Table workflow

Paul E. Johnson[1] [2]

[1]Department of Political Science

[2]Center for Research Methods and Data Analysis, University of Kansas

2019

KU

Outline

Outline

Why it is Bad to Type Tables by Hand

- Dangerous to type from computer output into a table
- It is dangerous because
  1. Typographical errors, and
  2. Laziness $\rightarrow$ resistance to correction/revision
- Remind me to tell you some horrible stories

Scenarios

- In the best case scenario, tables "pop out" automatically, never need hand massaging
- In the pretty-good case scenario, tables "pop out" almost ready, needing only minor formatting
- In real life, we usually live in the pretty-good scenario
  - because software to make tables almost never does exactly what you need
  - but it gets close to what you need.
- R (R Core Team, 2017) has may programming tools to facilitate the effort.

Outline

Scenarios

In the lecture about recoding, I saved 2 files that I need.

```
wdir23 <- "../summeR-2-3-recoding/workingdata"
list.files(wdir23)
```

```
[1] "nes2004-20190828.rds"          "nes2004-objects-20190828.RData"
```

```
file.copy(from = wdir23, to = ".", recursive =
    TRUE, overwrite = TRUE)
```

```
[1] TRUE
```

I will use R's "file.copy" function to get them and put them in the "`workingdata`" folder

```
wdir23 <- "../summeR-2-3-recoding/workingdata"
list.files(wdir23)
```

Scenarios ...

```
[1] "nes2004-20190828.rds"          "nes2004-objects-20190828.RData"
```

```
file.copy(from = wdir23, to = ".", recursive =
    TRUE, overwrite = TRUE)
```

```
[1] TRUE
```

- I don't have a good system worked out to only copy in the newest files, so I have to check what I got:

```
wdir <- "workingdata"
fl <- list.files(wdir, full.names = TRUE)
```

- See what we got:

```
fl
```

Scenarios ...

```
[1] "workingdata/nes2004-20190828.rds"
    "workingdata/nes2004-objects-20190828.RData"
```

- Check the workspace

```
ls()
```

```
[1] "fl"         "opts.orig" "par.orig"  "pjmar"      "tdir"
    "wdir"       "wdir23"
```

- Use `readRDS` to open the rds file (`readRDS` will only create object `anes2`)

```
anes2 <- readRDS(fl[1])
```

- Check the workspace

```
ls()
```

Scenarios ...

```
[1] "anes2"      "fl"         "opts.orig" "par.orig"   "pjmar"
    "tdir"       "wdir"       "wdir23"
```

- Using `load` to open the `RData` file can create many objects, whatever was in the saved file (and we have no control over object names)

```
load(fl[2])
```

- Check the workspace

```
ls()
```

```
[1] "anes2"      "fl"         "mod1"       "mod2"       "mod3"
    "opts.orig" "par.orig"   "pjmar"
[9] "tdir"       "wdir"       "wdir23"
```

- I'll remove the file list `fl`, to keep the workspace tidy

```
rm(fl)
```

Outline

mod1, mod2, mod3 were already estimated

```
summary(mod1)
```

```
Call:
lm(formula = th.bk ~ V043250, data = anes1)

Residuals:
     Min       1Q   Median       3Q      Max
-108.821  -42.753   -2.003   42.905  103.340

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -6.84133    4.59553  -1.489    0.137
V043250      0.18426    0.09181   2.007    0.045 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 53.89 on 1189 degrees of freedom
  (21 observations deleted due to missingness)
Multiple R-squared:  0.003376,  Adjusted R-squared:  0.002538
F-statistic: 4.028 on 1 and 1189 DF,  p-value: 0.04498
```

```
summary(mod2)
```

mod1, mod2, mod3 were already estimated ...

```
Call:
lm(formula = th.bk ~ V043250 + V041109A, data = anes1)

Residuals:
     Min       1Q   Median       3Q      Max
-113.174  -42.222   -2.782   42.478  107.164

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.08501    4.83135  -0.639   0.5232
V043250      0.19128    0.09166   2.087   0.0371 *
V041109AF   -7.71339    3.12318  -2.470   0.0137 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 53.77 on 1188 degrees of freedom
  (21 observations deleted due to missingness)
Multiple R-squared:  0.008467,  Adjusted R-squared:  0.006798
F-statistic: 5.072 on 2 and 1188 DF,  p-value: 0.006404
```

As we go along, I'm going to need parameter estimate names, so I'll collect them up here.

```
varLabs <- c("V043250" = "Age", "V041109AF" = "Gender Female",
    "V043116WD" = "Dem, Weak",  "V043116ID" = "Indep Lean Dem",
    "V043116I" = "Independent", "V043116IR" = "Indep Lean Repub",
    "V043116WR" = "Repub, Weak", "V043116R" = "Repub", "V043116SR" =
    "Repub, Str")
```

One regression model in a table, wide format

	Model 1	
	Estimate	(S.E.)
(Intercept)	-6.841	(4.596)
Age	0.184*	(0.092)
N	1191	
RMSE	53.885	
R^2	0.003	

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

One table with 2 of the Models, wide format

	Model 1 Estimate	(S.E.)	Model 2 Estimate	(S.E.)
(Intercept)	-6.841	(4.596)	-3.085	(4.831)
Age	0.184*	(0.092)	0.191*	(0.092)
Gender Female	_		-7.713*	(3.123)
N	1191		1191	
RMSE	53.885		53.770	
R^2	0.003		0.008	
adj R^2	0.003		0.007	

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

What Manner of Magic is This, You Ask?

```
library(rockchalk)
## The first model's table
or40b <- outreg(list("Model 1" = mod1), tight =
    FALSE, varLabels = varLabs)
## The second model's table
or45b <- outreg(list("Model 1" = mod1, "Model 2"
    = mod2), tight = FALSE, varLabels = varLabs,
    print.results = FALSE)
cat(or45b)
```

What other wonderful things are possible?

- Smart people prepare documents with LaTeX, Sweave (just ask us)
  http://pj.freefaculty.org/guides/stat/Regression/Multicollinearity/Multicollinearity-1-lecture.pdf
  http://pj.freefaculty.org/R/gloating
- Original table-making programs were designed by LaTeX users for LaTeX
- Now, reasonable HTML output can be obtained from several packages. That can be imported into MS Word.

Here's the nice looking table

	Model 1 Estimate	(S.E.)	Model 2 Estimate	(S.E.)	Model 3 Estimate	(S.E.)
(Intercept)	-6.841	(4.596)	-3.085	(4.831)	-53.310***	(10.350)
Age	0.184*	(0.092)	0.191*	(0.092)	0.029	(0.088)
Gender Female	_		-7.713*	(3.123)	-0.388	(2.947)
V043116L	_		_		-1.538	(10.465)
V043116SL	_		_		25.501*	(10.323)
V043116M	_		_		49.845***	(9.924)
V043116SC	_		_		79.650***	(10.236)
V043116C	_		_		103.708***	(10.146)
V043116EC	_		_		111.478***	(12.379)
N	1191		1191		859	
RMSE	53.885		53.770		42.795	
R^2	0.003		0.008		0.409	
adj R^2	0.003		0.007		0.404	

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

How to do that?

```
or70 <- outreg(list("Model 1" = mod1, "Model 2" =
    mod2, "Model 3" = mod3), tight = FALSE,
    varLabels = varLabs)
```

outreg is my entry in the regression table contest

There are plenty of packages that do the same

- xtable: an R classic. It can make nice looking output from any kind of table, but does not customize to regression so well
- apsrtable: from a former Washington University grad student (like me, much younger)
- memisc: another long standing R contributor who was far ahead of the rest of us on understanding social science applications of R (look for "mtable" and the function "toLatex".
- texreg: relatively newer package, author is writing "modules" to deal with all kinds of regression models he can find.
- More notes on this at http://pj.freefaculty.org/guides/Rcourse/regression-tables-1/regression-tables-1.pdf

KU

Vital to use Automatic Table Software!

- Fewer typographical errors
- More replicable research
- More easily revised term papers and dissertations

What if there's not magic-table-making function?

- In CRMDA, we've found 3 approaches.
  - Find an undergrad volunteer who's willing to type the tables
  - Manipulate table output to gather all of the coefficients and then write functions that create the tables we need.
    - sometimes sufficient to write out csv files, but
    - I'd rather produce an actual table
  - Find a function that is similar to our need, and then revise their code to do what we want
- We've been studying ways to create Structural Equation tables in kutils. Some exciting breakthroughs are illustrated in the SEM presentation.

KU

References

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Session

```
sessionInfo()
```

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 19.04

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
     LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=en_US.UTF-8     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

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

other attached packages:
[1] rockchalk_1.8.144
```

Session ...

```
loaded via a namespace (and not attached):
 [1] Rcpp_1.0.1        lattice_0.20-38 MASS_7.3-51.4   grid_3.6.0
     plyr_1.8.4        nlme_3.1-140
 [7] xtable_1.8-4      stats4_3.6.0    zip_2.0.2       carData_3.0-2
     minqa_1.2.4       nloptr_1.2.1
[13] Matrix_1.2-17     pbivnorm_0.6.0  boot_1.3-22     openxlsx_4.1.0
     splines_3.6.0     lme4_1.1-21
[19] tools_3.6.0       foreign_0.8-71  kutils_1.69     compiler_3.6.0
     mnormt_1.5-5      lavaan_0.6-3
```