TEXreg: Conversion of R regression output to LATEX tables

Philip Leifeld <philip.leifeld@eawag.ch>

June 19, 2012

1 Motivation

The TEXreg package for the statistical computing environment R was designed to convert regression model output from multiple models into tables for inclusion in LATEX documents. While similar packages exist – e.g., `xtable`, `outreg` and `memsic` –, they do not provide this functionality for `lme` objects (from the nlme package) and `ergm` objects (from the statnet suite). Moreover, most existing packages are not able to merge multiple models into one single table. The TEXreg package fills these gaps. Currently, `lm`, `lme`, `gls`, `glm` and `ergm` objects are supported. New model types can be easily implemented.

2 Installation

It should be possible to install TEXreg using a simple command:

```
> install.packages("texreg")
```

If this is not possible for some reason, the source files can be downloaded from http://www.philipleifeld.de or http://r-forge.r-project.org/projects/texreg/. To load the package in R once it has been installed, enter the following command:

```
> library(texreg)
```

3 Getting help

This R package vignette is part of the TEXreg package. It can be displayed in R by entering the command:

```
> vignette("texreg")
```

The help page of the package can be displayed as follows:

```
> help(package = "texreg")
```

More specific help on the `texreg` command can be obtained by entering the following command once the package has been loaded:

```
> help(texreg)
```

If all else fails, more help can be obtained from the homepage of the TEXreg package. Questions can be posted to a public forum at http://r-forge.r-project.org/projects/texreg/. A prior registration may be required.

4 TEXreg examples

Suppose you fit two simple OLS models. The following example was taken from the `lm()` help file.

```
> ctl <- c(4.17, 5.58, 5.18, 6.11, 4.5, 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)
> m1 <- lm(weight ~ group)
> m2 <- lm(weight ~ group - 1)
```

The coefficients, standard errors, p values etc. can be displayed as follows:

```
> summary(m2)

Call:
lm(formula = weight ~ group - 1)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0710 -0.4938  0.0685  0.2462  1.3690

Coefficients:
         Estimate Std. Error t value Pr(>|t|)
groupCtl   5.0320     0.2202   22.85 9.55e-15 ***
groupTrt   4.6610     0.2202   21.16 3.62e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6964 on 18 degrees of freedom
Multiple R-squared: 0.9818,        Adjusted R-squared: 0.9798
F-statistic: 485.1 on 2 and 18 DF,  p-value: < 2.2e-16
```

Now it is fairly tedious to copy every single coefficient and standard error to a LaTeX table when you design your academic paper. To improve the situation, the following commands can do this automatically (the LaTeX output code is shown below the R code, and the resulting table is shown in table 1):

```
> library(texreg)
> table <- texreg(m2)

\usepackage{booktabs}
\usepackage{dcolumn}

\begin{table}
\begin{center}
\begin{tabular}{l D{.}{.}{3.5} @{}}
\toprule
           & \multicolumn{1}{c}{Model 1} \\
\midrule
groupCtl   & 5.03^{***} \\
           & (0.22)     \\
groupTrt   & 4.66^{***} \\
           & (0.22)     \\
\midrule
R$^2$      & 0.98       \\
```

	Model 1
groupCtl	5.03***
	(0.22)
groupTrt	4.66***
	(0.22)
R^2	0.98
Adj. R^2	0.98
Num. obs.	20

$^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$

Table 1: Statistical models

	Model 1	Model 2
groupCtl		5.03***
		(0.22)
groupTrt	−0.37	4.66***
	(0.31)	(0.22)
(Intercept)	5.03***	
	(0.22)	
R^2	0.07	0.98
Adj. R^2	0.02	0.98
Num. obs.	20	20

$^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$

Table 2: Statistical models

```
Adj. R$^2$ & 0.98        \\
Num. obs.  & 20          \\
\bottomrule
\vspace{-2mm}\\
\multicolumn{2}{l}{\textsuperscript{***}$p<0.01$, \textsuperscript{**}$p<0.05$, \textsuperscript{*
\end{tabular}
\end{center}
\caption{Statistical models}
\label{table:coefficients}
\end{table}
```

The table is saved in the object `table`. Moreover, it is printed directly to the R console for easy copy & paste. In order to print it to the R console again, the following command can be used:

```
> cat(table)
```

The `texreg` command also accepts multiple models as a `list` and merges them in a table. The output of the following command is shown in table 2.

```
> table <- texreg(list(m1, m2))
```

The TEXreg package contains many customizations. Among other options, the `use.packages` argument can be used to switch off package loading at the beginning of the table code. Using the `label` argument, the label of the table can be set. In a similar way, the `caption` argument takes care of the caption. Activating the `scriptsize` option prints the table in a smaller font size. An example:

	Model 1	Model 2
groupCtl		5.03***
		(0.22)
groupTrt	−0.37	4.66***
	(0.31)	(0.22)
(Intercept)	5.03***	
	(0.22)	
R^2	0.07	0.98
Adj. R^2	0.02	0.98
Num. obs.	20	20

$^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$

Table 3: My regression table

```
> table <- texreg(list(m1, m2), use.packages = FALSE, label = "tab:3",
+     caption = "My regression table", scriptsize = TRUE)
```

The output of this command is shown as table 3. Another argument is `table`. By deactivating it, the plain `tabular` environment is printed, and the whole table environment and header is omitted from the output. The `no.margin` argument can be used to control the cell spacing of the table. If set to `TRUE`, regular margins are used. By default, no margins are used in order not to waste any horizontal space on the page.

TEXreg employs functions from the `booktabs` and `dcolumn` packages to generate beautiful tables. If these packages should not be used when generating tables, the arguments `booktabs` and `dcolumn`, respectively, can be set to `FALSE`.

The TEXreg package can also handle `ergm` objects (that is, exponential random graph models, which are used in social network analysis). Here is an example: the following code creates a network matrix.

```
> mat <- rbinom(400, 1, 0.16)
> mat <- matrix(mat, nrow = 20)
```

Using the `network` package, the matrix can be converted into a network object. The `ergm()` command from the `ergm` package can be used to fit some models:

```
> library(network)
> library(ergm)
> nw <- network(mat)
> m4 <- ergm(nw ~ edges)
> m5 <- ergm(nw ~ edges + mutual)
> m6 <- ergm(nw ~ edges + mutual + twopath)
```

The TEXreg command can then be used to create a table with the coefficients. Switching on `strong.signif` returns the significance levels used by the `ergm` package (three stars for p values smaller than 0.001 etc.) instead of using conventional significance stars:

```
> table <- texreg(list(m4, m5, m6), use.packages = FALSE, label = "tab:4",
+     scriptsize = FALSE, strong.signif = TRUE)
```

Table 4 shows the result of this command.

Most academic journals require tables where the coefficient and the standard error are stored in two separate rows of the table, as shown in tables 1 to 4. In some situations, however, it makes sense to accommodate them in a single row. The `single.row` argument can take care of this:

```
> table <- texreg(list(m4, m5, m6), use.packages = FALSE, label = "tab:5",
+     single.row = TRUE)
```

The result is shown in table 5. Note the difference between tables 4 and 5.

The TEXreg command can also combine the output of different model types in a single table. Consider the following example of an `lm` object, an `lme` (linear mixed-effects) model and an `ergm` object:

	Model 1	Model 2	Model 3
edges	-1.58^{***}	-1.43^{***}	-1.64^{**}
	(0.14)	(0.16)	(0.62)
mutual		$-1.30^{\cdot}$	$-1.29^{\cdot}$
		(0.76)	(0.75)
twopath			0.03
			(0.09)
AIC	349.74	347.72	349.53
BIC	353.68	355.60	361.35
Log Likelihood	−173.87	−171.86	−171.76

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$, $^{\cdot}p < 0.1$

Table 4: Statistical models

	Model 1	Model 2	Model 3
edges	-1.58 $(0.14)^{***}$	-1.43 $(0.16)^{***}$	-1.64 $(0.62)^{***}$
mutual		-1.30 $(0.76)^{*}$	-1.29 $(0.75)^{*}$
twopath			0.03 (0.09)
AIC	349.74	347.72	349.53
BIC	353.68	355.60	361.35
Log Likelihood	−173.87	−171.86	−171.76

$^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$

Table 5: Statistical models

	Model 1	Model 2	Model 3
age		0.66***	
		(0.06)	
edges			−1.64***
			(0.62)
groupCtl	5.03***		
	(0.22)		
groupTrt	4.66***		
	(0.22)		
(Intercept)		17.71***	
		(0.83)	
mutual			−1.29*
			(0.75)
SexFemale		−2.32***	
		(0.76)	
twopath			0.03
			(0.09)
R^2	0.98		
Adj. R^2	0.98		
Num. obs.	20	108	
AIC		447.51	349.53
BIC		460.78	361.35
Log Likelihood		−218.76	−171.76

$^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$

Table 6: Statistical models

```
> library(nlme)
> m3 <- lme(distance ~ age + Sex, data = Orthodont, random = ~1)

> table <- texreg(list(m2, m3, m6), label = "tab:6", use.packages = FALSE)
```

The output is shown in table 6. Note that different model types may report different kinds of goodness-of-fit statistics at the bottom of the table.

5 Creating templates for new model types

Currently, TEXreg supports `lm`, `lme`, `gls`, `glm` and `ergm` objects. However, implementing new kinds of statistical models is fairly easy (if you know how to modify R functions). For any model type, there exists a function which extracts the relevant information from a model. For example, `extract.lm()` provides coefficients and goodness-of-fit statistics for `lm` objects, `extract.ergm()` provides this information for `ergm` objects, etc. Any new function of this kind must return a list with two objects:

1. A matrix containing the coefficients. This matrix must have exactly three columns. The first column contains the coefficients, the second column contains the standard errors, and the third column contains the p values for any coefficient. The names of the coefficients or independent variables must be stored as row names of the matrix. Column names do not matter.

2. A matrix of goodness-of-fit statistics. This matrix must have exactly one column and as many rows as there are gof measures. For example, the `extract.lm()` function extracts R^2, Adj. R^2 and Num. obs. They are aggregated in a 3×1 matrix. The row names of this matrix should indicate what is being measured, for instance "Num. obs."

The following code is an example. It shows the `extract.lme()` function:

```
extract.lm <- function(model) {

  if (!class(model) == "lm") {
    stop("Internal error: Incorrect model type! Should be an lm object!")
  }

  tab <- summary(model)$coef[,-3]       #extract coefficient table
                                        #third column (t values) is omitted

  rs <- summary(model)$r.squared        #extract R-squared
  adj <- summary(model)$adj.r.squared   #extract adjusted R-squared
  n <- nobs(model)                      #extract number of observations

  gof <- matrix(c(rs, adj, n), ncol=1)  #put gof measures in a 1-column matrix
  row.names(gof) <- c("R$^2$", "Adj. R$^2$", "Num. obs.")  #set row names

  table.content <- list(tab, gof)       #put coefficients and gofs in a list
  return(table.content)                 #return the list object
}
```

After writing a custom function, this function has to be registered in the `texreg()` function. There are two locations where the code has to be slightly adjusted. These two locations are marked by a comment stating "IMPLEMENT NEW EXTENSIONS HERE".

If you write such a function, it would be very helpful to post them in the forum (see section 3) in order to let other users profit from it. If it works and if you can provide a self-contained example, the code can be implemented in a future version of TEXreg. Please send an inquiry if you are interested in joining the TEXreg project and working directly on the code.