Problems for reproducibility and some possible solutions

Ulrich Kohler

kohler@wz-berlin.de

Wissenschaftszentrum Berlin

(Presentation prepared with LaTeX)

Plan of the presentation

- Introduction: On reproducibility
- OS and environment dependency
- Software dependency
- Dataset confusion
- Transcription errors
- Outlook

On Reproducibility

To ensure reproducibility, *all* steps of an analysis should be *documented* in *program files* and be made *accessible* to others. In this definition

- “all steps” means all steps (!),
- “documented” means, that it should be easy to find the program file for a specific analysis and the program file should be easy to understand,
- “program files” means Do-file, SPS-File, etc, and
- “accessible” means, program files should be handled to others when requested and program files should produce the same results when running by others.

On Reproducibility

I have tried to make reproducible analyses in this sense. They are accessible on the web. When preparing these analyses I sometimes was caught in more or less hidden traps. This presentation shows how I escaped.

OS and environment dependency

With Stata, OS and environment dependency is a minor problem. Problems can arise if you specify filenames in do-files—especially if your research-group works in different institutions with different local network structures.

- Do not use the backslash as directory separator. Code `use subdir/mydata` instead of `use subdir\mydata`
- Do not use absolute pathnames in a do-file. Do not code `use c:/path/mydata`. Interactively `cd` to *c:/path* and code `use mydata` in the do-file

OS and environment dependency

- Major databases are used for many projects. They shouldn’t be copied to each project path. Absolute Pathnames could be circumvented by setting global macros in a project-setup file. This way, the pathnames must be only edited once.

master.do

```
global soepdir  "c:/important data/gsoep16" // <- Edit this
```

xyz.do

```
use "$soepdir/ap" //   <- Quotation marks are more secure
```

Software dependency

Stata

Obviously, a Stata do-file requires Stata. But with version control you can specify which Stata should be used:

xyz.do

```
* Analysis of the xyz-hypothesis
version 3
...
```

Cool Ados

User written programs are useful tools. But sharing do-files can getting painful if you often use them. And: User written programs sometimes change. Installing a new version of a user written program might break do-files.

- Install Ados within a do-file:

master.do

```
capture which mkdat
if _rc ~= 0 {
  net from http://www.sowi.uni-mannheim.de/lesas/ado
  net install mkdat
}
```

Software dependency

- Consider copying the program-definition into your do-file:

xyz.do

```
capture program drop soepren  // <- Note this
*! soepren.ado 0.1, ukohler@sowi.uni-mannheim.de
program define soepren
  ...
end

soepren ?bula, new(bul) w(1984/2000)
```

Software dependency

- Consider copying a renamed copy of the program in your project directory (and share the renamed copy with your colleagues):

mysoepren.ado

```
*! soepren.ado 0.1, ukohler@sowi.uni-mannheim.de
program define mysoepren  // <- Note new program-name
...
end
```

xyz.do

```
mysoepren ?bula, new(bul) w(1984/2000)
```

Dataset confusion

Many datasets regularly gets updated by their originators. Sometimes the update ships with the same name but it is a complete different file (for example the cumulated ALLBUS).

- Check the properties of such datasets:

mysoepren.do

```
describe using $allbdir/s1795, short
assert r(N) == 34956 & r(k) == 847
use $allbdir/s1795
```

Transcription errors

At one point of an analysis, results from the Stata output needs to be filled into a text document. This is nowadays often done by *copy* and *paste*. However transcription errors still arise, because

- pasted results needs to be “rearranged” in some way, and/or
- specific numbers needs to be “picked” from different command outputs.

Transcription errors

The likelihood of transcription errors raise if you did the copy-paste-rearrangement-picking procedure the fifth time after detecting data errors over and over again.

- Minimize copy-paste-rearrangement-picking procedures by
  1. using graphs
  2. using available specialized tools
  3. exporting datasets which contain the results
  4. using `file`

Transcription errors

Graphs

Stata graphs can be included into Ms-Word or LaTeX without copy-paste. This way the text contains always the latest version of the graph.

xyz.do

```
graph dot uc, over(country, sort((mean) uc))
graph export turnout.eps, replace
graph export turnout.wmf, replace
```

xyz.tex

```
\includegraphics{turnout}
```

xyz.doc

```
Einfuegen->Objekt->Aus Datei ertellen->
  "turnout.wmf" (Als Verknuepfung markieren)
```

Specialized Tools

There are several tools to get Stata output in a more publication ready format. Some of them will be described by Roger Newson in the next issue of the Stata Journal (2003, Nr. 3). Examples are

- `outreg`, `reformat`, `mktab`, `slist`, `fsum`, `ciform`, `xcontract`
- Tools to work with LaTeX: `sjlog`, `latab`, `est2tex`, `outtable`, `sutex`, `outtex`, `maketex`, `dotex`, `listtex`

Transcription errors

Export datasets containing the results

Exporting datasets containing the results provide a general way to get publication-ready output. I used this technique to produce the following table, which is quite different from the normal Stata output. The table has been fully produced within Stata and inserted into this presentation with the LaTeX-command `input{tables.tex}`.

	Kriminalitaetsfurcht					
	Gross	Mittel	Klein	Gueltig	Fehlend	Gesamt
Maennlich	57.4	38.4	4.2	6377	43	6420
Weiblich	63.5	33.5	3	6807	56	6863

Transcription errors

- To produce the table I constructed a Stata dataset which contains the numbers for the table and nothing else.

```
. list
```

gender	n1	n2	n3	n99	Nv	N
Maennlich	57.4	38.4	4.2	43	6377	6420
Weiblich	63.5	33.5	3	56	6807	6863

- This dataset can be exported to a format suited for Word-Processors and/or LaTeX.

Transcription errors

- To export to MS-Word, type
  `outsheet gender n1 n2 n3 Nv n99 N using xyz.tsv`
  inside Stata and use Word to import *xyz.tsv*:
  - Einfügen, Datei, Namen auswählen
  - Gesamten eingefügten Bereich markieren
  - Tabelle, Umwandeln, Text in Tabelle, Trennzeichen: Tabstops

  Now, dress up the table with Word. Unfortunatelly there seems no way to automate the Word-steps from within Stata (but `listtex` brings you somewhat closer).

Transcription errors

- To export to LaTeX type

tables.do

```
listtex gender n1 n2 n3 Nv n99 N using tables.tex, rstyle(tabular
  head("\begin{tabular}{lrrrrrrr}\hline"
       " & \multicolumn{6}{c}{\emph{Kriminalitaetsfurcht}} \\\\"
       " & Gross & Mittel & Klein & Gueltig & Missing & Valid \\\
  foot("\hline\end{tabular}")
```

- One can also use `listtex` to export tables to Word. However you still need to convert text to tables within word.

Transcription errors

- Finally, here is how I produced the Stata dataset:

tables.do

```
* Produce absolute Frequencies
by gender worries, sort: gen n = _N
by gender worries: gen Nv = sum(worries~=99)
by gender worries: keep if _n == _N
by gender (worries), sort: gen N = sum(n)
by gender (worries): replace Nv = sum(Nv)
by gender: replace N = N[_N]
by gender: replace Nv = Nv[_N]

* Reformat the data
reshape wide n, j(worries) i(gender)

* Calculate percentage values from valid n
forv i = 1/3 {
  replace n‘i’ = round(n‘i’/Nv*100,.1)
}
```

Transcription errors

- Producing publication ready output often is a data-management problem.
- There are specialized tools, but in the long run learning to use the general data-mangement tools will pay back.
  - `by varlist:`, `post`, `reshape`, `file`
  - Local-macros
  - Saved Results

Transcription errors

file

The command `file` is often convenient if you want to produce an output with numbers from different procedures. I used `file` to produce the following table, which picks Likelihood-Ratio-Tests from different models. The table has been fully produced within Stata and inserted into this presentation with `input{aninccchi2.tex}`.

	Baseline	Hypotheses
Country Main Effects	307.96	261.57
	(0)	(0)
Interaction Effects	25.08	26.32
	(0)	(0)

Transcription errors

- Here is how I produced the table:

aninc.do

```
file open an1 using anincchi2.tex, write text replace
file write an1 "\begin{tabular}{lrrr} \hline " _n
file write an1 "& Baseline & Hypotheses \\\\ \hline" _n
file write an1 "Country Main Effects & ‘lrctr1’  & ‘lrctr3’ \\\\"
file write an1 "& (‘pctr1’) & (‘pctr3’) \\\\" _n
file write an1 "Interaction Effects  & ‘lria1’   & ‘lria3’ \\\\"
file write an1 "& (‘pia1’)  & (‘pia3’) \\\\ \hline" _n
file write an1 "\end{tabular} "
file close an1
```

Outlook

Combining the tools mentioned so far, it is a natural step to produce a publication ready PDF-file with tables for all variables in a data-set. The general outline would be:

dreams.do

```
file open tables using alltables.tex, write text replace
file write tables "\documentclass{book}" _n
file write tables "\begin{document}" _n
foreach var of varlist _all {
  mytabs ‘var’ // <- produce table and save as ‘var’.tex
  local section: var lab ‘var’
  file write tables "\section*{\bf ‘var’: ‘section’}" _n
  file write tables "\input{‘var’.tex}" _n
  file write tables "\newpage" _n
}
file write tables "\end{document}" _n
file close tables
!pdflatex alltables
```

Outlook

- The crucial step in the program outline is `mytabs` which doesn't exist so far. At the WZB we have been quite successful in constructing a specialized version of `mytabs` for a single dataset with 658 variables. But a generalization needs some further efforts.
- Note also that `file` can read from arbitrary files. Therefore one may also add additional information from a database—i.e. questions from a questionnaire—to each of the tables.