Reproducible Research in Stata

Bill Rising

StataCorp LP

12th German Stata Users Group Meeting
Uni Hamburg
14 June 2014
It would be nice to make it easy to make nice documents containing Stata code and results
It would be nice to put them in a variety of forms
It would be nice to have this accessible to statistics lovers
It would be ideal to have this accessible to statistics lovers who do not like programming
What Type of Documents?

- Would like to produce documents of many types
  - HTML for web pages
  - \LaTeX{} for making presentations and handouts
- Would like non-programmers to use the tools
  - HTML and \LaTeX{} are painful for non-programmers
    - Actually \LaTeX{} is painful for everyone
- HTML could use markdown
- \LaTeX{} needs a front end
What We’ll See Here

- Partial success
  - \LaTeX\ lessons can be made quickly without much \LaTeX
  - Producing web pages (HTML) is working somewhat
    - Not ready for non-programmers
    - Not even quite ready for programmers
- Would like to show what can be possible
Using Reproducible Documents

- We want to work with reproducible documents
- A “reproducible” document contains both narrative and Stata commands
  - The commands get processed and their output is included in the final document
- This is the right way to work
  - The results in the documents are the actual results
  - Any changes in the data change the output as the document is made
  - There are no typos in the Stata code
Typical Goal for Document Creation

- Reproducibility
  - Results in the document must come from commands
  - Useful side-effect: automatic error-checking of Stata code
    - The code must run for the notes to be typeset
  - Must be quick for matching changes to output in Stata

- This should be simple
  - As most people work, it is not
Typically “reproducible research” is used as a term for reproducing published papers

- Published papers are unchanging

Teaching documents should be **reusable**

- Not just reproducible

They need to be flexible, among other goals
Uncommon Goals for Teaching, Part 1

- **Maintainability**
  - Must allow quick alterations
  - Otherwise there is a big disincentive to make changes
  - This encourages leaving bad notes as they are
  - Must take very little time for updating as Stata updates

- **Brevity and Completeness**
  - In training sessions, results are put on the screen
  - Hence they are not needed on the presentations
  - Would like handouts from lessons to contain more than slides
    - Notes for the reader
    - Alternative ways to achieve the same goals
  - Would like this to be simple to do
Uncommon Goals for Teaching, Part 2

- **Modularity**
  - Want to reuse lessons as much as possible
  - Would like flexible ordering of lessons
    - This allows customization of training
    - Adds a complication that there could be repeated material
We’ll open up a toy lesson to see how they are made

The application being used is OmniOutliner Pro

- This is commercial software available from omnigroup.com
- It is used because it can export its files in a way that they can be manipulated
Lesson as an Outline

- A lesson is an outline
- To put items in the lesson, put them in the outline
  - The first level defines sections
  - The second level defines subsections
  - The third level defines slides
- To add comments, use the comment field for the item
Adding Code

- Code is added as a comment
- To get the code evaluated, tick the code checkbox
- To put the code and/or its results should be in the handouts and/or presentation, use the hand and pres columns
Including Graphs

- Including graphs is simple enough:
  - Put in the graph code
  - Tick the *code* checkbox
  - Say where to include the command and/or graph
  - Give the figure a name
  - Select that a figure is present

- Including other graphics are used similarly
Including Results in Narrative

- A little \LaTeX{} is needed to put the results into the running narrative
  - Put \texttt{\Stataexpr{exp}} into the narrative
  - The \textit{exp} gets put in an display command, so use any display directives you want
- Typically, some formatting is needed to make things nice
Indexes are a strength of \LaTeX

They can be included by putting index entries in notes
  - This does require \LaTeX knowledge

Select whether the index comes before or after the outline item in the index column
  - This is needed because of indexing for a range of pages
Typesetting

- The lessons are typeset using a script
  - More specifically: an AppleScript

- Here is the short, hidden story
  - The outline gets put in OPML
    - OPML is a variant of XML which is made for outlines
  - The OPML gets translated to \LaTeX
  - The \LaTeX gets run through StatWeave
    - StatWeave is available from http://homepage.cs.uiowa.edu/~rlenth/StatWeave
    - Used because if can mix languages and because it can be extended

- The result gets typeset
End Result

- Typesetting produces
  - A handout, which can be long and detailed
  - A presentation, which helps people follow without full details
- All items are in both documents
- The handouts typically have all output
- Each slide in the presentation knows what page it is on in the handout
Making Changes

- Making changes is no different than adding new content
- Moving items is done, as expected, by dragging and dropping
- We’ll make some changes to the lesson now
Other Nice Features (not shown)

- It is possible to typeset many lessons as one course handout.
- Each lesson has its own presentation.
- Within the course handout, each lesson is one chapter.
- There is a single combined index.
- The page references on the slides refer to the pages in the combined handout.
- If lessons use datasets not included with Stata (or any other files), a download site gets made with links for each lesson.
More Features (not shown)

- It is possible to have conditional material
- Material can be excluded if it was covered in a previous lesson
  - This allows keeping overlap in lessons might all be used in one course
- Material can be included or excluded based on flags for the type of lesson
  - Deeper material can be included only in special cases, for example
Overall Conclusion

- This is very useful to me for outline-like presentations
- It allows using the strengths of \LaTeX\...
  - Programmability
  - Standardizing the look
- It is somewhat cobbled-together and hence needs careful installation documentation
  - This will make it more useful to others
Producing Web Pages

- This is doable, but not very friendly
- There will be one short example
Good News

- With the proper structure and files can put together lessons
- Only $\LaTeX$ needed is indexing (and Math typesetting if needed)
StatWeave can be used for arbitrary \LaTeX documents

- It can theoretically be used for ODT files produced by OpenOffice, LibreOffice, etc.

  - Sadly, these OpenOffice-based applications have put in security “features” which prevent opening documents with binary chunks changed by other applications

- Complicated tables and such can be made by including hidden commands and bringing the output as needed

- Using StatWeave in this form for \LaTeX requires fighting with \LaTeX
Bad News

- Lessons depend on OmniOutliner Pro, which is Mac only
  - Would love to hear about outliners on other platforms which can produce good OPML
- HTML is still weak
The World is Limitless

- Document generation can work well with enough programming behind the scenes
- Putting a friendly interface in front of the programming is critical
- We don’t want to end up with a Rube Goldberg contraption such as this:
  - Joseph Herscher’s Page Turner (click to view)