Package 'texreg'

July 21, 2014

Version 1.33

Date 2014-07-21

Title Conversion of R regression output to LaTeX or HTML tables.

Author Philip Leifeld

Maintainer Philip Leifeld <philip.leifeld@uni-konstanz.de>

Description texreg 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.

Suggests nlme, survival, network, ergm, lme4 (>= 1.0), gamlss

Depends R (>= 2.15.0)

Imports methods

License GPL-2 | GPL-3

NeedsCompilation no

Repository CRAN

Date/Publication 2014-07-21 20:08:05

R topics documented:

texreg-package *Conversion of statistical model output in R to LaTeX and HTML tables*

Description

texreg 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.

Details

Several packages like apsrtable, memisc, outreg, stargazer, or xtable are available for typesetting R regression output as LaTeX tables. However, texreg supports more flexible handling of new model types, supports multiple models side-by-side, supports confidence intervals and standard errors alike, has many options for customization, and beside LaTeX output, it can also print tables to the R console screen, save them as HTML or MS Word documents, or plot them as coefficient plots. If several models are submitted, they are merged by row labels of the coefficients, and they are inserted into the final table as separate columns. The package works with report generation tools like Sweave or knitr (including options for LaTeX, HTML, and Markdown). To display citation information, execute `citation("texreg")`.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

See Also

texreg plotreg

coeftostring *Convert coefficients into standardized strings*

Description

Convert coefficients into standardized strings.

Usage

```
coeftostring(x, lead.zero = FALSE, digits = 2)
```

Arguments

x	A numeric object, for example a coefficient resulting from a regression model.
lead.zero	If the number starts with "0" or "-0": should the zero be retained or removed? If true, the leading zero is kept.
digits	The number of decimal places to be used.

Details

This function converts numbers into strings and standardizes them according to some simple rules. The function is used by the texreg function.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

See Also

texreg-package texreg

Examples

```
coeftostring(-0.345, lead.zero = FALSE)  # this should give "-.34"
```

---

createTexreg *Create a texreg object*

Description

Create a texreg object with coefficients and GOF statistics.

Usage

```
createTexreg(coef.names, coef, se, 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

coef.names	A vector of coefficient names.
coef	The coefficient values.
se	The standard errors. This is optional if the ci.low and ci.up slots are filled.
pvalues	The p-values of the model. This is optional.
ci.low	Lower bound of confidence interval (the actual values, not the confidence level). This is optional as long as se is available, but if it is provided, ci.up must also be provided.

ci.up	Upper bound of confidence interval (the actual values, not the confidence level). This is optional as long as se is available, but if it is provided, ci.low must also be provided.
gof.names	A vector of names of the goodness-of-fit statistics.
gof	A vector of goodness-of-fit statistics.
gof.decimal	A vector of boolean/logical values indicating for each GOF statistic if decimal places shall be used. This is optional.
model.name	The name of the model. In some cases, models consist of two separate columns because two separate data-generating processes are modeled. In these cases, it may make sense to specify default names for the columns (that is, for each texreg object). This argument is optional.

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 command to create LaTeX tables and other representations of the model results.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

References

Leifeld, Philip (2013). texreg: Conversion of Statistical Model Output in R to LaTeX and HTML Tables. Journal of Statistical Software, 55(8), 1-24. http://www.jstatsoft.org/v55/i08/.

See Also

texreg-package texreg

Examples

```
library(nlme)  #load library for fitting linear mixed effects models
model <- lme(distance ~ age, data = Orthodont, random = ~ 1)  #estimate model
coefficient.names <- rownames(summary(model)$tTable)  #extract coefficient names
coefficients <- summary(model)$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)  #the 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 coefficients and GOF measures from a statistical object*

Description

Extract coefficients and GOF measures from a statistical object.

Usage

```
extract(model, ...)

extract.aftreg(model, include.loglik = TRUE, include.lr = TRUE,
    include.nobs = TRUE, include.events = TRUE, include.trisk = TRUE,
    ...)

extract.Arima(model, include.pvalues = FALSE, include.aic = TRUE,
    include.loglik = TRUE, ...)

extract.betareg(model, include.precision = TRUE,
    include.pseudors = TRUE, include.loglik = TRUE,
    include.nobs = TRUE, ...)

extract.brglm(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, ...)

extract.btergm(model, conf.level = 0.95, ...)

extract.clm(model, include.thresholds = TRUE, include.aic = TRUE,
    include.bic=TRUE, include.loglik = TRUE, include.nobs = TRUE,
    ...)

extract.clogit(model, include.aic = TRUE, include.rsquared = TRUE,
    include.maxrs = TRUE, include.events = TRUE,
    include.nobs = TRUE, include.missings = TRUE, ...)

extract.coeftest(model, ...)
```

```
extract.coxph(model, include.aic = TRUE, include.rsquared = TRUE,
    include.maxrs=TRUE, include.events = TRUE,
    include.nobs = TRUE, include.missings = TRUE,
    include.zph = TRUE, ...)

extract.coxph.penal(model, include.aic = TRUE,
    include.rsquared = TRUE, include.maxrs = TRUE,
    include.events = TRUE, include.nobs = TRUE,
    include.missings = TRUE, include.zph = TRUE, ...)

extract.dynlm(model, include.rsquared = TRUE, include.adjrs = TRUE,
    include.nobs = TRUE, include.fstatistic = FALSE, ...)

extract.ergm(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, ...)

extract.fGARCH(model, include.nobs = TRUE, include.aic = TRUE,
    include.loglik = TRUE, ...)

extract.gam(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, ...)

extract.gamlss(model, robust = FALSE, include.nobs = TRUE,
    include.nagelkerke = TRUE, include.gaic = TRUE, ...)

extract.gee(model, robust = TRUE, include.dispersion = TRUE,
    include.nobs = TRUE, ...)

extract.glm(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, ...)

extract.glmerMod(model, naive = FALSE, nsim = 1000,
    conf.level = 0.95, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, include.groups = TRUE,
    include.variance = TRUE, ...)

extract.glmmadmb(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, ...)
```

```
extract.glmrob(model, include.nobs = TRUE, ...)

extract.gls(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.nobs = TRUE, ...)

extract.gmm(model, include.obj.fcn = TRUE,
    include.overidentification = FALSE, include.nobs = TRUE, ...)

extract.hurdle(model, beside = FALSE, include.count = TRUE,
    include.zero = TRUE, include.aic = TRUE, include.loglik = TRUE,
    include.nobs = TRUE, ...)

extract.ivreg(model, include.rsquared = TRUE, include.adjrs = TRUE,
    include.nobs = TRUE, include.fstatistic = FALSE, ...)

extract.lm(model, include.rsquared = TRUE, include.adjrs = TRUE,
    include.nobs = TRUE, include.fstatistic = FALSE, ...)

extract.lme(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.nobs = TRUE,
    include.groups = TRUE, include.variance = FALSE, ...)

extract.lme4(model, naive = FALSE, nsim = 1000,
    conf.level = 0.95, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, include.groups = TRUE,
    include.variance = TRUE, ...)

extract.lmerMod(model, naive = FALSE, nsim = 1000,
    conf.level = 0.95, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, include.groups = TRUE,
    include.variance = TRUE, ...)

extract.lmrob(model, include.nobs = TRUE, ...)

extract.lnam(model, include.rsquared = TRUE, include.adjrs = TRUE,
    include.aic = TRUE, include.bic = TRUE, include.loglik = TRUE,
    ...)

extract.lrm(model, include.pseudors = TRUE, include.lr = TRUE,
    include.nobs = TRUE, ...)

extract.maBina(model, ...)

extract.mer(model, naive = FALSE, nsim = 1000,
    conf.level = 0.95, include.aic = TRUE, include.bic = TRUE,
```

```
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, include.groups = TRUE,
    include.variance = TRUE, ...)

extract.mnlogit(model, include.aic = TRUE, include.loglik = TRUE,
    include.nobs = TRUE, include.groups = TRUE,
    include.intercept = TRUE, include.iterations = FALSE, ...)

extract.multinom(model, include.pvalues = TRUE, include.aic = TRUE,
    include.bic = TRUE, include.loglik = TRUE,
    include.deviance = TRUE, include.nobs = TRUE, ...)

extract.negbin(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, ...)

extract.nlme(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.nobs = TRUE,
    include.groups = TRUE, include.variance = FALSE, ...)

extract.nlmerMod(model, naive = FALSE, nsim = 1000,
    conf.level = 0.95, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, include.groups = TRUE,
    include.variance = TRUE, ...)

extract.ols(model, include.nobs = TRUE, include.rsquared = TRUE,
    include.adjrs = TRUE, include.fstatistic = FALSE,
    include.lr = TRUE, ...)

extract.pgmm(model, include.nobs = TRUE, include.sargan = TRUE,
    include.wald = TRUE, ...)

extract.phreg(model, include.loglik = TRUE, include.lr = TRUE,
    include.nobs = TRUE, include.events = TRUE, include.trisk = TRUE,
    ...)

extract.plm(model, include.rsquared = TRUE, include.adjrs = TRUE,
    include.nobs = TRUE, ...)

extract.pmg(model, include.nobs = TRUE, ...)

extract.polr(model, include.thresholds = FALSE, include.aic = TRUE,
    include.bic = TRUE, include.loglik = TRUE,
    include.deviance = TRUE, include.nobs = TRUE, ...)

extract.rem.dyad(model, include.nvertices = TRUE,
    include.events = TRUE, include.aic = TRUE, include.aicc = TRUE,
```

```
    include.bic = TRUE, ...)

extract.rlm(model, include.nobs = TRUE, ...)

extract.rq(model, include.nobs = TRUE, include.percentile = TRUE,
    ...)

extract.sarlm(model, include.nobs = TRUE, include.lambda = TRUE,
    include.aic = TRUE, include.loglik = TRUE, include.wald = TRUE, ...)

extract.sclm(model, include.thresholds = TRUE, include.aic = TRUE,
    include.bic = TRUE, include.loglik = TRUE, include.nobs = TRUE,
    ...)

extract.sienaFit(model, include.iterations = TRUE, ...)

extract.simex(model, jackknife = TRUE, include.nobs = TRUE, ...)

extract.stergm(model, beside = FALSE, include.formation = TRUE,
    include.dissolution = TRUE, include.nvertices = TRUE,
    include.aic = FALSE, include.bic = FALSE,
    include.loglik = FALSE, ...)

extract.survreg(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = TRUE, ...)

extract.survreg.penal(model, include.aic = TRUE,
    include.bic = TRUE, include.loglik = TRUE,
    include.deviance = TRUE, include.nobs = TRUE, ...)

extract.svyglm(model, include.aic = FALSE, include.bic = FALSE,
    include.loglik = FALSE, include.deviance = TRUE,
    include.dispersion = TRUE, include.nobs = TRUE, ...)

extract.systemfit(model, include.rsquared = TRUE,
    include.adjrs = TRUE, include.nobs = TRUE, ...)

extract.texreg(model, ...)

extract.tobit(model, include.aic = TRUE, include.bic = TRUE,
    include.loglik = TRUE, include.deviance = TRUE,
    include.nobs = FALSE, include.censnobs = TRUE, include.wald=TRUE,
    ...)

extract.weibreg(model, include.loglik = TRUE, include.lr = TRUE,
    include.nobs = TRUE, include.events = TRUE, include.trisk = TRUE,
    ...)
```

```
extract.zelig(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,
    ...)

extract.zeroinfl(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`	If available: should the model terms be arranged below each other or beside each other (default)? For example, in a stergm model, the formation and dissolution coefficients can be arranged in two columns of the table.
`conf.level`	Confidence level (`1 - alpha`) for computing confidence intervals.
`include.adjrs`	If available: should the adjusted R-squared be reported?
`include.aic`	If available: should Akaike's information criterion (AIC) be reported?
`include.aicc`	If available: should AICc be reported? This is a version of AIC with a correction for finite sample sizes.
`include.bic`	If available: should the Bayesian information criterion (BIC) be reported?
`include.censnobs`	If available: should the total, right-censored, left-censored, and uncensored number of observations be reported?
`include.count`	If available: should the count model of a zero-inflated or hurdle regression be included in the coefficients block (before the zero-inflation or zero hurdle model)?
`include.dev.expl`	If available: should the deviance explained be reported?
`include.deviance`	If available: should the deviance be reported?
`include.dispersion`	If available: should the dispersion or scale parameter be reported?
`include.dissolution`	If available: should the coefficients for the dissolution phase in a STERGM be reported?
`include.events`	If available: should the number of events be reported (in survival models)?
`include.formation`	If available: should the coefficients for the formation phase in a STERGM be reported?
`include.fstatistic`	If available: should the F statistic be reported?
`include.gaic`	If available: should the Generalized Akaike's information criterion (GAIC) be reported?

include.gcv If available: should the GCV score be reported (in GAMs)?

include.groups If available: should the number of groups in a mixed-effects model (or k alternatives in a multinomial choice model) be reported?

include.intercept
If available: should the intercept be included in the GOF block?

include.iterations
If available: should the number of iterations be included?

include.lambda If available: should the lambda statistic and p-value be reported?

include.loglik If available: should the log-likelihood be reported?

include.lr If available: should the likelihood ratio test be reported?

include.maxrs If available: should the maximum possible R-squared be reported?

include.missings
If available: should the number of missing observations be reported (in survival models)?

include.nagelkerke
If available: should Nagelkerke's R-squared be reported?

include.nobs If available: should the number of observations be reported?

include.nsmooth
If available: should the number of smooth terms be reported (in GAMs)?

include.nvertices
If available: should the number of vertices be reported in a statistical network model?

include.obj.fcn
If available: should the value of the objective function (= criterion function) be reported (for gmm objects)? More precisely, this returns `E(g)var(g)^{-1}E(g)`.

include.overidentification
If available: should the J-test for overidentification be reported (for gmm objects)?

include.percentile
If available: should the percentile (tau) be reported?

include.precision
If available: should the precision estimates of a betareg fit (the phi coefficients) be reported as part of the coefficients block?

include.pseudors
If available: should the pseudo R-squared be reported?

include.pvalues
If available: should the p values be reported (naive p values are not recommended for lme4 models, but see also the `mcmc.pvalues` argument)?

include.rsquared
If available: should R-squared be reported?

include.sargan If available: should the Sargan test be reported?

include.smooth If available: should the smooth terms of a GAM be reported? 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.thresholds`
: If available: should the threshold parameters (that is, the intercepts for the class boundaries) be reported in ordinal models?

`include.trisk`
: If available: should the total time at risk be reported (in event-history models)?

`include.variance`
: If available: should group variances be reported?

`include.wald`
: If available: should the Wald statistic be included?

`include.zero`
: If available: should the zero-inflation model of a zero-inflated regression or the zero hurdle model of a hurdle regression be included in the coefficients block (after the count model)?

`include.zph`
: If available: should the Cox proportional hazards assumption be tested (resulting in a p value indicating whether the proportional hazards assumption of the model is violated)?

`jackknife`
: If available: use Jackknife variance instead of Asymptotic variance.

`naive`
: If available: use naive p values and standard errors.

`nsim`
: In linear mixed effects models: the MCMC sample size from which confidence intervals are derived (for old versions of the **lme4** package), or the number of simulations on which bootstrapped confidence intervals should be based (for newer versions of the **lme4** package; only if the argument `method = "boot"` is provided, otherwise CIs are based on profile likelihood). Note: high values may take considerable computing time.

`robust`
: If available: report robust instead of naive standard errors.

`...`
: Custom parameters.

Details

extract() is a generic function which extracts coefficients and GOF measures from statistical objects. There are several extract functions for the specific model types, which are called by the generic extract function if it encounters a model known to be handled by the specific function. The output is used by the texreg function.

The various extract functions can also be used directly on a statistical model to convert them into texreg objects.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

References

Leifeld, Philip (2013). texreg: Conversion of Statistical Model Output in R to LaTeX and HTML Tables. Journal of Statistical Software, 55(8), 1-24. http://www.jstatsoft.org/v55/i08/.

See Also

texreg-package texreg extract-methods

`extract-methods` *Methods for Function* `extract` *in Package* **texreg**

Description

Methods for function `extract` in package **texreg**.

Methods

`extract.aftreg` An extract function for aftreg objects from the **eha** package.

`extract.Arima` An extract function for Arima objects from the **stats** package.

`extract.betareg` An extract function for betareg objects from the **betareg** package.

`extract.brglm` An extract function for brglm objects from the **brglm** package.

`extract.btergm` An extract function for btergm objects from the **btergm** package.

`extract.clm` An extract function for clm objects from the **ordinal** package.

`extract.clogit` An extract function for clogit objects from the **survival** package.

`extract.coeftest` An extract function for coeftest objects from the **lmtest** package.

`extract.coxph` An extract function for coxph objects from the **survival** package.

`extract.coxph.penal` An extract function for coxph.penal objects from the **survival** package.

`extract.dynlm` An extract function for dynlm objects from the **dynlm** package.

`extract.ergm` An extract function for ergm objects from the **ergm** package.

`extract.fGARCH` An extract function for fGARCH objects from the **fGarch** package.

`extract.gam` An extract function for gam objects from the **mgcv** package.

`extract.gamlss` An extract function for gamlss objects from the **gamlss** package.

`extract.gee` An extract function for gee objects from the **gee** package.

`extract.glm` An extract function for glm objects from the **stats** package.

`extract.glmerMod` An extract function for glmerMod objects from the (old) **lme4** package.

`extract.glmmadmb` An extract function for glmmadmb objects from the **glmmADMB** package.

`extract.glmrob` An extract function for glmrob objects from the **robustbase** package.

`extract.gls` An extract function for gls objects from the **nlme** package.

`extract.gmm` An extract function for gmm objects from the **gmm** package.

`extract.ivreg` An extract function for ivreg objects from the **AER** package.

`extract.hurdle` An extract function for hurdle objects from the **pscl** package.

`extract.lm` An extract function for lm objects from the **stats** package.

`extract.lme` An extract function for lme objects from the **nlme** package.

`extract.lme4` An extract function for lme4 objects from the **lme4** package.

`extract.lmerMod` An extract function for lmerMod objects from the (old) **lme4** package.

`extract.lmrob` An extract function for lmrob objects from the **robustbase** package.

`extract.lnam` An extract function for lnam objects from the **sna** package.

`extract.lrm` An extract function for lrm objects from the **Design** or **rms** package.

`extract.maBina` An extract function for maBina objects from the **erer** package.

`extract.mer` An extract function for mer objects from the (old) **lme4** package.

`extract.mnlogit` An extract function for mnlogit objects from the **mnlogit** package.

`extract.multinom` An extract function for multinom objects from the **nnet** package.

`extract.negbin` An extract function for negbin objects from the **MASS** package.

`extract.nlme` An extract function for nlme objects from the **nlme** package.

`extract.nlmerMod` An extract function for nlmerMod objects from the (old) **lme4** package.

`extract.ols` An extract function for ols objects from the **rms** package.

`extract.pgmm` An extract function for pgmm objects from the **plm** package.

`extract.phreg` An extract function for phreg objects from the **eha** package.

`extract.plm` An extract function for plm objects from the **plm** package.

`extract.pmg` An extract function for pmg objects from the **plm** package.

`extract.polr` An extract function for polr objects from the **MASS** package.

`extract.rem.dyad` An extract function for rem.dyad objects from the **relevent** package.

`extract.rlm` An extract function for rlm objects from the **MASS** package.

`extract.rq` An extract function for rq objects from the **quantreg** package.

`extract.sarlm` An extract function for sarlm objects from the **spdep** package.

`extract.sclm` An extract function for sclm objects from the **ordinal** package.

`extract.sienaFit` An extract function for sienaFit objects from the **RSiena** package.

`extract.simex` An extract function for simex objects from the **simex** package.

`extract.stergm` An extract function for stergm objects from the **tergm** package.

`extract.survreg` An extract function for survreg objects from the **survival** package.

`extract.survreg.penal` An extract function for survreg.penal objects from the **survival** package.

`extract.svyglm` An extract function for svyglm objects from the **survey** package.

`extract.systemfit` An extract function for systemfit objects from the **systemfit** package.

`extract.texreg` An extract function for texreg objects from the **texreg** package. The purpose is to allow for easy manipulation of the output. `texreg` objects can be created using the createTexreg function or using the extract function. After manipulating the object, it can be handed back to the screenreg, texreg, or htmlreg functions for creating a table.

`extract.tobit` An extract function for tobit objects from the **AER** package.

`extract.weibreg` An extract function for weibreg objects from the **eha** package.

`extract.zelig` An extract function for zelig objects from the **Zelig** package. Has been tested with ls, logit and relogit models as implemented in the **Zelig** package.

`extract.zeroinfl` An extract function for zeroinfl objects from the **pscl** package.

References

Leifeld, Philip (2013). texreg: Conversion of Statistical Model Output in R to LaTeX and HTML Tables. Journal of Statistical Software, 55(8), 1-24. http://www.jstatsoft.org/v55/i08/.

See Also

texreg-package texreg extract

---

plotreg *Create coefficient plots from statistical model output*

Description

Create coefficient plots from statistical model output.

Usage

```
plotreg(l, file = NULL, custom.model.names = NULL,
    custom.coef.names = 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,
    use.se = FALSE, mfrow = TRUE, xlim = NULL, cex = 2.5,
    lwd.zerobar = 4, lwd.vbars = 1, lwd.inner = 7,
    lwd.outer = 5, signif.light = "#fbc9b9",
    signif.medium = "#f7523a", signif.dark = "#bd0017",
    insignif.light = "#c5dbe9", insignif.medium = "#5a9ecc",
    insignif.dark = "#1c5ba6", ...)

coefplot(labels, estimates, lower.inner = NULL,
    upper.inner = NULL, lower.outer = NULL,
    upper.outer = NULL, signif.outer = TRUE,
    xlab = "Coefficients and confidence intervals",
    main = "Coefficient plot", xlim = NULL,
    cex = 2.5, lwd.zerobar = 4, lwd.vbars = 1,
    lwd.inner = 7, lwd.outer = 5, signif.light = "#fbc9b9",
    signif.medium = "#f7523a", signif.dark = "#bd0017",
    insignif.light = "#c5dbe9", insignif.medium = "#5a9ecc",
    insignif.dark = "#1c5ba6", ...)
```

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. The file extension is automatically recognized. `pdf`, `ps`, `png`, `bmp`, `jpg`, and `tiff` are supported.

`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, `plotreg` 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 model shows a total of three coefficients (including the intercept), the argument `custom.coef.names = c("Intercept", "vari` will replace their names in this order.

`custom.note` With this argument, a replacement text for the `xlab` note below the diagram can be provided. If an empty character object is provided (`custom.note = ""`), the note will be omitted completely.

`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. This only has an effect where standard errors are converted into confidence intervals because no other CIs are present. 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(` 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.pval` Set custom values for the p values. This only has an effect where standard errors are converted into confidence intervals because no other CIs are present. In this case, significance is derived from the p values rather than the confidence intervals. 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.pval = list(c(0.1, 0.2, 0.3), c(0.05, 0.06,` If there is only one model, custom values can be provided as a plain vector (not embedded in a list). For example: `override.pval = 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 diagram 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.

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).

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 function.

mfrow If multiple models are handed over as the l argument, several plots are produced. If mfrow = TRUE is set, multiple diagrams are aligned on the same page. If mfrow = FALSE is set, each diagram per model comes out as a separate plot.

xlim Horizontal limits. In the coefplot function, they must be provided as a vector with two numeric, e.g., xlim = c(-5, 5) for displaying a range from -5 to +5. In the plotreg function, they can be provided either as such a vector with two values or as a list of vectors (with each entry corresponding to a model in l).

lwd.zerobar Line width of the vertical gray bar at the x value of 0. To remove the line, set lwd.zerobar = 0.

lwd.vbars Line width of the thin vertical gray bars. To remove them completely, set lwd.vbars = 0.

labels The names of the model terms. They are arranged on the left axis.

estimates The coefficients (point estimates) of the model terms. They are depicted as bold dots in the coefficient plot.

lower.inner	The lower bounds of the inner confidence intervals, provided as a vector. Inner CI means more relaxed (lower confidence level, higher alpha) because fewer observations have to fall into the CI, therefore the CI gets smaller.
upper.inner	The upper bounds of the inner confidence intervals, provided as a vector. Inner CI means more relaxed (lower confidence level, higher alpha) because fewer observations have to fall into the CI, therefore the CI gets smaller.
lower.outer	The lower bounds of the outer confidence intervals, provided as a vector. Outer CI means stricter or narrower (higher confidence level, lower alpha) because more observations have to fall into the CI, therefore the CI gets larger.
upper.outer	The upper bounds of the outer confidence intervals, provided as a vector. Outer CI means stricter or narrower (higher confidence level, lower alpha) because more observations have to fall into the CI, therefore the CI gets larger.
signif.outer	Different colors are used for significant estimates and confidence intervals. If `signif.outer = TRUE`, the outer CIs are used to evaluate significance, otherwise the inner CIs are used.
xlab	The label of the x axis.
main	The main title or heading of the plot.
cex	Size of the point representing the estimate.
lwd.inner	Line width of the inner confidence interval.
lwd.outer	Line width of the outer confidence interval.
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 or the graphics device. See the help entries of extract and extract-methods for more information.

Details

The `coefplot` function produces coefficient plots (i.e., forest plots applied to point estimates and confidence intervals). It accepts raw data (the lower and upper bounds of inner and outer confidence intervals as well as the point estimates and their names) as input data. Significant coefficients and intervals can be plotted in a different color.

The `plotreg` function is a wrapper for the `coefplot` function and works much like the `screenreg`, `texreg`, and `htmlreg` functions. It accepts a single 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 and extract-methods), the default standard errors are converted to confidence intervals. Most of the arguments work either like in the `screenreg`, `texreg`, and `htmlreg` functions, or they work like in the `coefplot` function.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

See Also

texreg-package extract extract-methods texreg

Examples

```
#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)
screenreg(lm.D9)  # print model output to the R console
plotreg(lm.D9)    # plot model output as a diagram
```

---

print.texregTable *Print the output of a screenreg, texreg, or htmlreg call*

Description

Print the output of a screenreg, texreg, or htmlreg call.

Usage

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

Arguments

x	A texregTable object. This is basically a simple character object with an additional class name called texregTable.
...	Additional arguments to be handed over to the cat function.

Details

This function prints a texregTable object, which results from a screenreg, texreg, or htmlreg call. Most of the time, this function is called implicitly by just entering the name of the object.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

See Also

texreg-package texreg

texreg *Convert regression output to LaTeX or HTML tables*

Description

Conversion of R regression output to LaTeX or HTML tables.

Usage

```
texreg(l, file = NULL, single.row = FALSE, stars = c(0.001,
    0.01, 0.05), custom.model.names = NULL,
    custom.coef.names = NULL, custom.gof.names = NULL,
    custom.note = NULL, digits = 2, leading.zero = TRUE,
    symbol = "\\cdot", override.coef = 0, override.se = 0,
    override.pval = 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, bold = 0.00, center = TRUE,
    caption = "Statistical models", caption.above = FALSE,
    label = "table:coefficients", booktabs = FALSE,
    dcolumn = FALSE, sideways = FALSE, use.packages = TRUE,
    table = TRUE, no.margin = TRUE, scriptsize = FALSE,
    float.pos = "", ...)

htmlreg(l, file = NULL, single.row = FALSE, stars = c(0.001,
    0.01, 0.05), custom.model.names = NULL,
    custom.coef.names = NULL, custom.gof.names = NULL,
    custom.note = NULL, digits = 2, leading.zero = TRUE,
    symbol = "&middot;", override.coef = 0, override.se = 0,
    override.pval = 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, bold = 0.00, center = TRUE,
    caption = "Statistical models", caption.above = FALSE,
    star.symbol = "*", inline.css = TRUE, doctype = TRUE,
    html.tag = FALSE, head.tag = FALSE, body.tag = FALSE, ...)

screenreg(l, file = NULL, single.row = FALSE, stars = c(0.001,
    0.01, 0.05), custom.model.names = NULL,
    custom.coef.names = NULL, custom.gof.names = NULL,
    custom.note = NULL, digits = 2, leading.zero = TRUE,
    symbol = ".", override.coef = 0, override.se = 0,
    override.pval = 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, column.spacing = 2, outer.rule = "=",
    inner.rule = "-", ...)
```

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 coef name and leave the first and third name as they are in the original model.
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.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".

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, "·" 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.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.pval = 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.pval = 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 or extract-methods.

`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, so the `custom.gof.names` argument should follow the original order.

`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 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)`).

`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`. If no star should be printed at all, `ci.test = NULL` can be used. 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)`.

`bold` [only in the `texreg` and `htmlreg` functions] The p value threshold below which the coefficient shall be formatted in a bold font. For example, `bold = 0.05` will cause all coefficients which are significant at the 95% level to be formatted in bold. Note that this is not compatible with the dcolumn argument in the `texreg` function. If both are `TRUE`, dcolumn is 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` [only in the `texreg` and `htmlreg` functions] Should the table be horizontally aligned at the center of the page?

`caption` [only in the `texreg` and `htmlreg` functions] Set the caption of the table.

`caption.above` [only in the `texreg` and `htmlreg` functions] Should the caption of the table be placed above the table? By default, it is placed below the table.

`label` [only in the `texreg` function] Set the label of the `table` environment.

`booktabs` [only in the `texreg` function] Use the `booktabs` LaTeX package to get thick horizontal rules in the output table (recommended).

`dcolumn` [only in the `texreg` function] Use the `dcolumn` LaTeX package to get a nice alignment of the coefficients (recommended).

`sideways` [only in the `texreg` function] If `sideways = TRUE` is set, the `table` floating environment is replaced by a `sidewaystable` float, and the `rotating` package is loaded in the preamble. The argument only has an effect if `table = TRUE` is also set.

`use.packages` [only in the `texreg` function] If this argument is set to `TRUE` (= the default behavior), the required LaTeX packages are loaded in the beginning. If set to `FALSE`, the use package statements are omitted from the output.

`table` [only in the `texreg` function] By default, texreg puts the actual `tabular` object in a `table` floating environment. To get only the `tabular` object without the whole table header, set `table = FALSE`.

`no.margin` [only in the `texreg` function] In order to save space, inner margins of tables are switched off by default. To reactivate the default table spacing, set `no.margin = FALSE`.

`scriptsize` [only in the `texreg` function] To save horizontal space on the page, the table can be set in script size instead of normal text size by setting `scriptsize = TRUE`.

`float.pos` [only in the `texreg` function] 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.

`star.symbol` [only in the `htmlreg` function] 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 `htmlreg` 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).

`inline.css` [only in the `htmlreg` function] 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` [only in the `htmlreg` function] 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** package is used to generate HTML code because **knitr** requires only the plain table, 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` [only in the `htmlreg` function] 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 recommended when the code is saved to a file, for example as an MS Word document.

`head.tag` [only in the `htmlreg` function] 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 recommended when the code is saved to a file, for example as an MS Word document.

`body.tag` [only in the `htmlreg` function] 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.

`column.spacing` [only in the `screenreg` function] 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` [only in the `screenreg` function] 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` [only in the `screenreg` function] The character which is 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 and extract-methods for more information.

Details

texreg converts coefficients, standard errors, significance stars, and goodness-of-fit statistics of statistical models into LaTeX tables 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.

The `texreg()` function creates LaTeX code for inclusion in a LaTeX document or for usage with **Sweave** or **knitr**.

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`, and `body.tag` arguments must be adjusted for the different purposes (see the description of the arguments).

The `screenreg()` function creates text representations of tables and prints them to the R console. This is an alternative to the `summary` method 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 three functions is identical.

Author(s)

Philip Leifeld (http://www.philipleifeld.de)

References

Leifeld, Philip (2013). texreg: Conversion of Statistical Model Output in R to LaTeX and HTML Tables. Journal of Statistical Software, 55(8), 1-24. http://www.jstatsoft.org/v55/i08/.

See Also

texreg-package extract extract-methods plotreg

Examples

```
#Linear mixed-effects models
library(nlme)
model.1 <- lme(distance ~ age, data = Orthodont, random = ~ 1)
model.2 <- lme(distance ~ age + Sex, data = Orthodont, random = ~ 1)
texreg(list(model.1, model.2), booktabs = TRUE, dcolumn = TRUE)

#Ordinary least squares model (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)
table.string <- texreg(lm.D9, return.string = TRUE)
cat(table.string)

#Create a 'fake' Office document containing a regression table
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")
```

Index