

Teaching with Self-Made Help-Files

Background

- The situation:
 - general: strong competition between Stata and R
 - university: “wiping replaces typing” & increasing divisions/distinctions
 - (body of) Stata commands becomes increasingly complex
- Important goals for running a statistical software:
 - being interdisciplinary
 - easy access for newcomers (=first-year students & female students)
 - being successful in competition with other software packages
- My role & my mission:
 - teacher’s perspective in social sciences = students don’t like statistics
 - enhancing entrance into world of statistics
 - increasing data literacy & programming skills

Agenda

- 1 Stata's Help-Files
- 2 Stata Markup and Control Language (SMCL)
- 3 Ways of learning Stata commands
- 4 Application Examples

1

Stata's Help-Files

Elements of Help Files

- Stata's help-files are read in the viewer window

```
help regress
```

```
view "C:\Program Files\StataNow19\ado\base\r\regress.sthlp"
```

- provide a lot of information in an encyclopaedical way
- usually you find text and links in help files
 - links to pdf-documentation
 - internal/jump links or links to other help files
 - external links (URL) to videos
- help-files are written in SMCL and can be edited

```
doedit "C:\Program Files\StataNow19\ado\base\r\regress.sthlp"
```

2

Stata Markup and Control Language (SMCL)

SMCL – Text Formatting Tags

- underlining `{ul:important text}`
- bold text `{bd:another important part of my text}`
- text in italics `{it:also but differently important part of my text}`
- combinations `{it:{ul:very important part of my text}}`
- pre-defined formatting directives
 - normal text `{text:}`
 - command `{cmd:}`
 - error messages `{error:}`
 - results `{result:}`
 - highlighted text `{hilite:}`

SMCL – Section Tags

- Standard section
 - `{pstd}`
This is a brief explanation
- Hanging sections
 - `{phang2}`
 - `{phang3}`
 - `{pmore3}`
- `[...]`
- type `help smcl` for all tags & directives

SMCL – Special Tags

- horizontal line `{hline}`
- special characters `{char A^}`
- executable Stata commands `{stata describe}`
- links to pdf-Documentation `({mansection R regress})`
- external links (videos, pictures etc.)
 - `{browse "http://www.youtube.com/watch?v=HafqFSB9x70"}`
 - `{browse "http://www.youtube.com/watch?v=HafqFSB9x70":Regression in Stata}`
 - `{browse "https://cloud.uni-hamburg.de/s/3ZLPogjRFs3eTNr"}`
 - `{browse "https://cloud.uni-hamburg.de/s/3ZLPogjRFs3eTNr":Data Says No}`

3

Ways of Learning Stata Commands

Ways of Learning Stata Commands

1. commands are shown on slides → students copy and type commands
 - disadvantages: **mistyping & frustration** (error messages are not always clear enough for beginners) & logic of command structure remains unclear
2. reading help-files → passive resource → students copy and type commands
 - relatively easy language appears complicated
3. do-files are distributed → students highlight commands and run them
 - disadvantage: **not typing** commands doesn't support cognitive learning process
 - in other words: complex commands appear too easy for students

Ways of Learning Stata Commands

4. trial-and-error using pull-down menus
 - disadvantage: **too complex** and unknown choices
5. watch Chuck Huber's Stata videos
 - disadvantage: uses pull-down menus & student's **self-deception** (students suggest themselves a deep learning effect)
6. using KI to create code
 - disadvantage: well, in worst case, **de-skilling** (students don't learn, but substitute)

Ways of Learning Stata Commands

- closing the gap by interactive commands in self-made help-files
- advantages:
 - no error-prone self-typing
 - immediate result response
 - command structure & relation between commands become visible
 - translation into other languages
 - addresses needs of first year students
 - inbetween commands are possible
 - teachers might use this as a start for writing a textbook

4

Application Examples