The Political Methodologist

Newsletter of the Political Methodology Section
American Political Science Association
Volume 20, Number 1, Fall 2012

Editors:
Jake Bowers, University of Illinois at Urbana-Champaign
jwbowers@illinois.edu
Wendy K. Tam Cho, University of Illinois at Urbana-Champaign
wendycho@illinois.edu
Brian J. Gaines, University of Illinois at Urbana-Champaign
bjgaines@illinois.edu

Editorial Assistant:
Ashly Adam Townsen, University of Illinois at Urbana-Champaign
townsen5@illinois.edu

Contents

It is autumn, and so election and football seasons are under way, and it is time for another issue of *The Political Methodologist*. This issue emphasizes software. Matthew Blackwell and Maya Sen provide a primer on how to tackle really large datasets, emphasizing some helpful R packages. Next, Stata users should be pleased to see advice from Eldad Davidov on how to estimate structural equation models in Stata 12. Is Stata 12 also the right product for those who like their variance in time, not space? Ellen M. Key and Matthew J. Lebo compare it to RATS, and many readers will surely appreciate the advice on whether or not they can, at last, make do without also buying specialized time-series software. Finally, our section president, who will not be facing off against any opponents in debates over the next few weeks, recaps what would have happened at the business meetings at the APSA meetings, had we actually had APSA meetings this year. Included is the list of the section's prize winners, and we editors heartily congratulate all of the deserving recipients. As always, we are grateful to the contributors, and we encourage readers to contact us with submissions, ideas for submissions, and other feedback or advice you have.

The Editors

Articles

Large Datasets and You: A Field Guide

Matthew Blackwell and Maya Sen
University of Rochester
m.blackwell@rochester.edu and msen@ur.rochester.edu

A wind of streaming data, social data and unstructured data is knocking at the door, and we're starting to let it in. It's a scary place at the moment.

Unidentified bank IT executive, as quoted by *The American Banker*

Error: cannot allocate vector of size 75.1 Mb

R

Introduction

The last five years have seen an explosion in the amount of data available to social scientists. Thanks to Twitter, blogs, online government databases, and advances in text analysis techniques, data sets with millions and millions of observations are no longer a rarity (Lohr, 2012). Although a blessing, these extremely large data sets can cause problems for political scientists working with standard statistical software programs, which are poorly suited to analyzing big data sets. At best, analyzing massive data sets can result in prohibitively long computing time; at worst, it can lead to repeated crashing, making anything beyond calculating the simplest of summary statistics impossible. The volume of data available to researchers is, however, growing faster than computational capacities, making developing techniques for how to handle "Big Data" is essential.

In this article, we describe a few approaches to handling these Big Data problems within the R programming language, both at the command line prior to R and after we fire up R.[1] We show that handling large datasets is about either (1) choosing tools that can shrink the problem or (2) fine-tuning R to handle massive data files.

Why Big Data Present Big Problems

It is no secret that current statistical software programs are not well equipped to handle extremely large datasets. R (R Development Core Team, 2012), for example, works by holding objects in its virtual memory, and big datasets are often larger then the size of the RAM that is available to researchers using their operating software. Many of these problems are compounded by the fact that not only do the raw loaded data take up RAM once loaded, but so do any analyses. Basic functions like `lm` and `glm` store multiple copies of the data within the workspace. Thus, even if the original data set is smaller than the allocated RAM, once multiple copies of the data are stored (via an `lm` function, for example), R will quickly run out of memory.

Purchasing more RAM is an option, as is moving to a server that can allocate more RAM. In addition, moving from a 32-bit to a 64-bit version of R can alleviate some problems. (Unix-like systems—e.g, Linux, Mac OS X—impose a 4Gb limit on 32-bit systems and no limit on 64-bit systems. On Windows, the limits are 2Gb and 4Gb for 32-bit and 64-bit respectively.) However, these fixes largely postpone the inevitable—scholars will (hopefully) continue to collect even larger datasets and push the boundaries of what is computationally possible. This will be compounded by running increasing numbers of more sophisticated analyses. In addition, all R builds have will have a maximum vector length of $2^{31} - 1$, or around two billion. A combination of any of these memory issues will result in the dreaded "cannot allocate vector size" error, which will swiftly derail a researcher's attempt at analyzing a large data set.

First Pass: Subset the Data

As simple as it sounds, the easiest work-around to the Big Data Problem is to avoid it if possible. After all, data files are often much larger than we need them to be; they usually contain more variables than we need for our analysis, or we plan to run our models on subsets of the data. In these cases, loading the excess data into the R workspace only to purge it (or ignore it) with a few commands later is incredibly wasteful in terms of memory. A better approach is to remove the excess data from the data file *before* loading it into R. This often appears difficult because we are used to performing data manipulation using R (this is probably why we are using R in the first place!). Luckily, there are a

[1] Here we focus on R, but this problem extends to other memory-based statistical environments, namely Stata. Other statistical packages, such as SAS and SPSS, have a file-based approach, which avoids some of these memory allocation issues.

handful of Unix command-line utilities that can help parse data files without running into memory issues.

We demonstrate this using a data file called `iris.tab`, which is tab-delimited and contains many rows. The dataset measures, in centimeters, (1) sepal length and width and (2) petal length and width for 50 flowers from three species of irises (Fisher, 1936). We can use the Unix `head` command to investigate the first ten lines of the data file:

```
$ head iris.tab
```

```
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
5.1          3.5         1.4          0.2         setosa
4.9          3           1.4          0.2         setosa
4.7          3.2         1.3          0.2         setosa
4.6          3.1         1.5          0.2         setosa
5            3.6         1.4          0.2         setosa
5.4          3.9         1.7          0.4         setosa
4.6          3.4         1.4          0.3         setosa
5            3.4         1.5          0.2         setosa
4.4          2.9         1.4          0.2         setosa
```

Suppose that we only need the first four numeric variables for our analysis (we don't care about the iris species). We can remove the `Species` variable using the `cut` utility, which takes in a data file and a set of column numbers and returns the data file with only those columns.[2] For example, the following command:

```
$ head iris.tab | cut -f1,2,3,4
```

will return the data without the `Species` variable:

```
Sepal.Length Sepal.Width Petal.Length Petal.Width
5.1          3.5         1.4          0.2
4.9          3           1.4          0.2
4.7          3.2         1.3          0.2
4.6          3.1         1.5          0.2
5            3.6         1.4          0.2
5.4          3.9         1.7          0.4
4.6          3.4         1.4          0.3
5            3.4         1.5          0.2
4.4          2.9         1.4          0.2
```

A few points of clarification. First, note that we are "piping" the output of the `head` command to the `cut` command to avoid running `cut` on the entire dataset.[3] This is useful for testing our approach at the command line. Once we have our syntax, we can run `cut` on the entire data as follows: `cut -f1,2,3,4 iris.tab >> iris-new.tab`. Note that this will create a new file, which may be quite large. Second, the `-f1,2,3,4` argument specifies which columns to keep and can be specified by ranges such as `-f1-4`.

In addition to removing variables, we often want to remove certain rows of the data (say, if we were running the analysis only on a subset of the data). To do this efficiently on large text-based data files, we can use `awk`, which comes standard on most Unix systems. The `awk` utility is a powerful data extraction tool, but we will only show its most basic features for selecting observations from a dataset. The command requires an expression that describes which rows of the data file to keep. For instance, if we wanted to keep the top row (with the variable names) and any row with a `Sepal.length` greater than 5, we could use the following:

```
$ head iris.tab | awk 'NR ==1 || $1 > 5'
```

This gives the following result:

```
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
5.1          3.5         1.4          0.2         setosa
5.4          3.9         1.7          0.4         setosa
```

Here, `NR` refers to the row number, so that `NR == 1` selects the first row of the file, which contains the variable names. The `$` operator refers to column numbers, so that `$1 > 5` selects any row where the first column is greater than 5. The `||` operator simply tells `awk` to select rows that match either of the two criteria.

There are many ways to preprocess our data before loading it into R to reduce its size and make it more manageable. Besides these Unix tools we've discussed there are more complicated approaches, including scripting languages such as Python or relational database interfaces with R such as `sqldf` or `RODBC`. These are more powerful approaches, but often simple one-line Unix commands can wrangle data as effectively and more efficiently. In any case, this approach can resolve many of the supposed Big Data problems out in the wild without any further complications. There are times, though, when Big Data problems remain, even after whittling away the data to only the necessary bits.

The Bite-Sized-Chunk Approach to Big Data

It's impossible to eat a big steak in one bite; instead, we cut our steak into smaller pieces and eat it one bite after another. Nearly all direct fixes to the Big Data conundrum rely on the same principle: if we need all the data (not just some subsets of it), we can break up the data into more manageable chunks that are then small enough to fit within the allocated memory. Essentially, we upload into the workspace only as much of the data as is necessary to run specific analyses. Indeed, many operations can be done piece-meal or sequentially on different chunks on data—e.g., a thousand rows a time, or only a few columns. For some simple calculations, such as the sample mean, this process is straightforward. For others, though, this is more daunting—how do we piece together regressions from different subsets of the data?

Fortunately, there are a handful of packages that have facilitated the use of big data in R and they work by automating and simplifying the bite-sized data approach.[4] Gener-

[2] We can also selectively load columns using the R command `read.table`; however, the approach we suggest is more efficient and is compatible with the `bigmemory` package below.

[3] In Unix environments, the "pipe character" (|) takes the output of one command and passes it as an input the next command.

[4] This bite-sized-chunk approach, sometimes called "split-apply-combine" (Wickham, 2011), has a strong history in computer science. Google's `MapReduce` programming model is essentially the same approach.

ally, they allow most of the data to stay in the working directory (on a file in the hard drive); this means that the data do not have to be loaded into the memory (thereby using up valuable allocated memory). They create an R object within the memory that acts like a `matrix` object, but in reality it's just a way for you to efficiently access different parts of the data file (still on the hard drive). In addition, they provide intuitive functions that allow users to access the data and calculate summary statistics of the entire data.

A Romp Through `bigmemory`

The `bigmemory` package (Kane and Emerson, 2011), along with its sister packages, allows users to interact with and analyze incredibly large datasets. To illustrate, we work through an example using U.S. lending data from 2006. Federal law mandates that all applications for a real estate mortgage be recorded by the lending agency and reported to the relevant U.S. agencies (who then make the data publicly available). This results in a wealth of data—some 11 million observations per year. However, the size of this data means that loading the data into an R workspace is essentially impossible, let alone running linear or generalized linear models.

To get started, we load the data into R as a `big.matrix` object. With large datasets, it is important to create a "backing file," which will reduce the amount of memory that R needs to access the data. To do this, we load the relevant packages and use the `read.big.matrix` function:

```
> library(bigmemory)

bigmemory >= 4.0 is a major revision since 3.1.2; please see package
biganalytics and http://www.bigmemory.org for more information.

> library(biglm)
Loading required package: DBI
> library(biganalytics)
> library(bigtabulate)
> mortgages <- read.big.matrix("allstates.txt", sep = "\t",
+                              header = TRUE,
+                              type = "double",
+                              backingfile = "allstates.bin",
+                              descriptor = "allstates.desc")
```

The resulting object, `mortgages`, is a `big.matrix` object and takes up very little memory in R. The process of creating this backing file takes around 25–30 minutes with these data, but after this is complete, it is fast and easy to load the `big.matrix` object in a fresh R session using the following:

```
> library(bigmemory)

bigmemory >= 4.0 is a major revision since 3.1.2; please see package
biganalytics and http://www.bigmemory.org for more information.

> library(biglm)
Loading required package: DBI
> library(biganalytics)
> library(bigtabulate)
> xdesc <- dget("allstates-clean.desc")
> mortgages <- attach.big.matrix(xdesc)
```

This process takes just seconds with same low memory overhead.

In many ways, we can interact with `big.matrix` objects in much the same way we do with `matrix` objects:

```
> dim(mortgages)
[1] 10875481       40
> head(mortgages)[,1:7]
     ID agency loan.type property loan.purpose occupancy loan.amount
[1,] 1      1         1        1            1         1          36
[2,] 1      1         1        1            1         1          61
[3,] 1      1         1        1            2         1          10
[4,] 1      1         1        1            3         1          76
[5,] 1      1         1        1            3         1         148
[6,] 1      1         1        1            3         1         132
```

We can see here that this dataset has over 10.8 million observations with 40 variables. Using functions from the sister package `biganalytics` (Emerson and Kane, 2010), we can quickly and easily find summary statistics on different columns:

```
> mean(mortgages[,"income"])
[1] 100.3027
> median(mortgages[,"income"])
[1] 76
```

These calculations take just a few seconds on Apple iMac with a 3.06 Ghz Intel Core 2 Duo processor and 4Gb of RAM. Note that `big.matrix` objects mimic `matrix` objects so we cannot use the `mortgages$income` syntax as we would with a `data.frame`.

Our data analyses often require more than simple summary statistics and `bigmemory` has a way to efficiently subset data. This is the `mwhich` command, which returns a vector of indices that match a set of criteria similar to the base `which` command. The function takes in a `big.matrix` object, a variable name, a value, and a comparison to perform on that value. For instance, with a tradition R `matrix`, we might choose the males with `which(mortgages[,"sex"] == 1)`, but the syntax is slightly different with a `big.matrix`:

```
> median(mortgages[mwhich(mortgages, "sex", 1, "eq"), "loan.amount"])
[1] 141
> median(mortgages[mwhich(mortgages, "sex", 2, "eq"), "loan.amount"])
[1] 127
```

The first call to `mwhich` selects the observations with `sex` equal (`eq`) to 1. The `mwhich` function can compare multiple variables or values at once. This allows us to create complex cross-tabulations on extremely large datasets with minimal memory or speed overhead.

Finally, the `biganalytics` also provides a method for passing `big.matrix` objects to the `biglm` function (Lumley, 2011), which efficiently computes ordinary least squares on large datasets. This approach is similar to running a regression with a normal `matrix`:

```
> mod1 <- biglm.big.matrix(high.rate ~ sex, data = mortgages)
> summary(mod1)
Large data regression model: biglm(formula = formula, data = data,...)
Sample size =  10875481
              Coef    (95%    CI)    SE p
(Intercept) 0.2331 0.2324 0.2338 4e-04 0
sex         0.0371 0.0367 0.0376 2e-04 0
```

Remarkably, this regression takes less than a minute to run (on all 10.8 million observations!). In general, the `bigmemory` suite of functions helps users with extremely large datasets avoid per-computation memory and speed issues by creating a file-backed version of the data. Further, they have put together a great set of functions to help users with the most common statistical tasks. These tasks can be sped up even more by parallel processing through the `foreach` package.

Conclusion

As journalists and public intellectuals have noted, the age of "Big Data" has dawned. However, advances in computational speed and memory size are not moving fast enough to allow analysis of this data with traditional techniques. To this end, researchers analyzing extremely large data sets will have to start start using different kinds of approaches—parallel processing, big data packages. Here, we have reviewed only some techniques in tackling "Big Data," but there are others. Ultimately, as the wealth of data only grows, those who can quickly and easily digest this information will be able to explore new and exciting research questions; those who can't will, unfortunately, be left out.

References

Emerson, John W. and Michael J. Kane. 2010. *biganalytics: A Library of Utilities for Big.matrix Objects of Package bigmemory*. R package version 1.0.14. http://CRAN.R-project.org/package=biganalytics

Fisher, R.A. 1936. "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Human Genetics* 7(2):179–188.

Kane, Michael J. and John W. Emerson. 2011. *bigmemory: Manage Massive Matrices with Shared Memory and Memory-mapped Files*. R package version 4.2.11. http://CRAN.R-project.org/package=bigmemory

Lohr, Steve. 2012. "The Age of Big Data." *The New York Times* p. SR1. http://www.nytimes.com/2012/02/12/sunday-review/big-datas-impact-in-the-world.html

Lumley, Thomas. 2011. *biglm: Bounded Memory Linear and Generalized Linear Models*. R package version 0.8. http://CRAN.R-project.org/package=biglm

R Development Core Team. 2012. *R: A Language and Environment for Statistical Computing*.Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-07-0. http://www.R-project.org/

Wickham, Hadley. 2011. "The Split-Apply-Combine Strategy for Data Analysis." *Journal of Statistical Software* 40(1):129. http://www.jstatsoft.org/v40/i01/

A Brief Introduction to the Structural Equation Modeling (SEM) Module in Stata 12

Eldad Davidov
University of Zurich
Davidov@soziologie.uzh.ch

In 2011 StataCorp launched Stata 12, the newest version of its flagship product, which now includes a large structural equation modeling (SEM) module. In this newest edition, the authors of the Stata manual state that "Structural equation modeling is a way of thinking, a way of writing, and a way of estimating" (StataCorp, 2011, p. 1). Thus, it is admirable that the designers of Stata took upon themselves to incorporate such a large, new, and useful module, and by doing this joined the ranks of both recent and established structural equation modeling software packages already available on the market (e.g., Amos, which has become an SPSS module: Arbuckle, 2009 or Byrne, 2009; Mplus: Muthén and Muthén, 1998–2010; Lisrel: Jöreskog and Sörbom, 1986; EQS: Bentler, 1995; Open MX: Boker et al., 2011). In the following, I will briefly sketch some of the SEM features offered by Stata 12. This is by no means a conclusive overview of everything that Stata offers in terms of SEM; neither do I try to provide a detailed discussion or explanation of terms from the SEM literature. Rather, I merely list some of its features. For a comprehensive overview of all the SEM features found in Stata 12, the reader is referred to the Stata manual.

First and foremost, Stata includes, in addition to its (easy to use) SEM syntax language, a graphical user interface, that lets one build structural equation models graphically. All users who dislike syntax commands and prefer visual model building will find this very good news. With its graphical interface, each and every path and parameter become visible, and it is easy to determine whether a path implied by the theory has been omitted or forgotten, or in contrast, whether there is a superfluous path that may be constrained. The graphical visibility of the model is also a useful tool for teaching SEM both in introductory and advanced level courses. This has been the case with the

excellent graphical interface provided by Amos that is now a welcome feature in Stata. Practically, all functions that can be accessed using the syntax are also available from the graphical user interface. One exception is that researchers can only set constraints that are common across groups using the graphical interface of Stata—setting different constraints for different groups may still require syntax commands. Especially researchers working in cross-cultural or comparative research may need this function rather often.

Stata provides four estimation methods. In addition to the common maximum likelihood (ML) method, it also offers maximum likelihood for missing values (MLMV). The latter method is preferred to deal with the problem of missing values. Stata uses, by default, listwise deletion (StataCorp, 2011, p. 201), which may lead to biased results, especially when there is a significant number of missing values (Schafer and Graham, 2002). MLMV assumes, however, that the data are missing at random (MAR) or completely at random (MCAR) (StataCorp, 2011, p. 203). This assumption may not always be realistic. Quasimaximum likelihood (QML) is a third estimation method that may be applied in Stata. It uses maximum likelihood, but relaxes the assumption that the data are normally distributed. Finally, the asymptotic distribution free (ADF) method is also offered by the program. It makes no assumptions about normality of the data. However, ADF may perform poorly when the sample size is not large enough (Hoogland and Boomsma, 1998, p. 335). Users may use either raw data or summary statistics data (SSD) such as means, variances, and covariances or correlations to run their models in Stata. The manual includes detailed explanations of how to create summary statistics data based on raw data and how to use both types of input files for model estimation.

Many typical SEM applications can also be performed now using Stata. These include, for example, estimating measurement models with several indicators and one or more factors (the factors may either be uncorrelated or correlated with each other), measurement models with or without correlated errors, models with second or higher order factors, or models that account for method factors (so-called multitrait-multimethod (MTMM) models, see, e.g., Cambell and Fiske, 1959). These models enable researchers to study the convergent and discriminant validity of their measurements. Once measurement issues have been investigated, one may turn to the structural models. In Stata, researchers may run structural models relating manifest and latent variables with each other in single or in multiple groups. Simple regressions and estimation of correlations or covariances are considered a subset of these general models. More advanced applications may include autoregressive cross-lagged (Finkel, 1995) or latent growth curve models (Bollen and Curran, 2006; Duncan, Duncan, and Strycker, 2006). Both can be performed in Stata to analyze panel data and study stability and change over time on the individual and group level. The Stata manual provides numerous examples of how such and other models may be performed. Similar to other SEM software packages, Stata offers users the possibility to fix model parameters to certain values or to constrain them to be equal to some reference parameter within or across groups easily. When no statement is made regarding parameters, the program will assume that they should be freely estimated.

One of the attractive features of SEM is the possibility to compare between groups. This application is very appealing, because it may be applied for comparative research, for instance, across cultural groups, countries, and social groups such as classes, age, or gender groups and also over time. It allows testing the equivalence (invariance) properties of the measurements across groups before drawing meaningful conclusions about similarities or differences in means or relations between variables across these groups. The measurement parts of the model, factor loadings, and item intercepts are constrained in Stata to be equivalent across groups by default. However, these constraints may be released. Furthermore, equality constraints on the structural parts of the model, variances, covariances, latent means, and intercepts, and structural relations between latent variables within and across groups may be imposed. The Stata manual provides examples of how this may be done.

Stata offers also various (in part), more advanced features. It allows tests to be performed on unstandardized and standardized coefficients and the decomposition of total effects into direct and indirect ones. It can provide standard errors under less restrictive assumptions, and perform bootstrap or jackknife procedures. It can estimate recursive and nonrecursive models and check the stability of nonrecursive models. Multiple-indicators multiple-causes (MIMIC) models, where various background variables, such as age, gender, education, income, or occupation may be used to predict other theoretical variables in the model, can also be easily performed in Stata. One may either draw paths from the background variables to the variables to be predicted, or formulate these relations via the syntax commands. When drawing the background variables (the so-called covariates), it is not necessary to define covariances between them because the program assumes they are associated by default. Furthermore, the program can create new variables of predicted observed endogenous variables as well as latent endogenous or exogenous variables, based on either factor scores or a linear prediction.

Like other SEM packages, Stata provides various types of output documentation. Fit statistics such as the standardized root mean squared residual (SRMR), root mean square error of approximation (RMSEA), indices for model comparison such as the Akaike information criterion (AIC) or the Bayesian information criterion (BIC), comparative fit indices such as the comparative fit index (CFI) or the Tucker-Lewis index (TLI), the chi-square value, the num-

ber of degrees of freedom, and the p-value are provided by the program. In the output the program also provides modification indices and unstandardized and standardized expected parameter changes (EPC) for the various types of constrained parameters in the model. It also reports tests results for model comparisons. Parameter estimates, standard errors, z statistics, and confidence intervals are reported alongside the standardized parameters. The user may choose to see the parameters in an alternative vector and matrix form. This may be done by asking the program to report the SEM results in the Bentler-Weeks form.

Notwithstanding these great features, the program lacks some other, useful ones. First, the user will not find, in the current Stata version, some of the advanced estimation procedures in SEM, such as Bayes (which is implemented, e.g., in Amos or Mplus) or WLSMV (which is implemented, e.g., in Mplus, see Muthén and Muthén 1998–2010) to analyze, for example, ordinal or dichotomous endogenous variables (manifest or latent). Furthermore, the possibility to analyze mixture modeling or specify interactions between latent variables is also missing. In comparison to Stata, Mplus,for example, covers these functions . The developers of Stata are looking into some of these features for possible additions to the program in the future. For many applications, Stata seems to offer excellent tools to analyze data and makes SEM more accessible for a large audience and this is a wonderful contribution to the research community.

A final word: Structural equation modeling covers a broad array of models: linear and nonlinear regressions, simultaneous equations, exploratory and confirmatory factor analysis, multilevel modeling, mixture modeling, interaction modeling, and various models to analyze panel data, just to name a few of the possible applications covered by SEM. The Stata manual provides a very good introduction to the features included in the SEM module and presents numerous examples that demonstrate how to apply them in Stata. For more basic reading about SEM the reader is referred to some excellent introductory and advanced books (e.g., Bollen, 1989; Brown, 2006; Hancock and Mller, 2006; Kline, 2011; Raykov and Marcoulides, 2006; Schumacker and Lomax, 2010, just to name a few).

References

Arbuckle, J. L. 2009. *AMOS 18.0 User's Guide*. Chicago, IL: SPSS.

Bentler, P. M. 1995. *EQS Structural Equations Program Manual*. Encino, CA: Multivariate Software, Inc.

Boker, S. M., M. Neale, H. Maes,M. Wilde, M. Spiegel, T. Brick,, et al. 2011. "OpenMx: An Open Source Extended Structural Equation Modeling Framework." *Psychometrika* 76(2): 306–17.

Bollen, K. A. 1989. *Structural Equations with Latent Variables*. New York, NY: Wiley.

Bollen, K. A. and P. J. Curran. 2006. *Latent Curve Models: A Structural Equation Perspective (1st ed.)*. Hoboken, NJ: Wiley-Interscience.

Brown, T. A. 2006. *Confirmatory Factor Analysis for Applied Research*. New York, NY: Guilford Press.

Byrne, B. M. 2009. *Structural Equation Modeling with AMOS: Basic Concepts, Applications, and Programming (2nd ed.)*. New York, NY: Routledge/Taylor & Francis.

Campbell, D. T. and D. W. Fiske. 1959. "Convergent and Discriminant Validation by the Multitrait-Multimethod Matrix." *Psychological Bulletin* 56 (2):81–105.

Duncan, T. E., S. C. Duncanand L. A. Strycker. 2006. *An Introduction to Latent Variable Growth Curve Modeling: Concepts, Issues, and Applications (2nd ed.)*. Mahwah, NJ: Erlbaum.

Finkel, S. 1995. *Causal Analysis with Panel Data*. London: Sage.

Hancock, G. R. and R. O. Müller, ed. 2006. *Structural Equation Modeling: A Second Course*. Charlotte, NC: Information Age Publishing.

Hoogland, Jefferey J. and Anne Boomsma. 1998. "Robustness Studies in Covariance Structure Modeling: An Overview and a Meta-Analysis." *Sociological Methods and Research* 26(3): 329–67.

Jöreskog, K. G. and D. Sörbom. 1986. *Lisrel VI: Analysis of Linear Structural Relationships by the Method of Maximum Likelihood*. Mooresville, IN: Scientific Software.

Kline, R. B. 2011. *Principles and Practice of Structural Equation Modeling. (3rd ed.)*. New York, NY: Guilford Press.

Little, T. D., J. A. Bovaird and K. F. Widaman.2006. "On the Merits of Orthogonalizing Powered and Product Terms: Implications for Modeling Interactions among Latent Variables." *Structural Equation Modeling* 13(4):497–519.

Marsh, H. W., Z. Wen and K. T. Hau. 2004. "Structural Equation Models of Latent Interactions: Evaluation of Alternative Estimation Strategies and Indicator Construction." *Psychological Methods* 9(3):275–300.

Muthén, L. K. and Muthén, B. O. 1998–2010. *Mplus User's Guide*. Los Angeles, CA: Muthén & Muthén.

Raykov, T. and G. A. Marcoulides. 2006. *A First Course in Structural Equation Modeling (2nd ed.)*. Mahwah, NJ: Erlbaum.

Schafer, J. L. and J. W. Graham. 2002. "Missing Data: Our View of the State of the Art." *Psychological Methods* 7(2): 147–77.

Schumacker, R. E. and R. G. Lomax 2010. *A Beginner's Guide to Structural Equation Modeling. (3rd ed.)*. Mahwah, NJ: Erlbaum.

StataCorp. 2011. *Stata Structural Equation Modeling Reference Manual. Release 12*. College Station, TX: A Stata Press Publication.

Time Series Software: Stata 12 versus RATS 8

Ellen M. Key and Matthew J. Lebo
Appalachian State University and Stony Brook University
keyem@appstate.edu and matthew.lebo@sunysb.edu

Stata's latest release, version 12, is a big leap forward in the programs time series capabilities. Always a popular program, Stata has had several areas where it lagged significantly behind the competition with "long-T" time series analysis being a prime example. This has been the case despite the fact that Stata's duration modeling and pooled-cross-sectional time series (PCSTS) capabilities have been excellent for several versions now. Practitioners and teachers of time series in political science have often resorted to more specialized software packages such as RATS (Regression Analysis for Time Series) and EViews if they wanted to get into advanced areas. The new Stata 12 finally has the ability to get deeper into recent (say, post-1995) advances in applied time series, making the program a viable substitute for dedicated time series programs. For teaching and many advanced models, Stata 12 can certainly do the job.

Two of Stata's advancements stand out to us as being the most pertinent to political scientists: improved multivariate GARCH estimation and ARFIMA modeling. Stata 12 can now estimate several members of the multivariate GARCH family including dynamic conditional correlations (DCC). Although the ability to estimate state-space models was first included in Stata 11, the conditional means and variances for each time period are predicted using a Kalman filter. As Lebo and Box-Steffensmeier (2008) show, DCC are better than Kalman filter estimates for determining the time-varying relationship between factors in the mean or variance equations. The native implementation of DCC along with the ability to estimate near-VARs with GARCH components (something that has not been implemented in other packages) is a great improvement for Stata 12.

Beyond including a host of multivariate GARCH, Stata now includes estimation of ARFIMA models as part of the base package. Previous releases of Stata had allowed the use of add-ons that could estimate fractional values of d in a Box-Jenkins style (p,d,q) model. This is now a built-in feature as is the ability to fractionally difference by the estimated value of d, obtained using the GPH procedure, making a series stationary. RATS 8.0 now allows fractional differencing as a command, but still relies on reading in a "source" file (similar to an ado file in Stata) for the estimation of d. In the details, however, is one spot where Stata clearly outperforms RATS. In ARFIMA models, one should be able to simultaneously estimate p, d, and q and Stata does this with little problem for multiple parameters. Try going beyond estimating a $(1, d, 0)$ model in RATS, though, and you will quickly hit trouble.

Other techniques covered in a standard 14-week time series course continue to be easily implemented in Stata. From Box-Jenkins to vector autoregression, all can be estimated using the commands included with Stata as shipped. Stata also has an impressive series of impulse response function and structural VAR commands to trace shocks throughout a system of equations and make inferences about causality. Although not included as "canned" commands, error correction mechanisms, granger causality, and exogeneity tests can all be implemented with a few lines of code.[1] Add in impressive PCSTS and event history capabilities and Stata 12 becomes a package that is ready to handle most anything in a typical political science time series course.

There are other advancements in Stata 12 that are of less interest to political scientists but may be useful, particularly with messy economic data. These include a set of four filters to de-trend data according to business or seasonal cycles. Stata also now includes a base unobserved-components model command to parse a series into trend, cyclical, and seasonal components as well as a new date function that allows you define your own business calendar. Again, most of these commands will be of little interest to the average political scientist but may be useful for those studying international financial markets.

For all the progress Stata has made, it is still not perfect. As in RATS, adding too many parameters to an advanced (e.g. GARCH) model can pose problems, from non-convergence to parameter instability. Since these problems become more prevalent as series get shorter, they are certainly big issues for political scientists. Some researchers may balk at data that do not like the model, but as the Stata documentation notes, these models are extremely complex and rarely have well-behaved likelihood functions. However, when your model will not converge in Stata its impossible to miss. RATS, on the other hand, will often return estimates with a note in the header that convergence was not achieved. This is another point in Stata's favor you certainly want to know when your model does not run. If you miss the warning, RATS can fool you with unreliable results. Both Stata and RATS offer options to tweak, including setting starting values and changing the maximization method, but users should be aware that there may be complex multivariate models that neither package can estimate.

As a specialty program, RATS had enjoyed some distinct advantages. Advanced programming, for example, was greatly aided by the community of users who archive and share their code generously. Also, although less customizable, the graphics produced by RATS are nicer than Stata's time series graphs and graphs from both programs can be exported in either PDF or EPS format for easy incorpora-

[1] Stata does include pre-programmed ECM and granger causality commands for use in estimating VARs, but these do not work with single equation models.

tion into Word and LaTeXdocuments. As of RATS 8, both programs are able to export estimation results in LaTeXas well.

On the other hand, RATS has always been a difficult program to use and to teach. It doesn't have a learning curve . . . just a long slow incline that remains slow as one gets into more advanced programming. Programming Box-Jenkins transfer functions, for example, involves using archaic holdovers from its mainframe FORTRAN origins. One gets blank stares when explaining to a graduate class that a line of code reading "# indvar 1 1 1" is asking RATS to estimate the effects of an independent variable on the dependent variable at lags of one and two periods (not at zero) and is also asking for an adjustment parameter to be estimated on the effect of the first lag. Moreover, students using MacRATS will have their own complaints of compatibility issues and crashes.

The Stata community is certainly large and active, but finding other researchers with long-T programming experience is not as easy there. That being said, the Stata time series community is growing and may do so at an exponential rate with the advancements in Stata 12. Stata's improvements in ARFIMA estimation and the expansion of the multivariate GARCH commands are especially welcome. Add to this a more intuitive programming language, easier data manipulation, and the ability to handle a variety of other data structures, including duration models and PC-STS, and Stata 12 becomes an even more attractive option for most instructors and users. Those with limited research budgets or who would prefer an all-in-one software package capable of running many advanced time series models will be very pleased with the improvements made in Stata 12. In sum, both programs are quite capable. That so much now comes down to taste is major progress for Stata. Stata 12 is now a very good alternative to RATS and one your students will ease into far more simply than they would add RATS to their catalogue of statistical packages.

A Note from Our Section President

Report from the President: Recap of the 2012 Business Meeting (that Never Happened)

Robert (Rob) J. Franzese, Jr.
University of Michigan
franzese@umich.edu

As all know, our annual business meeting fell prey to the weather cancellation of APSA 2012, so let me use these remarks as another chance (following on my PolMeth listserve emailed virtual meeting then) to communicate a summary of what would have transpired at the meeting.

At the business meeting, I would have reported on the states of the Society and the Section—which states, to summarize, were and are both grand—including a report on our finances from Treasurer, Luke Keele—he'd have told you our balances are positive and stable in both cases. To elaborate:

I would have thanked again Tom Carsey and Mike Ward and the teams at UNC and Duke for the successful summer meetings, and I would have announced that a host and venue for the 2015 PolMeth meetings has been secured, although an official letter of intent is pending, so I wouldn't have said whom and where. I would have mentioned that discussions with a possible 2016 host are ongoing, and emphasized that we're always interested in hearing from any and all interested parties in this regard.

I would have reminded the attendees about the call for proposals for small, thematic conferences, the winner of which was just announced (after anonymous review by a senior member of the Society and Section, for whose efforts I am extending this public, albeit parenthetical & still anonymous, thanks) as the "Causality in Political Networks Conference" proposed by Betsy Sinclair (Chicago), Guanglei Hong (Chicago), and Jake Bowers (Illinois).

I would have announced an upcoming contest to solicit a logo and letterhead for The Society for Political Methodology, the announcement of which contest is now very soon pending.

I would have recognized the *Political Methodologist* newsletter editors, Jake Bowers, Brian Gaines, and Wendy Tam Cho (all of Illinois), any of whom in attendance would have stood and said, to quote Jake: " 'please send submissions' and 'feel free to talk to me in person if you have ideas' and such."

I would have recognized Michelle Dion (McMaster), who'd have announced the going-live of OPOSSEM http:

//opossem.org.

I would also have noted and thanked the long terms of service of Diana O'Brien (WashU) and Jon Rogowski (Chicago) as moderators of the PolMeth email listerve, and congratulated, thanked, and welcomed Ethan Porter (Chicago) and Gregory Whitfield (WashU) as the new moderators.

I would have recognized someone from the contacts (Janet M. Box-Steffensmeier (Ohio State), Sara McLaughlin Mitchell (Iowa), and Caroline J. Tolbert (Iowa)) to report on the many great works over the past year of the women in methodology group, including the new "VIM: Visions in Methodology" website http://visionsinmethodology.org, the mentoring program directed by Meg Shannon (Florida State), and this year's edition of the highly successful VIM conference hosted by Sona Golder (Penn State).

And I would have introduced reports on Political Analysis from editors, Jonathan Katz and Michael Alvarez, and publisher, from our rep: Laura Bannon—those reports would have noted the now very high quality and yet still continuing improvements on submission and publication processing, and the continued strong circulation and exceptional citation records (still the #1 journal in political science by 5-year-impact rating). A call for you to submit your best work to PA would likely also have been given.

Penultimately, I would have—and, indeed, this one we actually did in our virtual, email meeting—noted that the terms of our Treasurer (then currently Luke Keele) and Member-at-Large (then Fred Boehmke) had expired, asked you to join me in thanking them, and announced the nominees running (uncontested) to replace them. For Treasurer, given the continuing complexities involved in the incorporation and non-profit status attainment of The Society, distinctly from The Section, the nomination committee rightly saw the critical need for continuity, and nominated Luke Keele to continue for another term as Treasurer. For Member-at-Large, the nomination committee expertly selected Betsy Sinclair. Our virtual meeting expired without any email opposed to either candidate, so I can now officially announce Betsy Sinclair as our current Member-at-Large and Luke Keele as (still) our current Treasurer.

Finally, I would have (and again did, virtually) announced & congratulated (again) this year's award winners, while handing them lovely small acrylic trophy/plaques and prize checks (also small), both of which were mailed instead. Please join me in congratulating one more time the tremendous set of 2012 awardees:

The Gosnell Prize for Excellence in Political Methodology is awarded for the best work in political methodology presented at any political science conference during the preceding year. 2012 Thomas Gschwend, James Lo, and Sven-Oliver Proksch, University of Mannheim, for "A Common Left-Right Scale for Voters and Parties in Europe."

The Miller Prize is awarded for the best work appearing in *Political Analysis* the preceding year. 2012 Devin Caughey and Jasjeet S. Sekhon, both at the University of California, Berkeley for "Elections and the Regression-Discontinuity Design: Lessons from Close U.S. House Races, 1942-2008." 18(4): 385–408.

The John T. Williams Dissertation Prize. In recognition of John T. Williams' contribution to graduate training, the John T. Williams Award has been established for the best dissertation proposal in the area of political methodology. 2012 Adriana Crespo-Tenorio, Washington University in St. Louis, for *Three Papers on the Political Consequences of Oil Price Volatility.*

The SPM Poster Award: The Society for Political Methodology Poster Award is given for the best poster presented at the annual summer Methodology Meeting. 2011 Brenton Kenkel, University of Rochester, for "Logistic Regression Coefficients with Nonignorable Missing Outcomes."

The Best Statistical Software Award recognizes individual(s) for developing statistical software that makes a significant research contribution. 2012 : Walter Mebane, University of Michigan, and Jasjeet Sekhon, University of California, Berkeley for genoud: (Genetic Optimization using Derivatives).

The Political Methodology Emerging-Scholar Award: This is designed to honor a young researcher, within ten years of their degree, who is making notable contributions to the field of political methodology. 2012 Jake Bowers, University of Illinois, Urbana-Champaign.

The Political Methodology Career-Achievement Award: 2012 Henry E. Brady, University of California, Berkeley

With that, this one more time, and finally, I hereby declare the 2012 Business Meeting (that Never Was) **ADJOURNED & CLOSED**.

University of Illinois at Urbana-Champaign
Department of Political Science
420 David Kinley Hall
1407 W. Gregory Drive
Urbana, IL 61801

The Political Methodologist is the newsletter of the Political Methodology Section of the American Political Science Association. The support of the Department of Political Science at the University of Illinois in helping to defray the editorial and production costs of the newsletter is gratefully acknowledged.

Subscriptions to *TPM* are free for members of the APSA's Methodology Section. Please contact APSA (202-483-2512) if you are interested in joining the section. Dues are $25.00 per year and include a free subscription to *Political Analysis*, the quarterly journal of the section.

Submissions to *TPM* are always welcome. Articles may be sent to any of the editors, by e-mail if possible. Alternatively, submissions can be made on diskette as plain ascii files sent to Wendy K. Tam Cho, 420 David Kinley Hall, 1407 W. Gregory Drive, Urbana, IL 61801. LaTeX format files are especially encouraged.

TPM was produced using LaTeX.