Stata Tutorial

Eric Hardy

January 10, 2019

Abstract

This tutorial provides a overview of Stata's basic functionality. This is accompanied by an executable Stata file and two datasets; run the do-file in Stata to better understand what it does

1 Overview

Use Stata for: data manipulation; data analysis; graphing; tables out to latex

1.1 Topics in this tutorial

- Basics: import data, browse dataset, create variables
- Dates: create date variables, create lead and lag variables
- Loops and "Macros": execute commands over a set of inputs, "by" groups
- Merge/Append: merge and append data from multiple datasets
- regression: basic syntax, lead/lag variables, storing regression output
- Tables to Latex:
- Graphing

1.2 Properties of Stata

- only one dataset in memory at a time
- (almost) everything you do will be represented as a column "variable" in the dataset
- variables cannot be overwritten; you must "drop" them before re-creating them
- numeric missing values are denoted by . which is the largest number Stata can hold
- character missing values are denoted by ""
- `if` statements evaluate to `1` if true, and `0` if false

Basic Syntax:

```
[by varlist:] command [arguments] [if] [in] [, options]
```

`[brackets]` indicate that the syntax is optional

1.3 Finding help

- type `help [command name]` at the stata prompt
- alternatively, google `stata [command name]`
- Help file layout:
  - syntax
  - options
  - option descriptions
  - examples
  - stored values

1.4 `.ado` files

- stata equivalent to packages
- to install, type `ssc install [.ado file name]` at the stata prompt
- ex: ssc install listtex
- once installed, these commands can be used like any other command

1.5 Miscellaneous

- to open GUI Stata from the command line, type `xstata` in the terminal
- to run a shell command from stata, type `! [command]`
- to continue a command on a second line, type `///` at the end of the first line
- to run a single command or group of commands from the do-file editor:
  - highlight the commands
  - Mac: command-shift-d; Windows/Linux: control-d

1.6 Editorializing

- give your saved datasets the same name as the do-file that created them
- if you are using lots of loops, you are doing something wrong
- keep variable names short and informative
- when creating new variables, indicate relationships to other variables with names
- ex: the varible `height_max` could store the maximum value of `height`

2 Basics

Remove all datasets, matrices, etc. from memory

```
. clear all
```

do not wait for user input after printing the first page of output

```
. set more off
```

Import files Import .dta file (Stata file type)

```
. use oecd_panel.dta, clear
```

Additional examples

```
. //import .csv file
. //insheet using oecd_panel.csv, comma names clear
. //see more here: http://www.stata.com/help.cgi?insheet
.
. //import .xlsx file
. //import excel using oecd_panel.xlsx, sheet("Sheet1") firstrow clear
. //see more here: http://www.stata.com/manuals13/dimportexcel.pdf
```

2.1 Viewing the Dataset

Observe changes to the dataset in the browse window after running each command
The `browse` command is commented out with `//` here for technical reasons only.
For displaying subsets of the dataset in this tutorial, we use `list` instead.

```
. //browse
```

Red columns are strings
Black columns are numbers (or dates!)
Blue columns are "encoded" variables; values in blue are not the actual values

```
. tab country
```

Country	Freq.	Percent	Cum.
Argentina	11	2.42	2.42
Australia	12	2.64	5.05
Austria	12	2.64	7.69
Belgium	12	2.64	10.33
Brazil	12	2.64	12.97
Canada	12	2.64	15.60
Chile	12	2.64	18.24
Czech Republic	12	2.64	20.88

Denmark	12	2.64	23.52
Estonia	12	2.64	26.15
Finland	12	2.64	28.79
France	12	2.64	31.43
Germany	12	2.64	34.07
Greece	4	0.88	34.95
Hungary	12	2.64	37.58
Iceland	12	2.64	40.22
Indonesia	12	2.64	42.86
Ireland	11	2.42	45.27
Israel	12	2.64	47.91
Italy	12	2.64	50.55
Japan	12	2.64	53.19
Korea	12	2.64	55.82
Luxembourg	11	2.42	58.24
Mexico	12	2.64	60.88
Netherlands	12	2.64	63.52
New Zealand	11	2.42	65.93
Norway	12	2.64	68.57
Poland	12	2.64	71.21
Portugal	12	2.64	73.85
Russian Federation	11	2.42	76.26
Slovak Republic	12	2.64	78.90
Slovenia	12	2.64	81.54
South Africa	12	2.64	84.18
Spain	12	2.64	86.81
Sweden	12	2.64	89.45
Switzerland	12	2.64	92.09
Turkey	12	2.64	94.73
United Kingdom	12	2.64	97.36
United States	12	2.64	100.00
Total	455	100.00	

```
. list country year quarter gdp chg_invtr if country == "Argentina"
```

	country	year	quarter	gdp	chg_in~r
1.	Argentina	2010	2	1392033	.
2.	Argentina	2010	3	1478053	.
3.	Argentina	2010	4	1584674	.
4.	Argentina	2011	1	1696802	.
5.	Argentina	2011	2	1822131	.
6.	Argentina	2011	3	1885742	.
7.	Argentina	2011	4	1963413	.
8.	Argentina	2012	1	2030251	.
9.	Argentina	2012	2	2099170	.
10.	Argentina	2012	3	2205873	.

```
     11. | Argentina   2012         4   2321690          . |
```

`if` statements evaluate to `1` if true, and `0` if false

```
. gen argentina_dummy = (country == "Argentina")
```

remove all observations from Belgium from the dataset

```
. drop if country == "Belgium"
(12 observations deleted)
```

remove `argentina_dummy` from the dataset

```
. drop argentina_dummy
```

2.2 Counting and Missing Values

count the number of observations in the dataset

```
. count
  443
```

count the number of `country` observations with non-missing values

```
. count if country != ""
  443
```

count the number of observations for Argentina

```
. count if country == "Argentina"
  11
```

be careful - stata will count missing values as greater than 1000000

```
. count if chg_invtr > 1000000
  385
```

exclude the missing values

```
. count if chg_invtr > 1000000 & chg_invtr != .
  0
```

2.3 Creating Variables

Create a dummy variable for each country (`c1`, `c2`, etc.)
The `quietly` prefix suppresses the terminal output of the command

```
. quietly tabulate country, gen(c)
```

```
. list country c1 c2 if inlist(country, "Argentina", "Austria")
```

	country	c1	c2
1.	Argentina	1	0
2.	Argentina	1	0
3.	Argentina	1	0
4.	Argentina	1	0
5.	Argentina	1	0
6.	Argentina	1	0
7.	Argentina	1	0
8.	Argentina	1	0
9.	Argentina	1	0
10.	Argentina	1	0
11.	Argentina	1	0
24.	Austria	0	0
25.	Austria	0	0
26.	Austria	0	0
27.	Austria	0	0
28.	Austria	0	0
29.	Austria	0	0
30.	Austria	0	0
31.	Austria	0	0
32.	Austria	0	0
33.	Austria	0	0
34.	Austria	0	0
35.	Austria	0	0

create a new variable

```
. gen double date = year + quarter/10
```

the special variable `_n` takes a value of `1` in the first row, `2` in the 2nd row, etc.

```
. gen counter = _n
```

the special variable `_N` denotes the number of observations

```
. gen num_obs = _N
```

variables cannot be overwritten; use `replace`

```
. replace counter = counter - 1
(443 real changes made)
```

replace the value of a variable in a specific row(s)

```
. replace counter = -5 in 2
(1 real change made)
. replace counter = -6 in 4/10
(7 real changes made)
. list counter if _n <= 12
```

	counter
1.	0
2.	-5
3.	2
4.	-6
5.	-6
6.	-6
7.	-6
8.	-6
9.	-6
10.	-6
11.	10
12.	11

3 Panel Data

create a variable containing an id number for each country

```
. sort country date
```

`id` has value `1` in the first observation for every country and `0` otherwise

```
. by country: gen id = (_n == 1)
```

sum the current observation of `id` with the previous observation
this yields a unique `id` value for each country
this is an example of looping functionality without explicitly writing a loop

```
. replace id = id + id[_n-1] if _n != 1
(442 real changes made)
```

"cross-tabulate" country names and id numbers ex: this shows 12 observations for Australia, all with id 2

```
. tab country id if inlist(country, "Argentina", "Australia", "Austria")
```

	id			
Country	1	2	3	Total

Argentina	11	0	0	11
Australia	0	12	0	12
Austria	0	0	12	12
Total	11	12	12	35

sort the data by country (ascending), and within country sort by date (ascending)

```
. sort country date
```

//sort the data by country (ascending), and within country sort by date (descending)

```
. gsort country -date
```

`by [varlist]` can only be used when the data is sorted by `[varlist]`
find the number of observations for each country

```
. by country: gen country_obs = _N
. tab country_obs
```

country_obs	Freq.	Percent	Cum.
4	4	0.90	0.90
11	55	12.42	13.32
12	384	86.68	100.00
Total	443	100.00	

From this crosstab, we see that one country has four observations, 5 countries have 11 observations each, and 32 countries have 12 observations each.

3.1 Lead and Lag Variables

Create `gdp` lagged one time period
This command does not account for the panel structure of the dataset
The first value of `gdp_1` for Austria is the last value of `gdp` for Australia

```
. sort country date
. gen gdp_1 = gdp[_n-1]
(1 missing value generated)
```

browse the dataset to see the problem

```
. list country date gdp gdp_1 if inlist(country, "Argentina", "Austria")
```

country	date	gdp	gdp_1

	country	date	gdp	gdp_1
1.	Argentina	2010.2	1392033	.
2.	Argentina	2010.3	1478053	1392033
3.	Argentina	2010.4	1584674	1478053
4.	Argentina	2011.1	1696802	1584674
5.	Argentina	2011.2	1822131	1696802
6.	Argentina	2011.3	1885742	1822131
7.	Argentina	2011.4	1963413	1885742
8.	Argentina	2012.1	2030251	1963413
9.	Argentina	2012.2	2099170	2030251
10.	Argentina	2012.3	2205873	2099170
11.	Argentina	2012.4	2321690	2205873
24.	Austria	2010.2	283165.3	1518220
25.	Austria	2010.3	289217.8	283165.3
26.	Austria	2010.4	294233.2	289217.8
27.	Austria	2011.1	297794.3	294233.2
28.	Austria	2011.2	300544.8	297794.3
29.	Austria	2011.3	302313.1	300544.8
30.	Austria	2011.4	304558.3	302313.1
31.	Austria	2012.1	307851.2	304558.3
32.	Austria	2012.2	310038.3	307851.2
33.	Austria	2012.3	311889.2	310038.3
34.	Austria	2012.4	313206.7	311889.2
35.	Austria	2013.1	314929.4	313206.7

```
. drop gdp_1
```

to fix this problem, re-generate the variable by country

```
. by country: gen gdp_1 = gdp[_n-1]
(38 missing values generated)
```

browse the dataset to be sure the problem no longer exists

```
. list country date gdp gdp_1 if inlist(country, "Argentina", "Austria")
```

	country	date	gdp	gdp_1
1.	Argentina	2010.2	1392033	.
2.	Argentina	2010.3	1478053	1392033
3.	Argentina	2010.4	1584674	1478053
4.	Argentina	2011.1	1696802	1584674
5.	Argentina	2011.2	1822131	1696802
6.	Argentina	2011.3	1885742	1822131

7.	Argentina	2011.4	1963413	1885742
8.	Argentina	2012.1	2030251	1963413
9.	Argentina	2012.2	2099170	2030251
10.	Argentina	2012.3	2205873	2099170
11.	Argentina	2012.4	2321690	2205873
24.	Austria	2010.2	283165.3	.
25.	Austria	2010.3	289217.8	283165.3
26.	Austria	2010.4	294233.2	289217.8
27.	Austria	2011.1	297794.3	294233.2
28.	Austria	2011.2	300544.8	297794.3
29.	Austria	2011.3	302313.1	300544.8
30.	Austria	2011.4	304558.3	302313.1
31.	Austria	2012.1	307851.2	304558.3
32.	Austria	2012.2	310038.3	307851.2
33.	Austria	2012.3	311889.2	310038.3
34.	Austria	2012.4	313206.7	311889.2
35.	Austria	2013.1	314929.4	313206.7

this is still not perfect - missing values will cause problems ex: suppose Argentina is missing its 2012 Q2 value for gdp a one-observation lag will record the value from 2012 Q1 as the lagged value for 2012 Q3. create correct lagged values by using dates

```
. drop gdp_1
```

miscellaneous panel data examples `egen` commands are user generated commands in practice they work just like the `gen` command list of egen commands (including `mean`, `min`, `max`, `sum`, `sd`, `rowtotal`, etc.) `https://www.stata.com/manuals13/degen.pdf`
note: there is a difference between `gen var_sum = sum()` and `egen var_sum = sum()` create a variable that is the running sum of the `id` variable

```
. by country: gen id_gen_sum = sum(id)
```

create a variable that sums the `id` variable by `country`

```
. by country: egen id_egen_sum = sum(id)
```

sum all rows with names beginning in `id_`

```
. egen id_rowtotal = rowtotal(id_*)
```

4 Dates

```
. drop time
```

the number of months since Q1, 1960

```
. gen time = yq(year, quarter)
```

`MMDDYYYY` dates are expressed as the number of days since 1960.
to let Stata treat observations as temporally related, use `tsset`

```
. sort id time
. tsset id time
       panel variable:  id (unbalanced)
        time variable:  time, 201 to 212
                delta:  1 unit
```

the command `tsset clear` will remove the current `tsset` settings
with `tsset`, we can construct correct leads and lags create `gdp` correctly lagged one time period

```
. gen gdp_1 = L.gdp
(38 missing values generated)
```

create gdp from two quarters in the future

```
. gen gdp_f2 = F2.gdp
(76 missing values generated)
```

create changes in `gdp`

```
. gen dgdp = gdp - L.gdp
(38 missing values generated)
. gen Dgdp = D.gdp
(38 missing values generated)
```

5 Loops and Macros

```
. local i = 0
. while `i´ < 5 {
  2.         local i = `i´ + 1
  3.         display `i´
  4. }
1
2
3
```

```
4
5

.
. forvalues i =0(2)10 {
  2.         display `i´
  3. }
0
2
4
6
8
10

.
. foreach num of numlist 1 8 7 6 {
  2.         display `num´
  3. }
1
8
7
6

.
. foreach var of varlist chg_invtr exp_gds exp_gds_svc exp_svc cons_expnd ///
>                        gross_cap_fmtn gdp imp_gds imp_gds_svc imp_svc {
  2.     //displays the variable name (string)
.     display "`var´"
  3.     //creates one and two period lags for each variable
.         gen `var´_l1 = L.`var´
  4.         gen `var´_l2 = L2.`var´
  5. }
chg_invtr
(388 missing values generated)
(393 missing values generated)
exp_gds
(399 missing values generated)
(403 missing values generated)
exp_gds_svc
(38 missing values generated)
(76 missing values generated)
exp_svc
(399 missing values generated)
(403 missing values generated)
cons_expnd
(378 missing values generated)
(384 missing values generated)
gross_cap_fmtn
(378 missing values generated)
(384 missing values generated)
gdp
```

```
(38 missing values generated)
(76 missing values generated)
imp_gds
(399 missing values generated)
(403 missing values generated)
imp_gds_svc
(38 missing values generated)
(76 missing values generated)
imp_svc
(399 missing values generated)
(403 missing values generated)
```

6 Merge/Append/Joinby

The current dataset is called `master`
The dataset to be merged in is called `using`

m:1 is "many to one"
each date exists for many countries in `master`; dates in `using` are unique

```
. merge m:1 year quarter using ten_year_treasury.dta
(note: variable quarter was byte, now float to accommodate using data´s values)

    Result                           # of obs.
    -----------------------------------------
    not matched                           229
        from master                         0  (_merge==1)
        from using                        229  (_merge==2)

    matched                               443  (_merge==3)
    -----------------------------------------
```

each merge command creates a `_merge` variable:
`_merge == 1`: `master` only
`_merge == 2`: `using` only
`_merge == 3`: in both `master` and `using`

```
. keep if _merge == 3
(229 observations deleted)
```

7 Regression

before regressing on `tsset` lagged variables, the dataset must be sorted

```
. sort id time
```

Regression syntax: `regress dependent_var indep_var_1 indep_var_2 indep_var_3`

```
. regress gdp L.gross_cap_fmtn L.imp_gds_svc L.exp_gds_svc
```

Source	SS	df	MS	Number of obs	=	65
				F(3, 61)	=	27101.03
Model	2.0027e+18	3	6.6756e+17	Prob > F	=	0.0000
Residual	1.5026e+15	61	2.4632e+13	R-squared	=	0.9993
				Adj R-squared	=	0.9992
Total	2.0042e+18	64	3.1315e+16	Root MSE	=	5.0e+06

gdp	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
gross_cap_fmtn L1.	4.214548	.4913233	8.58	0.000	3.232086	5.19701
imp_gds_svc L1.	-.1324612	.3399912	-0.39	0.698	-.8123155	.5473931
exp_gds_svc L1.	1.122472	.474752	2.36	0.021	.1731464	2.071797
_cons	-493976.9	803195.7	-0.62	0.541	-2100066	1112112

automatically create dummies for each country id

```
. quietly areg gdp L.gross_cap_fmtn L.imp_gds_svc L.exp_gds_svc, absorb(id)
```

create quarter and year dummies from the quarter and year variables

```
. quietly xi: areg gdp L.gross_cap_fmtn L.imp_gds_svc L.exp_gds_svc i.quarter i.year, abso
```

this is the same as the previous regression, but storing regressors in a global variable

```
. global controls = "L.imp_gds_svc L.exp_gds_svc i.quarter i.year"
. quietly areg gdp L.gross_cap_fmtn $controls , absorb(id)
```

Note: `controls` could also be defined as a local variable

look at the bottom of `help regress` to see stored values
retrieve coefficents and variance/covariance matrix:

```
. matrix b = e(b)
. matrix v = e(V)
```

display a matrix

```
. mat list b
```

```
b[1,12]
                L.            L.            L.           1b.            2.            3.
    gross_cap_~n   imp_gds_svc   exp_gds_svc       quarter       quarter       quarter
y1     .98436068    -.57815634     -.2665775             0    -325852.38     706927.13

                4.        2010b.          2011.         2012.         2013.
          quarter          year          year          year          year         _cons
y1     343274.96             0    -624434.66     358447.24     809559.59      81040767
```

display an element of a matrix

```
. di v[2,1]
-.03946835
```

store outreg options related to regression table formatting rescale `gdp`

```
. global outreg_options = "nor2 coefast se nolabel bracket"

. replace gdp = gdp/1000000
(443 real changes made)

. replace gdp_1 = gdp_1/1000000
(405 real changes made)
```

Regression for all countries

```
. quietly areg gdp L.gross_cap_fmtn $controls, absorb(id)

. outreg2 using table1.xls, $outreg_options addstat("R-squared", e(r2)) replace
table1.xls
dir : seeout
```

Regression for Argentina

```
. quietly areg gdp L.gross_cap_fmtn $controls if id == 1, absorb(id)

. outreg2 using table1.xls, $outreg_options addstat("R-squared", e(r2)) append
table1.xls
dir : seeout
```

8 Graphing

store the first coefficient from the last regression

```
. global coefficient = b[1,1]

. global coefficient = round($coefficient, 0.001)
```

set the width and height of the saved graph notice that the `width()` and `height()` commands come after the comma this indicates that they are optional commands. without them, Stata would create a graph using a default width and height

```
. global pngwidth = 800
. global pngheight = 600
```

the line beginning with `graphregion` makes the graph background white and removes the border from the legend
the `replace` option will replace an existing `graph.png` file with the current file

```
. twoway  (line gdp time, lcolor(red) lpattern(dash)) ///
>                 (line gdp_1 time, lcolor(blue) lpattern(dotdash)) if country == "Argentin
>                 title("This is a graph, and this is a Coefficient: $coefficient ") ///
>                 xtitle("Date") ytitle("GDP") ///
>                 graphregion(fcolor(white) lcolor(white) color(white)) legend(region(lcolo
> )
(note:  named style dotdash not found in class linepattern, default attributes used)
(note:  named style dotdash not found in class linepattern, default attributes used)

.
. graph export graph.png, replace width( $pngwidth ) height( $pngheight )
(file graph.png written in PNG format)
```

display the graph in a pdf created in latex
to do this, create a text file in stata containing the lines needed in a latex file:
the `_n` at the end of each line includes a "newline" character in the file

```
. file open f1 using "graph.tex", write replace
. file write f1 "\documentclass[10pt]{article}" _n
. file write f1 "\usepackage{graphicx}" _n
. file write f1 "\usepackage[margin=1in]{geometry}" _n
. file write f1 "\usepackage[labelformat=empty]{caption}" _n
. file write f1 "\newcommand{\fullscale}{.33}" _n
. file write f1 "\begin{document}" _n
. file write f1 "\clearpage" _n
. file write f1 "\begin{figure}" _n
. file write f1 "\begin{center}" _n
. file write f1 "Figure 1: What a figure!\\" _n
. file write f1 "\includegraphics[scale=\fullscale]{graph.png}" _n
. file write f1 "\end{center}" _n
. file write f1 "\caption{Notes: Write your notes here.}" _n
. file write f1 "\end{figure}" _n
. file write f1 "\end{document}" _n
. file close f1
```

compile the graph file in latex

```
. quietly ! pdflatex graph.tex
```

remove the auxiliary files created when compiling latex

```
. quietly ! rm graph.aux graph.log
```

9 Tables to Latex

Export stored regression results
Labels on variables in the regression will be included in the table

```
. label variable gdp "Gross Domestic Product"
. label variable gross_cap_fmtn "Gross Capital Formation"
. label variable imp_gds_svc "Imports of Goods and Services"
. label variable exp_gds_svc "Exports of Goods and Services"
```

run regressions, and store results using `eststo`

```
. quietly xi: areg gdp gross_cap_fmtn i.quarter i.year, absorb(id)
. eststo table_column_1
. quietly xi: areg gdp gross_cap_fmtn imp_gds_svc exp_gds_svc i.quarter i.year, absorb(id)
. eststo table_column_2
```

export the table of regressions to latex
`esttab table_column_* ...` includes all objects in `eststo` with names beginning in `table_column_`

```
.
. local notes_1 = "Put some notes here."
. local notes_2 = "Include more notes here if you would like."
. local significance = "  \\  Significance: * significant at 10\%; ** significant at 5\%; *
> icant at 1\%."
.
. esttab table_column_*  ///
> using gdp_table.tex, replace  ///
> star(* 0.10 ** 0.05 *** 0.01) /*plain*/ nogaps   ///
> nodepvars b(%9.3f) legend noabbrev style(tex) constant   ///
> title("Table 1: Two Nonsensical Regressions")   ///
> label stats(N r2_a, fmt(%9.0g %9.3g) labels("Observations" "Adjusted R-Squared")) se  //
> order(gross_cap_fmtn imp_gds_svc exp_gds_svc /*_cons*/)    ///
> keep( gross_cap_fmtn imp_gds_svc exp_gds_svc /*_cons*/)     ///
> nomtitles  ///
> posthead("Dependent Variable & GDP & GDP  \\"  ///
```

```
>                 "\hline")  ///
> prefoot("\hline" ///
>                 "Country FEs  & Yes & Yes \\"  ///
>                 "\hline"  ///
>                 "Sample       & Full & Full \\"  ///
>                 "\hline")  ///
> nonotes ///
> postfoot(       "\hline\hline" ///
>                         "\end{tabular}}"  ///
>                         "\captionsetup{justification=justified}"  ///
>                         "\caption{Notes: `notes_1´ `notes_2´ `significance´ }"  ///
>                         "\end{table}")  ///
> substitute(     \begin{table}[htbp]\centering \begin{table}[p]\centering\captionsetup{wi
> xtwidth}\footnotesize{ )
(output written to gdp_table.tex)
```

create latex file to "wrap" around the gdp table output

```
. file open f1 using "gdp_table_wrapper.tex", write replace
. file write f1 "\documentclass[10pt]{article}" _n
. file write f1 "\usepackage[margin=1in]{geometry}" _n
. file write f1 "\usepackage[labelformat=empty]{caption}" _n
. file write f1 "\begin{document}" _n
. file write f1 "\clearpage" _n
. file write f1 "\input{gdp_table.tex}" _n
. file write f1 "\end{document}" _n
. file close f1
```

compile the GDP table in latex

```
. quietly ! pdflatex gdp_table_wrapper.tex
```

remove the auxiliary files created when compiling latex

```
. quietly ! rm gdp_table_wrapper.aux gdp_table_wrapper.log
```

export a (nicely formatted) table from the dataset to do this, create a latex (`.tex`) file within Stata by using the to export a portion of the dataset as a nicely formatted table, we can use `listtex`
install listtex if you haven't already

```
. ssc install listtex
checking listtex consistency and verifying not already installed...
all files already exist and are up to date.

.
. listtex country year quarter gdp if country == "Argentina" using argentina_table.tex, re
```

```
> le(tabular)  ///
>                 head(   "\documentclass[10pt]{article}"  ///
>                                 "\usepackage[margin=1in]{geometry}"  ///
>                                 "\usepackage[labelformat=empty]{caption}"  ///
>                                 "\begin{document}"  ///
>                                 "\begin{table}[htbp]\centering\captionsetup{width=0.8\te
> ootnotesize{"  ///
>                                 "\caption{Table 2: Argentina}"  ///
>                                 "\begin{tabular}{l l l  r }"  ///
>                                 "\hline\hline"  ///
>                                 "Country     & Year & Quarter & GDP \\"  ///
>                                 "\hline")  ///
>                 foot(   "\hline"  ///
>                                 "\end{tabular}}"  ///
>                                 "\caption{Notes: Write your notes here.}"  ///
>                                 "\end{table}"  ///
>                                 "\end{document}")
```

compile the latex file created by listtex

```
. quietly ! pdflatex argentina_table.tex
```

remove the auxiliary files created when compiling latex

```
. quietly ! rm argentina_table.log argentina_table.aux
```