Regression Table Survey with outreg

Paul Johson, CRMDA, pauljohn@ku.edu

Guide No: 49

Keywords: R, rockchalk, outreg, LaTeX, regression tables

See https://crmda.ku.edu/guides for updates.

Aug. 26, 2024

Abstract

This is a comprehensive demonstration of the outreg() function in rockchalk. This vignette demonstrates the ability to integrate model estimates, to assign labels for models and variables, and to customize the selection of summary statistics. A major innovation in recent versions is the introduction of the ability to generate "decimal-centered" columns. This has been the most widely requested feature for a while.

Contents

The rockchalk package for R (R Core Team, 2018) was created to facilitate authors in stats classes who need to make presentable tables and graphs for regression models. It does not aspire to be a comprehensive table-maker, mainly because R package writers are not consistent in their methodology. Procedures called "regression" within R's base have a certain internal consistency, but user-contributed packages do not. The outreg function works for regression models that I use regularly, usually in the base of R, like lm, glm, and for lme4 objects. Other regression packages may work. I've tried to be somewhat general in the design.

The major new feature is "decimal-centered" columns. In the implementation of centering, we tested approaches based on two LaTeX packages, dcolumn and siunitx. dcolumn output it is mostly adequate, but it leaves problems in column headers and non-numeric content alignment. siunitx is a much more massive framework. There is much more danger of outright failure because of the immensity of siunitx itself. However, at the current time, all of the examples we have tested indicate that the siunitx-based tables are superior. there are hopeful signs it can deal with some

CRMDA, 1425 Jayhawk BLVD
University of Kansas, Waston Library Suite 470
Lawrence, KS 66045

Web: https://crmda.ku.edu
Email: pauljohn@ku.edu
Phone: 785-864-3353

very tough examples. See Table 30 I have a solution using `siunitx` for a troublesome table that failed badly with `dcolumn` (Table 29). The user can designate which method should be used by assigning the parameter `centering = "dcolumn"` or `centering = "siunitx"`. NOTE: it is necessary for the user to insert either `\usepackage{dcolumn}` or `\usepackage{siunitx}` in the preamble! The aim is for this whole thing to "just work" without any more attention than that. Please let me know if there is trouble.

1 Briefly: outreg()

First, run regression models. Second, give the fitted regression object(s) to `outreg`. When `outreg` runs, it will have 2 effects. First, it will write out LaTeX code to the screen (unless `print.results=FALSE`). A user might "copy/paste" that code into a LaTeX document. (Or, write it in a file and import it in the future. (Or, as in the case of this Sweaved document, the output goes directly into the result file.)

2 Briefly: table vs tabular

This is the most confusing thing for LaTeX beginners. *Put bluntly, a "table" is not a table.*

Terminology

tabular In LaTeX, a "`tablular`" object is a grid, a display like a "spreadsheet". You'd call that a "table", I believe.

table In LaTeX, a "`table`" is something else. It is a document "subsection" that "floats" around in the document. It is a container. A tabular is placed inside one of these table containers. Table objects are numbered, can be used in cross references (if they have labels)

If we are writing LaTeX code by hand, a tabular inside a table will look like

```
\begin{table}
    \caption{The title of the floating object is specified in
       caption}
    \label{tab:ex1}% cross referencing use \ref{tab:ex1}
\begin{tabular}
  ...code to create tablar defined here
\end{tabular}
\end{table}
```

The big question is "should my outreg function write the tabular only, or also the table container"?

I used to think the only reasonable answer was "tabular only", but now I see reasons why we might want to do both. So it is an option, as explained next.

rockchalk lets you choose, float or no float

The rockchalk outreg function has a parameter, `float=FALSE` or `float=TRUE`, to determine whether the output should include the floating table part along with the tabular object. If outreg users specify the argument title, then they are implicitly setting `float=TRUE` and the value for title is used as the LaTeX caption.

The default for `float` is `FALSE`, because I have usually preferred to control the floating table from my document. However, I recognize that many people who are better at LaTeX than I am disagree, they want to write the table and tabular structures into one file.

Why would a user prefer float=TRUE?

If you are not using LyX, then it may be convenient to let float=TRUE so the floating table can be handled from the outreg function. outreg allows the author to specify the caption and the label. However, I did not allow for any more fine-grained customizations, especially table placement details.

Why I don't generally use float=TRUE

LaTeX has options to control the placement of floating table structures that the rockchalk outreg function does not adjust. I think that if the LaTeX author wants to adjust those things, it is easier for the author to control them in the document itself, rather than the outreg function that writes a tabular object.

In LyX, the user interface has a pull down menu to create floating objects and the GUI includes a feature to set the title (sorry, the 'caption'), and it also includes a way to set a label. If I use this method, then LyX is aware of this thing and the LyX cross-referencing system is available.

Another reason to use float=FALSE

In an instructional document using Sweave, the code chunks will print out where they "are" in the input file.

In LyX, there are some cross-referencing tools. It is more desirable to create the table container in LyX, so then cross-references work correctly. If one creates a table (floating container), and then puts the tabular-creating code chunk inside, then the code will print out with the table. And cross references succeed using the LyX pull down menus. That is demonstrated in Tables 20, 21, 23, 31.

3 Torture test of "rockchalk" outreg function

In most of these examples, I've taken the one-step "float=TRUE" option, but I have some examples where I've manually created the LaTeX float and placed an outreg tabular inside it.

```
set.seed(2134234)
dat <- data.frame(x1 = rnorm(100), x2 = rnorm(100))
dat$y1 <- 30 + 5 * rnorm(100) + 3 * dat$x1 + 4 * dat$x2
```

Table 1: My One Tightly Printed Regression (uncentered)

	M1
	Estimate
	(S.E.)
Intercept	30.245***
	(0.618)
x1	1.546*
	(0.692)
N	100
RMSE	6.121
R^2	0.048

$*p \leq 0.05 ** p \leq 0.01 ***p \leq 0.001$

Table 2: My Wide Format "side-by-side" columns (uncentered)

	M1	
	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)
x1	1.546*	(0.692)
N	100	
RMSE	6.121	
R^2	0.048	

$*p \leq 0.05 ** p \leq 0.01 ***p \leq 0.001$

```
dat$y2 <- rnorm(100) + 5 * dat$x2
m1 <- lm(y1 ~ x1, data = dat)
m2 <- lm(y1 ~ x2, data = dat)
m3 <- lm(y1 ~ x1 + x2, data = dat)
gm1 <- glm(y1 ~ x1, family = Gamma, data = dat)
```

3.1 tight and wide styles for regression tables

In my terminology, a tight table has one "narrow" column for a regression (parameter and standard error stacked on top of each other). A table that is not in the tight format is wide, it has 2 columns for each table (parameter and standard error side by side).

A tight table is the default, it seems to be what many political science, sociology, and economics students will need. See Table 1. A wide format table is, in my opinion, more pleasant for the eyes and seems to be more popular in psychology. See Table 2.

```
library(rockchalk)
ex1w <- outreg(m1, title = "My Wide Format \"side-by-side\"
   columns (uncentered)", label = "tab:ex1w", tight = FALSE, float
   = TRUE, print.results = FALSE)
cat(ex1w)
```

Table 3: Tight column with centering = ”dcolumn”

	M1
	Estimate
	(S.E.)
(Intercept)	30.245***
	(0.618)
x1	1.546*
	(0.692)
N	100
RMSE	6.121
R^2	0.048

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 4: Tight column with centering = ”siunitx”

	M1
	Estimate
	(S.E.)
(Intercept)	30.245***
	(0.618)
x1	1.546*
	(0.692)
N	100
RMSE	6.121
R^2	0.048

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Decimal-centered test case 1

A decimal-centered version of the tight column format can be seen in Table 3 and the decimal-centered version of the wide format is seen in Table 5

```
library(rockchalk)
ex2d <- outreg(m1, title = 'Tight column with centering =
   "dcolumn"', label = "tab:ex2d", centering = "dcolumn", float =
   TRUE, print.results=FALSE)
cat(ex2d)
```

```
library(rockchalk)
ex2s <- outreg(m1, title = 'Tight column with centering =
   "siunitx"', label = "tab:ex2s", centering = "siunitx", float =
   TRUE)
```

```
library(rockchalk)
ex1wd <- outreg(m1, title = 'Wide (not tight) format with
   centering = "dcolumn"',  label = "tab:ex2wd", tight = FALSE,
   centering = "dcolumn",  float = TRUE, print.results = FALSE)
```

Table 5: Wide (not tight) format with centering = "dcolumn"

	M1	
	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)
x1	1.546*	(0.692)
N	100	
RMSE	6.121	
R^2	0.048	

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 6: Wide (not tight) format with centering = "siunitx"

	M1	
	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)
x1	1.546*	(0.692)
N	100	
RMSE	6.121	
R^2	0.048	

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

```
cat(ex1wd)
```

```
ex1ws <- outreg(m1, title = 'Wide (not tight) format with
   centering = "siunitx"',  label = "tab:ex2ws", tight = FALSE,
   centering = "siunitx", float = TRUE, print.results = FALSE)
cat(ex1ws)
```

alpha level adjustments

In Table 7, I demonstrate that the alpha parameter can be used to select different values for the critical region. In Tables 8 and 9, the centered-with-dcolumn and centered-with-siunitx are presented. I see some trouble here because the stars are not evenly spaced in the dcolumn-based Table 8. However, the siunitx based Table 9 may be adequate. This is one of the reasons I now *strongly* lean toward use of `siunitx` if centering is desired. Several alpha adjustments are scattered about in the examples in this essay, just to make sure they work.

```
ex2p <- outreg(list("Fingers" = m1), tight = FALSE, title =
   "Ability to change p values (not centered)", label =
   "tab:ex2p",  float = TRUE, alpha = c(0.1, 0.05, 0.01))
```

```
ex2pd <- outreg(list("Fingers" = m1), tight = FALSE,  title =
   "Ability to change p values (dcolumn)", label = "tab:ex2pd",
   centering = "dcolumn", float = TRUE, alpha = c(0.1, 0.05, 0.01))
```

Table 7: Ability to change p values (not centered)

	Fingers	
	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)
x1	1.546**	(0.692)
N	100	
RMSE	6.121	
R^2	0.048	

$*p \leq 0.1$** $p \leq 0.05$***$p \leq 0.01$

Table 8: Ability to change p values (dcolumn)

	Fingers	
	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)
x1	1.546 * *	(0.692)
N	100	
RMSE	6.121	
R^2	0.048	

$*p \leq 0.1$** $p \leq 0.05$***$p \leq 0.01$

```
ex2ps <- outreg(list("Fingers" = m1), tight = FALSE,  title =
   "Ability to change p values (siunitx)", label = "tab:ex2ps",
   centering = "siunitx", float = TRUE, alpha = c(0.1, 0.05, 0.01))
```

Several models in same table

One of the most valuable features of rockchalk is that one can align several models side by side and compare them. The rows are matched by the variable name.

About model names: If a list of regression fits is not named, then the names will be bland, "M1", "M2".

It is highly recommended that authors should name the regression models.

Table 9: Ability to change p values (siunitx)

	Fingers	
	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)
x1	1.546**	(0.692)
N	100	
RMSE	6.121	
R^2	0.048	

$*p \leq 0.1$** $p \leq 0.05$***$p \leq 0.01$

Table 10: My Two Linear Regressions (uncentered)

	Model A Estimate (S.E.)	Model B has a longer heading Estimate (S.E.)
(Intercept)	30.245*** (0.618)	29.774*** (0.522)
Billie	1.546* (0.692)	_
x2	_	3.413*** (0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305
F(df_{num},df_{denom})	4.98(1,98)*	44.4(1,98)***

$*p \leq 0.05 ** p \leq 0.01 *** p \leq 0.001$

Currently, I recommend that we provide the model names **in** the list that provides the fitted regressions, as you see here. That produces Table 10.

```
ex3 <- outreg(list("Model A" = m1, "Model B has a longer heading"
   = m2), varLabels = list(x1 = "Billie"),   title = "My Two
   Linear Regressions (uncentered)", label = "tab:ex3", request =
   c(fstatistic = "F"),  print.results = FALSE)
cat(ex3)
```

In the original rockchalk, I had it differently. It is possible to provide model names in a separate argument, `modelLabels`. That is shown in the code below, and in output Table 11. The modelLabels parameter takes precedence, it will replace the labels in the first argument. However, I found it confusing to write tables that way, so I made it work the other way too.

```
ex3b <- outreg(list("Model A" = m1, "Model B" = m2),   modelLabels
   = c("Overrides ModelA", "Overrides ModelB"), varLabels =
   list(x1 = "Billie"), title = "Note modelLabels Overrides model
   names (uncentered)",  label = "tab:ex3b"
 )
```

The column-centered versions are in Table 12 and Table 13.

```
ex3bd <- outreg(list("Model A" = m1, "Model B" = m2), modelLabels
   = c("Overrides ModelA", "Overrides ModelB"), varLabels =
   list(x1 = "Billie"), title = "Note modelLabels Overrides model
   names (dcolumn)", label = "tab:ex3bd", centering = "dcolumn")
```

```
ex3bs <- outreg(list("Model A" = m1, "Model B" = m2), modelLabels
   = c("Overrides ModelA", "Overrides ModelB"),  varLabels =
   list(x1 = "Billie"), title = "Note modelLabels Overrides model
   names (siunitx)", label = "tab:ex3bs", centering = "siunitx")
```

Table 11: Note modelLabels Overrides model names (uncentered)

	Overrides ModelA Estimate (S.E.)	Overrides ModelB Estimate (S.E.)
(Intercept)	30.245***	29.774***
	(0.618)	(0.522)
Billie	1.546*	_
	(0.692)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 12: Note modelLabels Overrides model names (dcolumn)

	Overrides ModelA Estimate (S.E.)	Overrides ModelB Estimate (S.E.)
(Intercept)	30.245***	29.774***
	(0.618)	(0.522)
Billie	1.546*	_
	(0.692)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 13: Note modelLabels Overrides model names (siunitx)

	Overrides ModelA	Overrides ModelB
	Estimate	Estimate
	(S.E.)	(S.E.)
(Intercept)	30.245***	29.774***
	(0.618)	(0.522)
Billie	1.546*	_
	(0.692)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

Specifying additional summary information

In the first version, I was thinking that everybody would be happy enough if the table included N, the standard error of the regression (which I dubbed RMSE in the old SAS style), and R-square. There were requests for other summaries.

At first, I was thinking I needed to add arguments for each request. I created argument "`showAIC=TRUE`" to request Akaike's information criterion. For example, see Table 14 produced by the following

```
ex5d <- outreg(list("Whichever" = m1, "Whatever" = m2), title =
   "Still have showAIC argument (uncentered)", label = "tab:ex5d",
   showAIC = TRUE, float = TRUE)
```

The column-centered versions are in Table 15 and 16.

```
ex5dd <- outreg(list("Whichever" = m1, "Whatever" = m2),  title =
   "Still have showAIC argument (dcolumn)", label = "tab:ex5dd",
   showAIC = TRUE, float = TRUE, centering = "dcolumn")
```

```
ex5ds <- outreg(list("Whichever" = m1, "Whatever" = m2), title =
   "Still have showAIC argument (siunitx)", label = "tab:ex5ds",
   showAIC = TRUE, float = TRUE, centering = "siunitx")
```

However, I anticipated that way might lead me down a bad path of writing a parameter for every possible summary statistic.

My first idea was to create a recipe book of particular summary items and make them available for requests by users. For example, I wrote a customized reporter for F statistics and a parameter to ask for that was called "request". For example, `request = c(fstatistic = "F")` asks for my special fstatistic and the label to be used for it in the table would be "F". I ended up not making very many of those fancy object, but in the semTable in the kutils package there is a similar approach for the model Chi-Square and now it seems to me I should come back and do more like that in outreg.

Table 14: Still have showAIC argument (uncentered)

	Whichever Estimate (S.E.)	Whatever Estimate (S.E.)
(Intercept)	30.245***	29.774***
	(0.618)	(0.522)
x1	1.546*	_
	(0.692)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305
AIC	650.109	617.694

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 15: Still have showAIC argument (dcolumn)

	Whichever Estimate (S.E.)	Whatever Estimate (S.E.)
(Intercept)	30.245***	29.774***
	(0.618)	(0.522)
x1	1.546*	_
	(0.692)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305
AIC	650.109	617.694

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 16: Still have showAIC argument (siunitx)

	Whichever Estimate (S.E.)	Whatever Estimate (S.E.)
(Intercept)	30.245***	29.774***
	(0.618)	(0.522)
x1	1.546*	_
	(0.692)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305
AIC	650.109	617.694

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

Because I expected that I would never have time to keep up with requests for specialized summary items, I created a "back door" though which users could use functions and include them in the summary.

However, I shot myself in the foot by creating that general purpose ability with a horrible, ungainly name for the function's argument, `runFuns`. I admit that is a horrible name, but I had good intentions. "runFuns" is short for "run functions". It will run a function named as the first argument and then label it with the right hand side argument. So, if one has a lot of request like AIC, BIC, and so forth, the R functions can be used without too much effort.

```
ex6d <- outreg(list("Whatever" = m1, "Whatever" =m2), title =
   "Another way to get AIC output", label="ex6d",  runFuns =
   c("AIC" = "Akaike IC"), centering = "dcolumn",
   print.results=FALSE)
cat(ex6d)
```

Insert more regressions in one table

This code produces Table 18, which is NOT decimal aligned

```
ex7 <- outreg(list("Amod" = m1, "Bmod" = m2, "Gmod" = m3), title =
   "My Three Linear Regressions", label="tab:ex7")
```

The column-aligned version of the same is found in Table 19, produced by the following code.

```
ex7d <- outreg(list("Amod" = m1, "Bmod" = m2, "Gmod" = m3),
   centering = "dcolumn", title = "My Three Linear Regressions
   (decimal aligned)", label="tab:ex7d")
```

I worried that users who are verbose might break the function. In Table 20, I show that some very long names, even ones with periods, do not seem to cause horrible trouble. They WILL run off the

Table 17: Another way to get AIC output

	Whatever Estimate (S.E.)	Whatever Estimate (S.E.)
(Intercept)	30.245∗∗∗	30.245∗∗∗
	(0.618)	(0.618)
x1	1.546∗	1.546∗
	(0.692)	(0.692)
x2	_	_
N	100	100
RMSE	6.121	5.205
R^2	0.048	0.312
adj R^2	0.039	0.305
Akaike IC	650.11	617.69

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 18: My Three Linear Regressions

	Amod Estimate (S.E.)	Bmod Estimate (S.E.)	Gmod Estimate (S.E.)
(Intercept)	30.245***	29.774***	30.013***
	(0.618)	(0.522)	(0.490)
x1	1.546*	_	2.217***
	(0.692)		(0.555)
x2	_	3.413***	3.717***
		(0.512)	(0.483)
N	100	100	100
RMSE	6.121	5.205	4.849
R^2	0.048	0.312	0.409
adj R^2	0.039	0.305	0.397

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 19: My Three Linear Regressions (decimal aligned)

	Amod Estimate (S.E.)	Bmod Estimate (S.E.)	Gmod Estimate (S.E.)
(Intercept)	30.245***	29.774***	30.013***
	(0.618)	(0.522)	(0.490)
x1	1.546*	_	2.217***
	(0.692)		(0.555)
x2	_	3.413***	3.717***
		(0.512)	(0.483)
N	100	100	100
RMSE	6.121	5.205	4.849
R^2	0.048	0.312	0.409
adj R^2	0.039	0.305	0.397

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

edge of the page if they get much longer. Author will need to change to the tight columns if that is an issue, as seen in Table 21.

Table 20: Stress test very long titles (uncentered)

```
ex11 <- outreg(list("I Love Long Titles" = m1, "Prefer Brevity" =
   m2, "Captain. Kirk. Named. This." = m3), tight = FALSE, float =
   FALSE, centering = "dcolumn")
```

	I Love Long Titles		Prefer Brevity		Captain. Kirk. Named. This.	
	Estimate	(S.E.)	Estimate	(S.E.)	Estimate	(S.E.)
(Intercept)	30.245***	(0.618)	29.774***	(0.522)	30.013***	(0.490)
x1	1.546*	(0.692)	_		2.217***	(0.555)
x2	_		3.413***	(0.512)	3.717***	(0.483)
N	100		100		100	
RMSE	6.121		5.205		4.849	
R^2	0.048		0.312		0.409	
adj R^2	0.039		0.305		0.397	

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 21: Stress test very long titles (dcolumn)

```
ex11td <- outreg(list("I Love Long Titles" = m1, "Prefer Brevity"
   = m2, "Captain. Kirk. Named. This" = m3), float = FALSE,
   centering = "dcolumn")
```

	I Love Long Titles	Prefer Brevity	Captain. Kirk. Named. This
	Estimate	Estimate	Estimate
	(S.E.)	(S.E.)	(S.E.)
(Intercept)	30.245***	29.774***	30.013***
	(0.618)	(0.522)	(0.490)
x1	1.546*	_	2.217***
	(0.692)		(0.555)
x2	_	3.413***	3.717***
		(0.512)	(0.483)
N	100	100	100
RMSE	6.121	5.205	4.849
R^2	0.048	0.312	0.409
adj R^2	0.039	0.305	0.397

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 22: Stress test very long titles (dcolumn)

```
ex11ts <- outreg(list("I Love Long Titles" = m1, "Prefer Brevity"
   = m2, "Captain. Kirk. Named. This" = m3), float = FALSE,
   centering = "siunitx")
```

	I Love Long Titles	Prefer Brevity	Captain. Kirk. Named. This
	Estimate	Estimate	Estimate
	(S.E.)	(S.E.)	(S.E.)
(Intercept)	30.245***	29.774***	30.013***
	(0.618)	(0.522)	(0.490)
x1	1.546*	_	2.217***
	(0.692)		(0.555)
x2	_	3.413***	3.717***
		(0.512)	(0.483)
N	100	100	100
RMSE	6.121	5.205	4.849
R^2	0.048	0.312	0.409
adj R^2	0.039	0.305	0.397

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Alternative standard errors

The original rockchalk took the standard errors from the fitted model. A student in Brazil wrote and asked me to make it possible for the author to supply "robust" standard errors. That was a

good idea. The code to demonstrate how to create an alternative vector of standard errors (in that case, Huber-White robust standard errors) will appear with the output in Table 23. The code is displayed there because I created the outreg table with float=FALSE, and then I manually created a table container into which I typed the R code chunk.

Table 23: Robust Standard Errors (uncentered)

```
if (require(car)){
    newSE <- sqrt(diag(car::hccm(m3)))
    ex8 <- outreg(list("Model A" = m1, "Model B" = m2, "Model C" =
       m3,
              "Model C w Robust SE" = m3),
              SElist= list("Model C w Robust SE" = newSE))
 }
```

	Model A Estimate (S.E.)	Model B Estimate (S.E.)	Model C Estimate (S.E.)	Model C w Robust SE Estimate (S.E.)
(Intercept)	30.245***	29.774***	30.013***	30.013***
	(0.618)	(0.522)	(0.490)	(0.481)
x1	1.546*	_	2.217***	2.217***
	(0.692)		(0.555)	(0.618)
x2	_	3.413***	3.717***	3.717***
		(0.512)	(0.483)	(0.464)
N	100	100	100	100
RMSE	6.121	5.205	4.849	4.849
R^2	0.048	0.312	0.409	0.409
adj R^2	0.039	0.305	0.397	0.397

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

Table 24: Robust Standard Errors (dcolumn)

```
if (require(car)){
    newSE <- sqrt(diag(car::hccm(m3)))
    ex8 <- outreg(list("Model A" = m1, "Model B" = m2, "Model C" =
       m3,
              "Model C w Robust SE" = m3),
              SElist= list("Model C w Robust SE" = newSE),
              centering = "dcolumn")
 }
```

	Model A Estimate (S.E.)	Model B Estimate (S.E.)	Model C Estimate (S.E.)	Model C w Robust SE Estimate (S.E.)
(Intercept)	30.245***	29.774***	30.013***	30.013***
	(0.618)	(0.522)	(0.490)	(0.481)
x1	1.546*	_	2.217***	2.217***
	(0.692)		(0.555)	(0.618)
x2	_	3.413***	3.717***	3.717***
		(0.512)	(0.483)	(0.464)
N	100	100	100	100
RMSE	6.121	5.205	4.849	4.849
R^2	0.048	0.312	0.409	0.409
adj R^2	0.039	0.305	0.397	0.397

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 25: Robust Standard Errors (siunitx)

```
if (require(car)){
    newSE <- sqrt(diag(car::hccm(m3)))
    ex8 <- outreg(list("Model A" = m1, "Model B" = m2, "Model C" =
       m3,
              "Model C w Robust SE" = m3),
              SElist= list("Model C w Robust SE" = newSE),
              centering = "siunitx")
 }
```

	Model A Estimate (S.E.)	Model B Estimate (S.E.)	Model C Estimate (S.E.)	Model C w Robust SE Estimate (S.E.)
(Intercept)	30.245***	29.774***	30.013***	30.013***
	(0.618)	(0.522)	(0.490)	(0.481)
x1	1.546*	_	2.217***	2.217***
	(0.692)		(0.555)	(0.618)
x2	_	3.413***	3.717***	3.717***
		(0.512)	(0.483)	(0.464)
N	100	100	100	100
RMSE	6.121	5.205	4.849	4.849
R^2	0.048	0.312	0.409	0.409
adj R^2	0.039	0.305	0.397	0.397

$*p \leq 0.05** p \leq 0.01***p \leq 0.001$

Aligning different kinds of fits

The output from lm and glm fits may sometimes be usefully compared. The parameter display from rockchalk will align same-named variables. The difference in the available summary statistics is apparent because the rows do not "line up". See Table 26 for the output from the following.

```
ex13 <- outreg(list("OLS" = m1, "GLM" = gm1), float = TRUE,
                title = "OLS and Logit in same table (dcolumn)",
                label="tab:ex13", alpha = c(0.05, 0.01), centering
                   = "dcolumn")
```

As seen in Table 27, we check if the supplemental parameter requests hold up with dcolumn

```
ex14 <- outreg(list(OLS = m1, GLM = gm1), float = TRUE,
          title = "OLS and Logit with summary report features
             (dcolumn)",
          label = "tab:ex14",
          request = c(fstatistic = "F"), runFuns = c("BIC" =
             "BIC"),
          centering = "dcolumn")
```

What if the number of digits is dialed up and alpha is altered? See Table 28.

Table 26: OLS and Logit in same table (dcolumn)

	OLS Estimate (S.E.)	GLM Estimate (S.E.)
(Intercept)	30.245**	0.033**
	(0.618)	(0.001)
x1	1.546*	−0.002*
	(0.692)	(0.001)
N	100	100
RMSE	6.121	
R^2	0.048	
Deviance		4.301
$-2LLR(Model\chi^2)$		0.208

$*p \leq 0.05$** $p \leq 0.01$

Table 27: OLS and Logit with summary report features (dcolumn)

	OLS Estimate (S.E.)	GLM Estimate (S.E.)
(Intercept)	30.245***	0.033***
	(0.618)	(0.001)
x1	1.546*	−0.002*
	(0.692)	(0.001)
N	100	100
RMSE	6.121	
R^2	0.048	
F(df_{num},df_{denom})	4.98(1,98)*	
Deviance		4.301
$-2LLR(Model\chi^2)$		0.208
BIC	657.92	659.82

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 28: OLS and GLM with more digits (digits)

	OLS Estimate (S.E.)	GLM Estimate (S.E.)
(Intercept)	30.24550∗	0.03313∗
	(0.61763)	(0.00068)
x1	1.54553	−0.00173
	(0.69242)	(0.00078)
N	100	100
RMSE	6.12090	
R^2	0.04838	
F(df_{num},df_{denom})	4.9821(1,98)	
Deviance		4.30066
$-2LLR(Model\chi^2)$		0.20827
BIC	657.92	659.82

$*p \leq 0.01$

```
ex15 <- outreg(list(OLS = m1, GLM = gm1), float = TRUE,
          title="OLS and GLM with more digits (digits)",
          label="tab:ex15",
          request = c(fstatistic = "F"), runFuns = c("BIC" =
             "BIC"),
          digits = 5, alpha = c(0.01), centering = "dcolumn")
```

In Table 29, output shows result when several runFuns are requested. Again, I'm very sorry that parameter name is so awful. I adjusted the alpha stars as well.

```
ex16d <- outreg(list("OLS 1" = m1, "OLS 2" = m2,  GLM = gm1),
   float = TRUE,
            title = "2 OLS and 1 Logit (dcolumn), additional
               runFuns",
            label="tab:ex16d",
            request = c(fstatistic = "F"),
            runFuns = c("BIC" = "BIC", "logLik" = "ll"),
            digits = 5, alpha = c(0.1, 0.05, 0.01), centering =
               "dcolumn")
```

```
ex16s <- outreg(list("OLS 1" = m1, "OLS 2" = m2,  GLM = gm1),
   float = TRUE,
            title = "2 OLS and 1 Logit (siunitx), additional
               runFuns",
            label="tab:ex16s",
            request = c(fstatistic = "F"),
            runFuns = c("BIC" = "BIC", "logLik" = "ll"),
            digits = 5, alpha = c(0.1, 0.05, 0.01), centering =
               "siunitx")
```

Table 29: 2 OLS and 1 Logit (dcolumn), additional runFuns

	OLS 1 Estimate (S.E.)	OLS 2 Estimate (S.E.)	GLM Estimate (S.E.)
(Intercept)	30.24550***	29.77420***	0.03313***
	(0.61763)	(0.52229)	(0.00068)
x1	1.54553 * *	_	−0.00173 * *
	(0.69242)		(0.00078)
x2	_	3.41342***	_
		(0.51222)	
N	100	100	100
RMSE	6.12090	5.20508	
R^2	0.04838	0.31184	
adj R^2	0.03867	0.30482	
F(df_{num},df_{denom})	4.9821(1,98)**	44.409(1,98)***	
Deviance			4.30066
$-2LLR(Model\chi^2)$			0.20827
BIC	657.92	625.51	659.82
ll	-322.05(3)	-305.85(3)	-323(3)

$*p \leq 0.1$** $p \leq 0.05$***$p \leq 0.01$

Table 30: 2 OLS and 1 Logit (siunitx), additional runFuns

	OLS 1 Estimate (S.E.)	OLS 2 Estimate (S.E.)	GLM Estimate (S.E.)
(Intercept)	30.24550***	29.77420***	0.03313***
	(0.61763)	(0.52229)	(0.00068)
x1	1.54553**	_	−0.00173**
	(0.69242)		(0.00078)
x2	_	3.41342***	_
		(0.51222)	
N	100	100	100
RMSE	6.12090	5.20508	
R^2	0.04838	0.31184	
adj R^2	0.03867	0.30482	
F(df_{num},df_{denom})	4.9821(1,98)**	44.409(1,98)***	
Deviance			4.30066
$-2LLR(Model\chi^2)$			0.20827
BIC	657.92	625.51	659.82
ll	-322.05(3)	-305.85(3)	-323(3)

$*p \leq 0.1$** $p \leq 0.05$***$p \leq 0.01$

After a while, I noticed that the left hand side of runFuns need not be in quotation marks. So my examples are not always consistent. For example, in Table 31 I have some in quotes, some not. And I also show that if authors want to create redundant rows, they are allowed to do so (N, for example).

Table 31: Additional test on summary stats (dcolumn)

```
ex17 <- outreg(list("Model A" = gm1, "Model B label with Spaces" =
   m2),
          request = c(fstatistic = "F"),
          runFuns = c("BIC" = "Schwarz IC", "AIC" = "Akaike IC",
             "logLik" = "ll",
          "nobs" = "N Again?"), centering = "dcolumn")
```

	Model A Estimate (S.E.)	Model B label with Spaces Estimate (S.E.)
(Intercept)	0.033***	29.774***
	(0.001)	(0.522)
x1	−0.002*	_
	(0.001)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE		5.205
R^2		0.312
adj R^2		0.305
F(df_{num},df_{denom})		44.4(1,98)***
Deviance	4.301	
$-2LLR(Model\chi^2)$	0.208	
Schwarz IC	659.82	625.51
Akaike IC	652.00	617.69
ll	-323(3)	-306(3)
N Again?	100	100

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

Table 32: Additional test on summary stats (siunitx)

```
ex17s <- outreg(list("Model A" = gm1, "Model B label with Spaces"
   = m2),
          request = c(fstatistic = "F"),
          runFuns = c("BIC" = "Schwarz IC", "AIC" = "Akaike IC",
             "logLik" = "ll",
          "nobs" = "N Again?"), centering = "siunitx")
```

	Model A Estimate (S.E.)	Model B label with Spaces Estimate (S.E.)
(Intercept)	0.033***	29.774***
	(0.001)	(0.522)
x1	−0.002*	_
	(0.001)	
x2	_	3.413***
		(0.512)
N	100	100
RMSE		5.205
R^2		0.312
adj R^2		0.305
F(df_{num},df_{denom})		44.4(1,98)***
Deviance	4.301	
$-2LLR(Model\chi^2)$	0.208	
Schwarz IC	659.82	625.51
Akaike IC	652.00	617.69
ll	-323(3)	-306(3)
N Again?	100	100

$*p \leq 0.05$** $p \leq 0.01$***$p \leq 0.001$

4 About examples here and your long-run work flow

In this document, I'm using a lot of short cuts to make it easier to produce examples with and without centering. I'm using the `float=TRUE` option (or, equivalently, giving a title) often here to cut down on the amount of work I'm doing.

If you are learning how to make tables, I usually suggest doing it in a different way. I would suggest instead that authors should create an outreg output file, with a sequence like this

```
lm1 <- lm(... regression commands ...)
lm1.out <- outreg(lm1, print.results=FALSE, ...parameters like
   float=FALSE, tight = FALSE ...)
cat(lm1.out, file = "lm1.out.tex")
```

I'd usually have an output directory where the tex file would be placed, but aside from this detail, that's my workflow. Then, when I want to use that file in my document, I use the LaTeX code "\input{lm1.out.tex}".

One reason for using that 2-step workflow is that the *automatically produced outreg table may not be exactly perfect*. Perhaps a variable label does not look right. Personally speaking, I wish the automatic table were always perfect. Practically, I accept it is not. So I leave open the opportunity that it might need to be revised by hand.

References

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Replication Information

```
R version 4.4.1 (2024-06-14)
Platform: x86_64-pc-linux-gnu
Running under: Ubuntu 24.04 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.26.so;  LAPACK version
    3.12.0

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=C               LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

time zone: Etc/UTC
tzcode source: system (glibc)

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] car_3.1-2         carData_3.0-5     rockchalk_1.8.157

loaded via a namespace (and not attached):
 [1] zip_2.3.1         nlme_3.1-166      knitr_1.48        xfun_0.47         stringi_1.8.4
 [6] minqa_1.2.8       xtable_1.8-4      buildtools_1.0.0  plyr_1.8.9        maketools_1.3.0
[11] sys_3.4.2         lme4_1.1-35.5     grid_4.4.1        abind_1.4-5       MASS_7.3-61
[16] openxlsx_4.2.6.1  compiler_4.4.1    Rcpp_1.0.13       lattice_0.22-6    nloptr_2.1.1
[21] foreign_0.8-87    kutils_1.73       splines_4.4.1     Matrix_1.7-0      tools_4.4.1
[26] boot_1.3-30
```

```
[1] "Warnings:"
```

```
## Don't delete this. It puts the interactive session options
## back the way they were. If this is compiled within a session
## it is vital to do this.
options(opts.orig)
```