Lecture: Quick introduction to Stata

222061-1617: Time Series Econometrics

Spring 2023

Jacek Suda

Outline

Outline:

1. Introduction
2. Getting data into Stata
3. Statistics and graphs
4. ARMA Estimation

Stata

- Stata is an integrated statistical analysis package
- Its main strengths are handling and manipulating large data sets
- Stata is a command-driven package: the best way to learn Stata is by typing in the commands.
- Commands can be entered in either of three ways:
  - Interactively: click through the menu on top of the screen
  - Manually: type the first command in the command window and execute it, then the next, then the next,...
  - Do-file: type up a list of commands in a "do-file", essentially a computer programme, and execute the do-file.

Starting the code

- [Optional] Start a log file
- Clear the workspace.
- Load the data...

  or generate your own.
- Examine, plot and [optionally, if needed] transform the data.
- Do the time series analysis.
- [Optional] Close the log file.

Log file

- We can record everything we do (both in command window and do file).
  - could be useful if we write commands directly into the command window rather than using do file
- Type
  `log using` *path\filename.log* `, replace text`
- At the end type
  `log close`

Help

- Four useful ways to look for help:
  1. type `help` *command*
  2. type `search` *command*
  3. use dialogue box
  4. google the command

Loading data

- Stata data sets usually identified with a .dta suffix
- ASCI data sets or flat files usually has an extension .raw (in Stata)
- Excel spreadsheets and other files can also be read
- Sometimes it is easier to paste data into the Stata editor and save
- Stata data set files read faster and can contain any labels or other information

Loading data: Example

- Consider the ASCI file `gdpq.prn` containing data on the US quarterly real GDP for the period 1954q1-2007q3
- Set the working path to the file:
  ```
  . cd c:\Stata\
  ```
- To load the variable *gdp.prn* into Stata, use `infile` command
  ```
  . infile gdp using gdpq.prn
  ```
- To save the data as a Stata datafile (with default extension *.dta*) write
  ```
  . save gdpq_data, all
  ```
  it saves data to file *gdp_data.dta* in our working directory.
- To overwrite an already existing file add option `replace` as in
  ```
  . save gdpq_data, all replace
  ```

Examining the data

- To calculate descriptive statistics of data set use the `summarize` command
  - `. summarize` to summarize all variables in the dataset
  - `. summarize gdp gdp2` to summarize only selected variables
- Use option `, detail` to obtain more descriptive statistics (percentiles, kurtosis, skewness and so on)
- To list all observations for GDP use command `list gdp`

Time Series in Stata

- Stata has a very particular set of functions that control time series commands
- After we loaded the data we can use these commands for our time series analysis...

  but before doing it we need to declare a dataset to be time-series data with command `tsset`, e.g.

  ```
  tsset datevar
  ```
- We can either use built in commands or write equations/formula on our own.

Time series

- To use the data for time series analysis we need it to tell Stata
- If there already exists variable (*date_variable*) in the dataset that correponds to time index (e.g. date index) we declare it with the command

  `tsset date_variable`
- If there is no such variable or we do not want to use it we must construct the time variable.
- We need to know:
  - what the date is for the first observation and
  - the frequency of the data (daily, weekly, quarterly or annual)
- Examples:

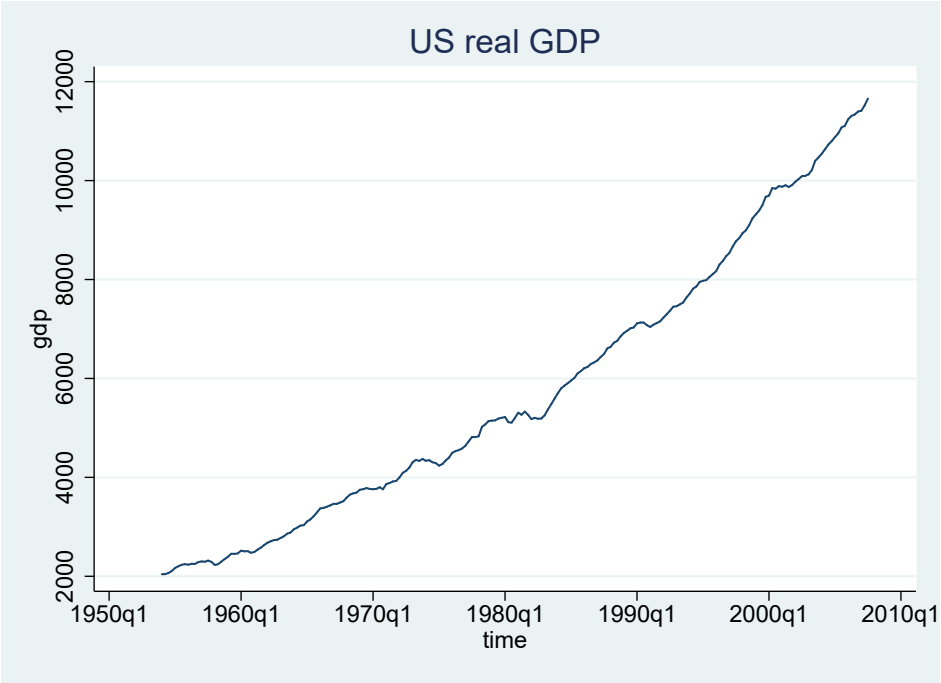
  `generate time = y(1991)+_n-1` — for annual data starting in 1991
  `format time %ty` — to set the format of time variable to annual
  `generate time = q(1954q1)+_n-1` — for quarterly data from 1954Q1
  `format time %tq` — to set the format of time variable to quarterly
  `generate time = m(1971m10)+_n-1` — for monthly data from 10/1971
  `format time %tm` — to set the format of time variable to monthly
- Finally, we need to declare the time series variable

  `tsset time`

Graphing

- With time series data, it is always a good idea to graph the data
- To do a simple line graph use `twoway` command
- Plotting our variable against time is done by the command

```
twoway line gdp time, title("US real GDP")
```



- The graph can be saved as Stata graph (*.gpf*) or exported (e.g. as *.pdf*)

```
graph save gdp.gph   or   graph export gdp.pdf
```

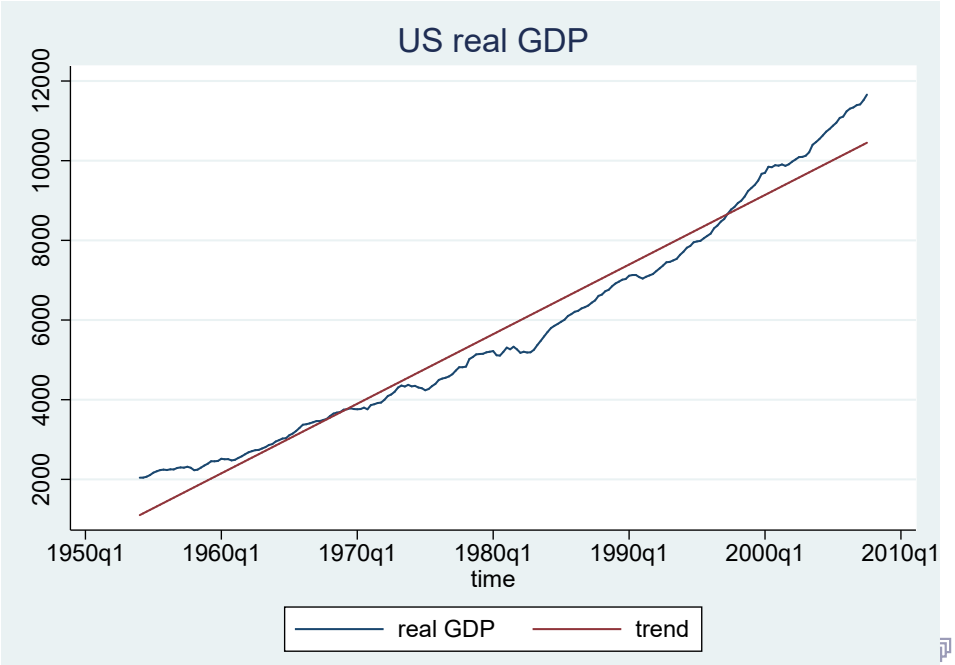
Graphing

- We can plot more than one series on one plot use.
- Example: adding a trend line
  1. Estimate a time trend for gdp: $gdp_t = c + t \times time_t + \varepsilon_t$

     ```
     regress gdp time
     ```
  2. Take $\widehat{gdp}_t$

     ```
     predict gdp_trend
     ```
  3. And plot both series: gdp and estimated time trend

     ```
     twoway (line gdp time) (line gdp_trend time), title("US real
     GDP") legend(label(1 "real GDP") label (2 "trend"))
     ```



Useful commands

- Once we defined time variable there are some useful Stata expressions
- For lags and leads, use the *L.varname* (to lag) and *F.varname* (to lead) commands

  `L.gdp` = *gdp(t-1)* and `L2.gdp`=*gdp(t-2)*

  `F.gdp` = *gdp(t+1)* and `F2.gdp` =*gdp(t+2)*
- For differencing the data use *D.varname* command to take the first difference, *D2.varname* to take the double difference (difference in difference), etc...

  `D.gdp` = *gdp(t)-gdp(t-1)*

  `D2.gdp` = *gdp(t)-gdp(t-1) - (gdp(t-1)-gdp(t-2)) = gdp(t)-2gdp(t-1)+gdp(t-2))*
- For seasonal differencing, the *S.varname* command is similar to the *D.varname* but the difference is always taken from the current observation to the $n-th$ observation:

  `S.gdp` = *gdp(t)-gdp(t-1)* and `S12.gdp` = *gdp(t)-gdp(t-12)*

Making data stationary

- The series for real GDP is clearly non-linear and non-stationary.
- To make it linear we can take $\log(GDP)$

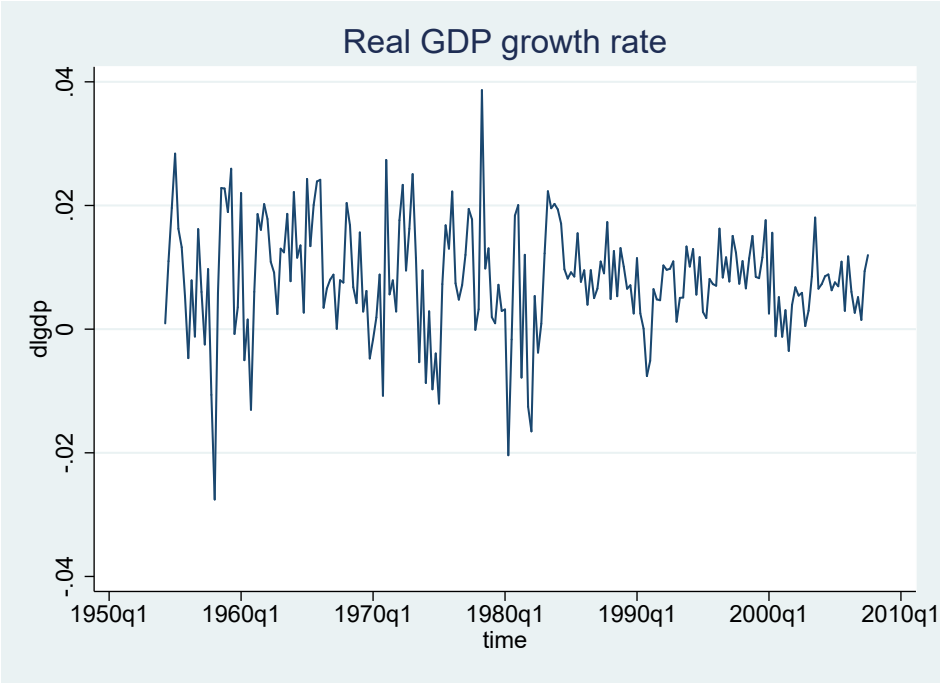
```
generate lgdp = log(gdp)
```

- We can generate first difference series as

```
generate dlgdp = d.lgdp
```

or we could use

```
generate ggdp = d.gdp/l.gdp
```



ACF, PACF, Q-test

- To see correlogram, i.e. table of autocorrelations and partial autocorrelation use command
  `corrgram` *varname*`, lags(x),`
  where `x` to number of lags for which we compute it.
- To plot ACF use
  `ac` *varname*`, lags(x)`
- To plot PACF use
  `pac` *varname*`, lags(x)`
- The Q-test for randomness is provided by
  `wntestq` *varname*`, lags(x)`

ARMA estimation

- The basic syntax for an ARMA(p,q) model is
  `arima` *varname*, `ar(1/p) ma(1/q)`
  or
  `arima` *varname*, `arima(p,0,q)`
- For pure AR(p) or MA(q) mode we can use
  `arima` *varname*, `ar(1/p)`
  or
  `arima` *varname*, `arima(p,0,0)`
- We could also write it as
  `arima` *varname*, `ar(1 2 ... p)`
- Writing only
  `arima` *varname*, `ar(p)`
  would estimate only

$$y_t = \alpha + \phi_p y_{t-p} + \varepsilon_t$$

ARMA estimation

- To estimate the model for a subset of data use `if tin(t0,t1)` where `t0` and `t1` are dates

  `arima` *varname* `if tin(1990q2,2010q4), ar(1/2)`

  - note that by leaving the first argument of `tin(,2010q4)` blank, we are including all available data through the second date (2010q4)
- Alternatively, use `in n0/n1` where `n0` and `n1` define the range of observations to be used

  `arima` *varname* `in 2/190, arima(2,0,0)`
- To estimate the model without a constant use `noconstant` option

  `arima` *varname*`, ar(1/p) ma(1/q) noconstant`
- To estimate conditional MLE instead of full MLE use `condition`

  `arima` *varname*`, ar(1/p) ma(1/q) condition`