second R-03

Paul E. Johnson$^1$  $^2$

$^1$Department of Political Science

$^2$Center for Research Methods and Data Analysis, University of Kansas

2015
Retrieve the working data we stached a little while ago

```r
mydata <- readRDS(paste0("../workingdata/nes2004--", "20150813", ".rds"))
load(paste0("../workingdata/nes2004--objects--", "20150813", ".RData"))
```
As we go along, I’m going to need parameter estimate names, so I’ll collect them up here.

```r
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")
```
In your paper, a regression table should look like this

<table>
<thead>
<tr>
<th>Model 1</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimate (S.E.)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Intercept) -6.841</td>
<td>4.596</td>
<td></td>
</tr>
<tr>
<td>Age 0.184*</td>
<td>0.092</td>
<td></td>
</tr>
<tr>
<td>N 1191</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RMSE 53.885</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R^2$ 0.003</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p \leq 0.05$ * $p \leq 0.01$ *** $p \leq 0.001$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Take 2 Previous Models, for example

<table>
<thead>
<tr>
<th></th>
<th>Model 1</th>
<th></th>
<th>Model 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Estimate</td>
<td>(S.E.)</td>
<td>Estimate</td>
<td>(S.E.)</td>
</tr>
<tr>
<td>(Intercept)</td>
<td>-6.841</td>
<td>(4.596)</td>
<td>-3.085</td>
<td>(4.831)</td>
</tr>
<tr>
<td>Age</td>
<td>0.184*</td>
<td>(0.092)</td>
<td>0.191*</td>
<td>(0.092)</td>
</tr>
<tr>
<td>Gender Female</td>
<td>.</td>
<td></td>
<td>-7.713*</td>
<td>(3.123)</td>
</tr>
<tr>
<td>N</td>
<td>1191</td>
<td></td>
<td>1191</td>
<td></td>
</tr>
<tr>
<td>RMSE</td>
<td>53.885</td>
<td></td>
<td>53.770</td>
<td></td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.003</td>
<td></td>
<td>0.008</td>
<td></td>
</tr>
<tr>
<td>adj $R^2$</td>
<td>0.003</td>
<td></td>
<td>0.007</td>
<td></td>
</tr>
</tbody>
</table>

*p ≤ 0.05 **p ≤ 0.01 ***p ≤ 0.001
What Manner of Magic is This, You Ask?

```r
library(rockchalk)
## The first model's table
outreg(list("Model 1" = mod1), tight = FALSE, varLabels = varLabs)
## The second model's table
library(rockchalk)
outreg(list("Model 1" = mod1, "Model 2" = mod2), tight = FALSE, varLabels = varLabs)
```
What other wonderful things are possible?

- Smart people prepare documents with \LaTeX, Sweave (just ask us)
  [http://pj.freefaculty.org/R/gloating](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

<table>
<thead>
<tr>
<th></th>
<th>Model 1</th>
<th>Model 2</th>
<th>Model 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Estimate</td>
<td>(S.E.)</td>
<td>Estimate</td>
</tr>
<tr>
<td>(Intercept)</td>
<td>-6.841</td>
<td>(4.596)</td>
<td>-3.085</td>
</tr>
<tr>
<td>Age</td>
<td>0.184*</td>
<td>(0.092)</td>
<td>0.191*</td>
</tr>
<tr>
<td>Gender Female</td>
<td>.</td>
<td>-.713*</td>
<td>(3.123)</td>
</tr>
<tr>
<td>Dem, Weak</td>
<td>.</td>
<td>.</td>
<td>32.542***</td>
</tr>
<tr>
<td>Indep Lean Dem</td>
<td>.</td>
<td>.</td>
<td>27.332***</td>
</tr>
<tr>
<td>Independent</td>
<td>.</td>
<td>.</td>
<td>54.114***</td>
</tr>
<tr>
<td>Indep Lean Repub</td>
<td>.</td>
<td>.</td>
<td>88.469***</td>
</tr>
<tr>
<td>Repub, Weak</td>
<td>.</td>
<td>.</td>
<td>93.245***</td>
</tr>
<tr>
<td>Repub, Str</td>
<td>.</td>
<td>.</td>
<td>120.191***</td>
</tr>
<tr>
<td>N</td>
<td>1191</td>
<td>1191</td>
<td>1174</td>
</tr>
<tr>
<td>RMSE</td>
<td>53.885</td>
<td>53.770</td>
<td>34.269</td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.003</td>
<td>0.008</td>
<td>0.600</td>
</tr>
<tr>
<td>adj $R^2$</td>
<td>0.003</td>
<td>0.007</td>
<td>0.598</td>
</tr>
</tbody>
</table>

* $p \leq 0.05$  ** $p \leq 0.01$  *** $p \leq 0.001$
How to do that?

```r
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.

Why is it Urgent to make Automatic Tables?

- Fewer typographical errors
- More replicable research
- More easily revised term papers and dissertations
- PJ: Don’t forget that tragic story about...
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
  - I’ve had standing offers to make table functions for Structural Equation Modelers, but nobody will work hard enough to help me understand what’s needed and find examples we should imitate.
On Screen versus In A File

- Confusing, but the graph you see on the screen is not easily and correctly saved into a file.
- Saving distorts margins, spaces between symbols.
- One of the primary reasons I wrote Rtips.html (http://pj.freefaculty.org/R/Rtips.html) was the difficulty I had in understanding advice about how to save graphs, and what formats ought to be used.
- Simply put,
  - this is difficult, and
  - you can make it easier by doing what I recommend.
Here’s my template, in R.

```r
if (SAVEME) pdf(file = "afilename.pdf", paper = "special", height = 7, width = 7, onefile = FALSE, family = "Times")
plot(y ~ x, data = dat, xlab = "My super plot", ylab = "My other variable", main = "smarter than you")
lines(z ~ x, data = dat, col = "green")
if (SAVEME) dev.off()
```

I realized that is best in 2010. First example:
These ones are more straightforward:
Get the basic idea. Run 3 lines

```r
df("myfirst.pdf")
hist(rnorm(1000), main = "1000 random normals")
dev.off()
```

- When you run that, nothing will show on the screen. Your graph is “showing” inside the file, however.
- That should make a file in a histogram pop out in your current working directory. Check:

```r
list.files()
```

- Use a PDF viewer to look at “myfirst.pdf”.
Why is my more elaborate way better?

```r
if (SAVEME) pdf(file = "afilename.pdf", paper = "special", height = 7, width = 7, onefile = FALSE, family = "Times")
plot(y ~ x, data = dat, xlab = "My super plot", ylab = "My other variable", main = "smarter than you")
lines(z ~ x, data = dat, col = "green")
if (SAVEME) dev.off()
```

First, it is relatively unobtrusive. IF SAVEME is either undefined or you run “SAVEME <- FALSE”, then the graph is just shown on the screen, as always.
When you set “SAVEME <- TRUE”, then the plotting commands that follow are drawn into a graph of just the right size.
That onefile argument is a little interesting

- If you put
  - onefile = TRUE, and
  - file = "mygreatplot.pdf"
  then ALL of the plots until dev.off() will be pages in one document.

- If you
  - onefile = FALSE, and
  - file = "mygreatplot-%03d.pdf"

Then you get separate plot files "mygreatplot-001.pdf", "mygreatplot-002.pdf"
You should want PDF graphs, but if you don’t …

- PDF is a scalable vector graphic format (the successor to “encapsulated postscript”)
- “Picture” formats, like “jpg” or “png” are “bitmaps”, they do not scale.
- R does have a device to create those formats

```r
if (SAVEME) png(file = "afilename.png", height = 800, width = 800)
## draw your plot here
if (SAVEME) dev.off()
```

A picture file requires height and width in pixels, NOT INCHES. pdf and postscript require inches.
I don’t do this often, I find it much nicer to export to pdf and convert to other formats from there with ImageMagick
**Saving Plots into Files**

My Real Workflow has a Couple of other Blandishments

# 1

- At the top of an R file that uses graphs, I set some default options.

```r
pdf.options(onefile = FALSE, family = "Times", paper = "special", height = 4, width = 6)
```

- In each (SAVEME) pdf command, I don’t have to type so much

```r
if (SAVEME) pdf(file = "afilename.pdf")
## make a plot, silly
if (SAVEME) dev.off()
```

- The pdf arguments can be re-inserted. Graphs are tall and narrow, a full page

```r
if (SAVEME) pdf(file = "afilename.pdf", height = 10, width = 7)
## make a plot, silly
if (SAVEME) dev.off()
```

- I don’t always remember to do this, but I think good students should.
My Real Workflow has a Couple of other Blandishments #2

- I don’t want all the plots to show up on my current working directory, R.
- In a real project, I’ll have these subfolders
  - R
  - output
  - writeup
  - workingdata
- And within my code, I’ll send plots directly over to the output folder (possibly a subfolder in there).
It is as simple as making sure the output folder exists, and then writing:

```r
if (SAVEME) pdf(file = "../output/afilename.pdf", paper = "special", height = 7, width = 7, onefile = FALSE, family = "Times")
```
My Real Workflow has a Couple of other Blandishments #2b

- Sometimes I need a GRA to run some code and I don’t know for sure if they have the output directory. So I put this at the top of the script

```r
outdir <- "../output/
if (!file.exists(outdir)) dir.create(outdir, recursive = TRUE)
```

- Later, when I want to write something, I create the file name on-the-fly with `paste0`:

```r
if (SAVEME) pdf(file = paste0(outdir, "afilename.pdf"), paper = "special", height = 7, width = 7, onefile = FALSE, family = "Times")
```

That uses the `paste0` function to paste together 2 character strings, `outdir` and the name.
My Real Workflow has a Couple of other Blandishments #2b ...

- Why `paste0()`? Answer: `paste()` inserts a space by default, but we could have written `paste(outdir, "afilename.pdf", sep = "")`.

- Why bother? I can adjust the output directory on-the-fly. Generate 10s or 100s of plots for separate parts of a project. I might want to place them into subdirectories.

```r
outdir <- "../output/part1/eastcoast/"
if (!file.exists(outdir)) dir.create(outdir, recursive = TRUE)
if (SAVEME) pdf(file = paste0(outdir, "afilename.pdf"), paper = "special", height = 7, width = 7, onefile = FALSE, family = "Times")
```

- That way’s good because you can easily change the output directory by editing one line. Plots will end up in “output/part1/eastcoast” and only line 1 must be altered.
Concluding Comments about Devices

- Times will change, new devices will come along. PDF now default, maybe tomorrow SVG or others.
- Other formats I use sometimes
  - xfig
  - svg
- SVG is appealing (browser compatibility, edit with Inkscape)
Sometimes a well placed \( \Gamma \) or \( Pi \) pushes your plot over the top

I don’t want to spend a lot of time on this, because it is almost mind-numbingly complicated in some ways, but let’s just run an example.

```r
plot(1:10, 1:10, type="n")
text(4, 5, expression(paste(alpha ,"," is alpha")))
text(7, 3, expression(paste(beta * alpha ,"," is beta * alpha")))
text(3, 6, expression(paste(frac(beta, alpha),"," is frac(beta, alpha"))))
text(2,9, expression(paste(hat(y)[i] == hat(beta)[0]+ hat(beta)[1]*x[1])))
```
Sometimes a well placed Γ or $Pi$ pushes your plot over the top

\[ \hat{y}_1 = \hat{\beta}_0 + \hat{\beta}_1 x_1 \]

$\frac{\beta}{\alpha}$ is frac(beta,alpha)

$\alpha$ is alpha

$\beta \alpha$ is beta * alpha
A Few plotmath Tips

- Two Equal Signs (== outputs =)
- Use hard brackets for subscripts, ^ for superscripts
- Want * to show? Type %*%
- Want · ? Type cdot
- Want \((x - 1, y_1)\)? Type group("(" , list(x[1], y[1]), ")")
What To Practice Today?

- In today's practice session, I suggest you walk through the examples from the WorkingExamples folder. `distributions-normalAndTCompared.R` and `distributions-GammaVersusNormal-1.R`. They will drop pdf or png output in your working directory, and you can practice creating another directory for graphs.