# Automated table generation and reporting with Stata

Ben Jann, ETH Zürich, jannb@ethz.ch

CEPS/INSTEAD, Differdange, Luxembourg

October 24, 2008

## Outline

- Introduction
- Low-level results processing
  - How to access results from Stata routines
  - Getting things out of Stata: The **file** command
  - Wrappers (example: **mat2txt**)
- Handling model estimation results
  - Archiving models using **eststo** and **estwrite**
  - Tabulating estimation results using **esttab** or **estout**
- Automatic reporting
  - Automation in LaTeX
  - Automation in Word/Excel

## Introduction

Statistical software packages are good at analyzing data, but they are usually weak when it comes to reporting.

- Output from statistical routines contains all sorts of details that are valuable to the researcher but are not so important for reporting.

  **=> select relevant results**

- Output from statistical routines sometimes contains results that are not well suited for interpretation or for presentation to a non-expert audience.

**=> transform results**

## Introduction

- Output from statistical routines is often not well formatted for presentation.

  **=> rearrange and reformat results**

- Various software packages might be used for further processing of results and for reporting.

  **=> transfer results to specific file formats**

- You might need to re-use results for other reports or extract additional results at a later point in time.

  **=> archive results**

## Introduction

**TWO MAXIMS**

**1) Never Copy/Paste results by hand**

You will almost sure make tons of mistakes!

**2) Do everything only once**

It is simply a waste of time to do things more than once.

## Introduction

- These two goals can be reached by **automation**.
- Automation has it's **price**:
  - initial investment of time and effort
  - reduced flexibility

- However, personally I find that automation almost always pays off.

- For example, although you are convinced that you do the tables in your research paper only once, you'll find yourself doing them over, and over, and over, ...

## Introduction

- Furthermore, automation increases **quality**:
  - no copy/paste errors
  - errors and possible improvements are often detected after everything is done; in a non-automated settings there are high barriers correcting such errors or implementing the improvements
  - the lack of flexibility leads to standardization (which is usually positive, but can sometimes also hinder innovation)
  - automation makes research more replicable

- Moreover, good tools can lower the costs of automation dramatically.

## Introduction

- Of course, there are also exceptions where automation might not pay off.

- Examples:
  - slides for presentations that are only used once or twice
  - numbers in text body (trick: only cite approximate values)

# Part 1: Low-level results processing

## Accessing results in Stata

- A prerequisite for automation is that the results from statistical routines can be accessed by the user.

- In Stata, most commands return their results in **r()** or **e()** (see **return**).

  - **r()** is used by "general" commands such as **summarize**
  - **e()** is used by "estimation" commands such as **regress**

- Returned are:

  - string scalars
  - numeric scalars
  - numeric matrices

  - For example, estimation commands return the number of observations in **e(N)**, the name of the command in **e(cmd)**, and the coefficients vector and the variance matrix in **e(b)** and **e(V)**.

## Accessing results in Stata

- Use **return list** or **ereturn list** to find out about available returns. Use **matrix list** to see the contents of a returned matrix.

```
sysuse auto
summarize price
return list
```
*<run>*

```
regress price mpg weight
ereturn list
```
*<run>*

- Use **matrix list** to see the contents of a returned matrix.

```
matrix list e(b)
matrix list e(V)
```
*<run>*

## Accessing results in Stata

- You can use the **e()** and **r()** scalars and matrices more or less as you would use any other scalar or matrix, although it is often advisable to first copy the results into regular macros, scalars, or matrices (see **macro**, **scalar**, and **matrix**).

- Examples:

```
display "BIC = " -2 * e(ll) + ln(e(N)) * (e(df_m)+1)
<run>
```

```
local BIC = -2 * e(ll) + ln(e(N)) * (e(df_m)+1)
display `BIC'
<run>
```

## Accessing results in Stata

- Example with matrices:

```
matrix X = 1                          /*the constant*/
foreach v of varlist weight mpg  {  /*reverse order*/
    summarize `v'
    matrix X = r(mean), X
}
matrix b = e(b)'
matrix Y = X * b
display Y[1,1]
<run>
```

```
adjust mpg weight
<run>
```

- Note that coefficients and standard errors can also be accessed as **_b[]** and **_se[]**:

```
display "t value = "_b[mpg] / _se[mpg]
<run>
```

## Getting things out of Stata: The file command

- The **file** command is used in Stata to write to (or read from) a file on disk.
- Use **file** to produce custom output files.
- **file** is a low level command. It just writes plain text, line by line. You have to do all formatting yourself.
- **file** may appear a bit clumsy: You have to

```
file open handle using filename, write /*initialize*/

file write handle ...                  /*write*/
...

file close handle                      /*done*/
```

- However, **file** can produce any desired output.

## Getting things out of Stata: The file command

- Example: Write a tab delimited file containing descriptive statistics

```
file open fh using example.txt, write replace
file write fh "variable" _tab "mean" _tab "sd"
foreach v of varlist price-foreign  {
    summarize `v'
    file write fh _n "`v'" _tab (r(mean)) _tab (r(sd))
}
file close fh
type example.txt
```

*<run>*  *<show>*

## Getting things out of Stata: The file command

- This can easily be turned into a program:

```
capture program drop mysumtab
program define mysumtab
    syntax varlist using [, replace append ]
    tempname fh
    file open `fh' `using', write `replace' `append'
    file write `fh' "variable" _tab "mean" _tab "sd"
    foreach v of local varlist {
        quietly summarize `v'
        file write `fh' _n "`v'" _tab (r(mean)) _tab (r(sd))
    }
    file close `fh'
end

sysuse nlsw88
mysumtab * using example.txt, replace
```

*<run>*  *<show>*

## Getting things out of Stata: The file command

- Or let's do HTML:

```
capture program drop mysumhtm
program define mysumhtm
    syntax varlist using [, replace append ]
    tempname fh
    file open `fh' `using', write `replace' `append'
    file write `fh' "<html><body><table>"
    file write `fh' _n "<thead><th>variable</th>" ///
        "<th>mean</th><th>sd</th></thead>"
    foreach v of local varlist {
        quietly summarize `v'
        file write `fh' _n "<tr><td>`v'y</td><td>" ///
             (r(mean)) "</td><td>" (r(sd)) "</td></tr>"
    }
    file write `fh' _n  "</table></body></html>"
    file close `fh'
```

```
end

mysumhtm * using example.html, replace
type example.html
```
<run> <show>

## Wrappers

- Of course you do not have to write a new program for everything.

- Check the SSC Archive to find out whether anything already exists that serves your needs (see **findit** and **ssc**).

- For example, **mat2txt** can be used to write a matrix to a tab-delimited file:

```
sysuse auto
regress price weight mpg for
mat V = e(V)
mat2txt, matrix(V) saving(example.txt) replace ///
    title(This is a variance matrix)
```
<run> <show>

# Part 2: Handling model estimation results

## Results from "estimation" commands are special

- Results from e-class commands are special because they share a common structure:

  - a coefficients vector: **e(b)**

  - and a variance matrix: **e(V)**

- There is, to some degree, a consensus/common practice of how to design tables containing model estimation results.

- Many models are estimated, usually, and estimation may be computationally intensive so that archiving the results is an issue.

## Archiving estimation results

- A good approach is to keep model estimation and reporting two separate processes.
- This requires that model estimates are stored for later tabulation.
- Estimating a new model replaces the **e()**-returns of a previous model. However, the results can be stored in memory under a specific name using **estimates store** or the **eststo** user command.
- In Stata 10, it is also possible to save the results of a model on disk using **estimates save**.
- A problem with **estimates save** is that it can only store one model at the time (i.e. each model is saved in a separate file). However, the **estwrite** user command overcomes this problem.

## Archiving estimation results

- Example:

```
clear all
sysuse auto
bysort foreign: eststo: regress price weight mpg
eststo dir
estwrite * using mymodels, replace
<run>

dir mymodels*
<run>
```

- Two weeks later:

```
clear all
sysuse auto
estread mymodels
<run>
```

## Tabulating estimation results

- Various commands exists to compile and export tables of model estimates. **estout** is one of them. Others are **outreg** (John Luke Gallup), **outreg2** (Roy Wada), **xml_tab** (Lokshin & Sajaia), **outtex** (Antoine Terracol), **est2tex** (Marc Muendler), **mktab** (Nicholas Winter), **parmest** (Roger Newson), of which all have their pros and cons.

- The **estout** package contains four commands:

  **esttab**: User-friendly command to produce publication-style regression tables for screen display or in various export formats such as such as CSV, RTF, HTML, or LaTeX.

  **estout**: Generic program to compile regression tables (the engine behind **esttab**).

  **estadd**: Program to add extra results (such as e.g., beta coefficients) to **e()** so that they can be tabulated.

  **eststo**: Improved version of **estimates store**.

## Tabulating estimation results

- **esttab** and **estout** are very flexible an can produce all sorts of regression tables.

- I will only show a few basic examples here. Many more examples cab be found at the following website:

  http://repec.org/bocode/e/estout

- The basic procedure is to store a number of models and then apply **esttab** (or **estout**) to tabulate them:

```
clear all
sysuse auto
eststo: regress price weight mpg
eststo: regress price weight mpg foreign
esttab
```
*<run>*

## Tabulating estimation results

- **esttab** can either display the table in Stata's results window or export it to a file on disk using one of several formats, such as

  - **fixed**: fixed-format ASCII
  - **tab**: tab-delimited ASCII
  - **csv**: CSV (Comma Separated Value format) for use with Excel
  - **rtf**: Rich Text Format for use with word processors
  - **tex**: LaTeX format
  - ...

## Use with Excel

- Excel: **csv** or **scsv**

```
esttab using example.csv
<run>

esttab using example.csv, scsv replace
<run>
```

(The **scsv** format uses a semi-colon as delimiter which is appropriate for certain language versions of Excel.)

- Use the **plain** option if you intend to do additional computations in Excel:

```
esttab using example.csv, scsv replace wide plain
<run>
```

(No Excel XML support. Sorry.)

## Use with Word

- Word: **rtf**

```
esttab using example.rtf
<run>
```

- Appending is possible. Furthermore, use **varwidth(#)** and **modelwidth(#)** to change column widths:

```
esttab using example.rtf, append wide label modelwidth(8)
<run>
```

- Including RTF literals:

```
esttab using example.rtf, replace                ///
    title({\b Table 1: This is a bold title})
<run>

esttab using example.rtf, replace                ///
    cells(b(fmt(a3)) t(par(\i( ))))
<run>
```

## Use with LaTeX

- LaTeX: **tex**

```
esttab using example1.tex, label nostar         ///
    title(Regression table\label{tab1}) page
<run>

!texify.exe --pdf example1.tex
winexec $Acrobat example1.pdf
<run>
```

- LaTeX: **booktabs**

```
esttab using example2.tex, label nostar replace booktabs ///
    title(Regression table\label{tab1}) page

!texify.exe --pdf example2.tex
winexec $Acrobat example2.pdf
```
<run>

## Use with LaTeX

- Improved LaTeX table using the *dcolumn* package:

```
esttab using example3.tex, label replace booktabs ///
    alignment(D{.}{.}{-1})                        ///
    page(dcolumn)                                 ///
    title(Regression table\label{tab1})

!texify.exe --pdf example3.tex
winexec $Acrobat example3.pdf
```
<run>

## Use with LaTeX

- Advanced example

```
eststo clear

eststo: reg weight mpg
eststo: reg weight mpg foreign

eststo: reg price weight mpg
eststo: reg price weight mpg foreign

esttab using example4.tex, booktabs replace label   ///
    mgroups(A B, pattern(1 0 1 0)                   ///
        prefix(\multicolumn{@span}{c}{) suffix(})   ///
        span erepeat(\cmidrule(lr){@span}))         ///
    alignment(D{.}{.}{-1}) page(dcolumn) nonumber
eststo clear

!texify.exe --pdf example4.tex
winexec $Acrobat example4.pdf
```

<run>

## Other uses

- **esttab** can be used to tabulate any results, not just regression models, as long as they are posted in **e()** in an appropriate way.

- Example: descriptives table

```
sysuse auto
generate y = uniform()
quietly regress y price weight mpg foreign, noconstant
estadd summ
esttab, cells("mean sd min max") nogap nomtitle nonumber
```
*<run>*

**Part 3:Automatic reporting**

## Automatic reporting

- Automatic reporting means that results and information on formatting should be separated.
- It has to be possible to replace the data without loosing the formatting.
- The usual approach is to maintain a hand-edited master file that structures the document and sets the formatting etc. and then dynamically link the files containing results into this file.

## Example with LaTeX

- Step 1: Set up master file

```
file open fh using example.tex, replace write
file write fh       "\documentclass{article}"
file write fh _n    "\begin{document}"
file write fh _n    "\section{My Tables}"
file write fh _n(2) "\input{_example1.tex}"
file write fh _n(2) "\input{_example2.tex}"
file write fh _n(2) "\end{document}"
file close fh
type example.tex
```
*<run>*

## Example with LaTeX

- Step 2: Generate table files

```
clear all
sysuse auto

eststo: reg weight mpg
eststo: reg weight mpg foreign
esttab using _example1.tex, replace title(Weight)

eststo clear
eststo: reg price weight mpg
eststo: reg price weight mpg foreign
esttab using _example2.tex, replace title(Price)
```
*<run>*

```
!texify.exe --pdf example.tex
winexec $Acrobat example.pdf
```
*<run>*

## Example with LaTeX

- You can now easily replace the tables and recompile the document

```
eststo clear
eststo: reg weight mpg, vce(bootstrap)
eststo: reg weight mpg foreign, vce(bootstrap)
esttab using _example1.tex, replace title(Weight) ///
    se wide label

eststo clear
eststo: reg price weight mpg, vce(bootstrap)
eststo: reg price weight mpg foreign, vce(bootstrap)
esttab using _example2.tex, replace title(Price) ///
    se wide label
```
*<run>*

```
!texify.exe --pdf example.tex
winexec $Acrobat example.pdf
```
*<run>*

## Example with Word and Excel

- Such automation does not seem to be directly possible with Word.

- However, you can link data files into Excel and then dynamically link Excel tables into Word.

## Example with Word and Excel

- Step 1: Generate results files

```
clear all
sysuse auto
eststo: reg price weight mpg
eststo: reg price weight mpg foreign
esttab using _example.txt, tab replace plain
```
*<run>*

- Step 2: Link data into Excel ("data" > "import external data"; make sure to uncheck the "prompt for file" under "properties" in the last dialog) and format table.

- Step 3: In Word, paste the table as an Excel object ("edit" > "paste special"; make sure to paste as link)

## Example with Word and Excel

- You can now replace the results files and update the Excel and Word files.

```
clear all
sysuse auto
eststo: reg price weight mpg, vce(bootstrap)
eststo: reg price weight mpg foreign, vce(bootstrap)
esttab using _example.txt, tab replace plain label
```
*<run>*

**Thank you for your attention!**

<*clean-up*>