Package ‘texreg’

June 18, 2020

Version 1.37.5
Date 2020-06-17
Title Conversion of R Regression Output to LaTeX or HTML Tables
Description Converts coefficients, standard errors, significance stars, and goodness-of-fit statistics of statistical models into LaTeX tables or HTML tables/MS Word documents or to nicely formatted screen output for the R console for easy model comparison. A list of several models can be combined in a single table. The output is highly customizable. New model types can be easily implemented. (If the Zelig package, which this package enhances, cannot be found on CRAN, you can find it at <https://github.com/IQSS/Zelig>.)

URL http://github.com/leifeld/texreg/

BugReports https://github.com/leifeld/texreg/issues

Suggests broom (>= 0.4.2), coda (>= 0.19.2), ggplot2 (>= 3.1.0), huxtable (>= 4.2.0), knitr (>= 1.22), rmarkdown (>= 1.12.3), sandwich (>= 2.3-1), testthat (>= 2.0.0), lme4 (>= 0.9-34)

Depends R (>= 3.5)
Imports methods, stats, httr
Enhances AER, alpaca, bife, biglm, brms (>= 2.8.8), btergm, dynlm, feisr (>= 1.0.1), forecast, gamlss, gamlss.inf, glmmTMB, lfe, lme4, miceadds, mlogit, mnlogit, MuMIn, nlme, nnet, ordinal, plm, speedglm, survival, VGAM, Zelig

SystemRequirements pandoc (>= 1.12.3) suggested for using wordreg function; LaTeX packages tikz, booktabs, dcolumn, rotating, thumbpdf, longtable, paralist for the vignette

License GPL-3
Encoding UTF-8

RoxygenNote 7.1.0

NeedsCompilation no

Author Philip Leifeld [aut, cre], Claudia Zucca [ctb]

Maintainer Philip Leifeld <philip.leifeld@essex.ac.uk>

Repository CRAN

Date/Publication 2020-06-18 11:00:02 UTC
### R topics documented:

- texreg-package .................................................. 4
- createTexreg ..................................................... 5
- extract ............................................................ 7
- extract.aftreg-method .......................................... 8
- extract.ANY-method ............................................ 9
- extract.Arima-method .......................................... 9
- extract.averaging-method ..................................... 10
- extract.bam-method ............................................. 10
- extract.betamfx-method ...................................... 11
- extract.betaor-method ....................................... 12
- extract.betareg-method ....................................... 13
- extract.bife-method .......................................... 13
- extract.biglm-method ........................................ 14
- extract.brglm-method ......................................... 15
- extract.brmsfit-method ....................................... 15
- extract.btergm-method ....................................... 17
- extract.censReg-method ...................................... 17
- extract.clm-method ........................................... 18
- extract.clmm-method ......................................... 19
- extract.clogit-method ........................................ 20
- extract.coeftest-method ..................................... 20
- extract.coxph-method ......................................... 21
- extract.coxph.penal-method ................................ 22
- extract.coxreg-method ........................................ 23
- extract.dynlm-method ......................................... 23
- extract.ergm-method .......................................... 24
- extract.ergmm-method ........................................ 25
- extract.ets-method ............................................. 25
- extract.feglm-method ......................................... 26
- extract.feis-method ............................................ 27
- extract.felm-method ........................................... 28
- extract.fGARCH-method ....................................... 29
- extract.forecast-method ..................................... 29
- extract.forecast_ARIMA-method ................................ 30
- extract.gam-method ............................................ 30
- extract.gamlss-method ....................................... 32
- extract.gamlssZadj-method .................................. 32
- extract.gee-method ............................................ 33
- extract.geeglm-method ....................................... 34
- extract.gel-method ............................................ 34
- extract glm-method ............................................ 35
- extract.glm.cluster-method .................................. 36
- extract.glmerMod-method ..................................... 37
- extract.glmmadmb-method .................................... 38
- extract.glmmPQL-method ...................................... 39
- extract.glmmTMB-method ...................................... 39
Description

texreg converts coefficients, standard errors, uncertainty measures, and goodness-of-fit statistics of statistical models into LaTeX or HTML tables or into nicely formatted screen output for the R console. A list of several models can be combined in a single table. The output is customizable. New model types can be easily implemented. Confidence intervals can be used instead of standard errors and p-values.
Author(s)

Philip Leifeld

References


See Also

extract texreg

createTexreg

Constructor for texreg objects

Description

Constructor for texreg objects.

Usage

createTexreg(
  coef.names,
  coef,
  se = numeric(0),
  pvalues = numeric(0),
  ci.low = numeric(0),
  ci.up = numeric(0),
  gof.names = character(0),
  gof = numeric(0),
  gof.decimal = logical(0),
  model.name = character(0)
)

Arguments

cogl.names The names for the covariates in a model as a character vector (= row names).
coef The coefficients as a numeric vector. Can have length zero.
se The standard errors as a numeric vector. Can have length zero.
pvalues The p-values as a numeric vector. Can have length zero.
ci.low The lower bounds of the confidence intervals as a numeric vector. Can have length zero.
ci.up The upper bounds of the confidence intervals as a numeric vector. Can have length zero.
gof.names  Names of the goodness-of-fit statistics as a character vector. Can have length zero.
gof  Goodness-of-fit statistics as a numeric vector. Can have length zero.
gof.decimal  A logical vector with as many elements as the gof argument, indicating whether the respective GOF statistic is a double (TRUE) or integer (FALSE) number or whether it is a character entry (NA).
model.name  A name for the statistical model. Can be a character vector of length zero if there is no model name.

Details

This function creates a texreg object. A texreg object contains information about coefficients, standard errors, p-values (optional), and about goodness-of-fit statistics. Instead of standard errors and p-values, a texreg object may also contain upper and lower bounds of a confidence interval. texreg objects are used by the texreg function to create LaTeX tables and other representations of the model results.

Value

A texreg object representing the statistical model.

Author(s)

Philip Leifeld

References


See Also

extract

Examples

library("nlme") # load library for fitting linear mixed effects models
model <- lme(distance ~ age, data = Orthodont, random =~ 1) # estimate
coefficient.names <- rownames(summary(model)$tTable) # extract coef names
coefficients <- summary(model)$tTable[, 1] # extract coefficient values
standard.errors <- summary(model)$tTable[, 2] # extract standard errors
significance <- summary(model)$tTable[, 5] # extract p-values
lik <- summary(model)$logLik # extract log likelihood
aic <- summary(model)$AIC # extract AIC
bic <- summary(model)$BIC # extract BIC
n <- nobs(model) # extract number of observations
gof <- c(aic, bic, lik, n) # create a vector of GOF statistics
gof.names <- c("AIC", "BIC", "Log Likelihood", "Num. obs.") # names of GOFs
decimal.places <- c(TRUE, TRUE, TRUE, FALSE) # last one is a count variable
# create the texreg object
tr <- createTexreg(coef.names = coefficient.names,
                   coef = coefficients,
                   se = standard.errors,
                   pvalues = significance,
                   gof.names = gof.names,
                   gof = gof,
                   gof.decimal = decimal.places)

---

**extract**  
*Extract details from statistical models for table construction*

**Description**

Extract details from statistical models for table construction. The function has methods for a range of statistical models.

**Usage**

`extract(model, ...)`

**Arguments**

- `model` A statistical model object.
- `...` Custom parameters, which are handed over to subroutines. The arguments are usually passed to the `summary` function, but in some cases to other functions.

**Details**

The `extract` function serves to retrieve coefficients, standard errors, p-values, confidence intervals, and goodness-of-fit statistics from statistical models in R. More than 100 `extract` methods ("extensions") for various statistical models are available. The function returns a `texreg` object.

`extract` is a generic function, which extracts coefficients and GOF measures from statistical model objects. `extract` methods for the specific model types are called by the generic `extract` function if it encounters a model known to be handled by the specific method. The output is a `texreg` object, which is subsequently used by the `texreg` function and related functions.

To list the model classes for which `extract` methods exist, type `showMethods("extract")` or `methods("extract")`. To show the method definition (i.e., the function body) of a specific `extract` method, use the `getMethod("extract","lm")` for linear models. To get help on a specific `extract` method, type something like `?extract,lm-method` (or something similar for other models, where "lm" needs to be replaced by the class name of the respective model).

Users can contribute their own extensions for additional statistical models. Examples are contained in the article in the Journal of Statistical Software referenced below. Suggestions can be submitted as pull requests at [https://github.com/leifeld/texreg/pulls](https://github.com/leifeld/texreg/pulls) or through the issue tracker at [https://github.com/leifeld/texreg/issues](https://github.com/leifeld/texreg/issues).
extract.aftreg-method

Value
The function returns a texreg object.

Author(s)
Philip Leifeld

References

See Also
createTexreg, matrixreg, screenreg, texreg

descr
extract.aftreg-method  extract method for aftreg objects

descr
Description
extract method for aftreg objects created by the aftreg function in the eha package.

Usage
## S4 method for signature 'aftreg'
extract(
  model,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.lr = TRUE,
  include.nobs = TRUE,
  include.events = TRUE,
  include.trisk = TRUE,
  ...
)

Arguments
model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.lr Report likelihood ratio test?
include.nobs Report the number of observations in the GOF block?
include.events Report the number of events in the GOF block?
include.trisk Report the total time at risk (in event-history models)?
... Custom parameters, which are handed over to subroutines. Currently not in use.
extract, ANY-method

extract, ANY-method  extract method for broom objects

Description

extract method for broom objects created by the broom function in the broom package.

Usage

## S4 method for signature 'ANY'
extract(model, ...)

Arguments

model  A statistical model object.
...

Custom parameters, which are handed over to subroutines. Currently not in use.

extract, Arima-method

extract, Arima-method  extract method for Arima objects

Description

extract method for Arima objects created by the arima function in the stats package.

Usage

## S4 method for signature 'Arima'
extract(
  model,
  include.pvalues = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model  A statistical model object.
include.pvalues  Report p-values?
include.aic  Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic  Report the Bayesian Information Criterion (BIC) in the GOF block?
extract, averaging-method

 extract method for averaging objects

Description

extract method for averaging objects created by the model.avg function in the MuMIn package.

Usage

```r
## S4 method for signature 'averaging'
extract(model, use.ci = FALSE, adjusted.se = FALSE, include.nobs = TRUE, ...)
```

Arguments

- `model`: A statistical model object.
- `use.ci`: Report confidence intervals in the GOF block?
- `adjusted.se`: Report adjusted standard error in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines. Currently not in use.

extract, bam-method

 extract method for bam objects

Description

extract method for bam objects created by the bam function in the mgcv package.

Usage

```r
## S4 method for signature 'bam'
extract(model,
    model,
    use.ci = TRUE,
    include.aic = TRUE,
    include.bic = TRUE,
    include.loglik = TRUE,
    include.deviance = TRUE,
    include.dev.expl = TRUE,
    include.dispersion = TRUE,
```
Arguments

model A statistical model object.
include.smooth Report the smooth terms of a GAM? If they are reported, the EDF value is reported as the coefficient, and DF is included in parentheses (not standard errors because a chi-square test is used for the smooth terms).
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.dev.expl Report the deviance explained?
include.dispersion Report the dispersion parameter?
include.rsquared Report $R^2$ in the GOF block?
include.gcv Report the GCV score?
include.nobs Report the number of observations in the GOF block?
include.nsmooth Report the number of smooth terms?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

extract method for betamfx objects created by the betamfx function in the mfx package.
Usage

## S4 method for signature 'betamfx'
extract(
  model,
  include.pseudors = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
include.pseudors Report pseudo R^2 in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Description

**extract** method for **betaor** objects created by the **betaor** function in the **mfx** package.

Usage

## S4 method for signature 'betaor'
extract(
  model,
  include.pseudors = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
include.pseudors Report pseudo R^2 in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.
Description

extract method for betareg objects created by the betareg function in the betareg package.

Usage

```r
## S4 method for signature 'betareg'
extract(
  model,
  include.precision = TRUE,
  include.pseudors = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- `model` A statistical model object.
- `include.precision` Report precision in the GOF block?
- `include.pseudors` Report pseudo R^2 in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.nobs` Report the number of observations in the GOF block?
- `...` Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

extract method for bife objects created by the bife function in the bife package.
## S4 method for signature 'bife'

```r
extract(
  model,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)
```

### Arguments
- `model`: A statistical model object.
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.deviance`: Report the residual deviance?
- `include.nobs`: Report the number of observations in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines. Currently not in use.

### Author(s)
Philip Leifeld, Christoph Riedl, Claudia Zucca

---

**extract, biglm-method**  
*extract method for biglm objects*

### Description
`extract` method for `biglm` objects created by the `biglm` function in the `biglm` package.

### Usage
```r
## S4 method for signature 'biglm'
extract(model, include.nobs = TRUE, include.aic = TRUE, use.ci = FALSE, ...)
```

### Arguments
- `model`: A statistical model object.
- `include.nobs`: Report the number of observations in the GOF block?
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `use.ci`: Report confidence intervals in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines. Currently not in use.

### Author(s)
Claudia Zucca, Philip Leifeld
**extract,brglm-method**  
*extract method for brglm objects*

**Description**

*extract* method for *brglm* objects created by the *brglm* function in the *brglm* package.

**Usage**

```r
## S4 method for signature 'brglm'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)
```

**Arguments**

- **model**  
  A statistical model object.
- **include.aic**  
  Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**  
  Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**  
  Report the log likelihood in the GOF block?
- **include.deviance**  
  Report the deviance?
- **include.nobs**  
  Report the number of observations in the GOF block?
- **...**  
  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

**extract,brmsfit-method**  
*extract method for brmsfit objects*

**Description**

*extract* method for *brmsfit* objects created by the *brm* function in the *brms* package.
Usage

```r
# S4 method for signature 'brmsfit'
extract(
  model,
  use.HDI = TRUE,
  level = 0.9,
  include.random = TRUE,
  include.r2 = TRUE,
  include.nobs = TRUE,
  include.1oo.ic = TRUE,
  reloo = FALSE,
  include.waic = TRUE,
  ...
)
```

Arguments

- `model` A statistical model object.
- `use.HDI` Report highest posterior density (HPD) intervals (HDI) using the `HPDinterval` function in the `coda` package, with the probability given in the `level` argument, instead of the default 95 percent posterior quantiles?
- `level` Significance level ($1 - \alpha$) for HPDs (in combination with the `use.HDI` argument).
- `include.random` Include random effects (standard deviations) in the GOF block of the table?
- `include.r2` Report R² in the GOF block?
- `include.nobs` Report the number of observations in the GOF block?
- `include.1oo.ic` Report Leave-One-Out Information Criterion?
- `reloo` Recompute exact cross-validation for problematic observations for which approximate leave-one-out cross-validation may return incorrect results? This is done using the `reloo.brmsfit` function and may take some time to compute.
- `include.waic` Report Widely Applicable Information Criterion (WAIC)?
- `...` Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Author(s)

Hyunjin (Jin) Song, Philip Leifeld
extract,btergm-method  

**extract method for btergm objects**

### Description

*extract* method for *btergm* objects created by the *btergm* function in the *btergm* package.

### Usage

```r
## S4 method for signature 'btergm'
extract(model, level = 0.95, include.nobs = TRUE, ...)
```

### Arguments

- **model**
  A statistical model object.
- **level**
  Significance or confidence level \((1 - \alpha)\) for computing confidence intervals.
- **include.nobs**
  Report the number of observations in the GOF block?
- **...**
  Custom parameters, which are handed over to subroutines. Currently not in use.

---

extract,censReg-method  

**extract method for censReg objects**

### Description

*extract* method for *censReg* objects created by the *censReg* function in the *censReg* package.

### Usage

```r
## S4 method for signature 'censReg'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```
Arguments

- **model**: A statistical model object.
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `clm` objects created by the `clm` function in the `ordinal` package.

Usage

```r
## S4 method for signature 'clm'
extract(
  model,
  include.thresholds = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **include.thresholds**: Report thresholds in the GOF block?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.
extract, clmm-method

**Description**

*extract* method for *clmm* objects created by the *clmm* function in the *ordinal* package.

**Usage**

```r
## S4 method for signature 'clmm'
extract(
  model,
  include.thresholds = TRUE,
  include.loglik = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.thresholds` Report thresholds in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.nobs` Report the number of observations in the GOF block?
- `include.groups` Report the number of groups?
- `include.variance` Report group variances?
- `...` Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
extract,clogit-method  extract method for clogit objects

Description

extract method for clogit objects created by the clogit function in the survival package.

Usage

## S4 method for signature 'clogit'
extract(
  model,
  include.aic = TRUE,
  include.rsquared = TRUE,
  include.maxrs = TRUE,
  include.events = TRUE,
  include.nobs = TRUE,
  include.missings = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.rsquared Report R^2 in the GOF block?
include.maxrs Report maximal R^2 in the GOF block?
include.events Report the number of events in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.missings Report number of missing data points in the GOF block?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

extract,coeftest-method  extract method for coeftest objects

Description

extract method for coeftest objects created by the coeftest function in the lmtest package.
**Usage**

```r
## S4 method for signature 'coeftest'
extract(model, ...)
```

**Arguments**

- `model`: A statistical model object.
- `...`: Custom parameters, which are handed over to subroutines. Currently not in use.

---

**Description**

`extract` method for `coxph` objects created by the `coxph` function in the `survival` package.

**Usage**

```r
## S4 method for signature 'coxph'
extract(
  model,
  include.aic = TRUE,
  include.rsquared = TRUE,
  include.maxrs = TRUE,
  include.events = TRUE,
  include.nobs = TRUE,
  include.missings = TRUE,
  include.zph = TRUE,
  ...
)
```

**Arguments**

- `model`: A statistical model object.
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.rsquared`: Report $R^2$ in the GOF block?
- `include.maxrs`: Report maximal $R^2$ in the GOF block?
- `include.events`: Report the number of events in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block?
- `include.missings`: Report number of missing data points in the GOF block?
- `include.zph`: Report proportional hazard test in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
Description

extract method for coxph.penal objects created by the coxph function in the survival package.

Usage

## S4 method for signature 'coxph.penal'
extract(
  model,
  include.aic = TRUE,
  include.rsquared = TRUE,
  include.maxrs = TRUE,
  include.events = TRUE,
  include.nobs = TRUE,
  include.missings = TRUE,
  include.zph = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike's Information Criterion (AIC) in the GOF block?
include.rsquared Report R^2 in the GOF block?
include.maxrs Report maximal R^2 in the GOF block?
include.events Report the number of events in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.missings Report number of missing data points in the GOF block?
include.zph Report proportional hazard test in the GOF block?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
extract,coxreg-method  

**extract method for coxreg objects**

**Description**

*extract* method for coxreg objects created by the *coxreg* function in the *eha* package.

**Usage**

```r
## S4 method for signature 'coxreg'
extract(
  model,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.lr = TRUE,
  include.nobs = TRUE,
  include.events = TRUE,
  include.trisk = TRUE,
  ...
)
```

**Arguments**

- *model*: A statistical model object.
- *include.aic*: Report Akaike’s Information Criterion (AIC) in the GOF block?
- *include.loglik*: Report the log likelihood in the GOF block?
- *include.lr*: Report likelihood ratio test?
- *include.nobs*: Report the number of observations in the GOF block?
- *include.events*: Report the number of events in the GOF block?
- *include.trisk*: Report the total time at risk (in event-history models)?
- *...*: Custom parameters, which are handed over to subroutines. Currently not in use.

---

extract,dynlm-method  

**extract method for dynlm objects**

**Description**

*extract* method for dynlm objects created by the *dynlm* function in the *dynlm* package.
Usage

```r
## S4 method for signature 'dynlm'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  include.fstatistic = FALSE,
  include.rmse = FALSE,
  ...
)
```

Arguments

- `model`: A statistical model object.
- `include.rsquared`: Report R^2 in the GOF block?
- `include.adjrs`: Report adjusted R^2 in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block?
- `include.fstatistic`: Report the F-statistic in the GOF block?
- `include.rmse`: Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `ergm` objects created by the `ergm` function in the `ergm` package.

Usage

```r
## S4 method for signature 'ergm'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  ...
)
```
**extract.ergmm-method**

**Arguments**

- **model**: A statistical model object.
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

**Description**

`extract` method for `ergmm` objects created by the `ergmm` function in the `latentnet` package.

**Usage**

```r
## S4 method for signature 'ergmm'
extract(model, include.bic = TRUE, ...)
```

**Arguments**

- **model**: A statistical model object.
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines. Currently not in use.

**extract.ets-method**

**Description**

`extract` method for `ets` objects created by the `ets` function in the `forecast` package.

**Usage**

```r
## S4 method for signature 'ets'
extract(model,
    include.pvalues = FALSE,
    include.aic = TRUE,
    include.aicc = TRUE,
    include.bic = TRUE,
    include.loglik = TRUE,
    ...
)
```
extract.feglm-method

Arguments

model  A statistical model object.
include.pvalues Report p-values?
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.aicc Report AICC in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Description

extract method for feglm objects created by the feglm function in the alpaca package.

Usage

## S4 method for signature 'feglm'
extract(
  model,
  include.deviance = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  ...
)

Arguments

model  A statistical model object.
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?
include.groups Report the number of groups?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Author(s)

Christoph Riedl, Oliver Reiter, Philip Leifeld
## S4 method for signature 'feis'
extract(model,
   include.rsquared = TRUE,
   include.adjrs = TRUE,
   include.nobs = TRUE,
   include.groups = TRUE,
   include.rmse = TRUE,
   ...
)

### Arguments

- **model**: A statistical model object.
- **include.rsquared**: Report $R^2$ in the GOF block?
- **include.adjrs**: Report adjusted $R^2$ in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **include.groups**: Report the number of groups?
- **include.rmse**: Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

### Author(s)

Tobias Rüttenauer, Philip Leifeld
**Extract, felm-method**

*extract method for felm objects*

**Description**

`extract` method for `felm` objects created by the `felm` function in the *lfe* package.

**Usage**

```r
## S4 method for signature 'felm'
extract(
  model,
  include.nobs = TRUE,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.fstatistic = FALSE,
  include.groups = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.nobs` Report the number of observations in the GOF block?
- `include.rsquared` Report $R^2$ in the GOF block?
- `include.adjrs` Report adjusted $R^2$ in the GOF block?
- `include.fstatistic` Report the F-statistic in the GOF block?
- `include.groups` Report the number of groups?
- `...` Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

**Author(s)**

Christoph Riedl, Claudia Zucca, Oliver Reiter, Philip Leifeld
extract,fGARCH-method  extract method for fGARCH objects

Description

extract method for fGARCH objects created by the garchFit function in the fGarch package.

Usage

## S4 method for signature 'fGARCH'
extract(
  model,
  include.nobs = TRUE,
  include.aic = TRUE,
  include.loglik = TRUE,
  ...
)

Arguments

model A statistical model object.
include.nobs Report the number of observations in the GOF block?
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.

extract,forecast-method  extract method for forecast objects

Description

extract method for forecast objects created by the forecast and holt functions in the forecast package.

Usage

## S4 method for signature 'forecast'
extract(model, ...)

Arguments

model A statistical model object.
... Custom parameters, which are handed over to subroutines. Currently not in use.
extract,forecast_ARIMA-method

extract method for forecast_ARIMA objects

Description

extract method for forecast_ARIMA objects created by the Arima function in the forecast package.

Usage

```r
## S4 method for signature 'forecast_ARIMA'
extract(
  model,
  include.pvalues = TRUE,
  include.aic = TRUE,
  include.aicc = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **include.pvalues**: Report p-values?
- **include.aic**: Report Akaike's Information Criterion (AIC) in the GOF block?
- **include.aicc**: Report AICC in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines. Currently not in use.

extract,gam-method

extract method for gam objects

Description

extract method for gam objects created by the gam function in the mgcv package.
Usage

```r
## S4 method for signature 'gam'
extract(
  model,
  include.smooth = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.dev.expl = TRUE,
  include.dispersion = TRUE,
  include.rsquared = TRUE,
  include.gcv = TRUE,
  include.nobs = TRUE,
  include.nsmooth = TRUE,
  ...
)
```

Arguments

- `model`: A statistical model object.
- `include.smooth`: Report the smooth terms of a GAM? If they are reported, the EDF value is reported as the coefficient, and DF is included in parentheses (not standard errors because a chi-square test is used for the smooth terms).
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic`: Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.deviance`: Report the deviance?
- `include.dev.expl`: Report the deviance explained?
- `include.dispersion`: Report the dispersion parameter?
- `include.rsquared`: Report R^2 in the GOF block?
- `include.gcv`: Report the GCV score?
- `include.nobs`: Report the number of observations in the GOF block?
- `include.nsmooth`: Report the number of smooth terms?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.
extract.gamlss-method  
**extract method for gamlss objects**

### Description

`extract` method for `gamlss` objects created by the `gamlss` function in the `gamlss` package.

### Usage

```r
## S4 method for signature 'gamlss'
extract(
  model,
  robust = FALSE,
  include.nobs = TRUE,
  include.nagelkerke = TRUE,
  include.gaic = TRUE,
  ...
)
```

### Arguments

- `model` A statistical model object.
- `robust` If TRUE computes robust standard errors in the variance-covariance matrix.
- `include.nobs` Report the number of observations in the GOF block?
- `include.nagelkerke` Report Nagelkerke R^2 in the GOF block?
- `include.gaic` Report Generalized Akaike's Information Criterion (AIC) in the GOF block?
- `...` Custom parameters, which are handed over to subroutines, in this case to the `vcov` method for the object.

extract.gamlssZadj-method  
**extract method for gamlssZadj objects**

### Description

`extract` method for `gamlssZadj` objects created by the `gamlssZadj` function in the `gamlss.inf` package.
Usage

## S4 method for signature 'gamlssZadj'
extract(
  model,
  type = c("qr", "vcov"),
  include.nobs = TRUE,
  include.gaic = TRUE,
  ...
)

Arguments

  model          A statistical model object.
  type           The type.
  include.nobs   Report the number of observations in the GOF block?
  include.gaic   Report Generalized Akaike's Information Criterion (AIC) in the GOF block?
  ...            Custom parameters, which are handed over to subroutines, in this case to the vcov method for the object.

Author(s)

Ricardo Graiff Garcia, Philip Leifeld

---

extract,gee-method  extract method for gee objects

Description

extract method for gee objects created by the gee function in the gee package.

Usage

## S4 method for signature 'gee'
extract(model, robust = TRUE, include.scale = TRUE, include.nobs = TRUE, ...)

Arguments

  model          A statistical model object.
  robust         If TRUE computes robust standard errors in the variance-covariance matrix.
  include.scale  Report the dispersion or scale parameter?
  include.nobs   Report the number of observations in the GOF block?
  ...            Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
**extract.geeglm-method**  
*extract method for geeglm objects*

**Description**

*extract* method for geeglm objects created by the `geeglm` function in the *geepack* package.

**Usage**

```r
## S4 method for signature 'geeglm'
extract(
  model,
  include.scale = TRUE,
  include.correlation = TRUE,
  include.nobs = TRUE,
  ...
)
```

**Arguments**

- `model`  
  A statistical model object.
- `include.scale`  
  Report the dispersion or scale parameter?
- `include.correlation`  
  Report the correlation parameter alpha and its standard error?
- `include.nobs`  
  Report the number of observations in the GOF block?
- `...`  
  Custom parameters, which are handed over to subroutines. Currently not in use.

---

**extract.gel-method**  
*extract method for gel objects*

**Description**

*extract* method for gel objects created by the `gel` function in the *gmm* package.

**Usage**

```r
## S4 method for signature 'gel'
extract(
  model,
  include.obj.fcn = TRUE,
  include.overidentification = FALSE,
  include.nobs = TRUE,
  overIdentTest = c("LR", "LM", "J"),
  ...
)
```

**Arguments**

- `model`  
  A statistical model object.
- `include.obj.fcn`  
  Report the dispersion or scale parameter?
- `include.overidentification`  
  Report the correlation parameter alpha and its standard error?
- `include.nobs`  
  Report the number of observations in the GOF block?
- `overIdentTest`  
  Custom parameters, which are handed over to subroutines. Currently not in use.
extract,glm-method

Arguments

model A statistical model object.
include.objfcn Report the value of the objective function (= criterion function)? More precisely, this returns \( E(g) \text{var}(g)^{-1}E(g) \).
include.overidentification Report the J-test for overidentification?
include.nobs Report the number of observations in the GOF block?
overIdentTest Which test statistics should be included in an overidentification test?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

extract,glm-method  extract method for glm objects

Description

extract method for glm objects created by the glm function in the stats package.

Usage

## S4 method for signature 'glm'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
**Description**

`extract` method for `glm.cluster` objects created by the `glm.cluster` function in the `miceadds` package.

**Usage**

```r
## S4 method for signature 'glm.cluster'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.deviance` Report the deviance?
- `include.nobs` Report the number of observations in the GOF block?
- `...` Custom parameters, which are handed over to subroutines. Currently not in use.

**Author(s)**

Alexander Staudt, Philip Leifeld
extract,glmerMod-method

**extract method for glmerMod objects**

**Description**

`extract` method for glmerMod objects created by the `glmer` function in the `lme4` package.

**Usage**

```r
## S4 method for signature 'glmerMod'
extract(
  model,
  method = c("naive", "profile", "boot", "Wald"),
  level = 0.95,
  nsim = 1000,
  include.aic = TRUE,
  include.bic = TRUE,
  include.dic = FALSE,
  include.deviance = FALSE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `method` The method used to compute confidence intervals or p-values. The default value "naive" computes naive p-values while the other methods compute confidence intervals using the `confint` function. See `confint.merMod`.
- `level` Significance or confidence level ($1 - \alpha$) for computing confidence intervals.
- `nsim` The MCMC sample size or number of bootstrapping replications on the basis of which confidence intervals are computed (only if the method argument does not specify "naive", which is the default behavior). Note: large values may take considerable computing time.
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.dic` Report the deviance information criterion (DIC)?
- `include.deviance` Report the deviance?
- `include.loglik` Report the log likelihood in the GOF block?
extract method for glmmadmb objects

Description

`extract` method for `glmmadmb` objects created by the `glmmadmb` function in the `glmmADMB` package.

Usage

```r
## S4 method for signature 'glmmadmb'
extract(
  model,
  include.variance = TRUE,
  include.dispersion = TRUE,
  include.zero = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  ...
)
```

Arguments

- `model` A statistical model object.
- `include.variance` Report group variances?
- `include.dispersion` Report the dispersion parameter?
- `include.zero` Should the binary part of a zero-inflated regression model or hurdle model be included in the coefficients block (after the count model)?
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.nobs` Report the number of observations in the GOF block?
- `include.groups` Report the number of groups?
- `...` Custom parameters, which are handed over to subroutines. Currently not in use.
Description

`extract` method for `glmmPQL` objects created by the `glmmPQL` function in the `MASS` package.

Usage

```r
## S4 method for signature 'glmmPQL'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = FALSE,
  ...
)
```

Arguments

- `model` A statistical model object.
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.nobs` Report the number of observations in the GOF block?
- `include.groups` Report the number of groups?
- `include.variance` Report group variances?
- `...` Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `glmmTMB` objects created by the `glmmTMB` function in the `glmmTMB` package.
Usage

## S4 method for signature 'glmmTMB'
extract(
  model,
  beside = FALSE,
  include.count = TRUE,
  include.zero = TRUE,
  include.aic = TRUE,
  include.groups = TRUE,
  include.variance = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
beside Arrange the model terms below each other or beside each other? The binary
  model parameters and the count parameters can be displayed in two separate
  columns of the table.
include.count Report the count parameters in the coefficients block (before the binary part for
  the zeros)?
include.zero Should the binary part of the model be included in the coefficients block (after
  the count parameters)?
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.groups Report the number of groups?
include.variance Report group variances?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines, in this case to the
  summary method for the object.

Author(s)

Ricardo Graiff Garcia, Philip Leifeld
extract,glmrob-method  

**extract method for glmrob objects**

**Description**

`extract` method for glmrob objects created by the `glmrob` function in the `robustbase` package.

**Usage**

```r
## S4 method for signature 'glmrob'
extract(model, include.nobs = TRUE, ...)
```

**Arguments**

- `model` A statistical model object.
- `include.nobs` Report the number of observations in the GOF block?
- `...` Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

---

extract,gls-method  

**extract method for gls objects**

**Description**

`extract` method for gls objects created by the `gls` function in the `nlme` package.

**Usage**

```r
## S4 method for signature 'gls'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.aic` Report Akaike's Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
Description

*extract* method for *gmm* objects created by the *gmm* function in the *gmm* package.

Usage

```r
## S4 method for signature 'gmm'
extract(
  model,
  include.obj.fcn = TRUE,
  include.overidentification = FALSE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **include.obj.fcn**: Report the value of the objective function (= criterion function)? More precisely, this returns $E(g) \text{var}(g)^{-1}E(g)$.
- **include.overidentification**: Report the J-test for overidentification?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

---

Description

*extract* method for *gnls* objects created by the *gnls* function in the *nlme* package.

Usage

```r
## S4 method for signature 'gnls'
extract(
  model, 
  include.obj.fcn = TRUE,
  include.nobs = TRUE,
  ... 
)
```

Arguments

- **model**: A statistical model object.
- **include.obj.fcn**: Report the value of the objective function (= criterion function)? More precisely, this returns $E(g) \text{var}(g)^{-1}E(g)$.
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
Usage

```r
## S4 method for signature 'gnls'
extract(
    model,
    include.aic = TRUE,
    include.bic = TRUE,
    include.loglik = TRUE,
    include.nobs = TRUE,
    ...
)
```

Arguments

- `model`: A statistical model object.
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic`: Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `gnm` objects created by the `gnm` function in the `gnm` package.

Usage

```r
## S4 method for signature 'gnm'
extract(
    model,
    include.aic = TRUE,
    include.bic = TRUE,
    include.loglik = TRUE,
    include.deviance = TRUE,
    include.nobs = TRUE,
    include.df = FALSE,
    include.chisq = FALSE,
    include.delta = FALSE,
    ...
)
```
extract,H2OBinomialModel-method

extract method for H2OBinomialModel objects

Arguments

model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?
include.df Report the degrees of freedom?
include.chisq Report the chi squared statistic?
include.delta Report the delta statistic?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Usage

## S4 method for signature 'H2OBinomialModel'
extract(model,
    standardized = FALSE,
    include.mse = TRUE,
    include.rsquared = TRUE,
    include.logloss = TRUE,
    include.meanerror = TRUE,
    include.auc = TRUE,
    include.gini = TRUE,
    include.deviance = TRUE,
    include.aic = TRUE,
    ...)

Description

eextract method for H2OBinomialModel objects created by the h2o.glm function in the h2o package.
extract.hurdle-method

Arguments

- **model**: A statistical model object.
- **standardized**: Report standardized coefficients instead of raw coefficients?
- **include.mse**: Report the mean squared error in the GOF block?
- **include.rsquared**: Report $R^2$ in the GOF block?
- **include.logloss**: Report the log loss?
- **include.meanerror**: Report the mean per-class error?
- **include.auc**: Report the area under the curve (AUC)?
- **include.gini**: Report the Gini coefficient?
- **include.deviance**: Report the deviance?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- ... Custom parameters, which are handed over to subroutines. Currently not in use.

extract.hurdle-method  extract method for hurdle objects

Description

extract method for hurdle objects created by the hurdle function in the pscl package.

Usage

```r
## S4 method for signature 'hurdle'
extract(
  model,
  beside = FALSE,
  include.count = TRUE,
  include.zero = TRUE,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **beside**: Arrange the model terms below each other or beside each other? The binary model parameters and the count parameters can be displayed in two separate columns of the table.
include.count  Report the count parameters in the coefficients block (before the binary part for the zeros)?
include.zero  Should the binary part of the model be included in the coefficients block (after the count parameters)?
include.aic  Report Akaike’s Information Criterion (AIC) in the GOF block?
include.loglik  Report the log likelihood in the GOF block?
include.nobs  Report the number of observations in the GOF block?
...
Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description
eextract method for ivreg objects

extract method for ivreg objects created by the ivreg function in the AER package.

Usage
## S4 method for signature 'ivreg'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  include.fstatistic = FALSE,
  include.rmse = FALSE,
  ...
)

Arguments
model  A statistical model object.
include.rsquared  Report R^2 in the GOF block?
include.adjrs  Report adjusted R^2 in the GOF block?
include.nobs  Report the number of observations in the GOF block?
include.fstatistic  Report the F-statistic in the GOF block?
include.rmse  Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?
...
Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
**Description**

*extract* method for *lm* objects created by the *lm* function in the *stats* package.

**Usage**

```r
## S4 method for signature 'lm'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  include.fstatistic = FALSE,
  include.rmse = FALSE,
  ...
)
```

**Arguments**

- **model**
  A statistical model object.

- **include.rsquared**
  Report $R^2$ in the GOF block?

- **include.adjrs**
  Report adjusted $R^2$ in the GOF block?

- **include.nobs**
  Report the number of observations in the GOF block?

- **include.fstatistic**
  Report the F-statistic in the GOF block?

- **include.rmse**
  Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?

- **...**
  Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

**Description**

*extract* method for *lm.cluster* objects created by the *lm.cluster* function in the *miceadds* package.
Usage

```r
## S4 method for signature 'lm.cluster'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  include.fstatistic = FALSE,
  include.rmse = FALSE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **include.rsquared**: Report R^2 in the GOF block?
- **include.adjrs**: Report adjusted R^2 in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **include.fstatistic**: Report the F-statistic in the GOF block?
- **include.rmse**: Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines. Currently not in use.

Author(s)

Alexander Staudt, Philip Leifeld

---

extract, lme-method

**extract method for lme objects**

Description

extract method for lme objects created by the lme function in the nlme package.

Usage

```r
## S4 method for signature 'lme'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
```
Arguments

model A statistical model object.
include.aic Report Akaike's Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.groups Report the number of groups?
include.variance Report group variances?
...

Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

extract method for lme4 objects

extract method for lme4 objects created by the lme4 package.

Usage

## S4 method for signature 'lme4'
extract(
  model,
  method = c("naive", "profile", "boot", "Wald"),
  level = 0.95,
  nsim = 1000,
  include.aic = TRUE,
  include.bic = TRUE,
  include.dic = FALSE,
  include.deviance = FALSE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = TRUE,
  ...)

Arguments

model  A statistical model object.
method The method used to compute confidence intervals or p-values. The default value
"naive" computes naive p-values while the other methods compute confidence
intervals using the confint function. See confint.merMod.
level Significance or confidence level (1 - alpha) for computing confidence intervals.
nsim The MCMC sample size or number of bootstrapping replications on the basis of
which confidence intervals are computed (only if the method argument does not
specify "naive", which is the default behavior). Note: large values may take
considerable computing time.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.dic Report the deviance information criterion (DIC)?
include.deviance Report the deviance?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.groups Report the number of groups?
include.variance Report group variances?
...  Arguments to be passed to the fixef function in the lme4 package.

Description

extract method for lmerMod objects

Usage

## S4 method for signature 'lmerMod'
extract(  
  model,  
  method = c("naive", "profile", "boot", "Wald"),  
  level = 0.95,  
  nsim = 1000,  
  include.aic = TRUE,  
  include.bic = TRUE,  
  include.dic = FALSE,  
  include.deviance = FALSE,  
  include.loglik = TRUE,  
  ...  
)
Arguments

model A statistical model object.

method The method used to compute confidence intervals or p-values. The default value "naive" computes naive p-values while the other methods compute confidence intervals using the `confint` function. See `confint.merMod`.

level Significance or confidence level (1 - \(\alpha\)) for computing confidence intervals.

nsim The MCMC sample size or number of bootstrapping replications on the basis of which confidence intervals are computed (only if the `method` argument does not specify "naive", which is the default behavior). Note: large values may take considerable computing time.

include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?

include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?

include.dic Report the deviance information criterion (DIC)?

include.deviance Report the deviance?

include.loglik Report the log likelihood in the GOF block?

include.nobs Report the number of observations in the GOF block?

include.groups Report the number of groups?

include.variance Report group variances?

... Arguments to be passed to the `fixef` function in the `lme4` package.

Description

`extract.lmrob-method` is an extract method for `lmrob` objects created by the `lmrob` function in the `robustbase` package.

`extract.lmRob-method` is an extract method for `lmRob` objects created by the `lmRob` function in the `robust` package.
Usage

```r
## S4 method for signature 'lmmrob'
extract(model, include.nobs = TRUE, ...)

## S4 method for signature 'lmRob'
extract(
  model,
  include.rsquared = TRUE,
  include.nobs = TRUE,
  include.rmse = TRUE,
  ...
)
```

Arguments

- `model`: A statistical model object.
- `include.nobs`: Report the number of observations in the GOF block?
- `include.rsquared`: Report $R^2$ in the GOF block?
- `include.rmse`: Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `lnam` objects created by the `lnam` function in the `sna` package.

Usage

```r
## S4 method for signature 'lnam'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  ...
)
```
extract.logitmfx-method

Arguments

model: A statistical model object.
include.rsquared: Report R^2 in the GOF block?
include.adjrs: Report adjusted R^2 in the GOF block?
include.aic: Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic: Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik: Report the log likelihood in the GOF block?
... Custom parameters, which are handed over to subroutines, in this case to the coef method for the object.

extract,logitmfx-method

extract method for logitmfx objects

Description

extract method for logitmfx objects created by the logitmfx function in the mfx package.

Usage

## S4 method for signature 'logitmfx'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)

Arguments

model: A statistical model object.
include.nobs: Report the number of observations in the GOF block?
include.loglik: Report the log likelihood in the GOF block?
include.deviance: Report the deviance?
include.aic: Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic: Report the Bayesian Information Criterion (BIC) in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.
extract, logitor-method

**extract method for logitor objects**

**Description**

`extract` method for `logitor` objects created by the `logitor` function in the `mfx` package.

**Usage**

```r
## S4 method for signature 'logitor'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.nobs` Report the number of observations in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.deviance` Report the deviance?
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `...` Custom parameters, which are handed over to subroutines. Currently not in use.

---

extract,lqmm-method

**extract method for lqmm objects**

**Description**

`extract` method for `lqmm` objects created by the `lqmm` function in the `lqmm` package.
Usage

## S4 method for signature 'lqmm'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.tau = FALSE,
  use.ci = FALSE,
  beside = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike's Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.groups Report the number of groups?
include.tau Report tau?
use.ci Report confidence intervals in the GOF block?
beside Arrange the model terms below each other or beside each other?
...
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

_extract method for lrm objects

Usage

## S4 method for signature 'lrm'
extract(
  model,
  include.pseudors = TRUE,
  include.lr = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
include.pseudors Report pseudo R^2 in the GOF block?
include.lr Report likelihood ratio test?
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Author(s)

Fabrice Le Lec

extract,maBina-method 

extract method for maBina objects

Description

extract method for maBina objects created by the maBina function in the erer package.

Usage

## S4 method for signature 'maBina'
extract(model, ...)

Arguments

model A statistical model object.
... Custom parameters, which are handed over to subroutines.

extract,merMod-method 

extract method for merMod objects

Description

extract method for merMod objects created by the lme4 package.
Usage

```r
## S4 method for signature 'merMod'
extract(
  model,
  method = c("naive", "profile", "boot", "Wald"),
  level = 0.95,
  nsim = 1000,
  include.aic = TRUE,
  include.bic = TRUE,
  include.dic = FALSE,
  include.deviance = FALSE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = TRUE,
  ...
)
```

Arguments

- `model` A statistical model object.
- `method` The method used to compute confidence intervals or p-values. The default value "naive" computes naive p-values while the other methods compute confidence intervals using the `confint` function. See `confint.merMod`.
- `level` Significance or confidence level (1 - alpha) for computing confidence intervals.
- `nsim` The MCMC sample size or number of bootstrapping replications on the basis of which confidence intervals are computed (only if the `method` argument does not specify "naive", which is the default behavior). Note: large values may take considerable computing time.
- `include.aic` Report Akaike's Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.dic` Report the deviance information criterion (DIC)?
- `include.deviance` Report the deviance?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.nobs` Report the number of observations in the GOF block?
- `include.groups` Report the number of groups?
- `include.variance` Report group variances?
- `...` Arguments to be passed to the `fixef` function in the `lme4` package.
extract,mhurdle-method

**extract method for mhurdle objects**

Description

`extract` method for mhurdle objects created by the `mhurdle` function in the `mhurdle` package.

Usage

```r
## S4 method for signature 'mhurdle'
extract(model, include.nobs = TRUE, include.loglik = TRUE, ...)
```

Arguments

- `model`: A statistical model object.
- `include.nobs`: Report the number of observations in the GOF block?
- `include.loglik`: Report the log likelihood in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

extract,mlogit-method

**extract method for mlogit objects**

Description

`extract` method for mlogit objects created by the `mlogit` function in the `mlogit` package.

Usage

```r
## S4 method for signature 'mlogit'
extract(
  model,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.order = FALSE,
  include.iterations = FALSE,
  beside = FALSE,
  ...
)
```
extract.mnlogit-method

extract method for mnlogit objects

Description

extract method for mnlogit objects created by the mnlogit function in the mnlogit package.

Usage

```r
## S4 method for signature 'mnlogit'
extract(
  model,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.iterations = FALSE,
  beside = FALSE,
  ...
)
```

Arguments

- `model`: A statistical model object.
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block?
- `include.groups`: Report the number of groups?
- `include.iterations`: Report the number of iterations?
- `beside`: Arrange the model terms below each other or beside each other?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
include.iterations
  Report the number of iterations?

beside
  Arrange the model terms below each other or beside each other?

...  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

---

extract, model.selection-method

extract method for model.selection objects

Description

extract method for model.selection objects created by the model.sel and dredge functions in the MuMIn package.

Usage

```r
## S4 method for signature 'model.selection'
extract(
  model,
  include.loglik = TRUE,
  include.aicc = TRUE,
  include.delta = TRUE,
  include.weight = TRUE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- `model`: A statistical model object.
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.aicc`: Report AICC in the GOF block?
- `include.delta`: Report the delta statistic?
- `include.weight`: Report Akaike weights?
- `include.nobs`: Report the number of observations in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines. Currently not in use.
extract.mtergm-method  

**Description**

*extract* method for *mtergm* objects created by the *mtergm* function in the *btergm* package.

**Usage**

```r
## S4 method for signature 'mtergm'
extract(
  model,
  include.nobs = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  ...
)
```

**Arguments**

- `model`  
  A statistical model object.
- `include.nobs`  
  Report the number of observations in the GOF block?
- `include.aic`  
  Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic`  
  Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik`  
  Report the log likelihood in the GOF block?
- `...`  
  Custom parameters, which are handed over to subroutines. Currently not in use.

extract.multinom-method

**Description**

*extract* method for *multinom* objects created by the *multinom* function in the *nnet* package.

**Usage**

```r
## S4 method for signature 'multinom'
extract(
  model,
  include.pvalues = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
      ...
)
```
include.loglik = TRUE,
include.deviance = TRUE,
include.nobs = TRUE,
include.groups = TRUE,
levels = model$lev,
beside = FALSE,
...
)

Arguments

model A statistical model object.
include.pvalues Report p-values?
include.aic Report Akaike's Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?
include.groups Report the number of groups?
levels The names of the levels of a multinomial model that should be included in the table. Should be provided as a vector of character strings.
beside Arrange the model terms below each other or beside each other?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

extract method for negbin objects created by the glm.nb function in the MASS package.

Usage

## S4 method for signature 'negbin'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)

extract,negbin-method  extract method for negbin objects
extract.negbinirr-method

Arguments

model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?
...

Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

extract method for negbinirr objects

Usage

## S4 method for signature 'negbinirr'
extract(
  model, 
  include.nobs = TRUE, 
  include.loglik = TRUE, 
  include.deviance = TRUE, 
  include.aic = TRUE, 
  include.bic = TRUE, 
  ...
)

Arguments

model A statistical model object.
include.nobs Report the number of observations in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
...

Custom parameters, which are handed over to subroutines. Currently not in use.
**extract,netlogit-method**

**extract method for netlogit objects**

---

**Description**

*extract* method for netlogit objects created by the *netlogit* function in the *sna* package.

**Usage**

```r
## S4 method for signature 'netlogit'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.nobs` Report the number of observations in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.deviance` Report the deviance?
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `...` Custom parameters, which are handed over to subroutines. Currently not in use.

---

**extract,negbinmfx-method**

**extract method for negbinmfx objects**

---

**Description**

*extract* method for negbinmfx objects created by the *negbinmfx* function in the *mfx* package.

**Usage**

```r
## S4 method for signature 'negbinmfx'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.nobs` Report the number of observations in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.deviance` Report the deviance?
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `...` Custom parameters, which are handed over to subroutines. Currently not in use.
extract.nlme-method

Usage

## S4 method for signature 'netlogit'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Description

extract method for nlme objects created by the nlme function in the nlme package.

Usage

## S4 method for signature 'nlme'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = FALSE,
  ...
)
Arguments

- **model**: A statistical model object.
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **include.groups**: Report the number of groups?
- **include.variance**: Report group variances?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

**extract** method for nlmerMod objects created by the nlmer function in the lme4 package.

Usage

```r
## S4 method for signature 'nlmerMod'
extract(
  model,
  method = c("naive", "profile", "boot", "Wald"),
  level = 0.95,
  nsim = 1000,
  include.aic = TRUE,
  include.bic = TRUE,
  include.dic = FALSE,
  include.deviance = FALSE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.groups = TRUE,
  include.variance = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **method**: The method used to compute confidence intervals or p-values. The default value "naive" computes naive p-values while the other methods compute confidence intervals using the confint function. See confint.merMod.
level  Significance or confidence level (1 - alpha) for computing confidence intervals.
nsim  The MCMC sample size or number of bootstrapping replications on the basis of which confidence intervals are computed (only if the method argument does not specify "naive", which is the default behavior). Note: large values may take considerable computing time.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.dic Report the deviance information criterion (DIC)?
include.deviance Report the deviance?
include.loglik Report the log likelihood in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.groups Report the number of groups?
include.variance Report group variances?
include.rsquared Arguments to be passed to the fixef function in the lme4 package.
include.iterations Report the number of iterations?

extract,oglmx-method  extract method for oglm objects

Description

extract method for oglm objects created by the oglm function in the oglmx package.

Usage

## S4 method for signature 'oglmx'
extract(
  model,
  include.aic = TRUE,
  include.iterations = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  include.rsquared = TRUE,
  ...
)

Arguments

model  A statistical model object.
include.aic  Report Akaike’s Information Criterion (AIC) in the GOF block?
include.iterations  Report the number of iterations?
include.loglik  Report the log likelihood in the GOF block?
include.nobs  Report the number of observations in the GOF block?
include.rsquared  Report R^2 in the GOF block?

...  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

---

**extract, ols-method**  
*extract method for ols objects*

**Description**

*extract method for ols objects created by the ols function in the rms package.*

**Usage**

```r
## S4 method for signature 'ols'
extract(
  model,
  include.nobs = TRUE,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.fstatistic = FALSE,
  include.lr = TRUE,
  ...
)
```

**Arguments**

- **model**  
A statistical model object.
- **include.nobs**  
Report the number of observations in the GOF block?
- **include.rsquared**  
Report R^2 in the GOF block?
- **include.adjrs**  
Report adjusted R^2 in the GOF block?
- **include.fstatistic**  
Report the F-statistic in the GOF block?
- **include.lr**  
Report likelihood ratio test?
- **...**  
Custom parameters, which are handed over to subroutines. Currently not in use.
### Description

The `extract` method for `panelAR` objects created by the `panelAR` function in the `panelAR` package.

#### Usage

```r
## S4 method for signature 'panelAR'
extract(model, include.rsquared = TRUE, include.nobs = TRUE, include.groups = TRUE, ...)
```

#### Arguments

- `model`: A statistical model object.
- `include.rsquared`: Report R^2 in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block?
- `include.groups`: Report the number of groups?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

### Description

The `extract` method for `pglm` objects created by the `pglm` function in the `pglm` package.

#### Usage

```r
## S4 method for signature 'pglm'
extract(model, include.aic = TRUE, include.loglik = TRUE, include.nobs = TRUE, ...)
```
extract,pacman-method

Arguments

model  A statistical model object.
include.aic  Report Akaike’s Information Criterion (AIC) in the GOF block?
include.loglik  Report the log likelihood in the GOF block?
include.nobs  Report the number of observations in the GOF block?
...  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

describe,pgmm-method  describe method for pgmm objects

description

describe method for pgmm objects created by the pgmm function in the plm package.

Usage

## S4 method for signature 'pgmm'
extract(
  model,
  include.nobs = TRUE,
  include.sargan = TRUE,
  include.wald = TRUE,
  ...
)

Arguments

model  A statistical model object.
include.nobs  Report the number of observations in the GOF block?
include.sargan  Report the Sargan test?
include.wald  Report the Wald statistic?
...  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
extract.phreg-method  

**extract** method for **phreg** objects

**Description**

**extract** method for **phreg** objects created by the **phreg** function in the **eha** package.

**Usage**

```r
## S4 method for signature 'phreg'
extract(
  model,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.lr = TRUE,
  include.nobs = TRUE,
  include.events = TRUE,
  include.trisk = TRUE,
  ...
)
```

**Arguments**

- `model` A statistical model object.
- `include.aic` Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.lr` Report likelihood ratio test?
- `include.nobs` Report the number of observations in the GOF block?
- `include.events` Report the number of events in the GOF block?
- `include.trisk` Report the total time at risk (in event-history models)?
- `...` Custom parameters, which are handed over to subroutines. Currently not in use.

extract.plm-method  

**extract** method for **plm** objects

**Description**

**extract** method for **plm** objects created by the **plm** function in the **plm** package.
### S4 method for signature 'plm'

```r
extract(
    model,
    include.rsquared = TRUE,
    include.adjrs = TRUE,
    include.nobs = TRUE,
    include.variance = TRUE,
    ...
)
```

#### Arguments

- **model**: A statistical model object.
- **include.rsquared**: Report R^2 in the GOF block?
- **include.adjrs**: Report adjusted R^2 in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **include.variance**: Report group variances?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

---

**extract,pmg-method**

*extract method for pmg objects*

#### Description

*extract* method for pmg objects created by the `pmg` function in the `plm` package.

#### Usage

```r
## S4 method for signature 'pmg'
extract(model, include.nobs = TRUE, ...)
```

#### Arguments

- **model**: A statistical model object.
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.
**extract.poissonirr-method**

**extract method for poissonirr objects**

---

**Description**

`extract` method for `poissonirr` objects created by the `poissonirr` function in the `mfx` package.

**Usage**

```r
## S4 method for signature 'poissonirr'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)
```

**Arguments**

- **model** A statistical model object.
- **include.nobs** Report the number of observations in the GOF block?
- **include.loglik** Report the log likelihood in the GOF block?
- **include.deviance** Report the deviance?
- **include.aic** Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic** Report the Bayesian Information Criterion (BIC) in the GOF block?
- **...** Custom parameters, which are handed over to subroutines. Currently not in use.

---

**extract.poissonmfx-method**

**extract method for poissonmfx objects**

---

**Description**

`extract` method for `poissonmfx` objects created by the `poissonmfx` function in the `mfx` package.
Usage

## S4 method for signature 'poissonmfx'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)

Arguments

model          A statistical model object.
include.nobs   Report the number of observations in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.aic    Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic    Report the Bayesian Information Criterion (BIC) in the GOF block?
...             Custom parameters, which are handed over to subroutines. Currently not in use.

extract, polr-method extract method for polr objects

Description

extract method for polr objects created by the polr function in the MASS package.

Usage

## S4 method for signature 'polr'
extract(
  model,
  include.thresholds = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)

extract.probitmfx-method

Arguments

- **model**: A statistical model object.
- **include.thresholds**: Report thresholds in the GOF block?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.deviance**: Report the deviance?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

`extract` method for `probitmfx` objects created by the `probitmfx` function in the `mfx` package.

Usage

```r
## S4 method for signature 'probitmfx'
extract(
  model,
  include.nobs = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **include.nobs**: Report the number of observations in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.deviance**: Report the deviance?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines. Currently not in use.
### Description

*extract* method for *rem.dyad* objects created by the *rem.dyad* function in the *relevent* package.

### Usage

```r
## S4 method for signature 'rem.dyad'
extract(
  model,
  include.nvertices = TRUE,
  include.events = TRUE,
  include.aic = TRUE,
  include.aicc = TRUE,
  include.bic = TRUE,
  ...
)
```

### Arguments

- **model**: A statistical model object.
- **include.nvertices**: Report the number of vertices in a STERGM?
- **include.events**: Report the number of events in the GOF block?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.aicc**: Report AICC in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines. Currently not in use.

### Description

*extract* method for *rlm* objects created by the *rlm* function in the *MASS* package.

### Usage

```r
## S4 method for signature 'rlm'
extract(model, include.nobs = TRUE, ...)
```
extract.rq-method

Arguments

- **model**: A statistical model object.
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `rq` objects created by the `rq` function in the `quantreg` package.

Usage

```r
## S4 method for signature 'rq'
extract(model, include.nobs = TRUE, include.percentile = TRUE, ...)
```

Arguments

- **model**: A statistical model object.
- **include.nobs**: Report the number of observations in the GOF block?
- **include.percentile**: Report the percentile (tau)?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

extract,sarlm-method

extract method for `sarlm` objects

Description

`extract` method for `sarlm` objects created by the `lagsarlm` function in the `spatialreg` package.

Usage

```r
## S4 method for signature 'sarlm'
exact(r
model,
include.nobs = TRUE,
include.loglik = TRUE,
include.aic = TRUE,
include.lr = TRUE,
include.wald = TRUE,
...)
```
Arguments

- **model**: A statistical model object.
- **include.nobs**: Report the number of observations in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.lr**: Report likelihood ratio test?
- **include.wald**: Report the Wald statistic?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

Description

`extract` method for `sclm` objects created by the `clm` function in the `ordinal` package.

Usage

```r
## S4 method for signature 'sclm'
extract(
  model,
  include.thresholds = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.nobs = TRUE,
  ...
)
```

Arguments

- **model**: A statistical model object.
- **include.thresholds**: Report thresholds in the GOF block?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.
**extract,selection-method**  
*extract method for selection objects*

**Description**

*extract* method for selection objects created by the *selection* function in the *sampleSelection* package.

**Usage**

```r
## S4 method for signature 'selection'
extract(
  model,
  prefix = TRUE,
  include.selection = TRUE,
  include.outcome = TRUE,
  include.errors = TRUE,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  ...
)
```

**Arguments**

- `model`: A statistical model object.
- `prefix`: Include prefix before the label of the coefficient in order to identify the current model component?
- `include.selection`: Report the selection component of a sample selection model?
- `include.outcome`: Report the outcome component of a sample selection model?
- `include.errors`: Report the error terms of a sample selection model?
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic`: Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.rsquared`: Report $R^2$ in the GOF block?
- `include.adjrs`: Report adjusted $R^2$ in the GOF block?
- `include.nobs`: Report the number of observations in the GOF block? block?
... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

---

**extract,sienaFit-method**

*extract method for sienaFit objects*

**Description**

*extract* method for *sienaFit* objects created by the *siena07* function in the *RSiena* package.

**Usage**

```r
## S4 method for signature 'sienaFit'
extract(model, include.iterations = TRUE, ...)
```

**Arguments**

- **model** A statistical model object.
- **include.iterations** Report the number of iterations?
- ... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

---

**extract,simex-method**

*extract method for simex objects*

**Description**

*extract* method for *simex* objects created by the *simex* function in the *simex* package.

**Usage**

```r
## S4 method for signature 'simex'
extract(model, jackknife = TRUE, include.nobs = TRUE, ...)
```

**Arguments**

- **model** A statistical model object.
- **jackknife** Use Jackknife variance instead of asymptotic variance?
- **include.nobs** Report the number of observations in the GOF block?
- ... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
extract.speedglm-method

**extract method for speedglm objects**

Description

`extract` method for `speedglm` objects created by the `speedglm` function in the `speedglm` package.

Usage

```r
## S4 method for signature 'speedglm'
extract(model,
         include.aic = TRUE,
         include.bic = TRUE,
         include.loglik = TRUE,
         include.deviance = TRUE,
         include.nobs = TRUE,
         ...
)
```

Arguments

- `model` A statistical model object.
- `include.aic` Report Akaike's Information Criterion (AIC) in the GOF block?
- `include.bic` Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik` Report the log likelihood in the GOF block?
- `include.deviance` Report the deviance?
- `include.nobs` Report the number of observations in the GOF block?
- `...` Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

extract.speedlm-method

**extract method for speedlm objects**

Description

`extract` method for `speedlm` objects created by the `speedlm` function in the `speedglm` package.
extract, stergm-method

### S4 method for signature 'speedlm'

```r
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  include.fstatistic = FALSE,
  include.rmse = FALSE,
  ...
)
```

#### Arguments

- **model**: A statistical model object.
- **include.rsquared**: Report $R^2$ in the GOF block?
- **include.adjrs**: Report adjusted $R^2$ in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **include.fstatistic**: Report the F-statistic in the GOF block?
- **include.rmse**: Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

---

**Description**

`extract` method for `stergm` objects created by the `stergm` function in the `tergm` package.

### Usage

```r
## S4 method for signature 'stergm'
extract(
  model,
  beside = FALSE,
  include.formation = TRUE,
  include.dissolution = TRUE,
  include.nvertices = TRUE,
  include.aic = FALSE,
  include.bic = FALSE,
  include.loglik = FALSE,
  ...
)
```
extract.summary.lm-method

**Arguments**

- `model`: A statistical model object.
- `beside`: Arrange the model terms below each other or beside each other? In a `stergm` model, the formation and dissolution coefficients can be arranged in two columns of the table.
- `include.formation`: Report the coefficients for the formation process in a `STERGM`?
- `include.dissolution`: Report the coefficients for the dissolution process in a `STERGM`?
- `include.nvertices`: Report the number of vertices in a `STERGM`?
- `include.aic`: Report Akaike’s Information Criterion (AIC) in the GOF block?
- `include.bic`: Report the Bayesian Information Criterion (BIC) in the GOF block?
- `include.loglik`: Report the log likelihood in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

**Description**

*extract method for summary.lm objects*

*extract* method for `summary.lm` objects created by the `summary` method for `lm` objects, defined in the `stats` package (see `summary.lm`).

**Usage**

```r
## S4 method for signature 'summary.lm'
extract(module, model,
     include.rsquared = TRUE,
     include.adjrs = TRUE,
     include.nobs = TRUE,
     include.fstatistic = FALSE,
     include.rmse = TRUE,
     ...
)
```

**Arguments**

- `model`: A statistical model object.
- `include.rsquared`: Report $R^2$ in the GOF block?
include.adjrs Report adjusted R^2 in the GOF block?
include.nobs Report the number of observations in the GOF block?
include.fstatistic Report the F-statistic in the GOF block?
include.rmse Report the root mean square error (RMSE; = residual standard deviation) in the GOF block?

... Custom parameters, which are handed over to subroutines. Currently not in use.

description

extract method for survreg objects

## S4 method for signature 'survreg'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike’s Information Criterion (AIC) in the GOF block?
include.bic Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.deviance Report the deviance?
include.nobs Report the number of observations in the GOF block?

... Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
### Description

**extract** method for `survreg.penal` objects created by the `survreg` function in the `survival` package.

### Usage

```r
## S4 method for signature 'survreg.penal'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  ...
)
```

### Arguments

- **model** : A statistical model object.
- **include.aic** : Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic** : Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik** : Report the log likelihood in the GOF block?
- **include.deviance** : Report the deviance?
- **include.nobs** : Report the number of observations in the GOF block?
- **...** : Custom parameters, which are handed over to subroutines, in this case to the `summary` method for the object.

### Description

**extract** method for `svyglm` objects created by the `svyglm` function in the `survey` package.
Usage

## S4 method for signature 'svyglm'
extract(
  model,
  include.aic = FALSE,
  include.bic = FALSE,
  include.loglik = FALSE,
  include.deviance = TRUE,
  include.dispersion = TRUE,
  include.nobs = TRUE,
  ...
)

Arguments

model  A statistical model object.
include.aic  Report Akaike's Information Criterion (AIC) in the GOF block?
include.bic  Report the Bayesian Information Criterion (BIC) in the GOF block?
include.loglik  Report the log likelihood in the GOF block?
include.deviance  Report the deviance?
include.dispersion  Report the dispersion parameter?
include.nobs  Report the number of observations in the GOF block?
...  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

Description

extract method for systemfit objects

Usage

## S4 method for signature 'systemfit'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  beside = FALSE,
include.suffix = FALSE,
...
)

Arguments

model A statistical model object.
include.rsquared Report R^2 in the GOF block?
include.adjrs Report adjusted R^2 in the GOF block?
include.nobs Report the number of observations in the GOF block?
beside Arrange the model terms below each other or beside each other, in separate columns?
include.suffix Report the name of the current model in parentheses after each model term (instead of before the model term)?

Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.

extract method for texreg objects

Description

extract method for texreg objects created by the extract function in the texreg package.

Usage

## S4 method for signature 'texreg'
extract(model, ...)

Arguments

model A statistical model object.
...

Custom parameters, which are handed over to subroutines. Currently not in use.
**extract.tobit-method**  
*extract method for tobit objects*

**Description**

`extract` method for `tobit` objects created by the `tobit` function in the **AER** package.

**Usage**

```r
## S4 method for signature 'tobit'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = FALSE,
  include.censnobs = TRUE,
  include.wald = TRUE,
  ...
)
```

**Arguments**

- `model`  
  A statistical model object.

- `include.aic`  
  Report Akaike’s Information Criterion (AIC) in the GOF block?

- `include.bic`  
  Report the Bayesian Information Criterion (BIC) in the GOF block?

- `include.loglik`  
  Report the log likelihood in the GOF block?

- `include.deviance`  
  Report the deviance?

- `include.nobs`  
  Report the number of observations in the GOF block?

- `include.censnobs`  
  Report the total, right-censored, left-censored, and uncensored number of observations?

- `include.wald`  
  Report the Wald statistic?

- `...`  
  Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
**Description**

`extract` method for `vglm` objects created by the `vglm` function in the **VGAM** package.

**Usage**

```r
## S4 method for signature 'vglm'
extract(
  model,
  include.loglik = TRUE,
  include.df = TRUE,
  include.nobs = TRUE,
  ...
)
```

**Arguments**

- `model`: A statistical model object.
- `include.loglik`: Report the log likelihood in the GOF block?
- `include.df`: Report the degrees of freedom?
- `include.nobs`: Report the number of observations in the GOF block?
- `...`: Custom parameters, which are handed over to subroutines. Currently not in use.

**Author(s)**

Christoph Riedl <c.riedl@neu.edu>

---

**Description**

`extract` method for `weibreg` objects created by the `weibreg` function in the **eha** package.
extract,wls-method

Usage

## S4 method for signature 'weibreg'
extract(
  model,
  include.aic = TRUE,
  include.loglik = TRUE,
  include.lr = TRUE,
  include.nobs = TRUE,
  include.events = TRUE,
  include.trisk = TRUE,
  ...
)

Arguments

model A statistical model object.
include.aic Report Akaike's Information Criterion (AIC) in the GOF block?
include.loglik Report the log likelihood in the GOF block?
include.lr Report likelihood ratio test?
include.nobs Report the number of observations in the GOF block?
include.events Report the number of events in the GOF block?
include.trisk Report the total time at risk (in event-history models)?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Description

extract method for wls objects created by the wls function in the metaSEM package.

Usage

## S4 method for signature 'wls'
extract(model, include.nobs = TRUE, ...)

Arguments

model A statistical model object.
include.nobs Report the number of observations in the GOF block?
... Custom parameters, which are handed over to subroutines. Currently not in use.

Author(s)

Christoph Riedl <c.riedl@neu.edu>
Description

extract method for zelig objects created by the **zelig** function in the **Zelig** package (version < 5.0).

extract method for Zelig objects created by the **zelig** function in the **Zelig** package (version >= 5.0).

Usage

```r
## S4 method for signature 'zelig'
extract(
  model,
  include.aic = TRUE,
  include.bic = TRUE,
  include.loglik = TRUE,
  include.deviance = TRUE,
  include.nobs = TRUE,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.fstatistic = TRUE,
  ...
)

## S4 method for signature 'Zelig'
extract(model, include.nobs = TRUE, include.nimp = TRUE, ...)
```

Arguments

- **model**: A statistical model object.
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.bic**: Report the Bayesian Information Criterion (BIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.deviance**: Report the deviance?
- **include.nobs**: Report the number of observations in the GOF block?
- **include.rsquared**: Report \( R^2 \) in the GOF block?
- **include.adjrs**: Report adjusted \( R^2 \) in the GOF block?
- **include.fstatistic**: Report the F-statistic in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines. Currently not in use.
- **include.nimp**: Report the number of multiple imputations (in Zelig models with imputed data)?
extract,zeroinfl-method

extract method for zeroinfl objects

Description

extract method for zeroinfl objects created by the zeroinfl function in the pscl package.

Usage

## S4 method for signature 'zeroinfl'
extract(
    model,
    beside = FALSE,
    include.count = TRUE,
    include.zero = TRUE,
    include.aic = TRUE,
    include.loglik = TRUE,
    include.nobs = TRUE,
    ...
)

Arguments

- **model**: A statistical model object.
- **beside**: Arrange the model terms below each other or beside each other? The binary model parameters and the count parameters can be displayed in two separate columns of the table.
- **include.count**: Report the count parameters in the coefficients block (before the binary part for the zeros)?
- **include.zero**: Should the binary part of the model be included in the coefficients block (after the count parameters)?
- **include.aic**: Report Akaike’s Information Criterion (AIC) in the GOF block?
- **include.loglik**: Report the log likelihood in the GOF block?
- **include.nobs**: Report the number of observations in the GOF block?
- **...**: Custom parameters, which are handed over to subroutines, in this case to the summary method for the object.
htmlreg

Convert regression output to a HTML table

Description

Conversion of R regression output to a HTML table.

Usage

```r
htmlreg(
  l,
  file = NULL,
  single.row = FALSE,
  stars = c(0.001, 0.01, 0.05),
  custom.header = NULL,
  custom.model.names = NULL,
  custom.coef.names = NULL,
  custom.coef.map = NULL,
  custom.gof.names = NULL,
  custom.gof.rows = NULL,
  custom.note = NULL,
  digits = 2,
  leading.zero = TRUE,
  star.symbol = "\star",
  symbol = "\cdot",
  override.coef = 0,
  override.se = 0,
  override.pvalues = 0,
  override.ci.low = 0,
  override.ci.up = 0,
  omit.coef = NULL,
  reorder.coef = NULL,
  reorder.gof = NULL,
  ci.force = FALSE,
  ci.force.level = 0.95,
  ci.test = 0,
  groups = NULL,
  custom.columns = NULL,
  custom.col.pos = NULL,
  bold = 0,
  center = TRUE,
  caption = "Statistical models",
  caption.above = FALSE,
  inline.css = TRUE,
  doctype = FALSE,
  html.tag = FALSE,
  head.tag = FALSE,
)```
Arguments

1

A statistical model or a list of statistical models. Lists of models can be specified as `l = list(model.1, model.2, ...)`. Different object types can also be mixed.

file

Using this argument, the resulting table is written to a file rather than to the R prompt. The file name can be specified as a character string. Writing a table to a file can be useful for working with MS Office or LibreOffice. For example, using the `htmlreg` function, an HTML table can be written to a file with the extension `.doc` and opened with MS Word. The table can then be simply copied into any Word document, retaining the formatting of the table. Note that LibreOffice can import only plain HTML; CSS decorations are not supported; the resulting tables do not retain the full formatting in LibreOffice.

single.row

By default, a model parameter takes up two lines of the table: the standard error is listed in parentheses under the coefficient. This saves a lot of horizontal space on the page and is the default table format in most academic journals. If `single.row = TRUE` is activated, however, both coefficient and standard error are placed in a single table cell in the same line.

stars

The significance levels to be used to draw stars. Between 0 and 4 threshold values can be provided as a numeric vector. For example, `stars = numeric(0)` will not print any stars and will not print any note about significance levels below the table. `stars = 0.05` will attach one single star to all coefficients where the p-value is below 0.05. `stars = c(0.001, 0.01, 0.05, 0.1)` will print one, two, or three stars, or a symbol as specified by the `symbol` argument depending on the p-values.

custom.header

An optional named list of multi-column headers that are placed above the model names. For example, `custom.header = list("abc" = 1:3, "ef" = 4:5)` will add the label "abc" to the first three models and "ef" to the fourth and fifth model. The column with coefficient names and any custom columns added by the "custom.columns" argument are not counted towards these positions. If `booktabs = TRUE`, `\cmidrule` rules are added below the respective labels; otherwise `\cline` lines are used.

custom.model.names

A character vector of labels for the models. By default, the models are named "Model 1", "Model 2", etc. Specifying `model.names = c("My name 1", "My name 2")` etc. overrides the default behavior.

custom.coef.names

By default, `texreg` uses the coefficient names which are stored in the models. The `custom.coef.names` argument can be used to replace them by other
character strings in the order of appearance. For example, if a table shows a total of three different coefficients (including the intercept), the argument `custom.coef.names = c("intercept","variable 1","variable 2")` will replace their names in this order.

Sometimes it happens that the same variable has a different name in different models. In this case, the user can use this function to assign identical names. If possible, the rows will then be merged into a single row unless both rows contain values in the same column.

Where the argument contains an `NA` value, the original name of the coefficient is kept. For example, `custom.coef.names = c(NA,"age",NA)` will only replace the second coefficient name and leave the first and third name as they are in the original model.

See also `custom.coef.map` for an easier and more comprehensive way to rename, omit, and reorder coefficients.

`custom.coef.map`  
The `custom.coef.map` argument can be used to select, omit, rename, and reorder coefficients.

Users must supply a named list of this form: `list("x" = "First variable", "y" = NA,"z" = "Third variable")`. With that particular example of `custom.coef.map`,

1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as "y", and variable "z" will appear as "Third variable".
3. all variables not named "x", "y", or "z" will be omitted from the table.

`custom.gof.names`  
A character vector which is used to replace the names of the goodness-of-fit statistics at the bottom of the table. The vector must have the same length as the number of GOF statistics in the final table. The argument works like the `custom.coef.names` argument, but for the GOF values. `NA` values can be included where the original GOF name should be kept.

`custom.gof.rows`  
A named list of vectors for new lines at the beginning of the GOF block of the table. For example, `list("Random effects" = c("YES","YES","NO"), Observations = c(25,25,26))` would insert two new rows into the table, at the beginning of the GOF block (i.e., after the coefficients). The rows can contain integer, numeric, or character objects. Note that this argument is processed after the `custom.gof.names` argument (meaning `custom.gof.names` should not include any of the new GOF rows) and before the `reorder.gof` argument (meaning that the new GOF order specified there should contain values for the new custom GOF rows). Arguments for custom columns are not affected because they only insert columns into the coefficient block.

`custom.note`  
With this argument, a replacement text for the significance note below the table can be provided. If an empty character object is provided (`custom.note = ""`), the note will be omitted completely. If some character string is provided (e.g., `custom.note = "My note"`), the significance legend is replaced by `My note`. The original significance legend can be included by inserting the `%stars` wildcard. For example, a custom note can be added right after the significance legend by providing `custom.note = "%stars. My note."`.
If the \texttt{threeparttable} argument is used, any note should be preceded by "\item", for example "\item %stars. \item Second note. \item Third note.", and it is possible to create line breaks in the formatted table by including "\\" and line breaks in the LaTeX code by including "\n", for example "\n\item %stars.\n\item Second line.\n".

\textbf{digits} \hspace{1cm} Set the number of decimal places for coefficients, standard errors and goodness-of-fit statistics. Do not use negative values! The argument works like the \texttt{digits} argument in the \texttt{round} function of the \texttt{base} package.

\textbf{leading.zero} \hspace{1cm} Most journals require leading zeros of coefficients and standard errors (for example, 0.35). This is also the default \texreg behavior. Some journals, however, require omission of leading zeros (for example, .35). This can be achieved by setting \texttt{leading.zero = FALSE}.

\textbf{star.symbol} \hspace{1cm} Alternative characters for the significance stars can be specified. This is useful if \texttt{knitr} and Markdown are used for HTML report generation. In Markdown, asterisks or stars are interpreted as special characters, so they have to be escaped. To make a HTML table compatible with Markdown, specify \texttt{star.symbol = "\*"}. Note that some other modifications are recommended for usage with \texttt{knitr} in combination with Markdown or HTML (see the \texttt{inline.css}, \texttt{doctype}, \texttt{html.tag}, \texttt{head.tag}, and \texttt{body.tag} arguments in the \texttt{htmlreg} function).

\textbf{symbol} \hspace{1cm} If four threshold values are handed over to the \texttt{stars} argument, \texttt{p-values} smaller than the largest threshold value but larger than the second-largest threshold value are denoted by this symbol. The default symbol is "\texttt{\textbullet}" for the LaTeX dot, "\texttt{&middot;}" for the HTML dot, or simply "." for the ASCII dot. If the \texttt{texreg} function is used, any other mathematical LaTeX symbol or plain text symbol can be used, for example \texttt{symbol = "\textcircled{c}" for a small circle} (note that backslashes must be escaped). If the \texttt{htmlreg} function is used, any other HTML character or symbol can be used. For the \texttt{screenreg} function, only plain text characters can be used.

\textbf{override.coef} \hspace{1cm} Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as \texttt{override.coef = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))}. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: \texttt{override.coef = c(0.05,0.06,0.07)}.

\textbf{override.se} \hspace{1cm} Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as \texttt{override.se = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))}. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: \texttt{override.se = c(0.05,0.06,0.07)}. Overriding standard errors can be useful for the implementation of robust SEs, for example.

\textbf{override.pvalues} \hspace{1cm} Set custom values for the \texttt{p-values}. New \texttt{p-values} are provided as a list of numeric vectors. The list contains vectors of \texttt{p-values} for each model. There must
be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.pvalues = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.pvalues = c(0.05,0.06,0.07)`. Overriding p-values can be useful for the implementation of robust SEs and p-values, for example.

`override.ci.low`  
Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.up` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

`override.ci.up`  
Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.low` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

`omit.coef`  
A character string which is used as a regular expression to remove coefficient rows from the table. For example, `omit.coef = "group"` deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example `omit.coef = "(thresh)|(ranef)"` to remove all model terms matching either "thresh" or "ranef". The `omit.coef` argument is processed after the `custom.coef.names` argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the extract functions instead (see the help entry of the `extract` function.

`reorder.coef`  
Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, `reorder.coef = c(3,2,1)` will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and after omitting coefficients, so the `custom.coef.names` and `omit.coef` arguments should follow the original order.

`reorder.gof`  
Reorder the rows of the goodness-of-fit block of the resulting table in a custom way. The argument takes a vector of the same length as the number of GOF statistics. For example, if there are three goodness-of-fit rows, `reorder.gof = c(3,2,1)` will exchange the first and the third row. Note: Reordering takes place after processing custom GOF names and after adding new custom GOF rows, so the `custom.gof.names` and `custom.gof.rows` arguments should follow the original order, and the `reorder.gof` argument should contain values for any rows that are added through the `custom.gof.rows` argument.

`ci.force`  
Should confidence intervals be used instead of the default standard errors and p-values? Most models implemented in the `texreg` package report standard errors and p-values by default while few models report confidence intervals. However, the functions in the `texreg` package can convert standard errors and into
confidence intervals using z-scores if desired. To enforce confidence intervals
instead of standard errors, the ci.force argument accepts either a logical value
indicating whether all models or none of the models should be forced to re-
port confidence intervals (ci.force = TRUE for all and ci.force = FALSE for
none) or a vector of logical values indicating for each model separately whether
the model should be forced to report confidence intervals (e.g., ci.force =
c(FALSE, TRUE, FALSE)). Confidence intervals are computed using the standard
normal distribution (z-values based on the qnorm function). The t-distribution
is currently not supported because this would require each extract method to
have an additional argument for the degrees of freedom.

**ci.force.level**
If the ci.force argument is used to convert standard errors to confidence inter-
vals, what confidence level should be used? By default, 0.95 is used (i.e., an
alpha value of 0.05).

**ci.test**
If confidence intervals are reported, the ci.test argument specifies the refer-
ence value to establish whether a coefficient/CI is significant. The default value
ci.test = 0, for example, will attach a significance star to coefficients if the
confidence interval does not contain 0. A value of ci.test = 1 could be use-
ful if coefficients are provided on the odds-ratio scale, for example. If no star
should be printed at all, ci.test = NA can be used. It is possible to provide a
single value for all models or a vector with a separate value for each model. The
ci.test argument works both for models with native support for confidence
intervals and in cases where the ci.force argument is used.

**groups**
This argument can be used to group the rows of the table into blocks. For ex-
ample, there could be one block for hypotheses and another block for control
variables. Each group has a heading, and the row labels within a group are in-
dented. The partitions must be handed over as a list of named numeric vectors,
where each number is a row index and each name is the heading of the group.
Example: groups = list("first group" = 1:4,"second group" = 7:8).

**custom.columns**
An optional list of additional text columns to be inserted into the coefficient
block of the table, for example coefficient types. The list should contain one
or more character vectors with as many character or numeric elements as there
are coefficients/model terms. If the vectors in the list are named, the names are
used as labels in the table header. For example, custom.columns = list(type
= c("a","b","c"),1:3) will add two columns; the first one is labeled while
the second one is not. Note that the numeric elements of the second column will
be converted to character objects in this example. The consequence is that
decimal alignment with the dcolumn package is switched off in these columns.
Note that this argument is processed after any arguments that affect the number
of rows.

**custom.col.pos**
An optional integer vector of positions for the columns given in the custom.columns
argument. For example, if there are three custom columns, custom.col.pos =
c(1,3,3) will insert the first custom column before the first column of the origi-
nal table and the remaining two custom columns after the second column of the
original table. By default, all custom columns are placed after the first column,
which usually contains the coefficient names.

**bold**
The p-value threshold below which the coefficient shall be formatted in a bold
font. For example, bold = 0.05 will cause all coefficients that are significant
at the 95% level to be formatted in bold. Note that this is not compatible with
the dcolumn or siunitx arguments in the texreg function. If both bold and
dcolumn or siunitx are TRUE, dcolumn and siunitx are switched off, and a
warning message appears. Note also that it is advisable to use stars = FALSE
together with the bold argument because having both bolded coefficients and
significance stars usually does not make any sense.

center Should the table be horizontally aligned at the center of the page?
caption Set the caption of the table.
caption.above Should the caption of the table be placed above the table? By default, it is placed
below the table.
inline.css Should the CSS stylesheets be embedded directly in the code of the table (inline.css
= TRUE), or should the CSS stylesheets be enclosed in the <head> tag, that is,
separated from the table code (inline.css = FALSE)? Having inline CSS code
makes the code of the table more complex, but sometimes it may be helpful
when only the table shall be printed, without the head of the HTML file (for
example when the table is embedded in a knitr report). As a rule of thumb: use
inline CSS if the table is not saved to a file.
doctype Should the first line of the HTML code contain the DOCTYPE definition? If
TRUE, the HTML 4 TRANSITIONAL version is used. If FALSE, no DOCTYPE
will be included. Omitting the DOCTYPE can be helpful when the knitr pack-
age is used to generate HTML code because knitr requires only the plain ta-
ble, not the whole HTML document including the document type declaration.
Including the DOCTYPE can be helpful when the code is saved to a file, for
example as an MS Word document.
html.tag Should the table code (and possibly the <body> and <head> tags) be enclosed
in an <html> tag? Suppressing this tag is recommended when knitr is used for
dynamic HTML or Markdown report generation. Including this tag is recom-
ended when the code is saved to a file, for example as an MS Word document.
head.tag Should the <head> tag (including CSS definitions and title/caption) be included
in the HTML code? Suppressing this tag is recommended when knitr is used
for dynamic HTML or Markdown report generation. Including this tag is recom-
ended when the code is saved to a file, for example as an MS Word document.
body.tag Should the table code be enclosed in a <body> HTML tag? Suppressing this tag
is recommended when knitr is used for dynamic HTML or Markdown report
generation. Including this tag is recommended when the code is saved to a file,
for example as an MS Word document.
indentation Characters used for indentation of the HTML code. By default, indentation =
" " uses no indentation. Any number of spaces or characters can be used instead.
For example, indentation = " " uses two spaces of (additional) indentation for
each subelement.
margin The margin around the table in pixels. This determines how much space there is
around the table. To remove all space around the table, set table.margin = 0.
padding The space on the left and right of each table cell in pixels.
color The color of the table, including text and rules or lines. This can be provided as
a hex RGB value or as a color string that is valid in HTML (e.g., "black").
outer.rules  The line width at the top and bottom of the table in pixels. Can be outer.rules = 0 to omit outer lines.

inner.rules  The horizontal line width before and after the coefficient block of the table in pixels. Can be outer.rules = 0 to omit inner lines.

...  Custom options to be passed on to the extract function. For example, most extract methods provide custom options for the inclusion or exclusion of specific goodness-of-fit statistics. See the help entries of extract for more information.

Details

The htmlreg function creates HTML code. Tables in HTML format can be saved with a "html" extension and displayed in a web browser. Alternatively, they can be saved with a "doc" extension and opened in MS Word for inclusion in office documents. htmlreg also works with knitr and HTML or Markdown. Note that the inline.css, doctype, html.tag, head.tag, body.tag, and star.symbol arguments must be adjusted for the different purposes (see the description of the arguments).

Author(s)

Philip Leifeld

References


See Also

texreg-package extract

Other texreg: huxtablereg(), knitreg(), matrixreg(), plotreg(), screenreg(), texreg, wordreg()

Examples

library("nlme")
model.1 <- lme(distance ~ age, data = Orthodont, random = ~ 1)
model.2 <- lme(distance ~ age + Sex, data = Orthodont, random = ~ 1)
htmlreg(list(model.1, model.2),
        file = "texreg.doc",
        inline.css = FALSE,
        doctype = TRUE,
        html.tag = TRUE,
        head.tag = TRUE,
        body.tag = TRUE)
unlink("texreg.doc")
Create a huxtable object from multiple statistical models.

Usage

huxtablereg(
  l,
  single.row = FALSE,
  stars = c(0.001, 0.01, 0.05),
  custom.model.names = NULL,
  custom.coef.names = NULL,
  custom.coef.map = NULL,
  custom.gof.names = NULL,
  custom.gof.rows = NULL,
  digits = 2,
  leading.zero = TRUE,
  star.symbol = star.symbol,
  symbol = "*",
  override.coef = 0,
  override.se = 0,
  override.pvalues = 0,
  override.ci.low = 0,
  override.ci.up = 0,
  omit.coef = NULL,
  reorder.coef = NULL,
  reorder.gof = NULL,
  ci.force = FALSE,
  ci.force.level = 0.95,
  ci.test = 0,
  groups = NULL,
  custom.columns = NULL,
  custom.col.pos = NULL,
  ...
)

Arguments

l A statistical model or a list of statistical models. Lists of models can be specified as l = list(model.1, model.2, ...). Different object types can also be mixed.

single.row By default, a model parameter takes up two lines of the table: the standard error is listed in parentheses under the coefficient. This saves a lot of horizontal space on the page and is the default table format in most academic journals. If
single.row = TRUE is activated, however, both coefficient and standard error are placed in a single table cell in the same line.

stars
The significance levels to be used to draw stars. Between 0 and 4 threshold values can be provided as a numeric vector. For example, stars = numeric(0) will not print any stars and will not print any note about significance levels below the table. stars = 0.05 will attach one single star to all coefficients where the p-value is below 0.05. stars = c(0.001, 0.01, 0.05, 0.1) will print one, two, or three stars, or a symbol as specified by the symbol argument depending on the p-values.

custom.model.names
A character vector of labels for the models. By default, the models are named "Model 1", "Model 2", etc. Specifying model.names = c("My name 1", "My name 2") etc. overrides the default behavior.

custom.coef.names
By default, texreg uses the coefficient names which are stored in the models. The custom.coef.names argument can be used to replace them by other character strings in the order of appearance. For example, if a table shows a total of three different coefficients (including the intercept), the argument custom.coef.names = c("Intercept", "variable 1", "variable 2") will replace their names in this order.

Sometimes it happens that the same variable has a different name in different models. In this case, the user can use this function to assign identical names. If possible, the rows will then be merged into a single row unless both rows contain values in the same column.

Where the argument contains an NA value, the original name of the coefficient is kept. For example, custom.coef.names = c(NA, "age", NA) will only replace the second coefficient name and leave the first and third name as they are in the original model.

See also custom.coef.map for an easier and more comprehensive way to rename, omit, and reorder coefficients.

custom.coef.map
The custom.coef.map argument can be used to select, omit, rename, and reorder coefficients.

Users must supply a named list of this form: list("x" = "First variable", "y" = NA, "z" = "Third variable"). With that particular example of custom.coef.map,

1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as "y", and variable "z" will appear as "Third variable".
3. all variables not named "x", "y", or "z" will be omitted from the table.

custom.gof.names
A character vector which is used to replace the names of the goodness-of-fit statistics at the bottom of the table. The vector must have the same length as the number of GOF statistics in the final table. The argument works like the custom.coef.names argument, but for the GOF values. NA values can be included where the original GOF name should be kept.
custom.gof.rows

A named list of vectors for new lines at the beginning of the GOF block of the table. For example, `list("Random effects" = c("YES","YES","NO"),Observations = c(25,25,26))` would insert two new rows into the table, at the beginning of the GOF block (i.e., after the coefficients). The rows can contain integer, numeric, or character objects. Note that this argument is processed after the `custom.gof.names` argument (meaning `custom.gof.names` should not include any of the new GOF rows) and before the `reorder.gof` argument (meaning that the new GOF order specified there should contain values for the new custom GOF rows). Arguments for custom columns are not affected because they only insert columns into the coefficient block.

digits

Set the number of decimal places for coefficients, standard errors and goodness-of-fit statistics. Do not use negative values! The argument works like the `digits` argument in the `round` function of the `base` package.

leading.zero

Most journals require leading zeros of coefficients and standard errors (for example, 0.35). This is also the default texreg behavior. Some journals, however, require omission of leading zeros (for example, .35). This can be achieved by setting `leading.zero = FALSE`.

star.symbol

Alternative characters for the significance stars can be specified. This is useful if `knitr` and Markdown are used for HTML report generation. In Markdown, asterisks or stars are interpreted as special characters, so they have to be escaped. To make a HTML table compatible with Markdown, specify `star.symbol = "\*"`. Note that some other modifications are recommended for usage with `knitr` in combination with Markdown or HTML (see the `inline.css`, `doctype`, `html.tag`, `head.tag`, and `body.tag` arguments in the `htmlreg` function).

symbol

If four threshold values are handed over to the `stars` argument, p-values smaller than the largest threshold value but larger than the second-largest threshold value are denoted by this symbol. The default symbol is "\cdot" for the LaTeX dot, "&middot;" for the HTML dot, or simply "." for the ASCII dot. If the `texreg` function is used, any other mathematical LaTeX symbol or plain text symbol can be used, for example `symbol = "\circ"` for a small circle (note that backslashes must be escaped). If the `htmlreg` function is used, any other HTML character or symbol can be used. For the `screenreg` function, only plain text characters can be used.

override.coef

Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as `override.coef = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.coef = c(0.05,0.06,0.07).

override.se

Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.se = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))`. If there is only one model, custom values can be provided as a plain vector (not
override.se
Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many lists of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as override.se = list(c(0.05, 0.06, 0.07), c(0.05, 0.06, 0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.se = c(0.05, 0.06, 0.07). Overriding standard errors can be useful for the implementation of robust SEs.

override.pvalues
Set custom values for the p-values. New p-values are provided as a list of numeric vectors. The list contains vectors of p-values for each model. There must be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as override.pvalues = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.pvalues = c(0.05, 0.06, 0.07). Overriding p-values can be useful for the implementation of robust SEs and p-values.

override.ci.low
Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the override.ci.up argument, the standard errors and p-values as well as the ci.force argument are ignored.

override.ci.up
Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the override.ci.low argument, the standard errors and p-values as well as the ci.force argument are ignored.

omit.coef
A character string which is used as a regular expression to remove coefficient rows from the table. For example, omit.coef = "group" deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example omit.coef = "(thresh)|(ranef)" to remove all model terms matching either "thresh" or "ranef". The omit.coef argument is processed after the custom.coef.names argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the extract functions instead (see the help entry of the extract function).

reorder.coef
Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, reorder.coef = c(3, 2, 1) will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and after omitting coefficients, so the custom.coef.names and omit.coef arguments should follow the original order.

reorder.gof
Reorder the rows of the goodness-of-fit block of the resulting table in a custom way. The argument takes a vector of the same length as the number of GOF statistics. For example, if there are three goodness-of-fit rows, reorder.gof = c(3, 2, 1) will exchange the first and the third row. Note: Reordering takes place after processing custom GOF names and after adding new custom GOF rows, so the custom.gof.names and custom.gof.rows arguments should follow the
original order, and the reorder.gof argument should contain values for any rows that are added through the custom.gof.rows argument.

ci.force Should confidence intervals be used instead of the default standard errors and p-values? Most models implemented in the texreg package report standard errors and p-values by default while few models report confidence intervals. However, the functions in the texreg package can convert standard errors and into confidence intervals using z-scores if desired. To enforce confidence intervals instead of standard errors, the ci.force argument accepts either a logical value indicating whether all models or none of the models should be forced to report confidence intervals (ci.force = TRUE for all and ci.force = FALSE for none) or a vector of logical values indicating for each model separately whether the model should be forced to report confidence intervals (e.g., ci.force = c(FALSE, TRUE, FALSE)). Confidence intervals are computed using the standard normal distribution (z-values based on the qnorm function). The t-distribution is currently not supported because this would require each extract method to have an additional argument for the degrees of freedom.

ci.force.level If the ci.force argument is used to convert standard errors to confidence intervals, what confidence level should be used? By default, 0.95 is used (i.e., an alpha value of 0.05).

ci.test If confidence intervals are reported, the ci.test argument specifies the reference value to establish whether a coefficient/CI is significant. The default value ci.test = 0, for example, will attach a significance star to coefficients if the confidence interval does not contain 0. A value of ci.test = 1 could be useful if coefficients are provided on the odds-ratio scale, for example. If no star should be printed at all, ci.test = NA can be used. It is possible to provide a single value for all models or a vector with a separate value for each model. The ci.test argument works both for models with native support for confidence intervals and in cases where the ci.force argument is used.

groups This argument can be used to group the rows of the table into blocks. For example, there could be one block for hypotheses and another block for control variables. Each group has a heading, and the row labels within a group are indented. The partitions must be handed over as a list of named numeric vectors, where each number is a row index and each name is the heading of the group. Example: groups = list("first group" = 1:4,"second group" = 7:8).

custom.columns An optional list of additional text columns to be inserted into the coefficient block of the table, for example coefficient types. The list should contain one or more character vectors with as many character or numeric elements as there are coefficients/model terms. If the vectors in the list are named, the names are used as labels in the table header. For example, custom.columns = list(type = c("a", "b", "c"), 1:3) will add two columns; the first one is labeled while the second one is not. Note that the numeric elements of the second column will be converted to character objects in this example. The consequence is that decimal alignment with the dcolumn package is switched off in these columns. Note that this argument is processed after any arguments that affect the number of rows.

custom.col.pos An optional integer vector of positions for the columns given in the custom.columns argument. For example, if there are three custom columns, custom.col.pos =
c(1, 3, 3) will insert the first custom column before the first column of the original table and the remaining two custom columns after the second column of the original table. By default, all custom columns are placed after the first column, which usually contains the coefficient names.

... Custom options to be passed on to the `extract` function. For example, most extract methods provide custom options for the inclusion or exclusion of specific goodness-of-fit statistics. See the help entries of `extract` for more information.

Details

The `huxtablereg` function creates a `huxtable` object using the `huxtable` package. This allows output to HTML, LaTeX, Word, Excel, Powerpoint, and RTF. The object can be formatted using `huxtable` package functions. See also `huxreg`.

Author(s)

David Hugh-Jones

See Also

texreg-package extract

Other texreg: htmlreg(), knitreg(), matrixreg(), plotreg(), screenreg(), texreg, wordreg()

Examples

```r
library("nlme")
model.1 <- lme(distance ~ age, data = Orthodont, random = ~ 1)
model.2 <- lme(distance ~ age + Sex, data = Orthodont, random = ~ 1)
if (requireNamespace("huxtable")) {
  hr <- huxtablereg(list(model.1, model.2))
  hr <- huxtable::set_bottom_border(hr, 1, -1, 0.4)
  hr <- huxtable::set_bold(hr, 1:nrow(hr), 1, TRUE)
  hr <- huxtable::set_bold(hr, 1, -1, TRUE)
  hr <- huxtable::set_all_borders(hr, 4, 2, 0.4)
  hr <- huxtable::set_all_border_colors(hr, 4, 2, "red")
  hr
  ## Not run: huxtable::quick_pdf(hr)
  huxtable::quick_docx(hr)
  # or use in a knitr document

  ## End(Not run)
}
```
KnitReg

Flexibly choose the right table output format for use with knitr.

Description

Flexibly choose the right table output format for use with knitr.

Usage

knitreg(...)

Arguments

... Arguments to be handed over to the texreg, htmlreg, screenreg, or matrixreg function. See the respective help page for details.

Details

This function automatically selects the right function (texreg, screenreg, htmlreg, or matrixreg) with the right set of arguments for use with the knitr package, for example in RStudio. The advantage of using this function with knitr is that the user does not need to replace the texreg, htmlreg etc. function call in the document when a different output format is selected.

knitreg works with...

- R HTML documents (.Rhtml extension)
- R Sweave documents (.Rnw extension) for PDF output via LaTeX, rendered using...
  - the knitr package
  - the Sweave package
- R Markdown documents (.Rmd extension), rendered as...
  - HTML documents
  - PDF documents
  - Word documents
  - Powerpoint presentations
  - Presentations (.Rpres extension, not .Rmd)
- R Notebooks, including preview

If Markdown and HTML rendering are selected, htmlreg arguments doctype = FALSE and star.symbol = "\*" are set to enable compatibility with Markdown. With R HTML documents (but not Markdown) or presentations (.Rpres extension), only doctype = FALSE is set.

For PDF/LaTeX documents, the texreg argument use.packages = FALSE is set to suppress any package loading instructions in the preamble. The user must load any packages manually in the preamble of the document.

The knitr and rmarkdown packages must be installed for this function to work.
Value
A table as a character string in the respective output format.

Author(s)
Philip Leifeld, with input from David Hugh-Jones

See Also
texreg-package extract
Other texreg: htmlreg(), huxtablereg(), matrixreg(), plotreg(), screenreg(), texreg, wordreg()

Examples
require("nlme")
model.1 <- lme(distance ~ age, data = Orthodont, random = ~ 1)
model.2 <- lme(distance ~ age + Sex, data = Orthodont, random = ~ 1)
knitreg(list(model.1, model.2), center = FALSE, caption = "", table = FALSE)

matrixreg Convert regression output to a character matrix

Description
Conversion of R regression output to a character matrix.

Usage
matrixreg(
  l,
  single.row = FALSE,
  stars = c(0.001, 0.01, 0.05),
  custom.model.names = NULL,
  custom.coef.names = NULL,
  custom.coef.map = NULL,
  custom.gof.names = NULL,
  custom.gof.rows = NULL,
  digits = 2,
  leading.zero = TRUE,
  star.symbol = "*",
  symbol = ".",
  override.coef = 0,
  override.se = 0,
  override.pvalues = 0,
  override.ci.low = 0,
override.ci.up = 0,
odmit.coef = NULL,
oreorder.coef = NULL,
oreorder.gof = NULL,
ci.force = FALSE,
ci.force.level = 0.95,
ci.test = 0,
bold = 0,
groups = NULL,
custom.columns = NULL,
custom.col.pos = NULL,
dcolumn = TRUE,
siunitx = FALSE,
output.type = c("ascii", "latex", "html"),
include.attributes = FALSE,
trim = FALSE,
... } 

**Arguments**

table_1

l A statistical model or a list of statistical models. Lists of models can be specified as `l = list(model.1, model.2, ...)` Different object types can also be mixed.

single.row By default, a model parameter takes up two lines of the table: the standard error is listed in parentheses under the coefficient. This saves a lot of horizontal space on the page and is the default table format in most academic journals. If `single.row = TRUE` is activated, however, both coefficient and standard error are placed in a single table cell in the same line.

stars The significance levels to be used to draw stars. Between 0 and 4 threshold values can be provided as a numeric vector. For example, `stars = numeric(0)` will not print any stars and will not print any note about significance levels below the table. `stars = 0.05` will attach one single star to all coefficients where the p-value is below 0.05. `stars = c(0.001, 0.01, 0.05, 0.1)` will print one, two, or three stars, or a symbol as specified by the `symbol` argument depending on the p-values.

custom.model.names A character vector of labels for the models. By default, the models are named "Model 1", "Model 2", etc. Specifying `model.names = c("My name 1", "My name 2")` etc. overrides the default behavior.

custom.coef.names By default, `texreg` uses the coefficient names which are stored in the models. The `custom.coef.names` argument can be used to replace them by other character strings in the order of appearance. For example, if a table shows a total of three different coefficients (including the intercept), the argument `custom.coef.names = c("Intercept", "variable 1", "variable 2")` will replace their names in this order. Sometimes it happens that the same variable has a different name in different models. In this case, the user can use this function to assign identical names. If
possible, the rows will then be merged into a single row unless both rows contain
values in the same column.
Where the argument contains an NA value, the original name of the coefficient is kept. For example, custom.coef.names = c(NA,"age",NA) will only replace
the second coefficient name and leave the first and third name as they are in the
original model.
See also custom.coef.map for an easier and more comprehensive way to re-
name, omit, and reorder coefficients.

custom.coef.map
The custom.coef.map argument can be used to select, omit, rename, and reorder coefficients.
Users must supply a named list of this form: list("x" = "First variable","y"
= NA,"z" = "Third variable"). With that particular example of custom.coef.map,
1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as
"y", and variable "z" will appear as "Third variable".
3. all variables not named "x", "y", or "z" will be omitted from the table.

custom.gof.names
A character vector which is used to replace the names of the goodness-of-fit
statistics at the bottom of the table. The vector must have the same length as
the number of GOF statistics in the final table. The argument works like the
custom.coef.names argument, but for the GOF values. NA values can be in-
cluded where the original GOF name should be kept.

custom.gof.rows
A named list of vectors for new lines at the beginning of the GOF block of the ta-
ble. For example, list("Random effects" = c("YES","YES","NO"),Observations
= c(25,25,26)) would insert two new rows into the table, at the beginning of
the GOF block (i.e., after the coefficients). The rows can contain integer, nu-
meric, or character objects. Note that this argument is processed after the
custom.gof.names argument (meaning custom.gof.names should not include
any of the new GOF rows) and before the reorder.gof argument (meaning that
the new GOF order specified there should contain values for the new custom
GOF rows). Arguments for custom columns are not affected because they only
insert columns into the coefficient block.

digits
Set the number of decimal places for coefficients, standard errors and goodness-
of-fit statistics. Do not use negative values! The argument works like the digits
argument in the round function of the base package.

leading.zero
Most journals require leading zeros of coefficients and standard errors (for ex-
ample, 0.35). This is also the default texreg behavior. Some journals, however,
require omission of leading zeros (for example, .35). This can be achieved by
setting leading.zero = FALSE.

star.symbol
Alternative characters for the significance stars can be specified. This is useful if knitir and Markdown are used for HTML report generation. In Markdown, asterisks or stars are interpreted as special characters, so they have to be escaped.
To make a HTML table compatible with Markdown, specify star.symbol = "\*". Note that some other modifications are recommended for usage with
**symbol**

If four threshold values are handed over to the `stars` argument, p-values smaller than the largest threshold value but larger than the second-largest threshold value are denoted by this symbol. The default symbol is "\cdot" for the LaTeX dot, "·" for the HTML dot, or simply "." for the ASCII dot. If the `texreg` function is used, any other mathematical LaTeX symbol or plain text symbol can be used, for example `symbol = \circ` for a small circle (note that backslashes must be escaped). If the `htmlreg` function is used, any other HTML character or symbol can be used. For the `screenreg` function, only plain text characters can be used.

**override.coef**

Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as `override.coef = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.coef = c(0.05,0.06,0.07)`.

**override.se**

Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.se = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.se = c(0.05,0.06,0.07)`. Overriding standard errors can be useful for the implementation of robust SEs, for example.

**override.pvalues**

Set custom values for the p-values. New p-values are provided as a list of numeric vectors. The list contains vectors of p-values for each model. There must be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.pvalues = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.pvalues = c(0.05,0.06,0.07)`. Overriding p-values can be useful for the implementation of robust SEs and p-values, for example.

**override.ci.low**

Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.up` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

**override.ci.up**

Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.low` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.
omit.coef

A character string which is used as a regular expression to remove coefficient rows from the table. For example, `omit.coef = "group"` deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example `omit.coef = "(thresh)|(ranef)"` to remove all model terms matching either "thresh" or "ranef". The `omit.coef` argument is processed after the `custom.coef.names` argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the `extract` functions instead (see the help entry of the `extract` function).

reorder.coef

Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, `reorder.coef = c(3,2,1)` will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and after omitting coefficients, so the `custom.coef.names` and `omit.coef` arguments should follow the original order.

reorder.gof

Reorder the rows of the goodness-of-fit block of the resulting table in a custom way. The argument takes a vector of the same length as the number of GOF statistics. For example, if there are three goodness-of-fit rows, `reorder.gof = c(3,2,1)` will exchange the first and the third row. Note: Reordering takes place after processing custom GOF names and after adding new custom GOF rows, so the `custom.gof.names` and `custom.gof.rows` arguments should follow the original order, and the `reorder.gof` argument should contain values for any rows that are added through the `custom.gof.rows` argument.

ci.force

Should confidence intervals be used instead of the default standard errors and p-values? Most models implemented in the `texreg` package report standard errors and p-values by default while few models report confidence intervals. However, the functions in the `texreg` package can convert standard errors into confidence intervals using z-scores if desired. To enforce confidence intervals instead of standard errors, the `ci.force` argument accepts either a logical value indicating whether all models or none of the models should be forced to report confidence intervals (`ci.force = TRUE` for all and `ci.force = FALSE` for none) or a vector of logical values indicating for each model separately whether the model should be forced to report confidence intervals (e.g., `ci.force = c(FALSE,TRUE,FALSE)`). Confidence intervals are computed using the standard normal distribution (z-values based on the `qnorm` function). The t-distribution is currently not supported because this would require each `extract` method to have an additional argument for the degrees of freedom.

ci.force.level

If the `ci.force` argument is used to convert standard errors to confidence intervals, what confidence level should be used? By default, 0.95 is used (i.e., an alpha value of 0.05).

ci.test

If confidence intervals are reported, the `ci.test` argument specifies the reference value to establish whether a coefficient/CI is significant. The default value `ci.test = 0`, for example, will attach a significance star to coefficients if the
confidence interval does not contain 0. A value of ci.test = 1 could be useful if coefficients are provided on the odds-ratio scale, for example. If no star should be printed at all, ci.test = NA can be used. It is possible to provide a single value for all models or a vector with a separate value for each model. The ci.test argument works both for models with native support for confidence intervals and in cases where the ci.force argument is used.

**bold**

The p-value threshold below which the coefficient shall be formatted in a bold font. For example, bold = 0.05 will cause all coefficients that are significant at the 95% level to be formatted in bold. Note that this is not compatible with the dcolumn or siunitx arguments in the texreg function. If both bold and dcolumn or siunitx are TRUE, dcolumn and siunitx are switched off, and a warning message appears. Note also that it is advisable to use stars = FALSE together with the bold argument because having both bolded coefficients and significance stars usually does not make any sense.

**groups**

This argument can be used to group the rows of the table into blocks. For example, there could be one block for hypotheses and another block for control variables. Each group has a heading, and the row labels within a group are indented. The partitions must be handed over as a list of named numeric vectors, where each number is a row index and each name is the heading of the group. Example: groups = list("first group" = 1:4,"second group" = 7:8).

**custom.columns**

An optional list of additional text columns to be inserted into the coefficient block of the table, for example coefficient types. The list should contain one or more character vectors with as many character or numeric elements as there are coefficients/model terms. If the vectors in the list are named, the names are used as labels in the table header. For example, custom.columns = list(type = c("a","b","c"),1:3) will add two columns; the first one is labeled while the second one is not. Note that the numeric elements of the second column will be converted to character objects in this example. The consequence is that decimal alignment with the dcolumn package is switched off in these columns. Note that this argument is processed after any arguments that affect the number of rows.

**custom.col.pos**

An optional integer vector of positions for the columns given in the custom.columns argument. For example, if there are three custom columns, custom.col.pos = c(1,3,3) will insert the first custom column before the first column of the original table and the remaining two custom columns after the second column of the original table. By default, all custom columns are placed after the first column, which usually contains the coefficient names.

**dcolumn**

Use the dcolumn LaTeX package to get a nice alignment of the coefficients at the decimal separator (recommended for use with the texreg function). Note that only one of the three arguments bold, dcolumn, and siunitx can be used at a time as they are mutually incompatible.

**siunitx**

Use the siunitx LaTeX package to get a nice alignment of the coefficients at the decimal separator (recommended for use with the texreg function). Note that only one of the three arguments bold, dcolumn, and siunitx can be used at a time as they are mutually incompatible.

**output.type**

Which type of output should be produced? Valid values are "ascii" (for plain text tables), "latex" (for LaTeX markup in the resulting table), and "html"
Create coefficient plots from statistical model output using \texttt{ggplot2}.

**Description**

Create coefficient plots of R regression output using \texttt{ggplot2}.

**Usage**

```r
plotreg(
  l,
  file = NULL,
  custom.model.names = NULL,
  custom.title = NULL,
  custom.coef.names = NULL,
  custom.coef.map = NULL,
```
custom.note = NULL,
override.coef = 0,
override.se = 0,
override.pval = 0,
override.ci.low = 0,
override.ci.up = 0,
omit.coef = NULL,
reorder.coef = NULL,
ci.level = 0.95,
ci.inner = 0.5,
use.se = FALSE,
type = "facet",
theme = NULL,
signif.light = "#FBC9B9",
signif.medium = "#F7523A",
signif.dark = "#BD0017",
insignif.light = "#C5DBE9",
insignif.medium = "#5A9ECC",
insignif.dark = "#1C5BA6",
...
)

Arguments

1
A statistical model or a list of statistical models. Lists of models can be specified as `l = list(model.1, model.2, ...)`. Different object types can also be mixed.

custom.model.names
A character vector of labels for the models. By default, the models are named "Model 1", "Model 2", etc. Specifying `model.names = c("My name 1", "My name 2")` etc. overrides the default behavior.

custom.title
With this argument, a replacement text for the `ggtitle`, which provides a title above the diagram, can be provided. If an empty character object is provided (`custom.title = ""`), the title will be omitted completely.

custom.coef.names
By default, `texreg` uses the coefficient names which are stored in the models. The `custom.coef.names` argument can be used to replace them by other character strings in the order of appearance. For example, if a table shows a total of three different coefficients (including the intercept), the argument `custom.coef.names = c("Intercept","variable 1","variable 2")` will replace their names in this order.
Sometimes it happens that the same variable has a different name in different models. In this case, the user can use this function to assign identical names. If possible, the rows will then be merged into a single row unless both rows contain values in the same column.

Where the argument contains an `NA` value, the original name of the coefficient is kept. For example, `custom.coef.names = c(NA, "age", NA)` will only replace the second coefficient name and leave the first and third name as they are in the original model.

See also `custom.coef.map` for an easier and more comprehensive way to rename, omit, and reorder coefficients.

**custom.coef.map**

The `custom.coef.map` argument can be used to select, omit, rename, and reorder coefficients.

Users must supply a named list of this form: `list("x" = "First variable", "y" = NA, "z" = "Third variable")`. With that particular example of `custom.coef.map`,

1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as "y", and variable "z" will appear as "Third variable".
3. all variables not named "x", "y", or "z" will be omitted from the table.

**custom.note**

With this argument, a replacement text for the significance note below the table can be provided. If an empty character object is provided (`custom.note = "")`, the note will be omitted completely. If some character string is provided (e.g., `custom.note = "My note"`), the significance legend is replaced by `My note`. The original significance legend can be included by inserting the `%stars` wildcard. For example, a custom note can be added right after the significance legend by providing `custom.note = "%stars. My note."`.

If the `threeparttable` argument is used, any note should be preceded by `\item`, for example `\item %stars. \item Second note. \item Third note.` and it is possible to create line breaks in the formatted table by including `\\` and line breaks in the LaTeX code by including `\n`, for example `\n%stars.\\\nSecond line.\n`.

**override.coef**

Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as `override.coef = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.coef = c(0.05, 0.06, 0.07)`.

**override.se**

Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.se = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.se = c(0.05, 0.06, 0.07)`. Overriding standard errors can be useful for the implementation of robust SEs, for example.
plotreg

override.pval  Set custom values for the p-values. New p-values are provided as a list of numeric vectors. The list contains vectors of p-values for each model. There must be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.pvalues = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.pvalues = c(0.05, 0.06, 0.07)`. Overriding p-values can be useful for the implementation of robust SEs and p-values, for example.

override.ci.low  Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.up` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

override.ci.up  Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.low` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

omit.coef  A character string which is used as a regular expression to remove coefficient rows from the table. For example, `omit.coef = "group"` deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example `omit.coef = "(thresh)|(ranef)"` to remove all model terms matching either "thresh" or "ranef". The `omit.coef` argument is processed after the `custom.coef.names` argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the extract functions instead (see the help entry of the `extract` function.

reorder.coef  Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, `reorder.coef = c(3, 2, 1)` will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and after omitting coefficients, so the `custom.coef.names` and `omit.coef` arguments should follow the original order.

ci.level  If standard errors are converted to confidence intervals (because a model does not natively support CIs), what confidence level should be used for the outer confidence interval? By default, 0.95 is used (i.e., an alpha value of 0.05).

ci.inner  If standard errors are converted to confidence intervals (because a model does not natively support CIs), what confidence level should be used for the inner confidence interval? By default, 0.50 is used (i.e., an alpha value of 0.5). Can be set to `ci.inner = 0` to suppress inner confidence CI bars.

use.se  Use one standard error for the inner horizontal bar and two standard errors from the estimate for the outer horizontal bar (instead of confidence intervals). Only
available if standard errors can be extracted from the model using the respective `extract` method.

**type**

The default option is `type = "facet"`. If only one model is specified, it will print one forest plot applied to point estimates and confidence intervals. If more than one model is specified, it will print as many facets as the number of models in a column of plots. Alternatively, if `type = "forest"` is specified, coefficients from one or more models will be grouped together and displayed as a single forest plot.

**theme**

The theme argument can be used to customize the appearance of the plot. The default theme is `theme_bw`. It can be replaced by any other `ggplot2` theme. See `ggtheme` for details.

**signif.light**

Color of outer confidence intervals for significant model terms.

**signif.medium**

Color of inner confidence intervals for significant model terms.

**signif.dark**

Color of point estimates and labels for significant model terms.

**insignif.light**

Color of outer confidence intervals for insignificant model terms.

**insignif.medium**

Color of inner confidence intervals for insignificant model terms.

**insignif.dark**

Color of point estimates and labels for insignificant model terms.

**...**

Custom options to be passed on to the `extract` function. For example, most extract methods provide custom options for the inclusion or exclusion of specific goodness-of-fit statistics. See the help entries of `extract` for more information.

**Details**

The `plotreg` function produces coefficient plots (i.e., forest plots applied to point estimates and confidence intervals) and works much like the `screenreg`, `texreg`, `htmlreg`, `matrixreg` and `wordreg` functions. It accepts a single model or multiple statistical models as input and internally extracts the relevant data from the models. If confidence intervals are not defined in the extract method of a statistical model (see `extract`), the default standard errors are converted to confidence intervals. Most of the arguments work like in the `screenreg`, `texreg`, and `htmlreg` `matrixreg`, and `wordreg` functions. It is possible to display the plots in two ways: using the `type = "facet"` argument, one forest plot applied to point estimates and confidence intervals will be visualized in case there is only one model. If there is more than one model, each one will be plotted next to the other as a separate facet; using the `type = "forest"` argument, coefficients from one or more models will be grouped together and displayed as a single forest plot.

**Value**

Coefficient plot as a `ggplot2` gg object if `file = FALSE`. NULL otherwise.

**Author(s)**

Claudia Zucca, Philip Leifeld
## Not run:
# example from the 'lm' help file:
ctl <- c(4.17, 5.58, 5.18, 6.11, 4.50, 4.61, 5.17, 4.53, 5.33, 5.14)
trt <- c(4.81, 4.17, 4.41, 3.59, 5.87, 3.83, 6.03, 4.89, 4.32, 4.69)
group <- gl(2, 10, 20, labels = c("Ctl", "Trt"))
weight <- c(ctl, trt)
lm.D9 <- lm(weight ~ group)
lm.D90 <- lm(weight ~ group - 1)
plotreg(lm.D9) # plot model output as a diagram

# customize theme and title and save as a PDF file.
plotreg(lm.D9, theme = theme_dark(), ggtitle = "my title", file = "myplot.pdf")
unlink("myplot.pdf")

# group coefficients from multiple models
plotreg(list(lm.D9, lm.D90), type = "forest")

## End(Not run)
Arguments

academic_user  Should be TRUE if you are at a university or public research institute. Should be FALSE if you are a private user, for example you are using texreg in your work for a firm, NGO, association, government department, as an individual user etc. We particularly need praise from non-academic users to demonstrate societal impact, but we can also make the case for academic usage to generate impact indirectly.

organization Please tell us the name of the organization for which you are using texreg. If we can show that the package is being employed in a number of different settings, this will help us demonstrate impact.

name (Optional) We would be delighted to to know who you are. After all, we can quote you much more effectively if we can tell the funders and employers who provided this praise! If possible, include your title.

general_praise Use this argument to provide general praise, for example about the way it was designed, the user support you have received, or just how much you enjoy using it. While this is useful, however, we would be even more interested in receiving statements in how texreg makes you more productive (in the increase_productivity argument) or how it increases the quality of your work or your reports (through the increase_quality argument). Note: you need to provide at least one of these three free-form text arguments.

increase_productivity
This is one of the fields we are most interested in. Please use this field to tell us how texreg is making you more productive. For example, does it speed up writing your articles or research reports? Does it enable you to skip manual work like copy and paste of your results into your reports, or to avoid fiddling with table formatting? How much time has it saved you so far? Are there any other benefits in terms of productivity you can think of? Note: you need to provide feedback using at least one of the three free-form arguments (general_praise, increase_productivity, or increase_quality).

increase_quality
This is one of the fields we are most interested in. Please use this argument to tell us how texreg increases the quality of your work or the quality of your reporting. For example, does the package generate tables that look more professional than the tables you used to create manually? Are you using screenreg to improve your workflow by understanding better how the results of multiple models compare? Are you using plotreg to visualize and present your statistical results in a more effective way? Can you think of any other ways in which texreg is helping you? Note: you need to provide feedback using at least one of
the three free-form arguments (general_praise, increase_productivity, or increase_quality).

start_using (Optional) When did you start using texreg? We are interested in the approximate time or year as a free-form text argument, for example "back in 2013 when the JSS article came out".

where_learn (Optional) Where or how did you learn about the texreg package?

contact_details (Optional) Tell us how we can contact you in case we would benefit from additional information. This might help us further down the road in compiling an impact case study or a similar report. Don’t worry, this information will not be displayed on the website!

models (Optional) Which kinds of statistical models do you use in your work? For example, "Mostly linear models, but also lme4 and ergm."

num_users (Optional) How many other texreg users do you know? In particular, if you are a non-academic user, would you mind telling us how many other non-academic users you are aware of and how many of them are in your organization? The more we know, the more convincing our evidence base will be. This argument accepts numeric values or more detailed responses as a character object.

return.response If TRUE, a website with the submitted data will be returned as a response object, as defined in the httr package. You can load the httr package and use the content function, possibly enclosed in an as.character call, to inspect the output and diagnose any problems with the transmission of the data. Only use this argument if instructed by the package authors.

Details

The praise_interactive function asks you 11 questions interactively on the R console. You can choose to answer or skip them. Some questions are mandatory but most are optional. After collecting your answers, it will call the praise function to submit your praise. You can also choose to use the praise function directly and supply your answers as arguments. Either way is fine.

Before your praise is submitted, the functions will present an interactive menu and ask if you want to submit the praise now. So do not worry about accidentally submitting feedback.

You can use these functions to praise the texreg package. Funders and academic employers are increasingly interested in seeing evidence for the impact academic research generates. For software, such as texreg, this is very hard to accomplish because the developers are usually disconnected from the users. The consequence is that incentives for developing packages like these are diminishing the more the funders and employers require evidence of impact on society, firms, or policy makers.

The praise and praise_interactive functions are our attempt at rectifying the situation. With these functions, you can provide positive feedback to the developers. The praise is saved to a database on the web server of the package maintainer and subsequently displayed at http://www.philipleifeld.com/praise for other users, funders, and employers to view. This will also enable the package authors to compile reports about how texreg is used by academic and non-academic users to increase their productivity and work quality, for example in the form of an impact case study for the next round of the UK Research Excellence Framework (REF).
We need many positive examples of how `texreg` has an impact on your work. We are especially interested in non-academic users, but welcome feedback from anyone. So please contribute by using the praise function! Tell us how cool this package is and how it has changed your work!

The minimal information we require from you is whether you are an academic or non-academic user, the name of your organization, and some free-form praise (of a general nature, or about how it makes you more productive, or about how it increases the quality of your work or reporting). But there are some additional fields. While we are happy with the basic information, of course we will be happier if we also know your name, how to contact you, what kinds of models you work with, and some other details. Your choice!

Please note that by using the `praise` or `praise_interactive` function you agree that the information you provide through the function, including your approximate location, is stored online in a database, displayed on the website of the package author, and used in reports to funders, employers etc. (This is the whole purpose of it.) You can contact the package maintainer any time to have your praise removed within a few days.

**Value**

If everything works well, no output is returned (but see the `return.response` argument to change this). If the submission of the praise to the maintainer fails, a `response` object (as defined in the `httr` package) will be returned. Should you have any problems, do feel free to e-mail your praise to the package maintainer directly.

**Author(s)**

Philip Leifeld

**Examples**

```r
## Not run:
praise(academic_user = TRUE,
      organization = "University of Happy Tables",
      increase_quality = "Man, I've never seen such pretty tables!")
## End(Not run)
```

---

### print.texregTable

*Prints a `texregTable` object.*

**Description**

Prints a `texregTable` object.

**Usage**

```r
## S3 method for class 'texregTable'
print(x, ...)
```
**Arguments**

- `x`: A `texregTable` argument, as produced by `texreg` and related functions.
- `...`: Additional arguments for the `cat` function.

**Author(s)**

Philip Leifeld

---

**screenreg**

*Convert regression output to an ASCII table*

**Description**

Conversion of R regression output to an ASCII table for display on screen.

**Usage**

```r
screenreg(
  l,
  file = NULL,
  single.row = FALSE,
  stars = c(0.001, 0.01, 0.05),
  custom.header = NULL,
  custom.model.names = NULL,
  custom.coef.names = NULL,
  custom.coef.map = NULL,
  custom.gof.names = NULL,
  custom.gof.rows = NULL,
  custom.note = NULL,
  digits = 2,
  leading.zero = TRUE,
  star.symbol = "*",
  symbol = ".",
  override.coef = 0,
  override.se = 0,
  override.pvalues = 0,
  override.ci.low = 0,
  override.ci.up = 0,
  omit.coef = NULL,
  reorder.coef = NULL,
  reorder.gof = NULL,
  ci.force = FALSE,
  ci.force.level = 0.95,
  ci.test = 0,
  groups = NULL,
  custom.columns = NULL,
)```
screenreg

custom.col.pos = NULL,
column.spacing = 2,
outer.rule = "=",
inner.rule = "-",
...
)

Arguments

1  A statistical model or a list of statistical models. Lists of models can be specified as \texttt{l = list(model.1,model.2,...)}. Different object types can also be mixed.

file  Using this argument, the resulting table is written to a file rather than to the R prompt. The file name can be specified as a character string. Writing a table to a file can be useful for working with MS Office or LibreOffice. For example, using the htmlreg function, an HTML table can be written to a file with the extension \texttt{.doc} and opened with MS Word. The table can then be simply copied into any Word document, retaining the formatting of the table. Note that LibreOffice can import only plain HTML; CSS decorations are not supported; the resulting tables do not retain the full formatting in LibreOffice.

single.row  By default, a model parameter takes up two lines of the table: the standard error is listed in parentheses under the coefficient. This saves a lot of horizontal space on the page and is the default table format in most academic journals. If \texttt{single.row = TRUE} is activated, however, both coefficient and standard error are placed in a single table cell in the same line.

stars  The significance levels to be used to draw stars. Between 0 and 4 threshold values can be provided as a numeric vector. For example, \texttt{stars = numeric(0)} will not print any stars and will not print any note about significance levels below the table. \texttt{stars = 0.05} will attach one single star to all coefficients where the p-value is below 0.05. \texttt{stars = c(0.001,0.01,0.05,0.1)} will print one, two, or three stars, or a symbol as specified by the symbol argument depending on the p-values.

custom.header  An optional named list of multi-column headers that are placed above the model names. For example, \texttt{custom.header = list("abc" = 1:3,"ef" = 4:5)} will add the label "abc" to the first three models and "ef" to the fourth and fifth model. The column with coefficient names and any custom columns added by the "custom.columns" argument are not counted towards these positions. If \texttt{booktabs = TRUE}, \texttt{\midrule} rules are added below the respective labels; otherwise \texttt{\cline} lines are used.

custom.model.names  A character vector of labels for the models. By default, the models are named "Model 1", "Model 2", etc. Specifying \texttt{model.names = c("My name 1","My name 2")} etc. overrides the default behavior.

custom.coef.names  By default, \texttt{texreg} uses the coefficient names which are stored in the models. The \texttt{custom.coef.names} argument can be used to replace them by other character strings in the order of appearance. For example, if a table shows a total of three different coefficients (including the intercept), the argument
custom.coef.names = c("Intercept","variable 1","variable 2") will replace their names in this order.

Sometimes it happens that the same variable has a different name in different models. In this case, the user can use this function to assign identical names. If possible, the rows will then be merged into a single row unless both rows contain values in the same column.

Where the argument contains an NA value, the original name of the coefficient is kept. For example, custom.coef.names = c(NA,"age",NA) will only replace the second coefficient name and leave the first and third name as they are in the original model.

See also custom.coef.map for an easier and more comprehensive way to rename, omit, and reorder coefficients.

custom.coef.map

The custom.coef.map argument can be used to select, omit, rename, and reorder coefficients.

Users must supply a named list of this form: list("x" = "First variable","y" = NA,"z" = "Third variable"). With that particular example of custom.coef.map,

1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as "y", and variable "z" will appear as "Third variable".
3. all variables not named "x", "y", or "z" will be omitted from the table.

custom.gof.names

A character vector which is used to replace the names of the goodness-of-fit statistics at the bottom of the table. The vector must have the same length as the number of GOF statistics in the final table. The argument works like the custom.coef.names argument, but for the GOF values. NA values can be included where the original GOF name should be kept.

custom.gof.rows

A named list of vectors for new lines at the beginning of the GOF block of the table. For example, list("Random effects" = c("YES","YES","NO"),Observations = c(25,25,26)) would insert two new rows into the table, at the beginning of the GOF block (i.e., after the coefficients). The rows can contain integer, numeric, or character objects. Note that this argument is processed after the custom.gof.names argument (meaning custom.gof.names should not include any of the new GOF rows) and before the reorder.gof argument (meaning that the new GOF order specified there should contain values for the new custom GOF rows). Arguments for custom columns are not affected because they only insert columns into the coefficient block.

custom.note

With this argument, a replacement text for the significance note below the table can be provided. If an empty character object is provided (custom.note = ""), the note will be omitted completely. If some character string is provided (e.g., custom.note = "My note"), the significance legend is replaced by My note. The original significance legend can be included by inserting the %stars wildcard. For example, a custom note can be added right after the significance legend by providing custom.note = "%stars. My note.".

If the threeparttable argument is used, any note should be preceded by \item, for example \item %stars. \item Second note. \item Third note.". and
it is possible to create line breaks in the formatted table by including "\\\n" and line breaks in the LaTeX code by including "\n", for example "\n\item %stars.\\\n\item Second line."

digits Set the number of decimal places for coefficients, standard errors and goodness-of-fit statistics. Do not use negative values! The argument works like the digits argument in the round function of the base package.

leading.zero Most journals require leading zeros of coefficients and standard errors (for example, 0.35). This is also the default texreg behavior. Some journals, however, require omission of leading zeros (for example, .35). This can be achieved by setting leading.zero = FALSE.

star.symbol Alternative characters for the significance stars can be specified. This is useful if knitr and Markdown are used for HTML report generation. In Markdown, asterisks or stars are interpreted as special characters, so they have to be escaped. To make a HTML table compatible with Markdown, specify star.symbol = "\*". Note that some other modifications are recommended for usage with knitr in combination with Markdown or HTML (see the inline.css, doctype, html.tag, head.tag, and body.tag arguments in the htmlreg function).

symbol If four threshold values are handed over to the stars argument, p-values smaller than the largest threshold value but larger than the second-largest threshold value are denoted by this symbol. The default symbol is "\cdot" for the LaTeX dot, "&middot;" for the HTML dot, or simply "." for the ASCII dot. If the texreg function is used, any other mathematical LaTeX symbol or plain text symbol can be used, for example symbol = "\circ" for a small circle (note that backslashes must be escaped). If the htmlreg function is used, any other HTML character or symbol can be used. For the screenreg function, only plain text characters can be used.

override.coef Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as override.coef = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.coef = c(0.05,0.06,0.07).

override.se Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as override.se = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.se = c(0.05,0.06,0.07). Overriding standard errors can be useful for the implementation of robust SEs, for example.

override.pvalues Set custom values for the p-values. New p-values are provided as a list of numeric vectors. The list contains vectors of p-values for each model. There must be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as
override.pvalues = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.pvalues = c(0.05, 0.06, 0.07). Overriding p-values can be useful for the implementation of robust SEs and p-values, for example.

override.ci.low
Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the override.ci.up argument, the standard errors and p-values as well as the ci.force argument are ignored.

override.ci.up
Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the override.ci.low argument, the standard errors and p values as well as the ci.force argument are ignored.

omit.coef
A character string which is used as a regular expression to remove coefficient rows from the table. For example, omit.coef = "group" deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example omit.coef = "(thresh)|(ranef)" to remove all model terms matching either "thresh" or "ranef". The omit.coef argument is processed after the custom.coef.names argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the extract functions instead (see the help entry of the extract function.

reorder.coef
Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, reorder.coef = c(3, 2, 1) will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and after omitting coefficients, so the custom.coef.names argument should follow the original order.

reorder.gof
Reorder the rows of the goodness-of-fit block of the resulting table in a custom way. The argument takes a vector of the same length as the number of GOF statistics. For example, if there are three goodness-of-fit rows, reorder.gof = c(3, 2, 1) will exchange the first and the third row. Note: Reordering takes place after processing custom GOF names and after adding new custom GOF rows, so the custom.gof.names and custom.gof.rows arguments should follow the original order, and the reorder.gof argument should contain values for any rows that are added through the custom.gof.rows argument.

ci.force
Should confidence intervals be used instead of the default standard errors and p-values? Most models implemented in the texreg package report standard errors and p-values by default while few models report confidence intervals. However, the functions in the texreg package can convert standard errors and into confidence intervals using z-scores if desired. To enforce confidence intervals instead of standard errors, the ci.force argument accepts either a logical value
indicating whether all models or none of the models should be forced to report confidence intervals (ci.force = TRUE for all and ci.force = FALSE for none) or a vector of logical values indicating for each model separately whether the model should be forced to report confidence intervals (e.g., ci.force = c(FALSE, TRUE, FALSE)). Confidence intervals are computed using the standard normal distribution (z-values based on the `qnorm` function). The t-distribution is currently not supported because this would require each `extract` method to have an additional argument for the degrees of freedom.

**ci.force.level** If the ci.force argument is used to convert standard errors to confidence intervals, what confidence level should be used? By default, 0.95 is used (i.e., an alpha value of 0.05).

**ci.test** If confidence intervals are reported, the ci.test argument specifies the reference value to establish whether a coefficient/CI is significant. The default value ci.test = 0, for example, will attach a significance star to coefficients if the confidence interval does not contain 0. A value of ci.test = 1 could be useful if coefficients are provided on the odds-ratio scale, for example. If no star should be printed at all, ci.test = NA can be used. It is possible to provide a single value for all models or a vector with a separate value for each model. The ci.test argument works both for models with native support for confidence intervals and in cases where the ci.force argument is used.

**groups** This argument can be used to group the rows of the table into blocks. For example, there could be one block for hypotheses and another block for control variables. Each group has a heading, and the row labels within a group are indented. The partitions must be handed over as a list of named numeric vectors, where each number is a row index and each name is the heading of the group. Example: `groups = list("first group" = 1:4,"second group" = 7:8)`.

**custom.columns** An optional list of additional text columns to be inserted into the coefficient block of the table, for example coefficient types. The list should contain one or more character vectors with as many character or numeric elements as there are coefficients/model terms. If the vectors in the list are named, the names are used as labels in the table header. For example, `custom.columns = list(type = c("a","b","c"),1:3)` will add two columns; the first one is labeled while the second one is not. Note that the numeric elements of the second column will be converted to character objects in this example. The consequence is that decimal alignment with the dcolumn package is switched off in these columns. Note that this argument is processed after any arguments that affect the number of rows.

**custom.col.pos** An optional integer vector of positions for the columns given in the custom.columns argument. For example, if there are three custom columns, `custom.col.pos = c(1,3,3)` will insert the first custom column before the first column of the original table and the remaining two custom columns after the second column of the original table. By default, all custom columns are placed after the first column, which usually contains the coefficient names.

**column.spacing** The amount of space between any two columns of a table. By default, two spaces are used. If the tables do not fit on a single page horizontally, the value can be set to 1 or 0.
outer.rule  The character which is used to draw the outer horizontal line above and below a table. If an empty character object is provided (i.e., `outer.rule = ""`), there will be no outer horizontal lines. Recommended values are "", "-", "_", or "#".

inner.rule  The character used to draw the inner horizontal line above and below a table. If an empty character object is provided (i.e., `outer.rule = ""`), there will be no inner horizontal lines. Recommended values are "", "-", or "_".

...  Custom options to be passed on to the `extract` function. For example, most extract methods provide custom options for the inclusion or exclusion of specific goodness-of-fit statistics. See the help entries of `extract` for more information.

Details

The `screenreg` function creates text representations of tables and prints them to the R console. This is an alternative to the `summary` function and serves easy model comparison. Moreover, once a table has been prepared in the R console, it can be later exported to LaTeX or HTML with little extra effort because the majority of arguments of the different functions are identical.

Author(s)

Philip Leifeld

References


See Also

texreg-package  `extract`

Other `texreg`: `htmlreg()`, `huxtablereg()`, `knitreg()`, `matrixreg()`, `plotreg()`, `texreg`, `wordreg()`

Examples

```r
# Display models from ?lm:
ctl <- c(4.17, 5.58, 5.18, 6.11, 4.50, 4.61, 5.17, 4.53, 5.33, 5.14)
trt <- c(4.81, 4.17, 4.41, 3.59, 5.87, 3.83, 6.03, 4.89, 4.32, 4.69)
group <- gl(2, 10, 20, labels = c("Ctl", "Trt"))
weight <- c(ctl, trt)
lm.D9 <- lm(weight ~ group)
lm.D90 <- lm(weight ~ group - 1)
screenreg(list(lm.D9, lm.D90))
```
show, texreg-method  

Show method for pretty output of texreg objects

Description

Show method for pretty output of texreg objects.

Usage

## S4 method for signature 'texreg'
show(object)

Arguments

object  
The texreg object to display.

Details

Print the different slots of texreg objects to the screen.

Author(s)

Philip Leifeld

References


See Also

extract, createTexreg, screenreg

texreg  

Convert regression output to a LaTeX table

Description

Conversion of \texttt{R} regression output to a LaTeX table.
Usage

texreg(
  l,
  file = NULL,
  single.row = FALSE,
  stars = c(0.001, 0.01, 0.05),
  custom.header = NULL,
  custom.model.names = NULL,
  custom.coef.names = NULL,
  custom.coef.map = NULL,
  custom.gof.names = NULL,
  custom.gof.rows = NULL,
  custom.note = NULL,
  digits = 2,
  leading.zero = TRUE,
  symbol = "\cdot",
  override.coef = 0,
  override.se = 0,
  override.pvalues = 0,
  override.ci.low = 0,
  override.ci.up = 0,
  omit.coef = NULL,
  reorder.coef = NULL,
  reorder.gof = NULL,
  ci.force = FALSE,
  ci.force.level = 0.95,
  ci.test = 0,
  groups = NULL,
  custom.columns = NULL,
  custom.col.pos = NULL,
  bold = 0,
  center = TRUE,
  caption = "Statistical models",
  caption.above = FALSE,
  label = "table:coefficients",
  booktabs = FALSE,
  dcolumn = FALSE,
  siunitx = FALSE,
  lyx = FALSE,
  sideways = FALSE,
  longtable = FALSE,
  threeparttable = FALSE,
  use.packages = TRUE,
  table = TRUE,
  tabular = TRUE,
  no.margin = FALSE,
  fontsize = NULL,
  scalebox = NULL,
Arguments

- float.pos = "",
- ...

A statistical model or a list of statistical models. Lists of models can be specified as 
1 = list(model.1, model.2, ...). Different object types can also be mixed.

file
- Using this argument, the resulting table is written to a file rather than to the R
  prompt. The file name can be specified as a character string. Writing a table to a
  file can be useful for working with MS Office or LibreOffice. For example, using
  the htmlreg function, an HTML table can be written to a file with the extension
  .doc and opened with MS Word. The table can then be simply copied into any
  Word document, retaining the formatting of the table. Note that LibreOffice
  can import only plain HTML; CSS decorations are not supported; the resulting
  tables do not retain the full formatting in LibreOffice.

single.row
- By default, a model parameter takes up two lines of the table: the standard
  error is listed in parentheses under the coefficient. This saves a lot of horizontal
  space on the page and is the default table format in most academic journals. If
  single.row = TRUE is activated, however, both coefficient and standard error
  are placed in a single table cell in the same line.

stars
- The significance levels to be used to draw stars. Between 0 and 4 threshold
  values can be provided as a numeric vector. For example, stars = numeric(0)
  will not print any stars and will not print any note about significance levels below
  the table. stars = 0.05 will attach one single star to all coefficients where the p
  value is below 0.05. stars = c(0.001, 0.01, 0.05, 0.1) will print one, two, or
  three stars, or a symbol as specified by the symbol argument depending on the
  p-values.

custom.header
- An optional named list of multi-column headers that are placed above the model
  names. For example, custom.header = list("abc" = 1:3, "ef" = 4:5) will
  add the label "abc" to the first three models and "ef" to the fourth and fifth
  model. The column with coefficient names and any custom columns added by the
  "custom.columns" argument are not counted towards these positions. If
  booktabs = TRUE, \cmidrule rules are added below the respective labels; other-
  wise \cline lines are used.

custom.model.names
- A character vector of labels for the models. By default, the models are named
  "Model 1", "Model 2", etc. Specifying model.names = c("My name 1", "My name
  2") etc. overrides the default behavior.

custom.coef.names
- By default, texreg uses the coefficient names which are stored in the mod-
  els. The custom.coef.names argument can be used to replace them by other
  character strings in the order of appearance. For example, if a table shows
  a total of three different coefficients (including the intercept), the argument
  custom.coef.names = c("Intercept", "variable 1", "variable 2") will re-
  place their names in this order.

Sometimes it happens that the same variable has a different name in different
models. In this case, the user can use this function to assign identical names. If
possible, the rows will then be merged into a single row unless both rows contain values in the same column.

Where the argument contains an NA value, the original name of the coefficient is kept. For example, `custom.coef.names = c(NA,"age",NA)` will only replace the second coefficient name and leave the first and third name as they are in the original model.

See also `custom.coef.map` for an easier and more comprehensive way to rename, omit, and reorder coefficients.

**custom.coef.map**

The `custom.coef.map` argument can be used to select, omit, rename, and reorder coefficients.

Users must supply a named list of this form: `list("x" = "First variable","y" = NA,"z" = "Third variable")`. With that particular example of `custom.coef.map`,

1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as "y", and variable "z" will appear as "Third variable".
3. all variables not named "x", "y", or "z" will be omitted from the table.

**custom.gof.names**

A character vector which is used to replace the names of the goodness-of-fit statistics at the bottom of the table. The vector must have the same length as the number of GOF statistics in the final table. The argument works like the `custom.coef.names` argument, but for the GOF values. NA values can be included where the original GOF name should be kept.

**custom.gof.rows**

A named list of vectors for new lines at the beginning of the GOF block of the table. For example, `list("Random effects" = c("YES","YES","NO"),Observations = c(25,25,26))` would insert two new rows into the table, at the beginning of the GOF block (i.e., after the coefficients). The rows can contain integer, numeric, or character objects. Note that this argument is processed after the `custom.gof.names` argument (meaning `custom.gof.names` should not include any of the new GOF rows) and before the `reorder.gof` argument (meaning that the new GOF order specified there should contain values for the new custom GOF rows). Arguments for custom columns are not affected because they only insert columns into the coefficient block.

**custom.note**

With this argument, a replacement text for the significance note below the table can be provided. If an empty character object is provided (`custom.note = ""`), the note will be omitted completely. If some character string is provided (e.g., `custom.note = "My note"`), the significance legend is replaced by `My note`. The original significance legend can be included by inserting the `%stars` wildcard. For example, a custom note can be added right after the significance legend by providing `custom.note = "%stars. My note."`.

If the `threeparttable` argument is used, any note should be preceded by "\item", for example "\item %stars. \item Second note. \item Third note.", and it is possible to create line breaks in the formatted table by including "\\" and line breaks in the LaTeX code by including "\n", for example "\n\item %stars.\\\n\item Second line.\n".
digits

Set the number of decimal places for coefficients, standard errors and goodness-of-fit statistics. Do not use negative values! The argument works like the digits argument in the round function of the base package.

leading.zero

Most journals require leading zeros of coefficients and standard errors (for example, 0.35). This is also the default texreg behavior. Some journals, however, require omission of leading zeros (for example, .35). This can be achieved by setting leading.zero = FALSE.

symbol

If four threshold values are handed over to the stars argument, p-values smaller than the largest threshold value but larger than the second-largest threshold value are denoted by this symbol. The default symbol is "\cdot" for the LaTeX dot, "&middot;" for the HTML dot, or simply "." for the ASCII dot. If the texreg function is used, any other mathematical LaTeX symbol or plain text symbol can be used, for example symbol = "\circ" for a small circle (note that backslashes must be escaped). If the htmlreg function is used, any other HTML character or symbol can be used. For the screenreg function, only plain text characters can be used.

override.coef

Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as override.coef = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.coef = c(0.05,0.06,0.07).

override.se

Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as override.se = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.se = c(0.05,0.06,0.07). Overriding standard errors can be useful for the implementation of robust SEs, for example.

override.pvalues

Set custom values for the p-values. New p-values are provided as a list of numeric vectors. The list contains vectors of p-values for each model. There must be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as override.pvalues = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07)). If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: override.pvalues = c(0.05,0.06,0.07). Overriding p-values can be useful for the implementation of robust SEs and p-values, for example.

override.ci.low

Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the override.ci.up argument, the standard errors and p-values as well as the ci.force argument are ignored.
override.ci.up Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the override.ci.low argument, the standard errors and p values as well as the ci.force argument are ignored.

omit.coef A character string which is used as a regular expression to remove coefficient rows from the table. For example, omit.coef = "group" deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example omit.coef = "(thresh)|(ranef)" to remove all model terms matching either "thresh" or "ranef". The omit.coef argument is processed after the custom.coef.names argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the extract functions instead (see the help entry of the extract function.

reorder.coef Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, reorder.coef = c(3, 2, 1) will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and after omitting coefficients, so the custom.coef.names and omit.coef arguments should follow the original order.

reorder.gof Reorder the rows of the goodness-of-fit block of the resulting table in a custom way. The argument takes a vector of the same length as the number of GOF statistics. For example, if there are three goodness-of-fit rows, reorder.gof = c(3, 2, 1) will exchange the first and the third row. Note: Reordering takes place after processing custom GOF names and after adding new custom GOF rows, so the custom.gof.names and custom.gof.rows arguments should follow the original order, and the reorder.gof argument should contain values for any rows that are added through the custom.gof.rows argument.

ci.force Should confidence intervals be used instead of the default standard errors and p-values? Most models implemented in the texreg package report standard errors and p-values by default while few models report confidence intervals. However, the functions in the texreg package can convert standard errors and into confidence intervals using z-scores if desired. To enforce confidence intervals instead of standard errors, the ci.force argument accepts either a logical value indicating whether all models or none of the models should be forced to report confidence intervals (ci.force = TRUE for all and ci.force = FALSE for none) or a vector of logical values indicating for each model separately whether the model should be forced to report confidence intervals (e.g., ci.force = c(FALSE, TRUE, FALSE)). Confidence intervals are computed using the standard normal distribution (z-values based on the qnorm function). The t-distribution is currently not supported because this would require each extract method to have an additional argument for the degrees of freedom.

ci.force.level If the ci.force argument is used to convert standard errors to confidence intervals, what confidence level should be used? By default, 0.95 is used (i.e., an alpha value of 0.05).
ci.test If confidence intervals are reported, the ci.test argument specifies the reference value to establish whether a coefficient/CI is significant. The default value ci.test = 0, for example, will attach a significance star to coefficients if the confidence interval does not contain 0. A value of ci.test = 1 could be useful if coefficients are provided on the odds-ratio scale, for example. If no star should be printed at all, ci.test = NA can be used. It is possible to provide a single value for all models or a vector with a separate value for each model. The ci.test argument works both for models with native support for confidence intervals and in cases where the ci.force argument is used.

groups This argument can be used to group the rows of the table into blocks. For example, there could be one block for hypotheses and another block for control variables. Each group has a heading, and the row labels within a group are indented. The partitions must be handed over as a list of named numeric vectors, where each number is a row index and each name is the heading of the group. Example: groups = list("first group" = 1:4,"second group" = 7:8).

custom.columns An optional list of additional text columns to be inserted into the coefficient block of the table, for example coefficient types. The list should contain one or more character vectors with as many character or numeric elements as there are coefficients/model terms. If the vectors in the list are named, the names are used as labels in the table header. For example, custom.columns = list(type = c("a","b","c"),1:3) will add two columns; the first one is labeled while the second one is not. Note that the numeric elements of the second column will be converted to character objects in this example. The consequence is that decimal alignment with the dcolumn package is switched off in these columns. Note that this argument is processed after any arguments that affect the number of rows.

custom.col.pos An optional integer vector of positions for the columns given in the custom.columns argument. For example, if there are three custom columns, custom.col.pos = c(1,3,3) will insert the first custom column before the first column of the original table and the remaining two custom columns after the second column of the original table. By default, all custom columns are placed after the first column, which usually contains the coefficient names.

bold The p-value threshold below which the coefficient shall be formatted in a bold font. For example, bold = 0.05 will cause all coefficients that are significant at the 95% level to be formatted in bold. Note that this is not compatible with the dcolumn or siunitx arguments in the texreg function. If both bold and dcolumn or siunitx are TRUE, dcolumn and siunitx are switched off, and a warning message appears. Note also that it is advisable to use stars = FALSE together with the bold argument because having both bolded coefficients and significance stars usually does not make any sense.

center Should the table be horizontally aligned at the center of the page?

caption Set the caption of the table.

caption.above Should the caption of the table be placed above the table? By default, it is placed below the table.

label Set the label of the table environment.

booktabs Use the booktabs LaTeX package to get thick horizontal rules in the output table (recommended).
dcolumn  Use the \texttt{dcolumn} LaTeX package to get a nice alignment of the coefficients at the decimal separator (recommended for use with the \texttt{texreg} function). Note that only one of the three arguments \texttt{bold}, \texttt{dcolumn}, and \texttt{siunitx} can be used at a time as they are mutually incompatible.

siunitx  Use the \texttt{siunitx} LaTeX package to get a nice alignment of the coefficients at the decimal separator (recommended for use with the \texttt{texreg} function). Note that only one of the three arguments \texttt{bold}, \texttt{dcolumn}, and \texttt{siunitx} can be used at a time as they are mutually incompatible.

lyx  logical; if \texttt{TRUE}, each new line in the output is doubled, which facilitates transferring the output into the LyX document processor.

sideways  If \texttt{sideways = TRUE} is set, the table floating environment is replaced by a \texttt{sidewaystable} float, and the \texttt{rotating} package is loaded in the preamble. The argument only has an effect if \texttt{table = TRUE} is also set.

longtable  If \texttt{longtable = TRUE} is set, the \texttt{longtable} environment from the \texttt{longtable} LaTeX package is used to set tables across multiple pages. Note that this argument is not compatible with the \texttt{sideways} and \texttt{scalebox} arguments. These arguments will be automatically switched off when \texttt{longtable = TRUE} is set.

threeparttable  If \texttt{threeparttable = TRUE} is set, the \texttt{threeparttable} environment will be used to enclose the \texttt{tabular} environment in the LaTeX code, and the significance note will be enclosed in a \texttt{tablenotes} environment. This permits word wrapping of long table notes and adequate spacing between multiple notes. See also the \texttt{custom.note} argument. If \texttt{longtable} is used, the \texttt{threeparttablex} LaTeX package is used instead of the \texttt{threeparttable} package.

use.packages  If this argument is set to \texttt{TRUE} (= the default behavior), the required LaTeX packages are loaded in the beginning. If set to \texttt{FALSE}, the use package statements are omitted from the output.

table  By default, \texttt{texreg} puts the actual \texttt{tabular} object in a table floating environment. To get only the \texttt{tabular} object without the whole table header, set \texttt{table = FALSE}.

tabular  By default, the table contents are wrapped in a \texttt{tabular} environment. To get only the contents for each row without the environment, set \texttt{tabular = FALSE}. Note that if \texttt{tabular = FALSE}, the \texttt{table} argument must also be \texttt{FALSE}, otherwise a warning is printed. Switching off the \texttt{tabular} environment may be useful for designing one's own table more flexibly, for example using \texttt{tabular*} or \texttt{tabularx} environments in LaTeX.

no.margin  In order to save space, inner margins of tables can be switched off.

fontsize  The \texttt{fontsize} argument serves to change the font size used in the table. Valid values are \texttt{"tiny"}, \texttt{"scriptsize"}, \texttt{"footnotesize"}, \texttt{"small"}, \texttt{"normalsize"}, \texttt{"large"}, \texttt{"Large"}, \texttt{"LARGE"}, \texttt{"huge"}, and \texttt{"Huge"}. Note that the \texttt{scalebox} argument often achieves better results when the goal is to change the size of the table.

scalebox  The \texttt{scalebox} argument serves to resize the table. For example, \texttt{scalebox = 1.0} is equivalent to the normal size, \texttt{scalebox = 0.5} decreases the size of the table by one half, and \texttt{scalebox = 2.0} doubles the space occupied by the table. Note that the \texttt{scalebox} argument does not work when the \texttt{longtable} argument is used.
float.pos

This argument specifies where the table should be located on the page or in the document. By default, no floating position is specified, and LaTeX takes care of the position automatically. Possible values include "h" (here), "p" (page), "t" (top), "b" (bottom), any combination thereof, e.g., "tb", or any of these values followed by an exclamation mark, e.g. "t!", in order to enforce this position. The square brackets do not have to be specified.

Custom options to be passed on to the extract function. For example, most extract methods provide custom options for the inclusion or exclusion of specific goodness-of-fit statistics. See the help entries of extract for more information.

Details

The texreg function creates LaTeX code for inclusion in a LaTeX document or for usage with Sweave or knitr, based on a list of statistical models.

Value

A character object with a regression table and LaTeX markup. The object has an additional "texregTable" class identifier, which causes the object to be formatted nicely on screen when printed.

Author(s)

Philip Leifeld

References


See Also

texreg-package extract
Other texreg: htmlreg(), huxtablereg(), knitreg(), matrixreg(), plotreg(), screenreg(), wordreg()
table.string <- texreg(lm.D9, return.string = TRUE)
cat(table.string)

---

texreg-class

An S4 class to represent a statistical model as a texreg object

Description

An S4 class to represent a statistical model as a texreg object.

Details

A texreg object stores details about a statistical model. It can be used for creating regression tables using `screenreg`, `texreg`, and similar functions.

Slots

coeff.names The covariate names.
coeff The coefficients.
se The standard errors.
pvalues The p-values.
ci.low The lower bounds of the confidence intervals.
ci.up The upper bounds of the confidence intervals.
gof.names The names of the goodness-of-fit statistics.
gof The goodness-of-fit statistics.
gof.decimal A vector describing for each GOF statistic whether it is a decimal value (TRUE) or an integer value (FALSE).
model.name An optional model name. Can be of length zero.

Author(s)

Philip Leifeld

References


See Also

extract createTexreg
wordreg  

Export regression output to an MS Word file

Description
Export regression output to an MS Word file.

Usage
wordreg(
  l, 
  file = NULL, 
  single.row = FALSE, 
  stars = c(0.001, 0.01, 0.05), 
  custom.model.names = NULL, 
  custom.coef.names = NULL, 
  custom.coef.map = NULL, 
  custom.gof.names = NULL, 
  custom.gof.rows = NULL, 
  digits = 2, 
  leading.zero = TRUE, 
  star.symbol = star.symbol, 
  symbol = ".", 
  override.coef = 0, 
  override.se = 0, 
  override.pvalues = 0, 
  override.ci.low = 0, 
  override.ci.up = 0, 
  omit.coef = NULL, 
  reorder.coef = NULL, 
  reorder.gof = NULL, 
  ci.force = FALSE, 
  ci.force.level = 0.95, 
  ci.test = 0, 
  groups = NULL, 
  custom.columns = NULL, 
  custom.col.pos = NULL, 
...)

Arguments

l  A statistical model or a list of statistical models. Lists of models can be specified as `l = list(model.1, model.2, ...)`. Different object types can also be mixed.

file  Using this argument, the resulting table is written to a file rather than to the R prompt. The file name can be specified as a character string. Writing a table to a file can be useful for working with MS Office or LibreOffice. For example, using
the htmlreg function, an HTML table can be written to a file with the extension .doc and opened with MS Word. The table can then be simply copied into any Word document, retaining the formatting of the table. Note that LibreOffice can import only plain HTML; CSS decorations are not supported; the resulting tables do not retain the full formatting in LibreOffice.

**single.row**

By default, a model parameter takes up two lines of the table: the standard error is listed in parentheses under the coefficient. This saves a lot of horizontal space on the page and is the default table format in most academic journals. If single.row = TRUE is activated, however, both coefficient and standard error are placed in a single table cell in the same line.

**stars**

The significance levels to be used to draw stars. Between 0 and 4 threshold values can be provided as a numeric vector. For example, stars = numeric(0) will not print any stars and will not print any note about significance levels below the table. stars = 0.05 will attach one single star to all coefficients where the p-value is below 0.05. stars = c(0.001, 0.01, 0.05, 0.1) will print one, two, or three stars, or a symbol as specified by the symbol argument depending on the p-values.

**custom.model.names**

A character vector of labels for the models. By default, the models are named "Model 1", "Model 2", etc. Specifying model.names = c("My name 1", "My name 2") etc. overrides the default behavior.

**custom.coef.names**

By default, texreg uses the coefficient names which are stored in the models. The custom.coef.names argument can be used to replace them by other character strings in the order of appearance. For example, if a table shows a total of three different coefficients (including the intercept), the argument custom.coef.names = c("Intercept","variable 1","variable 2") will replace their names in this order. Sometimes it happens that the same variable has a different name in different models. In this case, the user can use this function to assign identical names. If possible, the rows will then be merged into a single row unless both rows contain values in the same column.

Where the argument contains an NA value, the original name of the coefficient is kept. For example, custom.coef.names = c(NA,"age",NA) will only replace the second coefficient name and leave the first and third name as they are in the original model.

See also custom.coef.map for an easier and more comprehensive way to rename, omit, and reorder coefficients.

**custom.coef.map**

The custom.coef.map argument can be used to select, omit, rename, and reorder coefficients.

Users must supply a named list of this form: list("x" = "First variable", "y" = NA, "z" = "Third variable"). With that particular example of custom.coef.map,

1. coefficients will be presented in order: "x", "y", "z".
2. variable "x" will appear as "First variable", variable "y" will appear as "y", and variable "z" will appear as "Third variable".
3. All variables not named "x", "y", or "z" will be omitted from the table.

custom.gof.names
A character vector which is used to replace the names of the goodness-of-fit statistics at the bottom of the table. The vector must have the same length as the number of GOF statistics in the final table. The argument works like the custom.coef.names argument, but for the GOF values. NA values can be included where the original GOF name should be kept.

custom.gof.rows
A named list of vectors for new lines at the beginning of the GOF block of the table. For example, list("Random effects" = c("YES","YES","NO"),Observations = c(25,25,26)) would insert two new rows into the table, at the beginning of the GOF block (i.e., after the coefficients). The rows can contain integer, numeric, or character objects. Note that this argument is processed after the custom.gof.names argument (meaning custom.gof.names should not include any of the new GOF rows) and before the reorder.gof argument (meaning that the new GOF order specified there should contain values for the new custom GOF rows). Arguments for custom columns are not affected because they only insert columns into the coefficient block.

digits
Set the number of decimal places for coefficients, standard errors and goodness-of-fit statistics. Do not use negative values! The argument works like the digits argument in the round function of the base package.

leading.zero
Most journals require leading zeros of coefficients and standard errors (for example, 0.35). This is also the default texreg behavior. Some journals, however, require omission of leading zeros (for example, .35). This can be achieved by setting leading.zero = FALSE.

star.symbol
Alternative characters for the significance stars can be specified. This is useful if knitr and Markdown are used for HTML report generation. In Markdown, asterisks or stars are interpreted as special characters, so they have to be escaped. To make a HTML table compatible with Markdown, specify star.symbol = "\*". Note that some other modifications are recommended for usage with knitr in combination with Markdown or HTML (see the inline.css, doctype, html.tag, head.tag, and body.tag arguments in the htmlreg function).

symbol
If four threshold values are handed over to the stars argument, p-values smaller than the largest threshold value but larger than the second-largest threshold value are denoted by this symbol. The default symbol is "\cdot" for the LaTeX dot, ";middot;" for the HTML dot, or simply "." for the ASCII dot. If the texreg function is used, any other mathematical LaTeX symbol or plain text symbol can be used, for example symbol = "\circ" for a small circle (note that backslashes must be escaped). If the htmlreg function is used, any other HTML character or symbol can be used. For the screenreg function, only plain text characters can be used.

override.coef
Set custom values for the coefficients. New coefficients are provided as a list of numeric vectors. The list contains vectors of coefficients for each model. There must be as many vectors of coefficients as there are models. For example, if there are two models with three model terms each, the argument could be specified as override.coef = list(c(0.1,0.2,0.3),c(0.05,0.06,0.07)).
If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.coef = c(0.05, 0.06, 0.07).

**override.se**

Set custom values for the standard errors. New standard errors are provided as a list of numeric vectors. The list contains vectors of standard errors for each model. There must be as many vectors of standard errors as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.se = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.se = c(0.05, 0.06, 0.07)`. Overriding standard errors can be useful for the implementation of robust SEs, for example.

**override.pvalues**

Set custom values for the p-values. New p-values are provided as a list of numeric vectors. The list contains vectors of p-values for each model. There must be as many vectors of p-values as there are models. For example, if there are two models with three coefficients each, the argument could be specified as `override.pvalues = list(c(0.1, 0.2, 0.3), c(0.05, 0.06, 0.07))`. If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.pvalues = c(0.05, 0.06, 0.07)`. Overriding p-values can be useful for the implementation of robust SEs and p-values, for example.

**override.ci.low**

Set custom lower confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.up` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

**override.ci.up**

Set custom upper confidence interval bounds. This works like the other override arguments, with one exception: if confidence intervals are provided here and in the `override.ci.low` argument, the standard errors and p-values as well as the `ci.force` argument are ignored.

**omit.coef**

A character string which is used as a regular expression to remove coefficient rows from the table. For example, `omit.coef = "group"` deletes all coefficient rows from the table where the name of the coefficient contains the character sequence "group". More complex regular expressions can be used to filter out several kinds of model terms, for example `omit.coef = "(thresh)|(ranef)"` to remove all model terms matching either "thresh" or "ranef". The `omit.coef` argument is processed after the custom.coef.names argument, so the regular expression should refer to the custom coefficient names. To omit GOF entries instead of coefficient entries, use the custom arguments of the extract functions instead (see the help entry of the `extract` function.

**reorder.coef**

Reorder the rows of the coefficient block of the resulting table in a custom way. The argument takes a vector of the same length as the number of coefficients. For example, if there are three coefficients, `reorder.coef = c(3, 2, 1)` will put the third coefficient in the first row and the first coefficient in the third row. Reordering can be sensible because interaction effects are often added to the end of the model output although they were specified earlier in the model formula. Note: Reordering takes place after processing custom coefficient names and af-
ter omitting coefficients, so the custom.coef.names and omit.coef arguments should follow the original order.

**reorder.gof**
Reorder the rows of the goodness-of-fit block of the resulting table in a custom way. The argument takes a vector of the same length as the number of GOF statistics. For example, if there are three goodness-of-fit rows, reorder.gof = c(3,2,1) will exchange the first and the third row. Note: Reordering takes place after processing custom GOF names and after adding new custom GOF rows, so the custom.gof.names and custom.gof.rows arguments should follow the original order, and the reorder.gof argument should contain values for any rows that are added through the custom.gof.rows argument.

**ci.force**
Should confidence intervals be used instead of the default standard errors and p-values? Most models implemented in the texreg package report standard errors and p-values by default while few models report confidence intervals. However, the functions in the texreg package can convert standard errors and into confidence intervals using z-scores if desired. To enforce confidence intervals instead of standard errors, the ci.force argument accepts either a logical value indicating whether all models or none of the models should be forced to report confidence intervals (ci.force = TRUE for all and ci.force = FALSE for none) or a vector of logical values indicating for each model separately whether the model should be forced to report confidence intervals (e.g., ci.force = c(FALSE,TRUE,FALSE)). Confidence intervals are computed using the standard normal distribution (z-values based on the qnorm function). The t-distribution is currently not supported because this would require each extract method to have an additional argument for the degrees of freedom.

**ci.force.level**
If the ci.force argument is used to convert standard errors to confidence intervals, what confidence level should be used? By default, 0.95 is used (i.e., an alpha value of 0.05).

**ci.test**
If confidence intervals are reported, the ci.test argument specifies the reference value to establish whether a coefficient/CI is significant. The default value ci.test = 0, for example, will attach a significance star to coefficients if the confidence interval does not contain 0. A value of ci.test = 1 could be useful if coefficients are provided on the odds-ratio scale, for example. If no star should be printed at all, ci.test = NA can be used. It is possible to provide a single value for all models or a vector with a separate value for each model. The ci.test argument works both for models with native support for confidence intervals and in cases where the ci.force argument is used.

**groups**
This argument can be used to group the rows of the table into blocks. For example, there could be one block for hypotheses and another block for control variables. Each group has a heading, and the row labels within a group are indented. The partitions must be handed over as a list of named numeric vectors, where each number is a row index and each name is the heading of the group. Example: groups = list("first group" = 1:4,"second group" = 7:8).

**custom.columns**
An optional list of additional text columns to be inserted into the coefficient block of the table, for example coefficient types. The list should contain one or more character vectors with as many character or numeric elements as there are coefficients/model terms. If the vectors in the list are named, the names are used as labels in the table header. For example, custom.columns = list(type...
= c("a","b","c"),1:3) will add two columns; the first one is labeled while
the second one is not. Note that the numeric elements of the second column will
be converted to character objects in this example. The consequence is that
decimal alignment with the dcolumn package is switched off in these columns.
Note that this argument is processed after any arguments that affect the number
of rows.

custom.col.pos  An optional integer vector of positions for the columns given in the custom.columns
argument. For example, if there are three custom columns, custom.col.pos =
c(1,3,3) will insert the first custom column before the first column of the origi-
nal table and the remaining two custom columns after the second column of the
original table. By default, all custom columns are placed after the first column,
which usually contains the coefficient names.

...  Custom options to be passed on to the extract function. For example, most
extract methods provide custom options for the inclusion or exclusion of specific
goodness-of-fit statistics. See the help entries of extract for more information.

Details

The wordreg function creates a Microsoft Word document with the requested table.

Author(s)

Vincent Arel-Bundock

See Also

texreg-package extract
Other texreg: htmlreg(), huxtablereg(), knitreg(), matrixreg(), plotreg(), screenreg(),
texreg

Examples

## Not run:
# Use models from ?lm:
ctl <- c(4.17, 5.58, 5.18, 6.11, 4.50, 4.61, 5.17, 4.53, 5.33, 5.14)
trt <- c(4.81, 4.17, 4.41, 3.59, 5.87, 3.83, 6.03, 4.89, 4.32, 4.69)
group <- gl(2, 10, 20, labels = c("Ctl", "Trt"))
weight <- c(ctl, trt)
lm.D9 <- lm(weight ~ group)
lm.D90 <- lm(weight ~ group - 1)
wordreg(list(lm.D9, lm.D90), file = "testfile.doc")
unlink("testfile.doc")

## End(Not run)
Index

*Topic IO
texreg, 130

*Topic misc
texreg, 130

*Topic print
texreg, 130

*Topic utilities
texreg, 130

aftreg, 8
Arima, 30
arima, 9
as.character, 121

bam, 10
betamfx, 11
betaor, 12
betareg, 13
bife, 13
biglm, 14
brglm, 15
brm, 15
broom, 9
btergm, 17

cat, 123
censReg, 17
clm, 18, 78
clmm, 19
clogit, 20
coefftest, 20
confint.merMod, 37, 50, 51, 57, 66
content, 121
coxph, 21, 22
coxreg, 23
createTexreg, 5, 8, 130, 139
dredge, 60
dynlm, 23

ergmm, 25
etr, 25
extract, aftreg-method, 8
extract, ANY-method, 9
extract, Arima-method, 9
extract, averaging-method, 10
extract, bam-method, 10
extract, betamfx-method, 11
extract, betaor-method, 12
extract, betareg-method, 13
extract, bife-method, 13
extract, biglm-method, 14
extract, brglm-method, 15
extract, brmsfit-method, 15
extract, btergm-method, 17
extract, censReg-method, 17
extract, clm-method, 18
extract, clmm-method, 19
extract, clogit-method, 20
extract, coeftest-method, 20
extract, coxph-method, 21
extract, coxph.penal-method, 22
extract, coxreg-method, 23
extract, dynlm-method, 23
extract, ergmm-method, 24
extract, ergmm-method, 25
extract, ets-method, 25
extract, feglm-method, 26
extract, feis-method, 27
extract, felm-method, 28
extract, fGARCH-method, 29
extract, forecast-method, 29
extract, forecast_ARIMA-method, 30
extract, gam-method, 30
extract, gamlss-method, 32

146
extract, gamlssZadj-method, 32
extract, gee-method, 33
extract, geeglm-method, 34
extract, gel-method, 34
extract, glm-method, 35
extract, glm.cluster-method, 36
extract, glmerMod-method, 37
extract, glmmadmb-method, 38
extract, glmmPQL-method, 39
extract, glmmTMB-method, 39
extract, glmmrob-method, 41
extract, gts-method, 41
extract, gmm-method, 42
extract, gnls-method, 42
extract, gnm-method, 43
extract, H2OBinomialModel-method, 44
extract, hurdle-method, 45
extract, ivreg-method, 46
extract, lm-method, 47
extract, lm.cluster-method, 47
extract, lme-method, 48
extract, lme4-method, 49
extract, lmerMod-method, 50
extract, lmRob-method
(extract, lmrob-method), 51
extract, lmrob-method, 51
extract, lnam-method, 52
extract, logitmfx-method, 53
extract, logitor-method, 54
extract, lqmm-method, 54
extract, lrm-method, 55
extract, maBina-method, 56
extract, merMod-method, 56
extract, mhurdle-method, 58
extract, mlogit-method, 58
extract, mnlogit-method, 59
extract, model.selection-method, 60
extract, mtergm-method, 61
extract, multinom-method, 61
extract, negbin-method, 62
extract, negbinirr-method, 63
extract, negbinmfx-method, 64
extract, netlogit-method, 64
extract, nlme-method, 65
extract, nlmerMod-method, 66
extract, oglmx-method, 67
extract, ols-method, 68
extract, panelAR-method, 69
extract, pglm-method, 69
extract, pgmm-method, 70
extract, phreg-method, 71
extract, plm-method, 71
extract, pmg-method, 72
extract, poissonirr-method, 73
extract, poissonmfx-method, 73
extract, polr-method, 74
extract, probitmfx-method, 75
extract, rem.d1yad-method, 76
extract, rlm-method, 76
extract, rq-method, 77
extract, sarlm-method, 77
extract, sclm-method, 78
extract, selection-method, 79
extract, sienaFit-method, 80
extract, simex-method, 80
extract, speedglm-method, 81
extract, speedlm-method, 81
extract, stergm-method, 82
extract, summary.lm-method, 83
extract, survreg-method, 84
extract, survreg.penal-method, 85
extract, svyglm-method, 85
extract, systemfit-method, 86
extract, texreg-method, 87
extract, tobit-method, 88
extract, vglm-method, 89
extract, weibreg-method, 89
extract, wls-method, 90
extract, Zelig-method
(extract, zelig-method), 91
extract, zelig-method, 91
extract, zeroimfl-method, 92
extract.aftreg (extract, aftreg-method), 92
extract.ANY (extract, ANY-method), 93
extract.ANY-method
(extract, ANY-method), 93
extract.Arima (extract, Arima-method), 93
extract.averaging
(extract, averaging-method), 93
extract.bam (extract, bam-method), 93
extract.betafx
(extract, betamfx-method), 94
extract.betaor (extract, betaor-method), 94
extract.betaor
extract.betareg (extract, betareg-method), 13
extract.bife (extract, bife-method), 13
extract.biglm (extract, biglm-method), 14
extract.brglm (extract, brglm-method), 15
extract.brmsfit (extract, brmsfit-method), 15
extract.broom (extract, ANY-method), 9
extract.btergm (extract, btergm-method), 17
extract.censReg (extract, censReg-method), 17
extract.clm (extract, clm-method), 18
extract.clmm (extract, clmm-method), 19
extract.clogit (extract, clogit-method), 20
extract.coeftest (extract, coeftest-method), 20
extract.coxph (extract, coxph-method), 21
extract.coxph.penalty (extract, coxph.penalty-method), 22
extract.coxreg (extract, coxreg-method), 23
extract.dynlm (extract, dynlm-method), 23
extract.ergm (extract, ergm-method), 24
extract.ergm2mm (extract, ergm2mm-method), 25
extract.ets (extract, ets-method), 25
extract.feglm (extract, feglm-method), 26
extract.feis (extract, feis-method), 27
extract.felm (extract, felm-method), 28
extract.fGARCH (extract, fGARCH-method), 29
extract.forecast (extract, forecast-method), 29
extract.forecast_ARIMA (extract, forecast_ARIMA-method), 30
extract.gam (extract, gam-method), 30
extract.gamlss (extract, gamlss-method), 32
extract.gamlssZadj (extract, gamlssZadj-method), 32
extract.gee (extract, gee-method), 33
extract.geeglm (extract, geeglm-method), 34
extract.gel (extract, gel-method), 34
extract glm (extract, glm-method), 35
extract.glm.cluster (extract, glm.cluster-method), 36
extract.glmerMod (extract, glmerMod-method), 37
extract.glmmadmb (extract, glmmadmb-method), 38
extract.glmmPQL (extract, glmmPQL-method), 39
extract.glmmTMB (extract, glmmTMB-method), 39
extract.glmrob (extract, glmrob-method), 41
extract.gls (extract, gls-method), 41
extract.gmm (extract, gmm-method), 42
extract.gnls (extract, gnls-method), 42
extract.gnm (extract, gnm-method), 43
extract.H2OBinomialModel (extract, H2OBinomialModel-method), 44
extract.hurdle (extract, hurdle-method), 45
extract.ivreg (extract, ivreg-method), 46
extract.lm (extract, lm-method), 47
extract.lm.cluster (extract, lm.cluster-method), 47
extract.lme (extract, lme-method), 48
extract.lme4 (extract, lme4-method), 49
extract.lmerMod (extract, lmerMod-method), 50
extract.lmRob (extract, lmRob-method), 51
extract.lmrob (extract, lmrob-method), 51
extract.lnam (extract, lnam-method), 52
extract.logitmfx (extract, logitmfx-method), 53
extract.logitor (extract, logitor-method), 54
extract.lqmm (extract, lqmm-method), 54
extract.lrm (extract, lrm-method), 55
extract.maBina (extract, maBina-method), 56
extract.merMod (extract, merMod-method), 56
extract.mhurdle (extract, mhurdle-method), 58
extract.mlogit (extract, mlogit-method), 58
extract.mnlogit (extract, mnlogit-method), 59
extract.model.selection
(extract, model.selection-method), 60
extract.mtergm(extract,mtergm-method), 61
extract.multinom
(extract,multinom-method), 61
extract.negbin (extract,negbin-method), 62
extract.negbinirr
(extract,negbinirr-method), 63
extract.negbinmfx
(extract,negbinmfx-method), 64
extract.netlogit
(extract,netlogit-method), 64
extract.nlmerMod
(extract,nlmerMod-method), 66
extract.glmx (extract,glmx-method), 67
extract.ols (extract,ols-method), 68
extract.panelAR
(extract,panelAR-method), 69
extract.plml (extract,plm-method), 71
extract.pmg (extract,pmg-method), 72
extract.poissonirr
(extract,poissonirr-method), 73
extract.poissonmfx
(extract,poissonmfx-method), 73
extract.poissonmfx
(extract,poissonmfx-method), 73
extract.polr (extract,polr-method), 74
extract.probitmfx
(extract,probitmfx-method), 75
extract.rem.dyad
(extract,rem.dyad-method), 76
extract.rlm (extract,rlm-method), 76
extract.rq (extract,rq-method), 77
extract.sarlm (extract,sarlm-method), 77
extract.sclm (extract,sclm-method), 78
extract.selection
(extract,selection-method), 79
extract.sienaFit
(extract,sienaFit-method), 80
extract.simex (extract,simex-method), 80
extract.speedglm
(extract,speedglm-method), 81
extract.speedlm
(extract,speedlm-method), 81
extract.stergm(extract,stergm-method), 82
extract.summary.lm
(extract,summary.lm-method), 83
extract.survreg
(extract,survreg-method), 84
extract.survreg.penal
(extract,survreg.penal-method), 85
extract.svyglm (extract,svyglm-method), 85
extract.systemfit
(extract,systemfit-method), 86
extract.texreg (extract, texreg-method), 87
extract.tobit (extract,tobit-method), 88
extract.vglm (extract,vglm-method), 89
extract.weibreg
(extract,weibreg-method), 89
extract.wls (extract,wls-method), 90
extract.Zelig (extract,zelig-method), 91
extract.Zelig (extract,zelig-method), 91
extract.zeroinfl
(extract,zeroinfl-method), 92

feedback (praise), 119
feglm, 26
feis, 27
felm, 28
fixef, 38, 50, 51, 57, 67
forecast, 29

gam, 30
gamlss, 32
gamlssZadj, 32
garchFit, 29
gee, 33
geeGlm, 34
gel, 34
ggtheme, 118
glm, 35
glm.cluster, 36
glm.nb, 62
glmer, 37
glmmPQL, 39
glmmTMB, 39
glmRob, 41
gls, 41
gmm, 42
gnls, 42
gnm, 43
h2o.glm, 44
holt, 29
HPDinterval, 16
hurdle, 45
huxreg, 106
huxtable, 106
ivreg, 46
knitreg, 100, 106, 107, 107, 114, 119, 129, 138, 145
lagsarlm, 77
lm, 47
lm.cluster, 47
lme, 48
lmer, 50
lmRob, 51
lmrob, 51
lnam, 52
logitmfx, 53
logitor, 54
lqmm, 54
lrm, 55
maBina, 56
mhurdle, 58
mlogit, 58
mnlogit, 59
model.avg, 10
model.sel, 60
mtergm, 61
multinom, 61
negbinirr, 63
negbinmfx, 64
netlogit, 64
nlme, 65
nlmer, 66
oglmx, 67
ols, 68
panelAR, 69
pglm, 69
pgmm, 70
phreg, 71
plm, 71
pmg, 72
poissonirr, 73
poissonmfx, 73
polr, 74
praise, 119, 121, 122
praise_interactive, 121, 122
praise_interactive (praise), 119
print.texregTable, 122
probitmfx, 75
qnorm, 98, 105, 112, 128, 135, 144
reloo.brmsfit, 16
rem.dyad, 76
rlm, 76
round, 96, 103, 110, 126, 134, 142
rq, 77
screenreg, 8, 100, 106–108, 114, 118–120, 123, 129, 130, 138, 139, 145
selection, 79
show, texreg-method, 130
siena07, 80
simex, 80
speedglm, 81
speedlm, 81
stergm, 82
summary, 129
summary.lm, 83
survreg, 84, 85
svyglm, 85
systemfit, 86
texreg-class, 139
texreg-package, 4
tobit, 88
INDEX

vglm, 89

weibreg, 89
wls, 90

zelig, 91
zeroinfl, 92