim](#)

Examples

```
func <- function(x) {
  (x[1] - 2)^2 + (x[2] - 3)^2 + (x[3] - 8)^2
}

o <- optim(c(1, 1, 1), func)

tidy(o)
glance(o)
```

orcutt_tidiers	<i>Tidiers for Cochrane Orcutt object</i>
----------------	---

Description

Tidies a Cochrane Orcutt object, which estimates autocorrelation and beta coefficients in a linear fit.

Usage

```
## S3 method for class 'orcutt'
tidy(x, ...)

## S3 method for class 'orcutt'
glance(x, ...)
```

Arguments

`x` An "orcutt" object returned by `cochrane.orcutt`

`...` Extra arguments passed on to [tidy.lm](#)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy` returns the same information as [tidy.lm](#), though without confidence interval options.

`glance`

`glance` returns a one-row data frame with the following columns:

<code>r.squared</code>	R-squared
<code>adj.r.squared</code>	Adjusted R-squared
<code>rho</code>	Spearman's rho autocorrelation
<code>number.interaction</code>	Number of interactions

```

dw.original      Durbin-Watson statistic of original fit
p.value.original      P-value of original Durbin-Watson statistic
dw.transformed   Durbin-Watson statistic of transformed fit
p.value.transformed P-value of autocorrelation after transformation

```

Examples

```

reg <- lm(mpg ~ wt + qsec + disp, mtcars)
tidy(reg)

if (require("orcutt", quietly = TRUE)) {
  co <- cochrane.orcutt(reg)
  co

  tidy(co)
  glance(co)
}

```

plm_tidiers

Tidiers for panel regression linear models

Description

Tidiers for panel regression linear models

Usage

```

## S3 method for class 'plm'
tidy(x, conf.int = FALSE, conf.level = 0.95,
     exponentiate = FALSE, ...)

## S3 method for class 'plm'
augment(x, data = as.data.frame(stats::model.frame(x)), ...)

## S3 method for class 'plm'
glance(x, ...)

```

Arguments

x	a "plm" object representing a panel object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals
...	extra arguments, not used
data	original dataset

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy.plm returns a data frame with one row per coefficient, of the same form as [tidy.lm](#).

augment returns a data frame with one row for each initial observation, adding the columns

.fitted	predicted (fitted) values
.resid	residuals

glance returns a one-row data frame with columns

r.squared	The percent of variance explained by the model
adj.r.squared	r.squared adjusted based on the degrees of freedom
statistic	F-statistic
p.value	p-value from the F test, describing whether the full regression is significant
deviance	deviance
df.residual	residual degrees of freedom

See Also

[lm_tidiers](#)

Examples

```
if (require("plm", quietly = TRUE)) {
  data("Produc", package = "plm")
  zz <- plm(log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp,
            data = Produc, index = c("state", "year"))

  summary(zz)

  tidy(zz)
  tidy(zz, conf.int = TRUE)
  tidy(zz, conf.int = TRUE, conf.level = .9)

  head(augment(zz))

  glance(zz)
}
```

poLCA_tidiers

*Tidiers for poLCA objects***Description**

Tidiers for poLCA latent class regression models. Summarize the probabilities of each outcome for each variable within each class with `tidy`, add predictions to the data with `augment`, or find the log-likelihood/AIC/BIC with `glance`.

Usage

```
## S3 method for class 'poLCA'
tidy(x, ...)

## S3 method for class 'poLCA'
augment(x, data, ...)

## S3 method for class 'poLCA'
glance(x, ...)
```

Arguments

<code>x</code>	A poLCA object
<code>...</code>	Extra arguments, not used
<code>data</code>	For <code>augment</code> , the original dataset used to fit the latent class model. If not given, uses manifest variables in <code>x\$y</code> and, if applicable, covariates in <code>x\$x</code>

Value

All tidying methods return a `data.frame` without rownames, whose structure depends on the method chosen.

`tidy` returns a data frame with one row per variable-class-outcome combination, with columns:

- variable** Manifest variable
- class** Latent class ID, an integer
- outcome** Outcome of manifest variable
- estimate** Estimated class-conditional response probability
- std.error** Standard error of estimated probability

`augment` returns a data frame with one row for each original observation, augmented with the following columns:

- .class** Predicted class, using modal assignment
- .probability** Posterior probability of predicted class

If the data argument is given, those columns are included in the output (only rows for which predictions could be made). Otherwise, the y element of the poLCA object, which contains the manifest variables used to fit the model, are used, along with any covariates, if present, in x.

Note that while the probability of all the classes (not just the predicted modal class) can be found in the posterior element, these are not included in the augmented output, since it would result in potentially many additional columns, which `augment` tends to avoid.

`glance` returns a one-row data frame with the following columns:

logLik the data's log-likelihood under the model

AIC the Akaike Information Criterion

BIC the Bayesian Information Criterion

g.squared The likelihood ratio/deviance statistic

chi.squared The Pearson Chi-Square goodness of fit statistic for multiway tables

df Number of parameters estimated, and therefore degrees of freedom used

df.residual Number of residual degrees of freedom left

Examples

```
if (require("poLCA", quietly = TRUE)) {
  library(poLCA)
  library(dplyr)

  data(values)
  f <- cbind(A, B, C, D)~1
  M1 <- poLCA(f, values, nclass = 2, verbose = FALSE)

  M1
  tidy(M1)
  head(augment(M1))
  glance(M1)

  library(ggplot2)

  ggplot(tidy(M1), aes(factor(class), estimate, fill = factor(outcome))) +
    geom_bar(stat = "identity", width = 1) +
    facet_wrap(~ variable)

  set.seed(2016)
  # compare multiple
  mods <- data_frame(nclass = 1:3) %>%
    group_by(nclass) %>%
    do(mod = poLCA(f, values, nclass = .$nclass, verbose = FALSE))

  # compare log-likelihood and/or AIC, BIC
  mods %>%
    glance(mod)

  ## Three-class model with a single covariate.
```

```

data(election)
f2a <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,
            MORALB,CARESB,KNOWB,LEADB,DISHONB,INTELB)~PARTY
nes2a <- polCA(f2a, election, nclass = 3, nrep = 5, verbose = FALSE)

td <- tidy(nes2a)
head(td)

# show

ggplot(td, aes(outcome, estimate, color = factor(class), group = class)) +
  geom_line() +
  facet_wrap(~ variable, nrow = 2) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

au <- augment(nes2a)
head(au)
au %>%
  count(.class)

# if the original data is provided, it leads to NAs in new columns
# for rows that weren't predicted
au2 <- augment(nes2a, data = election)
head(au2)
dim(au2)
}

```

prcomp_tidiers

Tidying methods for principal components analysis via [prcomp](#)

Description

These tidiers operate on the results of a principal components analysis computed using `prcomp`. The `tidy` method returns a data frame with either the eigenvectors representing each row or each column.

Usage

```

## S3 method for class 'prcomp'
tidy(x, matrix = "u", ...)

## S3 method for class 'prcomp'
augment(x, data = NULL, newdata, ...)

```

Arguments

`x` an object of class "prcomp" resulting from a call to [prcomp](#)

<code>matrix</code>	character; Indicates which sets of eigenvectors are returned in tidy form. "v", "rotation", or "variables" will return information about each variable, while "u", "x", or "samples" (default) returns the loadings for each original row. "d" or "pcs" returns information about each principal component.
<code>...</code>	Extra arguments, not used
<code>data</code>	the original data on which principal components analysis was performed. This cannot be recovered from <code>x</code> . If <code>newdata</code> is supplied, <code>data</code> is ignored. If both <code>data</code> and <code>newdata</code> are missing, only the fitted locations on the principal components are returned.
<code>newdata</code>	data frame; new observations for which locations on principal components are sought.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

If `matrix` is "u", "samples", or "x", the tidy method returns

`row` The sample labels (rownames) of the data set on which PCA was performed

`PC` An integer vector indicating the principal component

`value` The value of the eigenvector (axis score) on the indicated principal component

If `matrix` is "v", "variables", or "rotation", the tidy method returns

`row` The variable labels (colnames) of the data set on which PCA was performed

`PC` An integer vector indicating the principal component

`value` The value of the eigenvector (axis score) on the indicated principal component

If `matrix` is "d" or "pcs", the tidy method returns

`PC` An integer vector indicating the principal component

`std.dev` Standard deviation explained by this PC

`percent` Percentage of variation explained

`cumulative` Cumulative percentage of variation explained

The `augment.prcomp` method returns a data frame containing fitted locations on the principal components for the observed data plus either the original data or the new data if supplied via `data` or `newdata` respectively.

Author(s)

Gavin L. Simpson

See Also

[prcomp](#), [svd_tidiers](#)

Examples

```

pc <- prcomp(USArrests, scale = TRUE)

# information about rotation
head(tidy(pc))

# information about samples (states)
head(tidy(pc, "samples"))

# information about PCs
tidy(pc, "pcs")

# state map
library(dplyr)
library(ggplot2)

pc %>%
  tidy(matrix = "samples") %>%
  mutate(region = tolower(row)) %>%
  inner_join(map_data("state"), by = "region") %>%
  ggplot(aes(long, lat, group = group, fill = value)) +
  geom_polygon() +
  facet_wrap(~ PC) +
  theme_void() +
  ggtitle("Principal components of arrest data")

au <- augment(pc, data = USArrests)
head(au)

ggplot(au, aes(.fittedPC1, .fittedPC2)) +
  geom_point() +
  geom_text(aes(label = .rownames), vjust = 1, hjust = 1)

```

process_ergm

helper function to process a tidied ergm object

Description

Optionally exponentiates the coefficients, and optionally adds a confidence interval, to a tidied ergm object.

Usage

```

process_ergm(ret, x, conf.int = FALSE, conf.level = 0.95,
  exponentiate = FALSE)

```

Arguments

ret	data frame with a tidied version of a coefficient matrix
x	an "ergm" object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals (typical for logistic regression)

process_geeglm *helper function to process a tidied geeglm object*

Description

Adds a confidence interval, and possibly exponentiates, a tidied object.

Usage

```
process_geeglm(ret, x, conf.int = FALSE, conf.level = 0.95,
  exponentiate = FALSE)
```

Arguments

ret	data frame with a tidied version of a coefficient matrix
x	a "geeglm" object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals (typical for log distributions)

process_lm *helper function to process a tidied lm object*

Description

Adds a confidence interval, and possibly exponentiates, a tidied object. Useful for operations shared between lm and biglm.

Usage

```
process_lm(ret, x, conf.int = FALSE, conf.level = 0.95,
  exponentiate = FALSE)
```

Arguments

ret	data frame with a tidied version of a coefficient matrix
x	an "lm", "glm", "biglm", or "bigglm" object
conf.int	whether to include a confidence interval
conf.level	confidence level of the interval, used only if conf.int=TRUE
exponentiate	whether to exponentiate the coefficient estimates and confidence intervals (typical for logistic regression)

process_rq	<i>Helper function for tidy.rq and tidy.rqs</i>
------------	---

Description

See documentation for `summary.rq` for complete description of the options for `se.type`, `conf.int`, etc.

Usage

```
process_rq(rq_obj, se.type = "rank", conf.int = TRUE, conf.level = 0.95,
  ...)
```

Arguments

rq_obj	an object returned by <code>summary.rq</code> or <code>summary.rqs</code>
se.type	type of standard errors used in <code>summary.rq</code> or <code>summary.rqs</code>
conf.int	whether to include a confidence interval
conf.level	confidence level for confidence interval
...	currently unused

pyears_tidiers	<i>Tidy person-year summaries</i>
----------------	-----------------------------------

Description

These tidy the output of `pyears`, a calculation of the person-years of follow-up time contributed by a cohort of subject. Since the output of `pyears$data` is already tidy (if the `data.frame = TRUE` argument is given), this does only a little work and should rarely be necessary.

Usage

```
## S3 method for class 'pyears'
tidy(x, ...)

## S3 method for class 'pyears'
glance(x, ...)
```


Arguments

x a "pyears" object
 ... extra arguments (not used)

Value

tidy returns a data.frame with the columns

pyears person-years of exposure
 n number of subjects contributing time
 event observed number of events
 expected expected number of events (present only if a ratetable term is present)

If the data.frame = TRUE argument is supplied to pyears, this is simply the contents of x\$data.

glance returns a one-row data frame with

total total number of person-years tabulated
 offtable total number of person-years off table

This contains the values printed by summary.pyears.

See Also

[pyears](#)

Examples

```
if (require("survival", quietly = TRUE)) {
  temp.yr <- tcut(mgus$dxyr, 55:92, labels=as.character(55:91))
  temp.age <- tcut(mgus$age, 34:101, labels=as.character(34:100))
  ptime <- ifelse(is.na(mgus$pctime), mgus$futime, mgus$pctime)
  pstat <- ifelse(is.na(mgus$pctime), 0, 1)
  pfit <- pyears(Surv(ptime/365.25, pstat) ~ temp.yr + temp.age + sex, mgus,
                data.frame=TRUE)
  head(tidy(pfit))
  glance(pfit)

  # if data.frame argument is not given, different information is present in
  # output
  pfit2 <- pyears(Surv(ptime/365.25, pstat) ~ temp.yr + temp.age + sex, mgus)
  head(tidy(pfit2))
  glance(pfit2)
}
```

rcorr_tidiers

Tidying methods for rcorr objects

Description

Tidies a correlation matrix from the `rcorr` function in the "Hmisc" package, including correlation estimates, p-values, and the number of observations in each pairwise correlation. Note that it returns these in "long", or "melted", format, with one row for each pair of columns being compared.

Usage

```
## S3 method for class 'rcorr'
tidy(x, diagonal = FALSE, ...)
```

Arguments

<code>x</code>	An object of class "rcorr"
<code>diagonal</code>	Whether to include diagonal elements (where estimate is 1 and p.value is NA), default FALSE
<code>...</code>	extra arguments (not used)

Details

Only half the symmetric matrix is shown.

Value

A data.frame with one row for each pairing in the correlation matrix. Columns are:

<code>column1</code>	Name or index of the first column being described
<code>column2</code>	Name or index of the second column being described
<code>estimate</code>	Estimate of Pearson's r or Spearman's rho
<code>n</code>	Number of observations used to compute the correlation
<code>p.value</code>	P-value of correlation

Examples

```
if (require("Hmisc", quietly = TRUE)) {
  mat <- replicate(52, rnorm(100))
  # add some NAs
  mat[sample(length(mat), 2000)] <- NA
  # also column names
  colnames(mat) <- c(LETTERS, letters)

  rc <- rcorr(mat)
```

```

    td <- tidy(rc)
    head(td)

    library(ggplot2)
    ggplot(td, aes(p.value)) +
      geom_histogram(binwidth = .1)

    ggplot(td, aes(estimate, p.value)) +
      geom_point() +
      scale_y_log10()
  }

```

ridgelm_tidiers

Tidying methods for ridgelm objects from the MASS package

Description

These methods tidies the coefficients of a ridge regression model chosen at each value of lambda into a data frame, or constructs a one-row glance of the model's choices of lambda (the ridge constant)

Usage

```

## S3 method for class 'ridgelm'
tidy(x, ...)

## S3 method for class 'ridgelm'
glance(x, ...)

```

Arguments

x	An object of class "ridgelm"
...	extra arguments (not used)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy.ridgelm returns one row for each combination of choice of lambda and term in the formula, with columns:

lambda	choice of lambda
GCV	generalized cross validation value for this lambda
term	the term in the ridge regression model being estimated
estimate	estimate of scaled coefficient using this lambda
scale	Scaling factor of estimated coefficient

`glance.ridge1m` returns a one-row data.frame with the columns

<code>kHKB</code>	modified HKB estimate of the ridge constant
<code>kLW</code>	modified L-W estimate of the ridge constant
<code>lambdaGCV</code>	choice of lambda that minimizes GCV

This is similar to the output of `select.ridge1m`, but it is returned rather than printed.

Examples

```
names(longley)[1] <- "y"
fit1 <- MASS::lm.ridge(y ~ ., longley)
tidy(fit1)

fit2 <- MASS::lm.ridge(y ~ ., longley, lambda = seq(0.001, .05, .001))
td2 <- tidy(fit2)
g2 <- glance(fit2)

# coefficient plot
library(ggplot2)
ggplot(td2, aes(lambda, estimate, color = term)) + geom_line()

# GCV plot
ggplot(td2, aes(lambda, GCV)) + geom_line()

# add line for the GCV minimizing estimate
ggplot(td2, aes(lambda, GCV)) + geom_line() +
  geom_vline(xintercept = g2$lambdaGCV, col = "red", lty = 2)
```

rlm_tidiers

Tidying methods for an rlm (robust linear model) object

Description

This method provides a glance of an "rlm" object. The `tidy` and `augment` methods are handled by [lm_tidiers](#).

Usage

```
## S3 method for class 'rlm'
glance(x, ...)
```

Arguments

<code>x</code>	rlm object
<code>...</code>	extra arguments (not used)

Value

`glance.rlm` returns a one-row `data.frame` with the columns

<code>sigma</code>	The square root of the estimated residual variance
<code>converged</code>	whether the IWLS converged
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>deviance</code>	deviance

See Also

[lm_tidiers](#)

Examples

```
library(MASS)

r <- rlm(stack.loss ~ ., stackloss)
tidy(r)
augment(r)
glance(r)
```

robust_tidiers

Tidiers for lmRob and glmRob objects

Description

Tidying robust regression objects from the robust package. The `tidy` and `augment` methods simply pass it on to the linear model tidiers.

Usage

```
## S3 method for class 'lmRob'
tidy(x, ...)

## S3 method for class 'lmRob'
augment(x, ...)

## S3 method for class 'lmRob'
glance(x, ...)

## S3 method for class 'glmRob'
tidy(x, ...)
```

```
## S3 method for class 'glmRob'
augment(x, ...)

## S3 method for class 'glmRob'
glance(x, ...)
```

Arguments

`x` An `lmRob` or `glmRob` object with a robust regression
`...` Extra arguments, not used

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy` and `augment` return the same results as [lm_tidiers](#).

On an `lmRob` `glance` returns a one-row data frame with the following columns:

<code>r.squared</code>	R-squared
<code>deviance</code>	Robust deviance
<code>sigma</code>	Residual scale estimate
<code>df.residual</code>	Number of residual degrees of freedom

On an `lmRob` `glance` returns a one-row data frame with the following columns:

<code>deviance</code>	Robust deviance
<code>null.deviance</code>	Deviance under the null model
<code>df.residual</code>	Number of residual degrees of freedom

See Also

[lm_tidiers](#), [lmRob](#), [glmRob](#)

Examples

```
if (require("robust", quietly = TRUE)) {
  m <- lmRob(mpg ~ wt, data = mtcars)

  tidy(m)
  augment(m)
  glance(m)

  gm <- glmRob(am ~ wt, data = mtcars, family = "binomial")
  glance(gm)
}
```

rowwise_df_tidiers	<i>Tidying methods for rowwise_dfs from dplyr, for tidying each row and recombining the results</i>
--------------------	---

Description

These tidy, augment and glance methods are for performing tidying on each row of a rowwise data frame created by dplyr's group_by and do operations. They first group a rowwise data frame based on all columns that are not lists, then perform the tidying operation on the specified column. This greatly shortens a common idiom of extracting tidy/augment/glance outputs after a do statement.

Usage

```
## S3 method for class 'rowwise_df'
tidy(x, object, ...)

## S3 method for class 'rowwise_df'
tidy_(x, object, ...)

## S3 method for class 'rowwise_df'
augment(x, object, ...)

## S3 method for class 'rowwise_df'
augment_(x, object, ...)

## S3 method for class 'rowwise_df'
glance(x, object, ...)

## S3 method for class 'rowwise_df'
glance_(x, object, ...)

## S3 method for class 'tbl_df'
tidy(x, ...)

## S3 method for class 'tbl_df'
augment(x, ...)

## S3 method for class 'tbl_df'
glance(x, ...)
```

Arguments

x	a rowwise_df
object	the column name of the column containing the models to be tidied. For tidy, augment, and glance it should be the bare name; for _ methods it should be quoted.
...	additional arguments to pass on to the respective tidying method

Details

Note that this functionality is not currently implemented for `data.tables`, since the result of the `do` operation is difficult to distinguish from a regular `data.table`.

Value

A "grouped_df", where the non-list columns of the original are used as grouping columns alongside the tidied outputs.

Examples

```
library(dplyr)
regressions <- mtcars %>%
  group_by(cyl) %>%
  do(mod = lm(mpg ~ wt, .))

regressions

regressions %>% tidy(mod)
regressions %>% augment(mod)
regressions %>% glance(mod)

# we can provide additional arguments to the tidying function
regressions %>% tidy(mod, conf.int = TRUE)

# we can also include the original dataset as a "data" argument
# to augment:
regressions <- mtcars %>%
  group_by(cyl) %>%
  do(mod = lm(mpg ~ wt, .), original = (.))

# this allows all the original columns to be included:
regressions %>% augment(mod) # doesn't include all original
regressions %>% augment(mod, data = original) # includes all original
```

Description

These methods tidy the coefficients of a quantile regression model into a summary, augment the original data with information on the fitted values and residuals, and construct a glance of the model's statistics.

Usage

```
## S3 method for class 'rq'
tidy(x, se.type = "rank", conf.int = TRUE, conf.level = 0.95,
     alpha = 1 - conf.level, ...)

## S3 method for class 'rqs'
tidy(x, se.type = "rank", conf.int = TRUE,
     conf.level = 0.95, alpha = 1 - conf.level, ...)

## S3 method for class 'nlrq'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)

## S3 method for class 'rq'
glance(x, ...)

## S3 method for class 'nlrq'
glance(x, ...)

## S3 method for class 'rq'
augment(x, data = model.frame(x), newdata, ...)

## S3 method for class 'rqs'
augment(x, data = model.frame(x), newdata, ...)

## S3 method for class 'nlrq'
augment(x, data = NULL, newdata = NULL, ...)
```

Arguments

x	model object returned by rq or nlrq
se.type	Type of standard errors to calculate; see <code>summary.rq</code>
conf.int	boolean; should confidence intervals be calculated, ignored if <code>se.type = "rank"</code>
conf.level	confidence level for intervals
alpha	confidence level when <code>se.type = "rank"</code> ; defaults to the same as <code>conf.level</code> although the specification is inverted
data	Original data, defaults to extracting it from the model
newdata	If provided, new data frame to use for predictions
...	other arguments passed on to <code>summary.rq</code>

Details

If `se.type != "rank"` and `conf.int = TRUE` confidence intervals are calculated by `summary.rq`. Otherwise they are standard t based intervals.

This simply calls `augment.nls` on the "nlrq" object.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy.rq` returns a data frame with one row for each coefficient. The columns depend upon the confidence interval method selected.

`tidy.rqs` returns a data frame with one row for each coefficient at each quantile that was estimated. The columns depend upon the confidence interval method selected.

`tidy.nlrq` returns one row for each coefficient in the model, with five columns:

<code>term</code>	The term in the nonlinear model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error from the linear model
<code>statistic</code>	t-statistic
<code>p.value</code>	two-sided p-value

`glance.rq` returns one row for each quantile (`tau`) with the columns:

<code>tau</code>	quantile estimated
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>df.residual</code>	residual degrees of freedom

`glance.rq` returns one row for each quantile (`tau`) with the columns:

<code>tau</code>	quantile estimated
<code>logLik</code>	the data's log-likelihood under the model
<code>AIC</code>	the Akaike Information Criterion
<code>BIC</code>	the Bayesian Information Criterion
<code>df.residual</code>	residual degrees of freedom

`augment.rq` returns a row for each original observation with the following columns added:

<code>.resid</code>	Residuals
<code>.fitted</code>	Fitted quantiles of the model
<code>.tau</code>	Quantile estimated

Depending on the arguments passed on to `predict.rq` via `...` a confidence interval is also calculated on the fitted values resulting in columns:

<code>.conf.low</code>	Lower confidence interval value
<code>.conf.high</code>	Upper confidence interval value

See `predict.rq` for details on additional arguments to specify confidence intervals. `predict.rq` does not provide confidence intervals when `newdata` is provided.

`augment.rqs` returns a row for each original observation and each estimated quantile (`tau`) with the following columns added:

.resid	Residuals
.fitted	Fitted quantiles of the model
.tau	Quantile estimated

predict.rqs does not return confidence interval estimates.

augment.rqs returns a row for each original observation with the following columns added:

.resid	Residuals
.fitted	Fitted quantiles of the model

rstanarm_tidiars	<i>Tidying methods for an rstanarm model</i>
------------------	--

Description

These methods tidy the estimates from [stanreg-objects](#) (fitted model objects from the **rstanarm** package) into a summary.

Usage

```
## S3 method for class 'stanreg'
tidy(x, parameters = "non-varying", intervals = FALSE,
     prob = 0.9, ...)
```

```
## S3 method for class 'stanreg'
glance(x, looic = FALSE, ...)
```

Arguments

x	Fitted model object from the rstanarm package. See stanreg-objects .
parameters	One or more of "non-varying", "varying", "hierarchical", "auxiliary" (can be abbreviated). See the Value section for details.
intervals	If TRUE columns for the lower and upper bounds of the 100*prob% posterior uncertainty intervals are included. See posterior_interval for details.
prob	See posterior_interval .
...	For glance, if looic=TRUE, optional arguments to loo.stanreg .
looic	Should the LOO Information Criterion (and related info) be included? See loo.stanreg for details. Note: for models fit to very large datasets this can be a slow computation.

Value

All tidying methods return a `data.frame` without rownames. The structure depends on the method chosen.

When `parameters="non-varying"` (the default), `tidy.stanreg` returns one row for each coefficient, with three columns:

<code>term</code>	The name of the corresponding term in the model.
<code>estimate</code>	A point estimate of the coefficient (posterior median).
<code>std.error</code>	A standard error for the point estimate based on <code>mad</code> . See the <i>Uncertainty estimates</i> section in <code>print.stanreg</code> for more details.

For models with group-specific parameters (e.g., models fit with `stan_glmmer`), setting `parameters="varying"` selects the group-level parameters instead of the non-varying regression coefficients. Additional columns are added indicating the level and group. Specifying `parameters="hierarchical"` selects the standard deviations and (for certain models) correlations of the group-level parameters.

Setting `parameters="auxiliary"` will select parameters other than those included by the other options. The particular parameters depend on which **rstanarm** modeling function was used to fit the model. For example, for models fit using `stan_glm.nb` the overdispersion parameter is included if `parameters="aux"`, for `stan_lm` the auxiliary parameters include the residual SD, R^2 , and `log(fit_ratio)`, etc.

If `intervals=TRUE`, columns for the lower and upper values of the posterior intervals computed with `posterior_interval` are also included.

`glance` returns one row with the columns

<code>algorithm</code>	The algorithm used to fit the model.
<code>pss</code>	The posterior sample size (except for models fit using optimization).
<code>nobs</code>	The number of observations used to fit the model.
<code>sigma</code>	The square root of the estimated residual variance, if applicable. If not applicable (e.g., for binomial GLMs), <code>sigma</code> will be given the value 1 in the returned object.

If `looic=TRUE`, then the following additional columns are also included:

<code>looic</code>	The LOO Information Criterion.
<code>elpd_loo</code>	The expected log predictive density ($elpd_loo = -2 * looic$).
<code>p_loo</code>	The effective number of parameters.

See Also

[summary.stanreg](#)

Examples

```
## Not run:
fit <- stan_glmmer(mpg ~ wt + (1|cyl) + (1+wt|gear), data = mtcars,
                  iter = 300, chains = 2)
```

```

# non-varying ("population") parameters
tidy(fit, intervals = TRUE, prob = 0.5)

# hierarchical sd & correlation parameters
tidy(fit, parameters = "hierarchical")

# group-specific deviations from "population" parameters
tidy(fit, parameters = "varying")

# glance method
glance(fit)
glance(fit, loaic = TRUE, cores = 1)

## End(Not run)

```

sexpfit_tidiers

Tidy an expected survival curve

Description

This constructs a summary across time points or overall of an expected survival curve. Note that this contains less information than most survfit objects.

Usage

```

## S3 method for class 'survexp'
tidy(x, ...)

## S3 method for class 'survexp'
glance(x, ...)

```

Arguments

x	"survexp" object
...	extra arguments (not used)

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy returns a one row for each time point, with columns

time	time point
estimate	estimated survival
n.risk	number of individuals at risk

glance returns a one-row data.frame with the columns:

n.max	maximum number of subjects at risk
n.start	starting number of subjects at risk
timepoints	number of timepoints

Examples

```
if (require("survival", quietly = TRUE)) {  
  sexpfit <- survexp(futime ~ 1, rmap=list(sex="male", year=accept.dt,  
                                          age=(accept.dt-birth.dt)),  
                  method='conditional', data=jasa)  
  
  tidy(sexpfit)  
  glance(sexpfit)  
}
```

smooth.spline_tidiers *tidying methods for smooth.spline objects*

Description

This combines the original data given to smooth.spline with the fit and residuals

Usage

```
## S3 method for class 'smooth.spline'  
augment(x, data = x$data, ...)  
  
## S3 method for class 'smooth.spline'  
glance(x, ...)
```

Arguments

x	a smooth.spline object
data	defaults to data used to fit model
...	not used in this method

Details

No tidy method is provided for smooth.spline objects.

Value

augment returns the original data with extra columns:

.fitted	Fitted values of model
.resid	Residuals

glance returns one row with columns

spar	smoothing parameter
lambda	choice of lambda corresponding to spar
df	equivalent degrees of freedom
crit	minimized criterion
pen.crit	penalized criterion
cv.crit	cross-validation score

Examples

```
spl <- smooth.spline(mtcars$wt, mtcars$mpg, df = 4)
head(augment(spl, mtcars))
head(augment(spl)) # calls original columns x and y

library(ggplot2)
ggplot(augment(spl, mtcars), aes(wt, mpg)) +
  geom_point() + geom_line(aes(y = .fitted))
```

sparse_tidiers

Tidy a sparseMatrix object from the Matrix package

Description

Tidy a sparseMatrix object from the Matrix package into a three-column data frame, row, column, and value (with zeros missing). If there are row names or column names, use those, otherwise use indices

Usage

```
## S3 method for class 'dgTMatrix'
tidy(x, ...)

## S3 method for class 'dgCMatrix'
tidy(x, ...)

## S3 method for class 'sparseMatrix'
tidy(x, ...)
```

Arguments

x	A Matrix object
...	Extra arguments, not used

sp_tidiers	<i>tidying methods for classes from the sp package.</i>
------------	---

Description

Tidy classes from the sp package to allow them to be plotted using ggplot2. To figure out the correct variable name for region, inspect as `.data.frame(x)`.

Usage

```
## S3 method for class 'SpatialPolygonsDataFrame'
tidy(x, region = NULL, ...)

## S3 method for class 'SpatialPolygons'
tidy(x, ...)

## S3 method for class 'Polygons'
tidy(x, ...)

## S3 method for class 'Polygon'
tidy(x, ...)

## S3 method for class 'SpatialLinesDataFrame'
tidy(x, ...)

## S3 method for class 'Lines'
tidy(x, ...)

## S3 method for class 'Line'
tidy(x, ...)
```

Arguments

x	SpatialPolygonsDataFrame to convert into a dataframe.
region	name of variable used to split up regions
...	not used by this method

Details

These functions originated in the ggplot2 package as "fortify" functions.

Examples

```

if (require("maptools")) {
  sids <- system.file("shapes/sids.shp", package="maptools")
  nc1 <- readShapePoly(sids,
    proj4string = CRS("+proj=longlat +datum=NAD27"))
  nc1_df <- tidy(nc1)
}

```

summary_tidiers

*Tidiers for summaryDefault objects***Description**

Tidy a summary of a vector.

Usage

```
## S3 method for class 'summaryDefault'
tidy(x, ...)
```

```
## S3 method for class 'summaryDefault'
glance(x, ...)
```

Arguments

x	summaryDefault object
...	extra arguments, not used

Value

Both tidy and glance return the same object: a one-row data frame with columns

minimum	smallest value in original vector
q1	value at the first quartile
median	median of original vector
mean	mean of original vector
q3	value at the third quartile
maximum	largest value in original vector
NAs	number of NA values (if any)

See Also

[summary](#)

Examples

```
v <- rnorm(1000)
s <- summary(v)
s

tidy(s)
glance(s)

v2 <- c(v,NA)
tidy(summary(v2))
```

survfit_tidiers	<i>tidy survival curve fits</i>
-----------------	---------------------------------

Description

Construct tidied data frames showing survival curves over time.

Usage

```
## S3 method for class 'survfit'
tidy(x, ...)

## S3 method for class 'survfit'
glance(x, ...)
```

Arguments

x	"survfit" object
...	extra arguments, not used

Details

glance does not work on multi-state survival curves, since the values glance outputs would be calculated for each state. tidy does work for multi-state survival objects, and includes a state column to distinguish between them.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy returns a row for each time point, with columns

time	timepoint
n.risk	number of subjects at risk at time t0

n.event	number of events at time t
n.censor	number of censored events
estimate	estimate of survival or cumulative incidence rate when multistate
std.error	standard error of estimate
conf.high	upper end of confidence interval
conf.low	lower end of confidence interval
state	state if multistate survfit object inputted
strata	strata if stratified survfit object inputted

glance returns one-row data.frame with the columns displayed by [print.survfit](#)

records	number of observations
n.max	n.max
n.start	n.start
events	number of events
rmean	Restricted mean (see print.survfit)
rmean.std.error	Restricted mean standard error
median	median survival
conf.low	lower end of confidence interval on median
conf.high	upper end of confidence interval on median

Examples

```
if (require("survival", quietly = TRUE)) {
  cfit <- coxph(Surv(time, status) ~ age + sex, lung)
  sfit <- survfit(cfit)

  head(tidy(sfit))
  glance(sfit)

  library(ggplot2)
  ggplot(tidy(sfit), aes(time, estimate)) + geom_line() +
    geom_ribbon(aes(ymin=conf.low, ymax=conf.high), alpha=.25)

  # multi-state
  fitCI <- survfit(Surv(stop, status * as.numeric(event), type = "mstate") ~ 1,
    data = mgus1, subset = (start == 0))
  td_multi <- tidy(fitCI)
  head(td_multi)
  tail(td_multi)
  ggplot(td_multi, aes(time, estimate, group = state)) +
    geom_line(aes(color = state)) +
    geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)

  # perform simple bootstrapping
```

```

library(dplyr)
bootstraps <- lung %>% bootstrap(100) %>%
  do(tidy(survfit(coxph(Surv(time, status) ~ age + sex, .))))

ggplot(bootstraps, aes(time, estimate, group = replicate)) +
  geom_line(alpha = .25)

bootstraps_bytime <- bootstraps %>% group_by(time) %>%
  summarize(median = median(estimate),
            low = quantile(estimate, .025),
            high = quantile(estimate, .975))

ggplot(bootstraps_bytime, aes(x = time, y = median)) + geom_line() +
  geom_ribbon(aes(ymin = low, ymax = high), alpha = .25)

# bootstrap for median survival
glances <- lung %>%
  bootstrap(100) %>%
  do(glance(survfit(coxph(Surv(time, status) ~ age + sex, .))))

glances

qplot(glances$median, binwidth = 15)
quantile(glances$median, c(.025, .975))
}

```

survreg_tidiers

Tidiers for a parametric regression survival model

Description

Tidies the coefficients of a parametric survival regression model, from the "survreg" function, adds fitted values and residuals, or summarizes the model statistics.

Usage

```

## S3 method for class 'survreg'
tidy(x, conf.level = 0.95, ...)

## S3 method for class 'survreg'
augment(x, data = stats::model.frame(x), newdata,
        type.predict = "response", type.residuals = "response", ...)

## S3 method for class 'survreg'
glance(x, conf.level = 0.95, ...)

```

Arguments

<code>x</code>	a "survreg" model
<code>conf.level</code>	confidence level for CI
<code>...</code>	extra arguments (not used)
<code>data</code>	original data; if it is not provided, it is reconstructed as best as possible with <code>model.frame</code>
<code>newdata</code>	New data to use for prediction; optional
<code>type.predict</code>	type of prediction, default "response"
<code>type.residuals</code>	type of residuals to calculate, default "response"

Details

When the modeling was performed with `na.action = "na.omit"` (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with `na.action = "na.exclude"`, one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to `augment` and `na.action = "na.exclude"`, a warning is raised and the incomplete rows are dropped.

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

`tidy` returns a data.frame with one row for each term

<code>term</code>	name of term
<code>estimate</code>	estimate of coefficient
<code>stderror</code>	standard error
<code>statistic</code>	Z statistic
<code>p.value</code>	p-value
<code>conf.low</code>	low end of confidence interval
<code>conf.high</code>	high end of confidence interval

`augment` returns the original data.frame with the following additional columns:

<code>.fitted</code>	Fitted values of model
<code>.se.fit</code>	Standard errors of fitted values
<code>.resid</code>	Residuals

`glance` returns a one-row data.frame with the columns:

<code>iter</code>	number of iterations
<code>df</code>	degrees of freedom
<code>statistic</code>	chi-squared statistic
<code>p.value</code>	p-value from chi-squared test

logLik	log likelihood
AIC	Akaike information criterion
BIC	Bayesian information criterion
df.residual	residual degrees of freedom

See Also

[na.action](#)

Examples

```
if (require("survival", quietly = TRUE)) {
  sr <- survreg(Surv(futime, fustat) ~ ecog.ps + rx, ovarian,
               dist="exponential")

  td <- tidy(sr)
  augment(sr, ovarian)
  augment(sr)
  glance(td)

  # coefficient plot
  library(ggplot2)
  ggplot(td, aes(estimate, term)) + geom_point() +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
    geom_vline(xintercept = 0)
}
```

svd_tidiers

Tidying methods for singular value decomposition

Description

These methods tidy the U, D, and V matrices returned by the [svd](#) function into a tidy format. Because `svd` returns a list without a class, this function has to be called by `tidy.list` when it recognizes a list as an SVD object.

Usage

```
tidy_svd(x, matrix = "u", ...)
```

Arguments

x	list containing d, u, v components, returned from <code>svd</code>
matrix	which of the u, d or v matrix to tidy
...	Extra arguments (not used)

Value

An SVD object contains a decomposition into u , d , and v matrices, such that $u \%*\% \text{diag}(d) \%*\% t(v)$ gives the original matrix. This tidier gives a choice of which matrix to tidy.

When `matrix = "u"`, each observation represents one pair of row and principal component, with variables:

<code>row</code>	Number of the row in the original data being described
<code>PC</code>	Principal component
<code>loading</code>	Loading of this principal component for this row

When `matrix = "d"`, each observation represents one principal component, with variables:

<code>PC</code>	Principal component
<code>d</code>	Value in the d vector
<code>percent</code>	Percent of variance explained by this PC, which is proportional to d^2

When `matrix = "v"`, each observation represents a pair of a principal component and a column of the original matrix, with variables:

<code>column</code>	Column of original matrix described
<code>PC</code>	Principal component
<code>value</code>	Value of this PC for this column

See Also

[svd, tidy.list](#)

Examples

```
mat <- as.matrix(iris[, 1:4])
s <- svd(mat)

tidy_u <- tidy(s, matrix = "u")
head(tidy_u)

tidy_d <- tidy(s, matrix = "d")
tidy_d

tidy_v <- tidy(s, matrix = "v")
head(tidy_v)

library(ggplot2)
library(dplyr)

ggplot(tidy_d, aes(PC, percent)) +
  geom_point() +
  ylab("% of variance explained")

tidy_u %>%
```

```
mutate(Species = iris$Species[row]) %>%
  ggplot(aes(Species, loading)) +
  geom_boxplot() +
  facet_wrap(~ PC, scale = "free_y")
```

tidy

Tidy the result of a test into a summary data.frame

Description

The output of tidy is always a data.frame with disposable row names. It is therefore suited for further manipulation by packages like dplyr, reshape2, ggplot2 and ggvis.

Usage

```
tidy(x, ...)
```

Arguments

x An object to be converted into a tidy data.frame
 ... extra arguments

Value

a data.frame

tidy.coefstest

Tidying methods for coefstest objects

Description

This tidies the result of a coefficient test, from the coefstest function in the lmtest package.

Usage

```
## S3 method for class 'coefstest'
tidy(x, ...)
```

Arguments

x coefstest object
 ... extra arguments (not used)

Value

A `data.frame` with one row for each coefficient, with five columns:

<code>term</code>	The term in the linear model being estimated and tested
<code>estimate</code>	The estimated coefficient
<code>std.error</code>	The standard error
<code>statistic</code>	test statistic
<code>p.value</code>	p-value

Examples

```
if (require("lmtest", quietly = TRUE)) {
  data(Mandible)
  fm <- lm(length ~ age, data=Mandible, subset=(age <= 28))

  coeftest(fm)
  tidy(coeftest(fm))
}
```

tidy.default

Default tidying method

Description

By default, `tidy` uses `as.data.frame` to convert its output. This is dangerous, as it may fail with an uninformative error message. Generally `tidy` is intended to be used on structured model objects such as `lm` or `htest` for which a specific S3 object exists.

Usage

```
## Default S3 method:
tidy(x, ...)
```

Arguments

<code>x</code>	an object to be tidied
<code>...</code>	extra arguments (not used)

Details

If you know that you want to use `as.data.frame` on your untidy object, just use it directly.

Value

A `data.frame`, from `as.data.frame` applied to the input `x`.

tidy.density	<i>tidy a density objet</i>
--------------	-----------------------------

Description

Given a "density" object, returns a tidy data frame with two columns: points x where the density is estimated, points y for the estimate

Usage

```
## S3 method for class 'density'
tidy(x, ...)
```

Arguments

x	an object of class "density"
...	extra arguments (not used)

Value

a data frame with "x" and "y" columns

```
d <- density(faithful$eruptions, bw = "sj") head(tidy(d))
library(ggplot2) ggplot(tidy(d), aes(x, y)) + geom_line()
```

See Also

[density](#)

tidy.dist	<i>Tidy a distance matrix</i>
-----------	-------------------------------

Description

Tidy a distance matrix, such as that computed by the [dist](#) function, into a one-row-per-pair table. If the distance matrix does not include an upper triangle and/or diagonal, this will not either.

Usage

```
## S3 method for class 'dist'
tidy(x, diag = attr(x, "Diag"), upper = attr(x, "Upper"),
     ...)
```

Arguments

x	A "dist" object
diag	Whether to include the diagonal of the distance matrix. Defaults to whether the distance matrix includes it
upper	Whether to include the upper right triangle of the distance matrix. Defaults to whether the distance matrix includes it
...	Extra arguments, not used

Value

A data frame with one row for each pair of item distances, with columns:

item1	First item
item2	Second item
distance	Distance between items

Examples

```
iris_dist <- dist(t(iris[, 1:4]))
iris_dist

tidy(iris_dist)
tidy(iris_dist, upper = TRUE)
tidy(iris_dist, diag = TRUE)
```

tidy.ftable	<i>tidy an ftable object</i>
-------------	------------------------------

Description

An ftable contains a "flat" contingency table. This melts it into a data.frame with one column for each variable, then a Freq column. It directly uses the `stats:::as.data.frame.ftable` function

Usage

```
## S3 method for class 'ftable'
tidy(x, ...)
```

Arguments

x	An object of class "ftable"
...	Extra arguments (not used)

See Also[ftable](#)**Examples**

```
tidy(ftable(Titanic, row.vars = 1:3))
```

tidy.manova	<i>tidy a MANOVA object</i>
-------------	-----------------------------

Description

Constructs a data frame with one row for each of the terms in the model, containing the information from [summary.manova](#).

Usage

```
## S3 method for class 'manova'
tidy(x, test = "Pillai", ...)
```

Arguments

x	object of class "manova"
test	one of "Pillai" (Pillai's trace), "Wilks" (Wilk's lambda), "Hotelling-Lawley" (Hotelling-Lawley trace) or "Roy" (Roy's greatest root) indicating which test statistic should be used. Defaults to "Pillai"
...	additional arguments passed on to <code>summary.manova</code>

Value

A data.frame with the columns

term	Term in design
statistic	Approximate F statistic
num.df	Degrees of freedom
p.value	P-value

Depending on which test statistic is specified, one of the following columns is also included:

pillai	Pillai's trace
wilks	Wilk's lambda
hl	Hotelling-Lawley trace
roy	Roy's greatest root

See Also[summary.manova](#)**Examples**

```
npk2 <- within(npk, foo <- rnorm(24))
npk2.aov <- manova(cbind(yield, foo) ~ block + N*P*K, npk2)
```

`tidy.map`*Tidy method for map objects.*

Description

This function turns a map into a data frame.

Usage

```
## S3 method for class 'map'
tidy(x, ...)
```

Arguments

<code>x</code>	map object
<code>...</code>	not used by this method

Details

This code and documentation originated in `ggplot2`, but was called "fortify." In `broom`, "fortify" became "augment", which is reserved for functions that *add* columns to existing data (based on a model fit, for example) so these functions were renamed as "tidy."

Examples

```
if (require("maps") && require("ggplot2")) {
  ca <- map("county", "ca", plot = FALSE, fill = TRUE)
  head(tidy(ca))
  qplot(long, lat, data = ca, geom = "polygon", group = group)

  tx <- map("county", "texas", plot = FALSE, fill = TRUE)
  head(tidy(tx))
  qplot(long, lat, data = tx, geom = "polygon", group = group,
        colour = I("white"))
}
```

tidy.NULL	<i>tidy on a NULL input</i>
-----------	-----------------------------

Description

tidy on a NULL input returns an empty data frame, which means it can be combined with other data frames (treated as "empty")

Usage

```
## S3 method for class 'NULL'
tidy(x, ...)
```

Arguments

x	A value NULL
...	extra arguments (not used)

Value

An empty data.frame

tidy.numeric	<i>Tidy atomic vectors</i>
--------------	----------------------------

Description

Turn atomic vectors into data frames, where the names of the vector (if they exist) are a column and the values of the vector are a column.

Usage

```
## S3 method for class 'numeric'
tidy(x, ...)

## S3 method for class 'character'
tidy(x, ...)

## S3 method for class 'logical'
tidy(x, ...)
```

Arguments

x	An object of class "numeric", "integer", "character", or "logical". Most likely a named vector
...	Extra arguments (not used)

Examples

```
x <- 1:5
names(x) <- letters[1:5]
tidy(x)
```

tidy.pairwise.htest *tidy a pairwise hypothesis test*

Description

Tidy a pairwise.htest object, containing (adjusted) p-values for multiple pairwise hypothesis tests.

Usage

```
## S3 method for class 'pairwise.htest'
tidy(x, ...)
```

Arguments

x	a "pairwise.htest" object
...	extra arguments (not used)

Details

Note that in one-sided tests, the alternative hypothesis of each test can be stated as "group1 is greater/less than group2".

Note also that the columns of group1 and group2 will always be a factor, even if the original input is (e.g.) numeric.

Value

A data frame with one row per group/group comparison, with columns

group1	First group being compared
group2	Second group being compared
p.value	(Adjusted) p-value of comparison

See Also

[pairwise.t.test](#), [pairwise.wilcox.test](#)

Examples

```
attach(airquality)
Month <- factor(Month, labels = month.abb[5:9])
ptt <- pairwise.t.test(Ozone, Month)
tidy(ptt)

attach(iris)
ptt2 <- pairwise.t.test(Petal.Length, Species)
tidy(ptt2)

tidy(pairwise.t.test(Petal.Length, Species, alternative = "greater"))
tidy(pairwise.t.test(Petal.Length, Species, alternative = "less"))

tidy(pairwise.wilcox.test(Petal.Length, Species))
```

tidy.power.htest	<i>tidy a power.htest</i>
------------------	---------------------------

Description

tidy a power.htest

Usage

```
## S3 method for class 'power.htest'
tidy(x, ...)
```

Arguments

x	a power.htest object
...	extra arguments, not used

Value

A data frame with one row per parameter passed in, with columns n, delta, sd, sig.level, and power (from the power.htest object).

See Also

[power.t.test](#)

Examples

```
ptt <- power.t.test(n = 2:30, delta = 1)
tidy(ptt)

library(ggplot2)
ggplot(tidy(ptt), aes(n, power)) + geom_line()
```

tidy.spec	<i>tidy a spec objet</i>
-----------	--------------------------

Description

Given a "spec" object, which shows a spectrum across a range of frequencies, returns a tidy data frame with two columns: "freq" and "spec"

Usage

```
## S3 method for class 'spec'
tidy(x, ...)
```

Arguments

x	an object of class "spec"
...	extra arguments (not used)

Value

a data frame with "freq" and "spec" columns

Examples

```
spc <- spectrum(lh)
tidy(spc)

library(ggplot2)
ggplot(tidy(spc), aes(freq, spec)) + geom_line()
```

tidy.table	<i>tidy a table object</i>
------------	----------------------------

Description

A table, typically created by the [table](#) function, contains a contingency table of frequencies across multiple vectors. This directly calls the [as.data.frame.table](#) method, which melts it into a data frame with one column for each variable and a Freq column.

Usage

```
## S3 method for class 'table'
tidy(x, ...)
```

Arguments

x	An object of class "table"
...	Extra arguments (not used)

See Also

[as.data.frame.table](#)

Examples

```
tab <- with(airquality, table(cut(Temp, quantile(Temp)), Month))
tidy(tab)
```

tidy.ts	<i>tidy a ts timeseries object</i>
---------	------------------------------------

Description

Turn a ts object into a tidy data frame. Right now simply uses `as.data.frame.ts`.

Usage

```
## S3 method for class 'ts'
tidy(x, ...)
```

Arguments

x	a "ts" object
...	extra arguments (not used)

Value

a tidy data frame

See Also

[as.data.frame.ts](#)

tidy.TukeyHSD	<i>tidy a TukeyHSD object</i>
---------------	-------------------------------

Description

Returns a data.frame with one row for each pairwise comparison

Usage

```
## S3 method for class 'TukeyHSD'
tidy(x, separate.levels = FALSE, ...)
```

Arguments

x	object of class "TukeyHSD"
separate.levels	Whether to separate comparison into level1 and level2 columns
...	additional arguments (not used)

Value

A data.frame with one row per comparison, containing columns

term	Term for which levels are being compared
comparison	Levels being compared, separated by -
estimate	Estimate of difference
conf.low	Low end of confidence interval of difference
conf.high	High end of confidence interval of difference
adj.p.value	P-value adjusted for multiple comparisons

If `separate.levels = TRUE`, the comparison column will be split up into level1 and level2.

See Also

[TukeyHSD](#)

Examples

```

fm1 <- aov(breaks ~ wool + tension, data = warpbreaks)
thsd <- TukeyHSD(fm1, "tension", ordered = TRUE)
tidy(thsd)
tidy(thsd, separate.levels = TRUE)

# may include comparisons on multiple terms
fm2 <- aov(mpg ~ as.factor(gear) * as.factor(cyl), data = mtcars)
tidy(TukeyHSD(fm2))

```

unrowname	<i>strip rownames from an object</i>
-----------	--------------------------------------

Description

strip rownames from an object

Usage

```
unrowname(x)
```

Arguments

x	a data frame
---	--------------

xyz_tidiers	<i>Tidiers for x, y, z lists suitable for persp, image, etc.</i>
-------------	--

Description

Tidies lists with components x, y (vector of coordinates) and z (matrix of values) which are typically used by functions such as [persp](#) or [image](#) and returned by interpolation functions such as [interp](#).

Usage

```
tidy_xyz(x, ...)
```

Arguments

x	list with components x, y and z
...	extra arguments

Value

All tidying methods return a data.frame without rownames, whose structure depends on the method chosen.

tidy returns a data frame with columns x, y and z and one row per value in matrix z.

Examples

```
A <- list(x=1:5, y=1:3, z=matrix(runif(5*3), nrow=5))
image(A)
tidy(A)
```

zoo_tidiers

Tidying methods for a zoo object

Description

Tidies zoo (Z's ordered observations) time series objects. zoo objects are not tidy by default because they contain one row for each index and one series per column, rather than one row per observation per series.

Usage

```
## S3 method for class 'zoo'
tidy(x, ...)
```

Arguments

x	An object of class "zoo"
...	extra arguments (not used)

Value

tidy returns a data frame with one row for each observation in each series, with the following columns:

index	Index (usually date) for the zoo object
series	Name of the series
value	Value of the observation

Examples

```
if (require("zoo", quietly = TRUE)) {
  set.seed(1071)

  # data generated as shown in the zoo vignette
  Z.index <- as.Date(sample(12450:12500, 10))
  Z.data <- matrix(rnorm(30), ncol = 3)
  colnames(Z.data) <- c("Aa", "Bb", "Cc")
  Z <- zoo(Z.data, Z.index)

  tidy(Z)

  if (require("ggplot2", quietly = TRUE)) {
    ggplot(tidy(Z), aes(index, value, color = series)) + geom_line()
    ggplot(tidy(Z), aes(index, value)) + geom_line() +
      facet_wrap(~ series, ncol = 1)

    Zrolled <- rollmean(Z, 5)
    ggplot(tidy(Zrolled), aes(index, value, color = series)) + geom_line()
  }
}
```

Index

aareg_tidiers, 4
acf_tidiers, 5
anova_tidiers, 6, 40
arima, 9
Arima_tidiers, 8
as.data.frame.table, 122
as.data.frame.ts, 123
auc_tidiers, 9
augment, 10
augment.betareg (betareg_tidiers), 11
augment.coxph (coxph_tidiers), 26
augment.data.frame
 (data.frame_tidiers), 30
augment.felm (felm_tidiers), 34
augment.glmRob (robust_tidiers), 93
augment.kmeans (kmeans_tidiers), 52
augment.lm, 10, 45
augment.lm (lm_tidiers), 59
augment.lme (nlme_tidiers), 74
augment.lmRob (robust_tidiers), 93
augment.loess (loess_tidiers), 62
augment.Mclust (mclust_tidiers), 67
augment.merMod (lme4_tidiers), 55
augment.nlrq (rq_tidiers), 96
augment.nls (nls_tidiers), 76
augment.plm (plm_tidiers), 80
augment.poLCA (poLCA_tidiers), 82
augment.prcomp (prcomp_tidiers), 84
augment.rowwise_df
 (rowwise_df_tidiers), 95
augment.rq (rq_tidiers), 96
augment.rqs (rq_tidiers), 96
augment.smooth.spline
 (smooth.spline_tidiers), 102
augment.survreg (survreg_tidiers), 108
augment.tbl_df (rowwise_df_tidiers), 95
augment_.rowwise_df
 (rowwise_df_tidiers), 95
augment_columns, 11
betareg_tidiers, 11
biglm_tidiers, 13
binDesign_tidiers, 15
binWidth_tidiers, 16
boot, 18
boot.ci, 18
boot_tidiers, 18
bootstrap, 17
brms, 20
brms_tidiers, 19
broom, 21
broom-package (broom), 21
btergm, 21, 22
btergm_tidiers, 21
cch, 24
cch_tidiers, 23
cld, 71
cohen.kappa, 50, 51
compact, 25
confint, 14, 26, 35, 46, 60, 72, 73
confint.geeglm, 25, 41, 42
confint_tidy, 26
control.ergm, 34
cooks.distance, 12, 60
coxph_tidiers, 26
cv.glmnet_tidiers, 28
data.frame_tidiers, 30
density, 114
describe, 31
dist, 114
ergm, 34
ergm_tidiers, 32
felm_tidiers, 34
finish_glance, 36
fitdistr_tidiers, 37
fix_data_frame, 38

- ftable, [116](#)
- gam_tidiers, [40](#)
- gamlss_tidiers, [39](#)
- geeglm_tidiers, [41](#)
- glance, [42](#)
- glance.aareg (aareg_tidiers), [4](#)
- glance.Arima (Arima_tidiers), [8](#)
- glance.betareg (betareg_tidiers), [11](#)
- glance.biglm (biglm_tidiers), [13](#)
- glance.binDesign (binDesign_tidiers), [15](#)
- glance.cch (cch_tidiers), [23](#)
- glance.coxph (coxph_tidiers), [26](#)
- glance.cv.glmnet (cv.glmnet_tidiers), [28](#)
- glance.data.frame (data.frame_tidiers), [30](#)
- glance.ergm (ergm_tidiers), [32](#)
- glance.felm (felm_tidiers), [34](#)
- glance.fitdistr (fitdistr_tidiers), [37](#)
- glance.gam (gam_tidiers), [40](#)
- glance.glm (glm_tidiers), [44](#)
- glance.glmnet (glmnet_tidiers), [43](#)
- glance.glmRob (robust_tidiers), [93](#)
- glance.gmm (gmm_tidiers), [45](#)
- glance.htest (htest_tidiers), [48](#)
- glance.kmeans (kmeans_tidiers), [52](#)
- glance.list (list_tidiers), [54](#)
- glance.lm (lm_tidiers), [59](#)
- glance.lme (nlme_tidiers), [74](#)
- glance.lmodel2 (lmodel2_tidiers), [57](#)
- glance.lmRob (robust_tidiers), [93](#)
- glance.matrix (matrix_tidiers), [66](#)
- glance.Mclust (mclust_tidiers), [67](#)
- glance.merMod (lme4_tidiers), [55](#)
- glance.multinom (multinom_tidiers), [72](#)
- glance.nlrq (rq_tidiers), [96](#)
- glance.nls (nls_tidiers), [76](#)
- glance.orcutt (orcutt_tidiers), [79](#)
- glance.plm (plm_tidiers), [80](#)
- glance.poLCA (poLCA_tidiers), [82](#)
- glance.pyears (pyears_tidiers), [88](#)
- glance.ridgelm (ridgelm_tidiers), [91](#)
- glance.rlm (rlm_tidiers), [92](#)
- glance.rowwise_df (rowwise_df_tidiers), [95](#)
- glance.rq (rq_tidiers), [96](#)
- glance.smooth.spline (smooth.spline_tidiers), [102](#)
- glance.stanreg (rstanarm_tidiers), [99](#)
- glance.summary.lm (lm_tidiers), [59](#)
- glance.summaryDefault (summary_tidiers), [105](#)
- glance.survexp (sexpfit_tidiers), [101](#)
- glance.survfit (survfit_tidiers), [106](#)
- glance.survreg (survreg_tidiers), [108](#)
- glance.tbl_df (rowwise_df_tidiers), [95](#)
- glance_.rowwise_df (rowwise_df_tidiers), [95](#)
- glance_optim (optim_tidiers), [78](#)
- glm, [45](#)
- glm_tidiers, [44](#)
- glmnet_tidiers, [43](#)
- glmRob, [94](#)
- gmm_tidiers, [45](#)
- htest_tidiers, [48](#)
- image, [124](#)
- inflate, [49](#)
- insert_NAs, [50](#)
- interp, [54](#), [124](#)
- kappa_tidiers, [50](#)
- kde_tidiers, [51](#)
- kmeans, [53](#)
- kmeans_tidiers, [52](#)
- list_tidiers, [54](#)
- lm_tidiers, [12](#), [13](#), [40](#), [44](#), [59](#), [81](#), [92–94](#)
- lme4_tidiers, [55](#)
- lmodel2_tidiers, [57](#)
- lmRob, [94](#)
- loess_tidiers, [62](#)
- loo.stanreg, [99](#)
- lsmeans_tidiers, [64](#)
- mad, [100](#)
- matrix_tidiers, [66](#)
- Mclust, [68](#)
- mclust_tidiers, [67](#)
- mcmc_tidiers, [68](#)
- mle2_tidiers, [70](#)
- model.frame, [109](#)
- multcomp_tidiers, [71](#)
- multinom_tidiers, [72](#)
- na.action, [27](#), [56](#), [61](#), [63](#), [75](#), [77](#), [110](#)
- nlme_tidiers, [74](#)
- nls, [77](#)

- nls_tidiers, 76
- optim, 78
- optim_tidiers, 54, 78
- orcutt_tidiers, 54, 79
- pairwise.t.test, 119
- pairwise.wilcox.test, 119
- persp, 124
- plm_tidiers, 80
- poLCA_tidiers, 82
- posterior_interval, 99, 100
- power.t.test, 120
- prcomp, 84, 85
- prcomp_tidiers, 84
- predict.coxph, 27
- predict.glm, 59
- print.stanreg, 100
- print.survfit, 107
- process_ergm, 86
- process_geeglm, 87
- process_lm, 87
- process_rq, 88
- pyears, 89
- pyears_tidiers, 88
- rcorr_tidiers, 90
- residuals.coxph, 27
- residuals.glm, 59
- ridgelm_tidiers, 91
- rlm_tidiers, 92
- robust_tidiers, 93
- rowwise_df_tidiers, 10, 95
- rq_tidiers, 96
- rstanarm_tidiers, 99
- sexpfit_tidiers, 101
- smooth.spline_tidiers, 102
- sp_tidiers, 104
- sparse_tidiers, 103
- stan_glm.nb, 100
- stan_glmer, 100
- stan_lm, 100
- summary, 105
- summary.ergm, 32, 34
- summary.lm, 61
- summary.manova, 116, 117
- summary.nls, 77
- summary.ref.grid, 64
- summary.stanreg, 100
- summary_tidiers, 105
- survfit_tidiers, 106
- survreg_tidiers, 108
- svd, 110, 111
- svd_tidiers, 54, 85, 110
- table, 122
- tidy, 112
- tidy.aareg (aareg_tidiers), 4
- tidy.acf (acf_tidiers), 5
- tidy.anova (anova_tidiers), 6
- tidy.aov (anova_tidiers), 6
- tidy.aovlist (anova_tidiers), 6
- tidy.Arima (Arima_tidiers), 8
- tidy.betareg (betareg_tidiers), 11
- tidy.biglm (biglm_tidiers), 13
- tidy.binDesign (binDesign_tidiers), 15
- tidy.binWidth (binWidth_tidiers), 16
- tidy.boot (boot_tidiers), 18
- tidy.brmsfit (brms_tidiers), 19
- tidy.btergm (btergm_tidiers), 21
- tidy.cch (cch_tidiers), 23
- tidy.character (tidy.numeric), 118
- tidy.cld (multcomp_tidiers), 71
- tidy.coefstest, 112
- tidy.confint.glht (multcomp_tidiers), 71
- tidy.coxph (coxph_tidiers), 26
- tidy.cv.glmnet (cv.glmnet_tidiers), 28
- tidy.data.frame (data.frame_tidiers), 30
- tidy.default, 113
- tidy.density, 114
- tidy.dgCMatrix (sparse_tidiers), 103
- tidy.dgTMatrix (sparse_tidiers), 103
- tidy.dist, 114
- tidy.ergm (ergm_tidiers), 32
- tidy.felm (felm_tidiers), 34
- tidy.fitdistr (fitdistr_tidiers), 37
- tidy.ftable, 115
- tidy.gam (gam_tidiers), 40
- tidy.gamlss (gamlss_tidiers), 39
- tidy.geeglm (geeglm_tidiers), 41
- tidy.glht (multcomp_tidiers), 71
- tidy.glmnet (glmnet_tidiers), 43
- tidy.glmRob (robust_tidiers), 93
- tidy.gmm (gmm_tidiers), 45
- tidy.htest (htest_tidiers), 48
- tidy.kappa (kappa_tidiers), 50
- tidy.kde (kde_tidiers), 51

- tidy.kmeans (kmeans_tidiers), 52
- tidy.Line (sp_tidiers), 104
- tidy.Lines (sp_tidiers), 104
- tidy.list, 110, 111
- tidy.list (list_tidiers), 54
- tidy.lm, 45, 79, 81
- tidy.lm (lm_tidiers), 59
- tidy.lme (nlme_tidiers), 74
- tidy.lmodel2 (lmodel2_tidiers), 57
- tidy.lmRob (robust_tidiers), 93
- tidy.logical (tidy.numeric), 118
- tidy.lsmobj (lsmeans_tidiers), 64
- tidy.manova, 116
- tidy.map, 117
- tidy.matrix (matrix_tidiers), 66
- tidy.Mclust (mclust_tidiers), 67
- tidy.merMod (lme4_tidiers), 55
- tidy.mle2 (mle2_tidiers), 70
- tidy.multinom (multinom_tidiers), 72
- tidy.nlrq (rq_tidiers), 96
- tidy.nls (nls_tidiers), 76
- tidy.NULL, 118
- tidy.numeric, 118
- tidy.orcutt (orcutt_tidiers), 79
- tidy.pairwise.htest, 119
- tidy.plm (plm_tidiers), 80
- tidy.poLCA (poLCA_tidiers), 82
- tidy.Polygon (sp_tidiers), 104
- tidy.Polygons (sp_tidiers), 104
- tidy.power.htest, 120
- tidy.prcomp (prcomp_tidiers), 84
- tidy.pyears (pyears_tidiers), 88
- tidy.rcorr (rcorr_tidiers), 90
- tidy.ref.grid (lsmeans_tidiers), 64
- tidy.ridgeIm (ridgeIm_tidiers), 91
- tidy.rjags (mcmc_tidiers), 68
- tidy.roc (auc_tidiers), 9
- tidy.rowwise_df (rowwise_df_tidiers), 95
- tidy.rq (rq_tidiers), 96
- tidy.rqs (rq_tidiers), 96
- tidy.sparseMatrix (sparse_tidiers), 103
- tidy.SpatialLinesDataFrame
(sp_tidiers), 104
- tidy.SpatialPolygons (sp_tidiers), 104
- tidy.SpatialPolygonsDataFrame
(sp_tidiers), 104
- tidy.spec, 121
- tidy.stanfit (mcmc_tidiers), 68
- tidy.stanreg (rstanarm_tidiers), 99
- tidy.summary.glht (multcomp_tidiers), 71
- tidy.summary.lm (lm_tidiers), 59
- tidy.summaryDefault (summary_tidiers),
105
- tidy.survexp (sexpfit_tidiers), 101
- tidy.survfit (survfit_tidiers), 106
- tidy.survreg (survreg_tidiers), 108
- tidy.table, 122
- tidy.tbl_df (rowwise_df_tidiers), 95
- tidy.ts, 122
- tidy.TukeyHSD, 123
- tidy.zoo (zoo_tidiers), 125
- tidy_.rowwise_df (rowwise_df_tidiers),
95
- tidy_optim (optim_tidiers), 78
- tidy_svd (svd_tidiers), 110
- tidy_xyz (xyz_tidiers), 124
- tidyMCMC (mcmc_tidiers), 68
- TukeyHSD, 123
- unrowname, 124
- xyz_tidiers, 54, 124
- zoo_tidiers, 125