Regression Table Demo

S-043 (Fall 2019)

This document demonstrates the different regression table methods, and talks about some weird wrinkles with using them. We use data from Project A, and fit data to the 8th grade students only. We have made a High School dummy variable.

Our two models we use for demo purposes have a HS term and no HS term:

```
modA <- lmer( esafe ~ 1 + (1 | ID), data=dat.g8)
modB <- lmer( esafe ~ 1 + HS + (1 | ID), data=dat.g8)
```

BTW, the libraries we focus on in this are

```
library(texreg)
library(stargazer)
library( lmerTest )
```

We next show how to get better summary output (according to some folks) and then we walk through some ways of making regression tables.

Getting p-values

The lmerTest package is a way of making R give you more complete output. We are going to load it, and then put the new lmer models into new variables so we can see how the different model fitting packages work with the regression table packages below.

```
library( lmerTest )
modB.T <- lmer( esafe ~ 1 + HS + (1 | ID), data=dat.g8)
modA.T <- lmer( esafe ~ 1 + (1 | ID), data=dat.g8)

summary( modB.T )
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: esafe ~ 1 + HS + (1 | ID)
##    Data: dat.g8
##
## REML criterion at convergence: 2746.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3883 -0.6156  0.2021  0.7628  1.7331
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  ID       (Intercept) 0.04809  0.2193
##  Residual             0.46459  0.6816
## Number of obs: 1305, groups:  ID, 26
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)  3.52798    0.08637 29.91033  40.846   <2e-16 ***
```

```
## HSTRUE      -0.29480    0.10787 25.77814  -2.733   0.0112 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##        (Intr)
## HSTRUE -0.801
```

Screenreg

Screenreg is fine. It looks a bit like raw output, but it is clear and clean. It will take models fit using lmer or lmerTest no problem.

```
screenreg(list(modA,modB))
```

```
##
## ================================================
##                        Model 1       Model 2
## ------------------------------------------------
## (Intercept)              3.35 ***      3.53 ***
##                         (0.06)        (0.09)
## HSTRUE                                -0.29 **
##                                       (0.11)
## ------------------------------------------------
## AIC                    2756.78       2754.79
## BIC                    2772.30       2775.49
## Log Likelihood        -1375.39      -1373.40
## Num. obs.              1305          1305
## Num. groups: ID          26            26
## Var: ID (Intercept)       0.07          0.05
## Var: Residual             0.46          0.46
## ================================================
## *** p < 0.001, ** p < 0.01, * p < 0.05
```

Comment: Note that the number of stars are different for the display vs the summary output! (Look at the HS coefficient for example.) Not good, it would seem.

This is because the p-values are calculated using the normal approximation by the screenreg command, and using the t-test with approximate degrees of freedom by `lmerTest`. This makes a difference. Consider the following, using the t statistics for the HS variable:

```
2 * pt( -2.733, df=25.77814 )
```

```
## [1] 0.0111831
```

```
2 * pnorm( -2.733 )
```

```
## [1] 0.006276033
```

One is below 0.01, and one is not. An extra star!

Using texreg and latex

This uses latex (which you would need to install). Then when you compile to a pdf, all is well. In the R block you need to include `results="asis"` to get the latex to compile right. E.g., "`r, results="asis"`" when

you declare a code block.

```
texreg(list(modA,modB), table=FALSE)
```

	Model 1	Model 2
(Intercept)	3.35***	3.53***
	(0.06)	(0.09)
HSTRUE		−0.29**
		(0.11)
AIC	2756.78	2754.79
BIC	2772.30	2775.49
Log Likelihood	-1375.39	-1373.40
Num. obs.	1305	1305
Num. groups: ID	26	26
Var: ID (Intercept)	0.07	0.05
Var: Residual	0.46	0.46

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Note that the `table=FALSE` puts the table right where you want it, not at some random spot latex things is nice. Latex likes to have "floating tables" which it puts where there is space.

Stargazer

```
library( stargazer )
stargazer(modA,modB, header=FALSE, type='latex')
```

Table 1:

	Dependent variable:	
	esafe	
	(1)	(2)
HS		−0.295***
		(0.108)
Constant	3.346***	3.528***
	(0.059)	(0.086)
Observations	1,305	1,305
Log Likelihood	−1,375.388	−1,373.397
Akaike Inf. Crit.	2,756.775	2,754.795
Bayesian Inf. Crit.	2,772.297	2,775.491
Note:	*p<0.1; **p<0.05; ***p<0.01	

One issue is stargazer does not include the random effect variances, so the output is quite limited for multilevel modeling. It also has less stringent conditions for when to put down stars. One star is below 0.10, two is below 0.05, and three is below 0.01. This is quite generous. Also it is using the normal approximation.

Stargazer with lmerTest

Stargazer with lmerTest is a bit fussy. This shows how to make it work if you have loaded the lmerTest package. Recall the lmerTest package makes your lmer commands have p-values and whatnot. But this means your new `lmer()` command is not quite the same as the old—and stargazer is expecting the old. You gix this by lying to R, telling it the new thing is the old thing. This basically works.

Now for stargazer, we need to tell it that our models are the right type. First note:

```
class( modB )
```

```
## [1] "lmerMod"
## attr(,"package")
## [1] "lme4"
```

```
class( modB.T)
```

```
## [1] "lmerModLmerTest"
## attr(,"package")
## [1] "lmerTest"
```

So we fix as follows:

```
library( stargazer )
class( modB.T ) = "lmerMod"
class( modA.T ) = "lmerMod"
stargazer(modA.T, modB.T, header=FALSE, type='latex' )
```

Table 2:

	Dependent variable:	
	esafe	
	(1)	(2)
HS		−0.295***
		(0.108)
Constant	3.346***	3.528***
	(0.059)	(0.086)
Observations	1,305	1,305
Log Likelihood	−1,375.388	−1,373.397
Akaike Inf. Crit.	2,756.775	2,754.795
Bayesian Inf. Crit.	2,772.297	2,775.491
Note:	*p<0.1; **p<0.05; ***p<0.01	