Duke GLOBAL HEALTH INSTITUTE | Research Design & Analysis Core

Core Guide: Reproducible Research Software Tutorial – Part 1: Stata Markdoc

Part of a series addressing common issues in statistical and epidemiological design and analysis

Background and overview

In statistics, reproducibility refers to the ability of an independent researcher to reproduce the same results as the original study if we apply identical methods to the same dataset. Statistical programming tools are available to enhance reproducibility, allowing users to summarize the code and results of an analysis in a single document so that others can easily execute the same analysis and obtain the same results. The scope of Part 1 of this Core Guide is to provide technical and programming support for reproducible research using software package Stata Markdoc.[1]

Markdoc is a "general-purpose literate programming package for Stata", which helps a user generate dynamic documents by writing documentation directly within code, and weaving statistics output directly from code into the documentation (Haghish, 2016). Markdoc uses the markdown markup language to insert statistics, tables, and figures directly from the data analysis into a formatted document. Using Stata Markdoc[2] in this way facilitates reproducibility and manuscript version control by allowing the analyst to output results in an easily readable format and document the analytical code that produces the results.

In this core guide, we show how to use Stata Markdoc to generate a statistical report. Familiarity with the basics of Stata will be needed to understand this core guide.

Tutorial

Data – the "Informed Health Choices" Podcast Study

The data used for this tutorial come from a randomized controlled trial assessing the effects of the "Informed Health Choices" podcast on the ability of parents of primary school children in Uganda to assess claims about the benefits and harms of treatments (Semakula et al., 2017). The paper was published in volume 390 of The Lancet (July 2017). The raw data (in .xlsx format) for the study are available in Supplementary Appendix 2 on the journal website, and are provided with this core guide in Supplemental

[1] Read more on the context and thought process of reproducible research in another Core Guide – Reproducing the Results of a Published Trial

[2] Other literate programming packages are available in Stata. One user-written package is called Markstat (http://data.princeton.edu/stata/markdown/). This package has similar features to Markdoc, and now (as of 2022) also allows for output to Word format, but has not been explored further by us. As of 2017, Stata also has an inbuilt command called dyndoc for creating dynamic documents. While this command allows the user to output to Word format, the coding is a bit more clunky and time-consuming than Markdoc. However, it does hold promise, as it is inbuilt, so Stata's coding team is working on it directly. This will likely mean more bug testing, more updates in perpetuity, and more thorough updates than user-written commands. Stata has introduced more features to their inbuilt reproducibility programs as of Stata version 17, so it seems likely that in-built functionality will continue to improve over time. In fact, there is now a Markdown command which reads in plain text files with Stata code and text, but seems a bit more difficult to use than Markdoc.

Material 1. The purpose of the tutorial is to reproduce the results in the paper and demonstrate the use of Stata Markdoc.

The randomized controlled trial was conducted in central Uganda. A total of 675 parents of children in their fifth year of school at 35 primary schools were enrolled and 561 of them were followed up. The parents were randomly allocated to listen to the Informed Health Choices podcast (intervention arm), or typical public service announcements about health issues (control arm). The primary outcome, measured after listening to the entire podcast, was the mean score and the proportion of parents with passing scores (correctly answering at least 11 questions) for an 18-question test measuring competency to assess the accuracy of claims about the benefits of health treatments. The secondary outcome was the proportion of parents achieving a mastery score (correctly answering 15 or more questions). The analysis was based on the intention-to-treat principle (Platt and Turner, 2019).

Supporting materials

We will walk through the use of Stata Markdoc in the context of analyzing the "Informed Health Choices" Podcast data. We introduce the basics and highlight two ways of using Stata Markdoc in this tutorial. To help users go through the details interactively, we provide the following in Supplemental Material 1:

- ***mmc2_data.xlsx*** – "Informed Health Choices" Podcast Study dataset
- ***mmc1.pdf*** – data dictionary for mmc2_data.xlsx
- ***Statistical_Report_Stata.do*** – do file for this Tutorial
- ***Formatted_Log_File_Stata.do*** – additional do file for Formatted Log File
- ***Markdoc Call.do*** – do file that executes the analysis files
- ***mystyle.docx*** – the template file that tells Stata how to format the output

Uses for Markdoc

There are multiple uses for Markdoc. One use for Markdoc is to create a summary results section for a statistical report or a paper, where the reported numbers are filled in by Stata. This use is the primary focus of this tutorial. For the "Informed Health Choices" Podcast study, the tutorial writes a draft methods and results section entirely in Stata using Markdoc. The output is available in Supplemental Material 3. The code to run the program in its entirety, ***Statistical_Report_Stata.do***, is provided in a separate zipped file included with this guide (Supplemental Material 1).

Another use of Markdoc is to create a formatted log file. The Stata log files generated using the `log` command, while useful to statisticians and programmers, can be aesthetically unappealing and difficult to read. Markdoc can be used to output a log file with narrative summaries that have statistics inserted directly into the summaries, with code sections separated by headers. This approach can enhance reproducibility by annotating the code with clear descriptions of the programmer's intent. The supporting file ***Formatted_Log_File_Stata.do*** shows how to do this.

The remainder of this tutorial will focus on the details of ***Statistical_Report_Stata.do***.

Installing Markdoc

Since Markdoc is a user-written command, the user must install Markdoc into Stata. Instructions for installing Markdoc can be found online at Haghish's github page (https://github.com/haghish/markdoc/wiki/Installation). As of May 2022, the following code works. Type the following within Stata, and you will be ready to use Markdoc.

```
net install github, from("https://haghish.github.io/github/")
github install haghish/markdoc, stable
```

Running Markdoc

In order to create Markdoc output, two separate Stata do files are needed. First, an analysis do file (e.g. ***Statistical_Report_Stata.do***) written in Markdoc syntax, which will be described in the next section. Second, a separate Stata do file which calls the Markdoc analysis do file.

The second do file (e.g. ***Markdoc Call.do***) should include a change directory (`cd`) command to change the directory to the location where the Markdoc analysis code is located. (Note that as of last testing, including a network drive path in this change directory command will throw an error. The directory must start with the letter of the mapped drive.) In *Stata code snippet 1*, we use the cd "~" shortcut (replace "~" with your own path) to set the directory to where the Markdoc call do file is located, which is also where the analysis code is located. This will be followed by a `markdoc` command which calls the Markdoc analysis file. The following code includes the "author", "affiliation", and "date" options, which puts all this information on the first page of the output document. In addition, we include the Word template (***mystyle.docx***) created using the procedure outlined in Supplemental Material 2, which allows for page breaks in the document, and specify docx (Word) as the output file type.

```
cd "~"
markdoc " Statistical_Report_Stata.do", replace export(docx) template(mystyle.docx) ///
author("Research Design and Analysis Core") affiliation("Duke Global Health Institute") date
```

Stata code snippet 1

If you have changed the directory (using the cd command) at any point within the Markdoc analysis do file, you will need to change it back to the directory where the analysis file is located before the end of the analysis file in order for Markdoc to compile. To successfully run the `markdoc` command, Stata may prompt you to install the Pandoc and wkhtmltopdf software if they have not been installed on your machine.

Basic instructions

Now we create the analysis do file (***Statistical_Report_Stata.do***) to be executed by the `markdoc` command. The final compiled output is shown in Appendix B to visualize the features of the code described in this section. Markdoc works with //ON and //OFF code sections. Anything following //OFF will not be output to the Word or PDF file. Anything between //ON and //OFF will be output. In *Stata code snippet 2*, the code begins with an //OFF, so as not to show the preamble in the Markdoc output.

```
//OFF

/*\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
 TOP MATTER
 Program: Statistical Report
 Author: RESEARCH DESIGN AND ANALYSIS CORE
 Created: January, 2022
 Purpose: Analyzing PODCAST dataset as an example of using
                      Markdoc for reproducible research.
 Modified:

 /* variables created:

*/

******* OBJECTS READ IN ***************

******* OBJECTS OUTPUT ****************


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\*/
set more off
version 16
```

Stata code snippet 2

Once we are ready to display output, we can use //ON to start displaying results in the output, as shown in *Stata code snippet 3*.

```
//ON
/***
# Background

The ability to obtain, process, and understand basic health information is crucial for
making sound health choices. Many people overestimate the benefits and underestimate the
harms of treatments which can result in poor health outcomes. Health choices are especially
important in low-income countries where people cannot afford waste. The podcast called the
Health Choices Programme was developed to help people understand how to make beneficial
health choices. The aim of this study was to assess the effects of the podcast on the
ability of parents of primary school children in Uganda to assess claims about the effects
of treatments.
***/
```

Stata code snippet 3

Note a few features in *Stata code snippet 3*. First, any regular text to be output to the Word file should be enclosed within /*** and ***/. The pound symbol (#) is used for headers. One # indicates a top-level header, two ## indicate a second-level header, etc.

The code in *Stata code snippet 4* shows the use of * to create an unordered list of primary objectives and corresponding statistical hypotheses.

```
/***
# Primary Objectives and Hypotheses

* Objective 1: Determine if mean test score is different between study arms.

       Hypothesis 1: Mean test score is not different between study arms.

* Objective 2: Determine if the proportion of passing score is different between study arms.

       Hypothesis 1: The proportion of passing score is not different between study arms.

* Objective 3: Determine if the proportion of mastery score is different between study arms.

       Hypothesis 1: The proportion of mastery score is not different between study arms.
***/
```

Stata code snippet 4

Stata code snippet 5 includes an unsorted list of primary and secondary outcomes created with * and corresponding headers specified using the pound symbol (#). In addition, the five-pound symbol (#####) is a method to force a page break in the Markdoc Word output. This method is discussed further in Supplemental Material 2.

```
/***
#####

# Primary and Secondary Outcomes

## Primary Outcomes
* Mean score (score): Number of correct answers on the test of 18 questions

* Passing score (pass): Proportion of participants with a passing score, 11 or more correct

## Secondary Outcome
* Mastery score (master): Proportion of participants with a mastery score, 15 or more
correct

***/
```

Stata code snippet 5

Additionally, graphs can easily be output using the `img` command. *Stata code snippet 6* inserts the histogram of the primary outcome, Score, into the Word document created by Markdoc. Note that the graph will not be displayed in the output file if you have a separate log file open, so be sure to include a "log close" command before the `img` command if a log file is open.

When code has /***/ in front of it, only the Stata command (no output) will be shown. Alternatively, when code has /**/ in front of it, only the output (no Stata command) will be shown. Finally, if there is nothing in front of the code, both the Stata command and output will be shown.

```
/**/capture gen scorecontrol = score if group == 2
/**/capture gen scoreintervention = score if group == 1

/**/twoway histogram score, width(1) legend(size(*.7) col(1)) xtitle("Score")
subtitle("Distribution of Score") ///
/**/xscale(range(0 20)) xlabel(0(5) 20) || kdensity scorecontrol, lpattern(solid)
lcolor(red) legend(label(2 "Control")) ///
/**/|| kdensity scoreintervention, lpattern(dash) lcolor(green) legend(label(3 "Podcast"))
name(Score, replace) ///
/**/graphregion(color(white))

/**/img
```

Stata code snippet 6

After the t-test or z-test code is run, we use the `local` command to save the test statistics and confidence intervals to local macros. Number of digits is controlled by `%3.1f`. After the regression code is run, we use the `lincom` command to extract the linear contrast of interest, and then use the `local` command to save the regression coefficients and confidence intervals into local macros (see *Stata code snippet 7*).

```
//OFF
* Compare Mean Differences
* Unadjusted mean difference for first co-primary outcome, using ttest
ttest per_correct, by(group)
       * Store test results
       local unadjscore: di %3.1f `r(mu_1)'-`r(mu_2)'
       local unadjscorelow: di %3.1f `unadjscore'+invt(`r(df_t)',.025)*`r(se)'
       local unadjscoreup: di %3.1f `unadjscore'+invt(`r(df_t)',.975)*`r(se)'

* Unadjusted mean difference for secondary co-primary outcome and secondary outcomes
* z-tests
* passing score
prtest pass, by(group)
       * Store test results
       local unadjpass: di %3.1f 100*`r(P_diff)'
       local unadjpasslow: di %3.1f 100*`r(lb_diff)'
       local unadjpassup: di %3.1f 100*`r(ub_diff)'
* mastery score
prtest master, by(group)
       * Store test results
       local unadjmaster: di %3.1f 100*`r(P_diff)'
       local unadjmasterlow: di %3.1f 100*`r(lb_diff)'
local unadjmasterup: di %3.1f 100*`r(ub_diff)'

* Adjusted mean difference for first co-primary outcome, using linear regression
regress per_correct i.groupF i.education
       * Stotre regression results
       lincom 2.groupF
       local adjscore: di %3.1f r(estimate)
       local adjscorelow: di %3.1f r(lb)
       local adjscoreup: di %3.1f r(ub)

* Adjusted mean difference for second co-primary outcome and secondary outcome, using linear
regression
* passing score
regress pass i.groupF i.education
       * Stotre regression results
       lincom 2.groupF
       local adjpass: di %3.1f 100*r(estimate)
       local adjpasslow: di %3.1f 100*r(lb)
       local adjpassup: di %3.1f 100*r(ub)
* mastery score
regress master i.groupF i.education
       * Stotre regression results
       lincom 2.groupF
       local adjmaster: di %3.1f 100*r(estimate)
       local adjmasterlow: di %3.1f 100*r(lb)
       local adjmasterup: di %3.1f 100*r(ub)
```

Stata code snippet 7

In the *Stata code snippet 8*, we have text such as <!`adjscore'!> and <!`adjscorelow'!>. In Markdoc, the procedure for inserting statistics directly into the text is to enclose the scalar or macro containing those statistics in <! and !>.

```
//ON

/***
We can see that the adjusted difference in mean scores (podcast group – control group) is
<!`adjscore'!>%, with a confidence interval (CI) from <!`adjscorelow'!>% to
<!`adjscoreup'!>%. This means, conditional on their level of education, parents who listened
to the podcast achieved, on average, a score <!`adjscore'!>% percentage points higher than
parents who only listened to the public service announcements.***/
```

Stata code snippet 8

In *Stata code snippet 7*, we store the adjusted mean difference in score in a local macro called `adjscore`, and the lower and upper bounds of the confidence interval in local macros called `adjscorelow` and `adjscoreup`, respectively. To refer to the local macros in the Markdoc text chunk, we use `` ` ' ``, while the scalars do not need these quotation marks. So, *Stata code snippet 8* becomes the following when Markdoc compiles:

> We can see that the adjusted difference in mean scores (podcast group – control group) is 15.6%, with a confidence interval (CI) from 12.5% to 18.6%. This means, conditional on their level of education, parents who listened to the podcast answered, on average, 15.6% more of the questions correctly that parents who only listened to the public service announcements.

In *Stata code snippet 9*, we output the local macros, including `adjscore`, `adjscorelow`, and `adjscoreup`, to a table using `tbl` command. The content of the table is included in `()`. Each row of the table is separated by `\`, and each column is separated by `,`. Additional options can be used to specify the title, location, etc.

```
//ON

tbl ("Outcome", "Podcast Group", "Control Group", "Unadjusted Mean Difference (95% CI)",
"Adjusted Mean Difference (95% CI)"    \ ///
       "Mean Score (Mean (S.D.))", `scoremean1'" ("`scoresd1'")", `scoremean0'"
("`scoresd0'")" , `unadjscore'" ("`unadjscorelow'" -- " `unadjscoreup'")", `adjscore'"
("`adjscorelow'" -- " `adjscoreup'")" \ ///
       "Passing Score (N (%))", `pass1'" ("`passperc1'"%)", `pass0'" ("`passperc0'"%)",
`unadjpass'"% ("`unadjpasslow'"% -- " `unadjpassup'"%)", `adjpass'"% ("`adjpasslow'"% -- "
`adjpassup'"%)" \ ///
       "Mastery Score (N (%))", `master1'" ("`masterperc1'"%)", `master0'"
("`masterperc0'"%)", `unadjmaster'"% ("`unadjmasterlow'"% -- " `unadjmasterup'"%)",
`adjmaster'"% ("`adjmasterlow'"% -- " `adjmasterup'"%)" ), ///
       title("_Table 1. Summary of outcomes in each group, with unadjusted and adjusted mean
differences. Adjusted mean differences derived from a linear regression model with parental
education as an adjustment factor_")  center
```

Stata code snippet 9

Stata code snippet 9 becomes the following table when Markdoc compiles:

Table 1. Summary of outcomes in each group, with unadjusted and adjusted mean differences. Adjusted mean differences derived from a linear regression model with parental education as an adjustment factor

Outcome	Podcast Group	Control Group	Unadjusted Mean Difference (95% CI)	Adjusted Mean Difference (95% CI)
Mean Score (Mean (S.D.))	67.8 (19.6)	52.4 (17.6)	15.4 (12.3 – 18.5)	15.6 (12.5 – 18.6)
Passing Score (N (%))	203 (70.5%)	103 (37.7%)	32.8% (25.0% – 40.6%)	33.4% (25.6% – 41.1%)
Mastery Score (N (%))	91 (31.6%)	17 (6.2%)	25.4% (19.3% – 31.5%)	25.6% (19.4% – 31.8%)

Additional Options

We have covered many options of Markdoc, but have not included many text formatting options. Additional options can be found in Haghish (2016). Since these are based on markdown language, users of R Markdown will already be familiar with many of them. For example, text can be **bolded** using **text** or *italicized* using *text*. See Supplemental Material 4 for a listing of some of these options.

Conclusion

We have walked through simple reproducible analysis examples using Stata Markdoc. Additional instructions for Markdoc, including help with installation and example vignettes, can be found on Markdoc's github site (https://github.com/haghish/markdoc/wiki).

References

Haghish, E.F. (2016). Markdoc: literate programming in stata. The Stata Journal *16*, 964–988.

Platt, A.C., and Turner, E.L. (2019). The implications of non-compliance: Randomised controlled trials: the intention-to-treat principle. BJOG *126*, 1337.

Semakula, D., Nsangi, A., Oxman, A.D., Oxman, M., Austvoll-Dahlgren, A., Rosenbaum, S., Morelli, A., Glenton, C., Lewin, S., Kaseje, M., et al. (2017). Effects of the Informed Health Choices podcast on the ability of parents of primary school children in Uganda to assess claims about treatment effects: a randomised controlled trial. Lancet *390*, 389–398.

Prepared by: John Gallis, MSc and Yunji Zhou, MB
Reviewed by: Duke Global Health Institute | Research Design & Analysis Core
Version: 4.0, last updated 09/02/2022