Dynamic Document Generation in Stata

Bill Rising

StataCorp LLC

2017 Brazilian Stata Users Group meeting
São Paulo, SP
8 December 2017
The Good and Bad of Creating Documents

- Think of documents you’ve made in the past
- Think of good and bad things which happened the first time you thought you were done
The Bad

- Questions on methods for reaching particular numerical results
- Needing updated analyses because of new or improved data
- The report was nice enough you were asked to do it repeatedly, say, every month
- Needing to fix transcription errors
- All in all, the document created maintenance costs
The Good

- Reusing ideas
- Reusing lessons for teaching
  - Better: polishing lessons to shining perfection
- Gaining utility from reproducing a near-copy of the document
General Idea

- What gets done once often gets done twice
  - Similar projects
  - Updated datasets
  - Datasets arriving over time or from various sources
  - Teaching
- The second and later repetitions should not start from scratch
- There should be protection against mistakes
Dynamic Documents

- Needed: reproducible and reusable documents, aka dynamic documents
  - Documents should be reproducible
    - No magic required or desired
  - Documents should be reusable
    - This is especially necessary for teaching
- Both of these are easy for pure narratives
- Including computational results is trickier
- Making this nice for all collaborative parties is even trickier
Best Possible Process

- One underlying file for producing a final document, including narrative and stats
  - If not a single document, a single folder with easily-related files
- The final document can be reliably reproduced from scratch
- Drafts of the final document can be passed around to all collaborators
  - Topic experts as well as statistical experts as well as writers
  - Those comfortable with programmerish work and those who are not
- The final document could be in a variety of forms
  - Different audiences prefer different forms (web, print, etc.)
What We’ll See Here

- Several tools for producing dynamic documents
- Some way of deciding between complexity, completeness, and comprehension
General Needs

- Bare Necessities for Teaching
  - Commands
  - Results
  - Graphs

- Bare Necessities for Reports
  - Results without commands
  - Inline results
    - Results often show up within the narrative
  - Invisible commands
Overview of Official Stata Software

- **New Stata 15 commands**
  - dyndoc
  - putdocx
  - putpdf
Terminology

- It will help to have some defined jargon here to refer to files
  - A *source* file gets processed by the software
  - Sometimes, the result of the processing is an *interim* file, which requires more processing
  - When the processing is done, the result is a *final* file, which can be opened in the proper application
    - This is not final as in “final draft”
dyndoc: Markdown to html

- dyndoc takes a source Markdown document containing Stata code and turns it into a final html file (aka a web page)
  - There are no interim files
- Markdown is a simple way to make a structured document
  - A text file with a few rules for common construction
- This does require learning Markdown (which is simple) and Stata’s dynamic tags (which is fairly simple)
- The document is typically in narrative mode except when dynamic tags switch to Stata commands
  - Not your typical do-file
Quick Introduction to Markdown

- Markdown was intended for an easy way for bloggers to write
  - Since it was written by a programmer, it is also made for easy ways to blog about programming
- Paragraphs are separated by blank lines
- Inline code gets put `between left quotes`
- Block code is put between sets of four tildes
- Emphasis comes _between underscores_ (or asterisks)
- Boldface comes **between double asterisks** (or underscores)
- List items start with either *, - or a number with a period
Quick Introduction to Stata Dynamic Tags

- `<<dd_do>>` starts Stata code blocks
- `</dd_do>>` ends Stata code blocks
- `<<dd_display: [fmt] exp >>` puts Stata results in the running text
- `<<dd_graph>>` puts in graphs
  - Some extra attributes are needed

Some extra attributes are needed.
An Example of dyndoc

- We should look at an example
- Open up this file to take a peek
  - doedit dyndoc_ex.md
    - The .md extension is for Markdown files
    - This can help some text editors highlight the file better
- We can make this into a web page
  - dyndoc dyndoc_ex.md, replace
- This creates the web page dyndoc_ex.html
- Take a look!
Comments on dyndoc

- **Good News:**
  - Simple to use
  - Uses fairly readable source documents
  - Generally quite nice

- **Bad News:**
  - Only produces html
putdocx: Creating docx Documents

- putdocx makes docx documents
  - Close but not exactly MS Word documents
  - Exactly Open Office documents
  - Generally very compatible with MS Word
- Works directly from a do-file, as all commands are Stata commands
- All text or tables are enclosed in commands
  - No split between narrative and Stata modes
It is useful to define some macros for common text or paragraph types

Start writing to the document with `putdocx begin`

Start new paragraphs with `putdocx paragraph`

Include text with `putdocx text`

Include graphs (or other images) with `putdocx image`

Build tables with `putdocx table`

Write actual docx document with `putdocx save`
An Example of putdocx

- Here is a short example for putdocx
  . doedit putdocx_ex.do

- Creating the document is done by doing the do-file
  . do putdocx_ex
Comments on putdocx

- **Good News:**
  - Can make docx documents
  - There is a fair amount of control over table construction
    - Though it can take a bit of work
  - Can be used for mass production of reports

- **Bad News:**
  - The source file is difficult to read
  - This is not made for teaching Stata, because including commands and output is not simple
Creating PDF documents with `putpdf`

- `putpdf` creates PDF documents directly
- It is similar in kind to `putdocx`
- The source file is a do-file
- There are no interim files
General Structure

- It is useful to define some macros for common text or paragraph types
- Start writing to the document with \texttt{putpdf begin}
- Start new paragraphs with \texttt{putpdf paragraph}
- Include text with \texttt{putpdf text}
- Include graphs (or other images) with \texttt{putpdf image}
- Build tables with \texttt{putpdf table}
- Write actual docx document with \texttt{putpdf save}
Similarity to putdocx

- putpdf is very similar to putdocx
- The names of options for commands often differ, however
  - The terminology used for putpdf is related to how people talk about PDF files
  - The terminology used for putdocx is what the docx format uses
  - These are a bit different
An Example of putpdf

- Here is a short example for putpdf
  . doedit putpdf_ex.do
- Creating the document is done by doing the do-file
  . do putpdf_ex
Comments on putpdf

- **Good News:**
  - Can make pdf documents
  - There is a fair amount of control over table construction
    - Though it can take a bit of work
  - Can be used for mass production of reports

- **Bad News:**
  - The source file is difficult to read
  - This is not made for teaching Stata, because including commands and output is not simple
  - The base PDF definitions are not as rich as those for putdocx, so there is less control over the final look
There are many user-written commands and packages for dynamic documents.

The two that will be covered here are:
- `putwrap`
- `markstat`
putwrap: More Readable

- putwrap is a simple wrapper which allows putdocx and putdocx documents to be more readable
- Paragraphs are separated by blank lines
  - The source document has narrative and code modes
- Everything else is like putdocx and putpdf
Looking at an Example

- There is nothing new to define here
- Here is an example source file
  . doedit putwrap_docx_ex.do
- To make the interim file, use
  . putwrap using putwrap_docx_ex.do, replace
- This creates the do-file putwrap_docx_ex_conv.do
  - The _conv gets added because the original interim file has a .do extension
- To create the document, use
  . do putwrap_docx_ex_conv
Comments on putwrap

Good News:
- All the good news from putdocx and putpdf
- If the document has a lot of standard narrative, then this makes things much more readable and easier to edit
- Can be used for mass production of reports

Bad News:
- The source file becomes difficult to read if there is a lot of mixing of fonts
- This is still not made for teaching Stata, because including commands and output is not simple

Will be up on the SSC next week
The general way to work is via markdown

- There is a simple syntax, which is less flexible
- There is a strict syntax, which allows more features but is harder to read

The user-written `markstat`, written by Germán Rodríguez can write html, docx, and pdf from the same document

- This is done via pandoc, which is a general package which must be installed outside of Stata

There are no explicit interim files which must be tracked

- There are many interim files if needed for debugging, however
General Structure, Simple Syntax

- Typical Markdown, except
  - Stata code gets indented either 4 spaces or one tab
  - There are no nested lists
- Stata results get included in the narrative using `s [fmt] exp`
- Mata results use the same, except for an `m` instead of an `s`
- Otherwise this is very much like the official dyndoc
An Example of Simple Syntax

- Here is a short example for markdown with a simple syntax
  . doedit markstat_ex.stmd
  - The stmd extension is required!
- Creating the document is done by using markstat
  . markstat using markstat_ex
- By default, this creates an html document: markstat_ex.html
- This, however, will create a docx document
  . markstat using markstat_ex, docx
- This, however, will create a docx document
  . markstat using markstat_ex, pdf
General Structure, Strict Syntax

- Strict syntax is needed if you wish to squelch commands or use nested lists
- Stata code blocks start with ````s and end with ````
- Mata code blocks start with ````m and end with ````
An Example of Strict Syntax

- Here is a short example for markdown with a simple syntax
  . doedit markstat_strict_ex.stmd
    - The stmd extension is required!
- Creating the document is done by using markstat
  . markstat using markstat_strict_ex, strict
- Just like the simple syntax, this creates an html file by default
- You can make docx and pdf files just as before
Comments on markstat

- **Good**
  - For simple source documents, this is the most readable
    - This is good for example Stata documents, for both instructors and students
    - The Stata output looks a bit more polished
    - Many different output types can be made from a single source

- **Bad**
  - Requires both pandoc (from outside Stata) and `whereis`, another package written by Germán
  - Though it produces `.docx` documents, it does not have the fine control over tables found in `putdocx`
Other Software Not Covered

- Ben Jann’s `texdoc` package
  - For creating `\LaTeX` documents
  - Unfortunately has no inline results

- Haghish’s `markdown` package
  - Very flexible but often a moving target

- Russ Lenth’s `StatWeave` package
  - Not a Stata package, but can be used to make `\LaTeX` or html documents for Stata, SAS, bash, Matlab among other languages
  - Will be available on github by early 2018
Perhaps this will get you curious about producing dynamic documents
Perhaps this will get you asking us for features in our own document generation tools