
Tutorial:

Output processing and making tables in Stata

Ben Jann, University of Bern, ben.jann@soz.unibe.ch

Odense, December 6, 2018

Required user packages:

- **estout** *<install>*
- **estwrite** *<install>*
- **mat2txt** *<install>*

<next>

Outline

<next> *<top>*

Introduction I

Statistical software packages are good at analyzing data, but they are often 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.

  => you have to 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.

  => you have to transform results

<next> *<overview>*

Introduction II

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

  => you have to rearrange and reformat results

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

  => you have to 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.

  => you have to archive results

<next> *<overview>*

Introduction III

TWO MAXIMS

1) Never Copy/Paste results by hand

You will almost surely make tons of mistakes!

2) Do everything only once

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

<next> *<overview>*

Introduction IV

- These two goals can be reached by **automation.**
- Automation has its **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, ...

<next> *<overview>*

Introduction V

- 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 against 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.

<next> *<overview>*

Part 1: General issues

<next> *<overview>*

Accessing results in Stata I

- 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)**.

<next> <overview>

Accessing results in Stata II

- 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, clear
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>

<next> <overview>

Accessing results in Stata III

- 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>

<next> *<overview>*

Accessing results in Stata IV

- 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>

```
margins   /*same result*/
```
<run>

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

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

<next> *<overview>*

Getting things out of Stata: The file command I

- 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.

<next> *<overview>*

Getting things out of Stata: The file command II

- Example: Write a tab delimited file containing descriptive statistics

```
sysuse auto, clear
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>*

<next> *<overview>*

Getting things out of Stata: The file command III

- 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, clear
mysumtab * using example.txt, replace
type example.txt
```

<run> *<show>*

<next> *<overview>*

Getting things out of Stata: The file command IV

- 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'</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>*

<next> *<overview>*

Getting things out of Stata: The file command V

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

- Check the SSC Archive to find out whether any output routine 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:

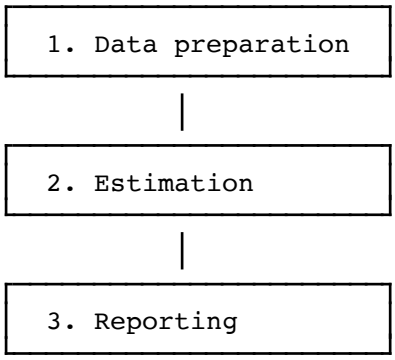
```
sysuse auto, clear
regress price weight mpg for
mat V = e(V)
mat2txt, matrix(V) saving(example.txt) replace ///
    title(This is a variance matrix)
```

<run> *<show>*

<next> *<overview>*

Data analysis in three steps

- Data analysis can be conceptualized as a three step process:



- The three steps are often done within different sessions and on different machines.

- Hence, results from step 2 need to be saved so that they can be further processed in step 3.

<next> *<overview>*

Part 2: How to deal with model estimates

<next> *<overview>*

Results from "estimation" commands

- 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.

<next> *<overview>*

Archiving models I

- 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.

- Since Stata 10, it is also possible to save the results of a model on disk using **estimates save**.

- A user-friendly command to store multiple sets in one file is provided by the **estwrite** user command.

<next> *<overview>*

Archiving models II

- Store estimation sets on disk:

  estwrite [*namelist* **using**] *filename* [**,** *options*]

- See what's in a file:

  estread *filename*, **describe**

- Import estimation sets from a file:

  estread [*namelist* **using**] *filename* [**,** *options*]

<next> *<overview>*

Archiving models III

- Example:

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

dir mymodels*
estread mymodels, describe
<run>
```

- Two weeks later:

```
estimates clear
sysuse auto, clear
estread mymodels
<run>
```

<next> <overview>

Tabulating estimation results I

- 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

  esttab: Produce publication-style regression tables for screen display or export into CSV, RTF, HTML, LaTeX, etc.

  estout: Engine behind **esttab**.

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

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

  estpost: Post results from non-e() commands in e().

<next> <overview>

Tabulating estimation results II

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

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

  http://repec.sowi.unibe.ch/stata/estout/index.html

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

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

<next> *<overview>*

Tabulating estimation results III

- **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 MS Excel

  - **rtf:** Rich Text Format for use with word processors

  - **tex:** LaTeX format

  - **...**

<next> *<overview>*

Tabulating estimation results IV

- Use with MS Excel: **csv** or **scsv**

  ```
  esttab using example.csv, replace
  ```
  <run> *<show>*

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

  (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 MS Excel:

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

  (No XML support. Sorry.)

<next> *<overview>*

Tabulating estimation results V

- Use with MS Word: **rtf**

  ```
  esttab using example.rtf, replace
  ```
  <run> *<show>*

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

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

- Including RTF literals:

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

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

<next> *<overview>*

Tabulating estimation results VI

- Use with LaTeX: **tex**

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

<run>

Compile the LaTeX file document:

<texify>

View the result:

<show PDF>

<next> *<overview>*

Tabulating estimation results VII

- Improved LaTeX table using the *booktabs* package:

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

<run>

<texify> *<show PDF>*

- Improved LaTeX table using the *dcolumn* package:

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

<run>

<texify> *<show PDF>*

<next> *<overview>*

Tabulating estimation results VIII

- Advanced LaTeX 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 example.tex, replace                    ///
    label booktabs nonumber                          ///
    page(dcolumn)                                    ///
    alignment(D{.}{.}{-1})                           ///
    mgroups(A B, pattern(1 0 1 0)                    ///
        prefix(\multicolumn{@span}{c}{) suffix(})    ///
        span erepeat(\cmidrule(lr){@span}))
```

<run>

<texify> *<show PDF>*

<next> *<overview>*

Tabulating results from non-estimation commands I

- Unfortunately, many commands do not post their results in **e()**, but we still might wish to tabulate the results using a command such as **esttab**.

- Two approaches:

  A) For results from post-estimation commands it is usually most convenient to add the results to the model's existing **e()**-returns. This can be done using **estadd**.

     For example, **estadd** provides support for Scott Long's SPost commands.

  B) For results from other commands such as, say, **summarize** or **tabulate** it makes sense to post the results in a new **e()**-set. The **estpost** command was written for this purpose.

<next> *<overview>*

Tabulating results from non-estimation commands II

- **estpost** supports the following commands:

summarize	summary statistics
tabstat	summary statistics
ttest	two-group mean-comparison tests
prtest	two-group tests of proportions
tabulate	one-way or two-way frequency table
svy: tabulate	frequency table for survey data
correlate	correlations
ci	confidence intervals for means, proportions, or counts
stci	confidence intervals for means and percentiles of survival time

- The basic syntax is:

  estpost *command* [*arguments*] [, *options*]

<*next*> <*overview*>

Tabulating results from non-estimation commands III

- Example: Post summary statistics using **estpost summarize**

```
eststo clear
sysuse auto, clear

estpost summarize price mpg rep78 foreign, listwise

esttab, cells("mean sd min max") nomtitle nonumber
```
<*run*>

- Example: Post summary statistics using **estpost tabstat**

```
estpost tabstat price mpg rep78, by(foreign) ///
    statistics(mean sd) columns(statistics) listwise

esttab, main(mean) aux(sd) nostar unstack ///
    noobs nonote nomtitle nonumber label
```
<*run*>

<*next*> <*overview*>

Tabulating results from non-estimation commands IV

- Example: One-way frequency table

```
estpost tabulate foreign

esttab, cells("b(lab(freq)) pct(fmt(2)) cumpct(fmt(2))") ///
    nonumber nomtitle noobs
```

<run>

- Example: One-way frequency table with funny labels

```
lab def origin 0 "Car type: domestic" ///
               1 "Car type: foreign", modify

estpost tabulate foreign

esttab, cells("b(lab(freq)) pct(fmt(2)) cumpct(fmt(2))") ///
    varlabels(`e(labels)')                                ///
    varwidth(20) nonumber nomtitle noobs
```

<run>

<next> *<overview>*

Tabulating results from non-estimation commands V

- Example: Two-way table with complex survey data

```
webuse nhanes2b, clear

svyset psuid [pweight=finalwgt], strata(stratid)

estpost svy: tabulate race diabetes, row percent

esttab ., se nostar nostar unstack
```

<run>

<next> *<overview>*

Tabulating results from non-estimation commands V

- Example: Tabulate correlation coefficients

  ```
  sysuse auto, clear

  estpost correlate price turn foreign rep78

  esttab, cell("rho p count") noobs
  ```
  <run>

- Example: Correlation matrix

  ```
  estpost correlate price turn foreign rep78, matrix listwise

  esttab, unstack not noobs nonum nomti compress
  ```
  <run>

<next> *<overview>*

End of tutorial

- Clean-up: erase working files

  ```
  capture erase mymodels.sters
  capture erase mymodels.dta
  capture erase example.txt
  capture erase example.html
  capture erase example.csv
  capture erase example.rtf
  capture erase example.tex
  capture erase example.pdf
  capture erase example.aux
  capture erase example.log
  capture erase _example.txt
  capture erase _example1.tex
  capture erase _example2.tex
  ```
  <run>

<top>