

Gretl Mini-Manual

John E. Floyd
University of Toronto
September 26, 2011

Obtain your copy of Gretl by going to <http://gretl.sourceforge.net/win32/> and downloading the self-extracting zip file `gretl-1.9.5.exe`. Then run this file and follow the prompts, making sure that you end up with a `Gretl` icon on your desktop.

To **load your data**, take the spreadsheet file into which you obtained your data off the web or from elsewhere and copy a version that has as its first row the variable names along the top of the data columns and a left-most column providing some sort of identification of the observations—for time-series data this will indicate the dates of the observations. Make sure that the observation column has a variable name in the upper-left cell of the spreadsheet. After saving this spreadsheet as an `.xls` file, load Gretl by clicking on its icon. Click on `File`, then `Open data`, and then `Import` and then choose `Excel`. In the window that appears, go to the directory you are working in and click on the Excel spreadsheet file. For illustration we will use the file `usmondatt.xls`. Follow the prompts and you will see another window with a list of variables in it and another little window telling you that the imported data has been interpreted as undated (cross-sectional) and asking if you would like to give it a time-series or panel interpretation. Answer yes, choose `Time series`, and then click on `Forward`. Choose quarterly in the present case, click on `Forward` again and set the date as `1959:1` in the window that appears. Then click again on `Forward` and choose `Yes` if the beginning and ending dates that appear are the correct ones. Now delete the variable that has the name of your column of observations by clicking on it and then pressing the right mouse button and selecting `Delete`. You can examine the remaining variables by double-clicking on them with the left mouse button.

At this point it would be desirable for you to **add descriptions to the data series** in the data file. Do this by highlighting each variable in turn using the left mouse button, then clicking on `Variable`, choosing `Edit Attributes` and typing the appropriate descriptive material in the window that will be presented to you.

You can also **plot any variable** by highlighting it and choosing `Variable` and then `Time-series plot`. Alternatively, you can **plot two or more variables** together by clicking on `View` and then `Graph specified vars`, selecting the type of plot, and then specifying in the resulting window the variables you want to plot.

It should be obvious how you can perform many statistical operations by pointing and clicking. Before going further, however, you should **save your data** file by clicking on `File`, then `Save data` and giving your data file a name with the suffix `.gdt`. You should probably

choose the name Gretl suggests, which will be the same name as your Excel file with the suffix `.gdt` instead of `.xls`.

The fastest way to work in Gretl is by feeding it prepared script files. Exit Gretl, ignoring any prompt to save things (assuming that you have already saved your data file), and then load Gretl again. When you click on the Gretl icon, a screen will appear on which you should select **File** and then **Open data** and you will be prompted to load the data set that you had previously saved. Click on that name and the Data Window will again appear. Next, click on **Tools** and then **Command log** and a window will appear containing the following information.

```
# Log started 2010-08-19 15:57
# Record of session commands. Please note that this will
# likely require editing if it is to be run as a script.
open C:\DIRECTORY PATH\usmondatt.gdt
```

which you should highlight, then **Copy** by selecting the third icon from the right along the command line. Now open up your text editor and save this material in a new file—call it `usdemon.inp`. From now on, you can add commands to this file and load it into Gretl by clicking on **File**, then on **Script files**, and then on the name of the file. A script window will appear containing the material in the file. From this point forward, you can add material to the file in the script window and then click on the sixth icon from the left along the top of the window to execute the commands it contains. You have no choice but run the whole file unless you do as I do and insert your favorite single-word profanity in the file where you want the processing to stop—the program, being clean minded, will object to the profanity and processing will cease.

We now present an **illustrative script** that analyzes the data in `usmondatt.gdt` and then estimates a demand function for United States M1. The meaning of the code should be obvious, although I have provided prompts by making comments to the right of the code lines. Gretl ignores all material on a line to the right of the character `#`. At any time in Gretl you can click on **help** in the top-right corner of the Data Window to access detailed information about the program. Also, when you want to find the code for a particular procedure it is often useful to click on that procedure in the Data Window and then check the command log. You can then copy the code from the command log into your script file. And, of course, the best way to remember what coding to use is to steal the code from your previously written script files.

```
# GRETLL SCRIPT FOR STATISTICAL COMPUTING ANALYSIS
#
open C:\DIRECTORY PATH\usmondatt.gdt
#
# EXPLORE ALTERNATIVE PRICE LEVEL SERIES
#
lags 4; USIPD USCPI # Generate lags of 1 through 4 periods.
# Lags are denoted by the characters _1, _2, _3, and _4 added
# to the variable names.
#
gener YGCPI = 100*(USCPI - USCPI_4)/USCPI_4 # Generate year-over-year
gener YGIPD = 100*(USIPD-USIPD_4)/USIPD_4 # inflation rates.
```

```

#
summary YYGCPPI    # Provide information about the
summary YYGIPD    # two series.
QCPI05 = quantile(YYGCPPI,0.05) # Calculate various quantiles
QCPI25 = quantile(YYGCPPI,0.25)
QCPI50 = quantile(YYGCPPI,0.50)
QCPI75 = quantile(YYGCPPI,0.75)
QCPI95 = quantile(YYGCPPI,0.95)
#
QIPD05 = quantile(YYGIPD,0.05)
QIPD25 = quantile(YYGIPD,0.25)
QIPD50 = quantile(YYGIPD,0.50)
QIPD75 = quantile(YYGIPD,0.75)
QIPD95 = quantile(YYGIPD,0.95)
#
# RUN DEMAND FOR MONEY REGRESSION
# after putting M1 and GDP in real terms
# and taking the logarithms
#
genr USRM1 = 100*(USM1/USIPD)    # Calculate real values by deflating
genr USRGDP = 100*(USGDP/USIPD) # the series by the GDP deflator.
genr LUSRM1 = log(USRM1)        # Generate logarithms of the
genr LUSRGDP = log(USRGDP)     # two series.
#
ols LUSRM1 const US3MTBR LUSRGDP # Run the regression
RESIDS = $uhat # Extract regression residuals and
FITTED = $yhat # fitted values for subsequent plotting
#
modtest --breusch-pagan    # test for heteroskedasticity
modtest 4 --autocorr      # tests for autocorrelation in residuals
#

```

The last two commands use the function `modtest`, giving it the instruction to first run a Breusch-Pagan test and then do tests for autocorrelation in the residuals using 4 lags, a number chosen here for purposes of illustration. The results are printed on the next three pages.

```

gretl version 1.9.1
Current session: 2011-09-18 10:29
# GRETL SCRIPT FOR STATISTICAL COMPUTING ANALYSIS
#
? open C:\DIRECTORY PATH\usmondatt.gdt

Read datafile C:\DIRECTORY PATH\usmondatt.gdt
periodicity: 4, maxobs: 205
observations range: 1959:1-2010:1

Listing 7 variables:
  0) const      1) USGDP      2) US3MTBR    3) USIPD      4) USCPI
  5) USM1      6) USM2

#
? lags 4; USIPD USCPI # Generate lags of 1 through 4 periods.
Listing 15 variables:
  0) const      1) USGDP      2) US3MTBR    3) USIPD      4) USCPI
  5) USM1      6) USM2      7) USIPD_1    8) USIPD_2    9) USIPD_3
 10) USIPD_4   11) USCPI_1   12) USCPI_2   13) USCPI_3   14) USCPI_4

# Lags are denoted by the characters _1, _2, _3, and _4 added
# to the variable names.
#
? genr YYGCP1 = 100*(USCPI - USCPI_4)/USCPI_4 # Generate year-over-year
Generated series YYGCP1 (ID 15) # inflation rates.
? genr YYGIPD = 100*(USIPD-USIPD_4)/USIPD_4
Generated series YYGIPD (ID 16)
#
? summary YYGCP1 # Provide information about the two series

Summary statistics, using the observations 1959:1 - 2010:1
for the variable 'YYGCP1' (201 valid observations)

Mean                4.1040
Median              3.2998
Minimum             -1.5952
Maximum             14.426
Standard deviation  2.9098
C.V.                0.70901
Skewness            1.4524
Ex. kurtosis        1.9968

```

```
? summary YYGIPD
```

```
Summary statistics, using the observations 1959:1 - 2010:1  
for the variable 'YYGIPD' (201 valid observations)
```

Mean	3.6540
Median	2.9926
Minimum	0.48682
Maximum	11.085
Standard deviation	2.3670
C.V.	0.64778
Skewness	1.2175
Ex. kurtosis	0.77655

```
? QCPI05 = quantile(YYGCPI,0.05) # Calculate various quantiles
```

```
Generated scalar QCPI05 = 1.20246
```

```
? QCPI25 = quantile(YYGCPI,0.25)
```

```
Generated scalar QCPI25 = 2.23928
```

```
? QCPI50 = quantile(YYGCPI,0.50)
```

```
Generated scalar QCPI50 = 3.29976
```

```
? QCPI75 = quantile(YYGCPI,0.75)
```

```
Generated scalar QCPI75 = 5.17358
```

```
? QCPI95 = quantile(YYGCPI,0.95)
```

```
Generated scalar QCPI95 = 11.1064
```

```
#
```

```
? QIPD05 = quantile(YYGIPD,0.05)
```

```
Generated scalar QIPD05 = 1.11431
```

```
? QIPD25 = quantile(YYGIPD,0.25)
```

```
Generated scalar QIPD25 = 1.97793
```

```
? QIPD50 = quantile(YYGIPD,0.50)
```

```
Generated scalar QIPD50 = 2.9926
```

```
? QIPD75 = quantile(YYGIPD,0.75)
```

```
Generated scalar QIPD75 = 4.73358
```

```
? QIPD95 = quantile(YYGIPD,0.95)
```

```
Generated scalar QIPD95 = 8.93689
```

```
#
```

```
# RUN DEMAND FOR MONEY REGRESSION
```

```
# after putting M1 and GDP in real terms
```

```
# and taking the logarithms
```

```
#
```

```
? genr USRM1 = 100*(USM1/USIPD) # Calculate real values by deflating
```

```
Generated series USRM1 (ID 17)
```

```
? genr USRGDP = 100*(USGDP/USIPD) # the series by the GDP deflator.
```

```
Generated series USRGDP (ID 18)
```

```
? genr LUSRM1 = log(USRM1)          # Generate logarithms of the
Generated series LUSRM1 (ID 19)
? genr LUSRGDP = log(USRGDP)        # two series.
Generated series LUSRGDP (ID 20)
#
? ols LUSRM1 const US3MTBR LUSRGDP # Run the regression
```

Model 1: OLS, using observations 1959:1-2010:1 (T = 205)
 Dependent variable: LUSRM1

	coefficient	std. error	t-ratio	p-value	
const	3.34776	0.0930368	35.98	8.49e-090	***
US3MTBR	-0.0194151	0.00172099	-11.28	3.31e-023	***
LUSRGDP	0.419451	0.0103621	40.48	7.14e-099	***

```
Mean dependent var    6.931613    S.D. dependent var    0.222799
Sum squared resid     0.947369    S.E. of regression    0.068483
R-squared              0.906446    Adjusted R-squared    0.905520
F(2, 202)             978.5879    P-value(F)            1.2e-104
Log-likelihood         260.2679    Akaike criterion      -514.5359
Schwarz criterion     -504.5668    Hannan-Quinn          -510.5036
rho                   0.965846    Durbin-Watson         0.067818
```

Log-likelihood for USRM1 = -1160.71

```
? RESIDS = $uhat
Generated series RESIDS (ID 21)
? FITTED = $yhat
Generated series FITTED (ID 22)
#
? modtest --breusch-pagan # test for heteroskedasticity
```

Breusch-Pagan test for heteroskedasticity
 OLS, using observations 1959:1-2010:1 (T = 205)
 Dependent variable: scaled uhat^2

	coefficient	std. error	t-ratio	p-value	
const	-7.19539	2.19599	-3.277	0.0012	***
US3MTBR	-0.00569001	0.0406214	-0.1401	0.8887	
LUSRGDP	0.935988	0.244581	3.827	0.0002	***

Explained sum of squares = 39.7335

Test statistic: LM = 19.866746,
with p-value = P(Chi-square(2) > 19.866746) = 0.000049

? modtest 4 --autocorr # tests for autocorrelation in residuals

Breusch-Godfrey test for autocorrelation up to order 4
OLS, using observations 1959:1-2010:1 (T = 205)
Dependent variable: uhat

	coefficient	std. error	t-ratio	p-value	
const	0.00429298	0.0224893	0.1909	0.8488	
US3MTBR	-0.000431786	0.000416595	-1.036	0.3012	
LUSRGDP	-0.000220610	0.00250467	-0.08808	0.9299	
uhat_1	1.25031	0.0679086	18.41	9.02e-045	***
uhat_2	-0.440575	0.107538	-4.097	6.10e-05	***
uhat_3	0.429152	0.107642	3.987	9.41e-05	***
uhat_4	-0.289821	0.0680998	-4.256	3.21e-05	***

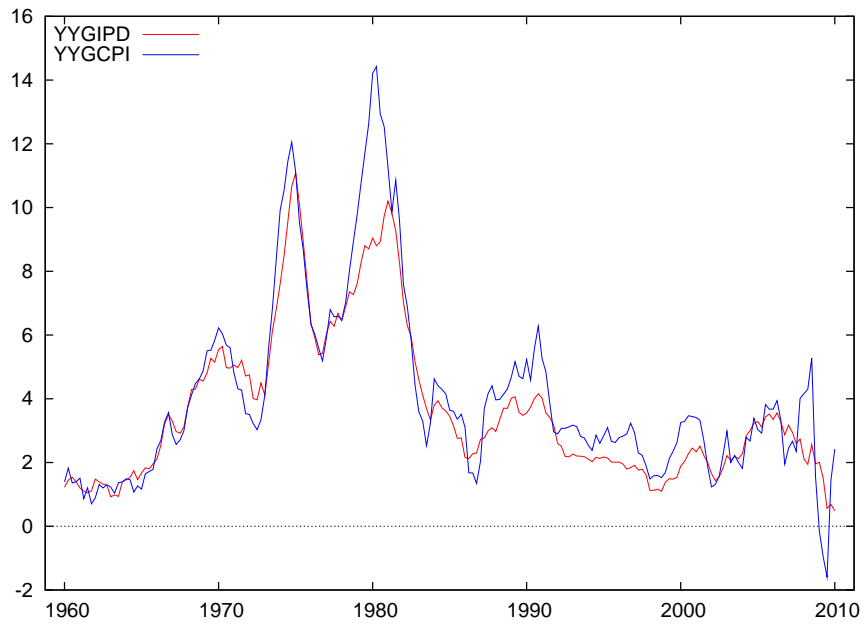
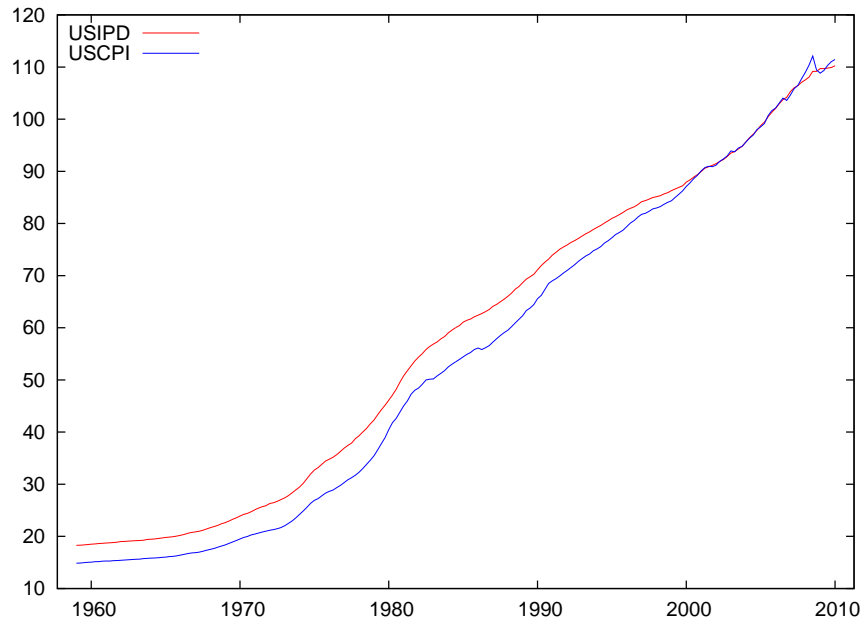
Unadjusted R-squared = 0.942736

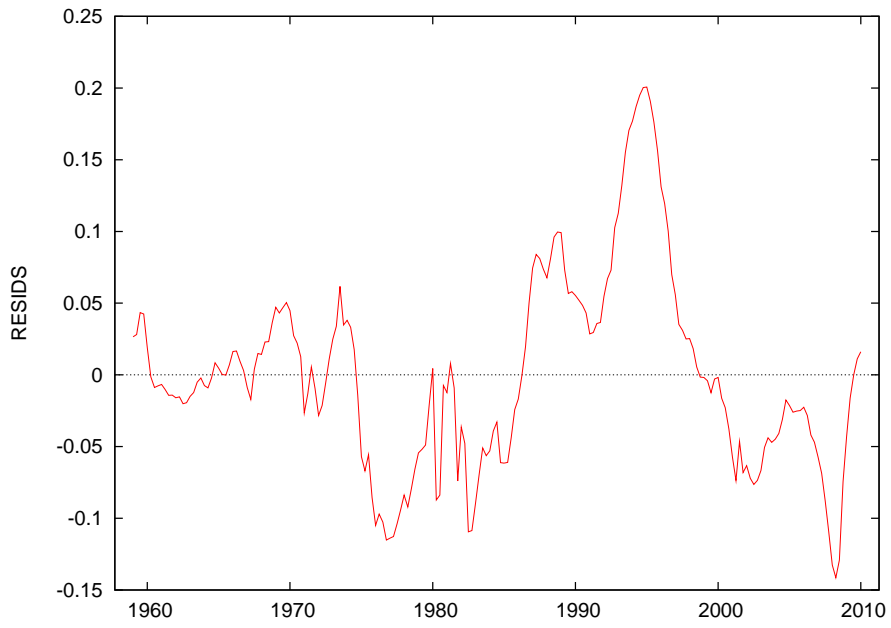
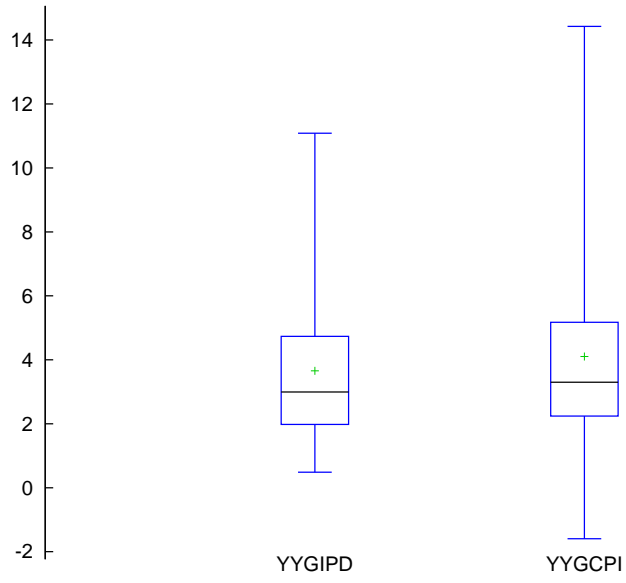
Test statistic: LMF = 814.914971,
with p-value = P(F(4,198) > 814.915) = 1.01e-121

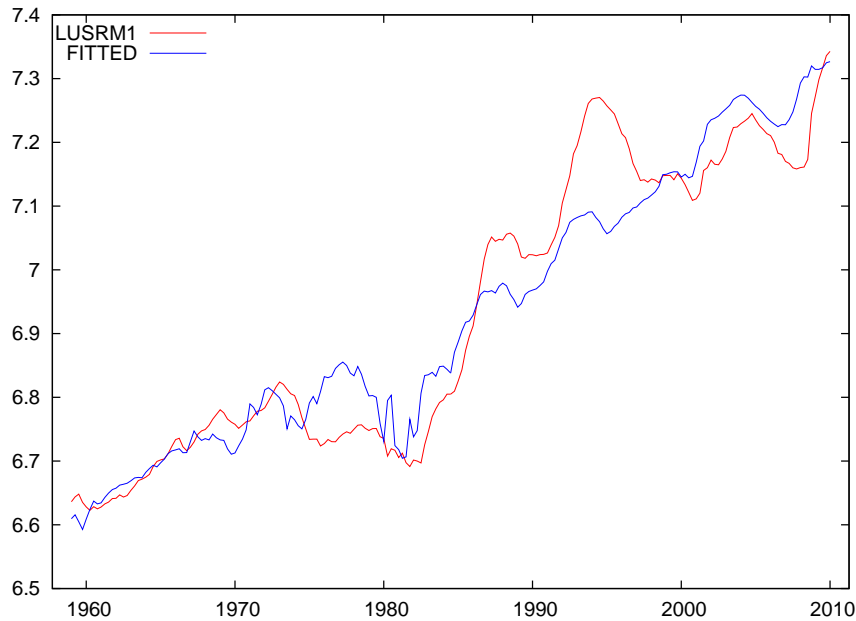
Alternative statistic: TR² = 193.260847,
with p-value = P(Chi-square(4) > 193.261) = 1.06e-040

Ljung-Box Q' = 664.583,
with p-value = P(Chi-square(4) > 664.583) = 1.62e-142

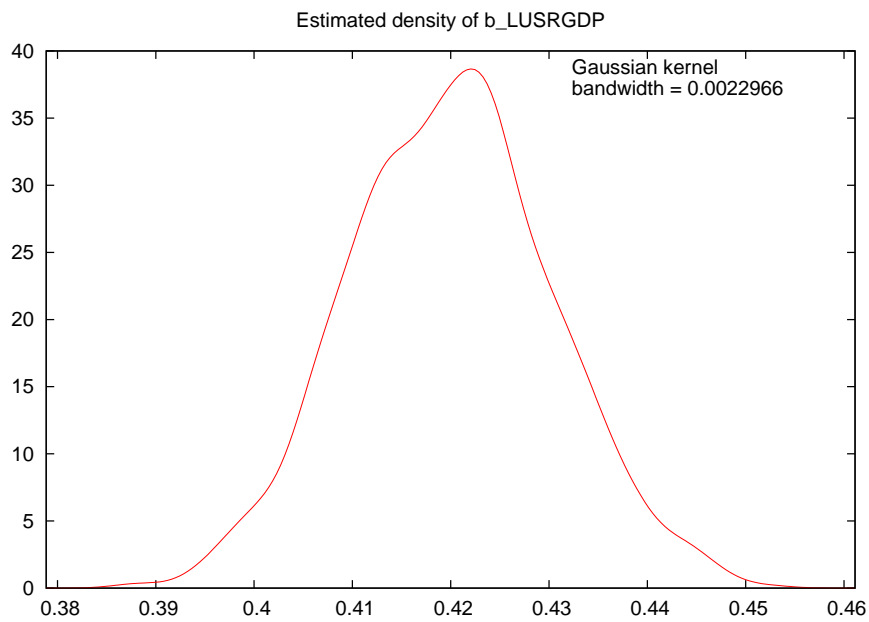
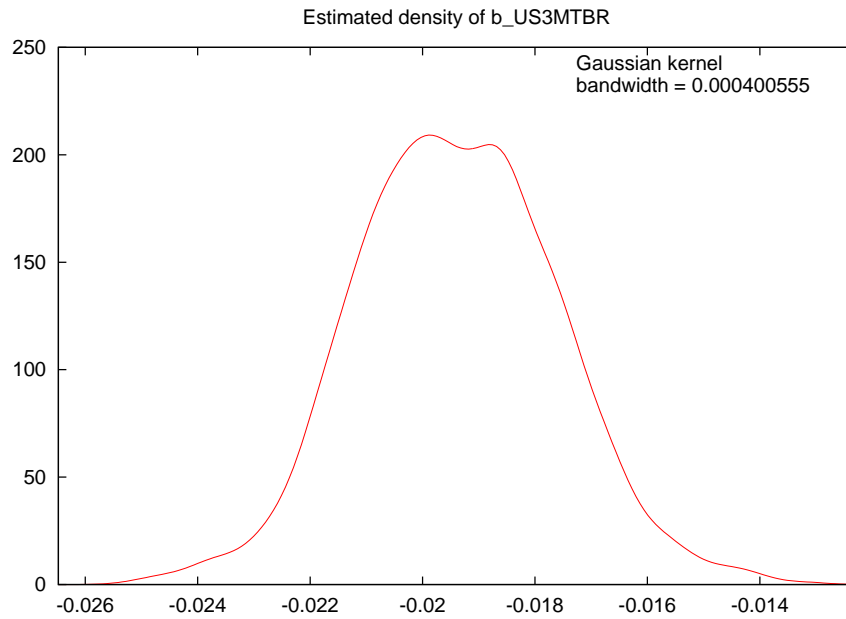
You will notice that there are no commands in our script file that deal with the plotting of particular variables of interest. This is because the easiest way to create plots is to do so from the data-file page after the script file has been executed. All series produced by the execution of the script will appear among the variables in the modified data file. To plot these, we just click on **View**, then on **Graph specific vars** and then choose the type of graph and the series to be included by following the prompts. When a graph appears, we can add a title, change the labels, and so forth by holding down the right-mouse button and choosing **Edit**. And we can save each of the plots by clicking on one of the **Save** commands. The five plots of special interest are presented below without editing the raw graphs.







In cases where your sample size is small it is desirable to produce bootstrapped values for the regression coefficients to provide a range over which the regression coefficients might possibly lie. To do this, first run the script, and then run the main regression again by clicking in the Data Window on **Model**, then **Ordinary Least Squares**, and selecting the appropriate variables in the window that results. After you click on **OK** to run the regression, a window will appear containing the regression results alone. Click on **Analysis** along the top of that window and then on **Bootstrap** and select the options **Confidence interval**, **Resample residuals** and **Save bootstrap data to file**. Then, after selecting the coefficient you want bootstrapped estimates of, click on **OK**. A window will appear with print delineating a 95% confidence interval for the coefficient, on top of which will be pasted another window asking you for the file name in which to save the bootstrap results. After choosing a directory and file name for the bootstrap results for that coefficient, leaving off the suffix because Gretl will add `.gdt`, the top window will disappear and you can make note of or cut and paste the confidence interval material somewhere. Now go back to the regression model window and proceed to obtain bootstrap results for the coefficient of the next variable, following the same procedure as in the case of the first coefficient and saving its bootstrap results in a different file to which Gretl will again attach the suffix `.gdt`. Repeat this process for all remaining coefficients you want obtain bootstrap estimates of. The final step is to restart **Gretl** and access one of the bootstrap coefficient files. In the Data Window that appears, select **File** and then **Append data** and sequentially append the other bootstrap coefficient files to the first one. Then save all the bootstrapped coefficients in the resulting file in standard format under a more general name—here, we can use `bootres.gdt`. The individual coefficient files can then be deleted, and you can load `bootres.gdt` and save summary statistics and plot and save kernel density estimates of the bootstrapped coefficients. Plots of the kernel density estimates of the bootstrapped treasury bill and log real GDP coefficients in our demand for money regression are presented below.



Our regression results above reveal substantial heteroskedasticity and serial correlation in the regression residuals. This raises the possibility that the coefficient standard errors indicated by our regression may be distorted sufficiently to make the variables appear statistically significant when in truth they are not. Procedures have therefore been developed to **adjust the coefficient standard errors for the presence of both heteroskedasticity and autocorrelation in the regression residuals** so that our conclusions as to the statistical significance

of the included independent variables based on those new coefficient standard errors will be more likely to be correct. To run regressions with heterkedasticity-and-autocorrelation-adjusted (HAC) coefficient standard errors using Gretl interactively we simply highlight the term **robust standard errors** near the bottom of the **specify model** window. To do this in a script, we add the code `--robust` at the end of the line specifying the regression to be run.

```
ols LUSRM1 const US3MTBR LUSRGDP --robust
```

The adjustment made by Gretl is based on the White and Newey-West procedures.

Model 2: OLS, using observations 1959:1-2010:1 (T = 205)

Dependent variable: LUSRM1

HAC standard errors, bandwidth 4 (Bartlett kernel)

	coefficient	std. error	t-ratio	p-value
const	3.34776	0.133043	25.16	3.54e-064 ***
US3MTBR	-0.0194151	0.00248172	-7.823	2.82e-013 ***
LUSRGDP	0.419451	0.0162121	25.87	4.83e-066 ***
Mean dependent var	6.931613	S.D. dependent var	0.222799	
Sum squared resid	0.947369	S.E. of regression	0.068483	
R-squared	0.906446	Adjusted R-squared	0.905520	
F(2, 202)	334.7330	P-value(F)	7.50e-65	
Log-likelihood	260.2679	Akaike criterion	-514.5359	
Schwarz criterion	-504.5668	Hannan-Quinn	-510.5036	
rho	0.965846	Durbin-Watson	0.067818	

Log-likelihood for USRM1 = -1160.71

Another important issue that must be dealt with is the possibility that our **regression may be spurious**. For this to happen, all the variables in the regression must be non-stationary and, in addition, they must not be cointegrated. The first task is therefore to determine whether the time-series processes that can reasonably describe the evolution of our time-series variables are stationary. This involves running **Dickey-Fuller tests**. This is done by highlighting each variable in turn and then clicking on **variable** in the Data Window and then scrolling down to and clicking on **Augmented Dickey-Fuller test**. A window will appear in which we give Gretl the appropriate instructions. We should highlight the following options

```
test without constant
test with constant
test with constant and trend
test down from maximum lag order
use level of variable
```

and then, acting as if we did not know how many lags to use, set the lag-order at a large number—say, 10. Alternatively, we can simply add the following code lines to our script.

```
adf 10 LUSRGDP --nc --c --ct --test-down
adf 10 LUSRM1 --nc --c --ct --test-down
adf 10 US3MTBR --nc --c --ct --test-down
```

This procedure produces the the following results.

```
? adf 10 LUSRGDP --nc --c --ct --test-down
```

```
Augmented Dickey-Fuller test for LUSRGDP
including 2 lags of (1-L)LUSRGDP
sample size 202
unit-root null hypothesis: a = 1
```

```
test without constant
model: (1-L)y = (a-1)*y(-1) + ... + e
1st-order autocorrelation coeff. for e: -0.000
lagged differences: F(2, 199) = 15.241 [0.0000]
estimated value of (a - 1): 0.000486135
test statistic: tau_nc(1) = 5.05466
asymptotic p-value 1
```

```
test with constant
model: (1-L)y = b0 + (a-1)*y(-1) + ... + e
1st-order autocorrelation coeff. for e: -0.001
lagged differences: F(2, 198) = 11.982 [0.0000]
estimated value of (a - 1): -0.00219991
test statistic: tau_c(1) = -1.73143
asymptotic p-value 0.4153
```

```
with constant and trend
model: (1-L)y = b0 + b1*t + (a-1)*y(-1) + ... + e
1st-order autocorrelation coeff. for e: -0.012
lagged differences: F(2, 197) = 14.461 [0.0000]
estimated value of (a - 1): -0.0396875
test statistic: tau_ct(1) = -2.47903
asymptotic p-value 0.3387
```

```
? adf 10 LUSRM1 --nc --c --ct --test-down
```

```
Augmented Dickey-Fuller test for LUSRM1
including 8 lags of (1-L)LUSRM1
sample size 196
unit-root null hypothesis: a = 1
```

```
test without constant
model: (1-L)y = (a-1)*y(-1) + ... + e
```

1st-order autocorrelation coeff. for e: -0.009
lagged differences: F(8, 187) = 18.700 [0.0000]
estimated value of (a - 1): 0.000199843
test statistic: tau_nc(1) = 1.85655
asymptotic p-value 0.9853

test with constant
model: $(1-L)y = b_0 + (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: -0.009
lagged differences: F(8, 186) = 18.659 [0.0000]
estimated value of (a - 1): -0.00163501
test statistic: tau_c(1) = -0.499691
asymptotic p-value 0.889

with constant and trend
model: $(1-L)y = b_0 + b_1*t + (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: -0.011
lagged differences: F(7, 187) = 21.674 [0.0000]
estimated value of (a - 1): -0.0223141
test statistic: tau_ct(1) = -2.46265
asymptotic p-value 0.347

? adf 10 US3MTBR --nc --c --ct --test-down

Augmented Dickey-Fuller test for US3MTBR
including 7 lags of (1-L)US3MTBR
sample size 197
unit-root null hypothesis: a = 1

test without constant
model: $(1-L)y = (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: 0.015
lagged differences: F(7, 189) = 10.649 [0.0000]
estimated value of (a - 1): -0.00722627
test statistic: tau_nc(1) = -0.906071
asymptotic p-value 0.3239

test with constant
model: $(1-L)y = b_0 + (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: 0.017
lagged differences: F(7, 188) = 10.797 [0.0000]
estimated value of (a - 1): -0.0363103
test statistic: tau_c(1) = -1.89242
asymptotic p-value 0.3362

```

with constant and trend
model: (1-L)y = b0 + b1*t + (a-1)*y(-1) + ... + e
1st-order autocorrelation coeff. for e: 0.016
lagged differences: F(7, 187) = 10.699 [0.0000]
estimated value of (a - 1): -0.0421692
test statistic: tau_ct(1) = -2.17285
asymptotic p-value 0.5043

```

As indicated by the P-values, the hypothesis of non-stationarity can be rejected in all three cases.

In order to pick the best number of lagged differences ourselves we can have Gretl **estimate the autocorrelations and partial-autocorrelations** by clicking on **variable** and then **correlogram**. As an example, we do this with our LUSRM1 variable. First we must set up the first-differences of our variable by clicking on **add** and then on **Define new variable**, inserting the code line

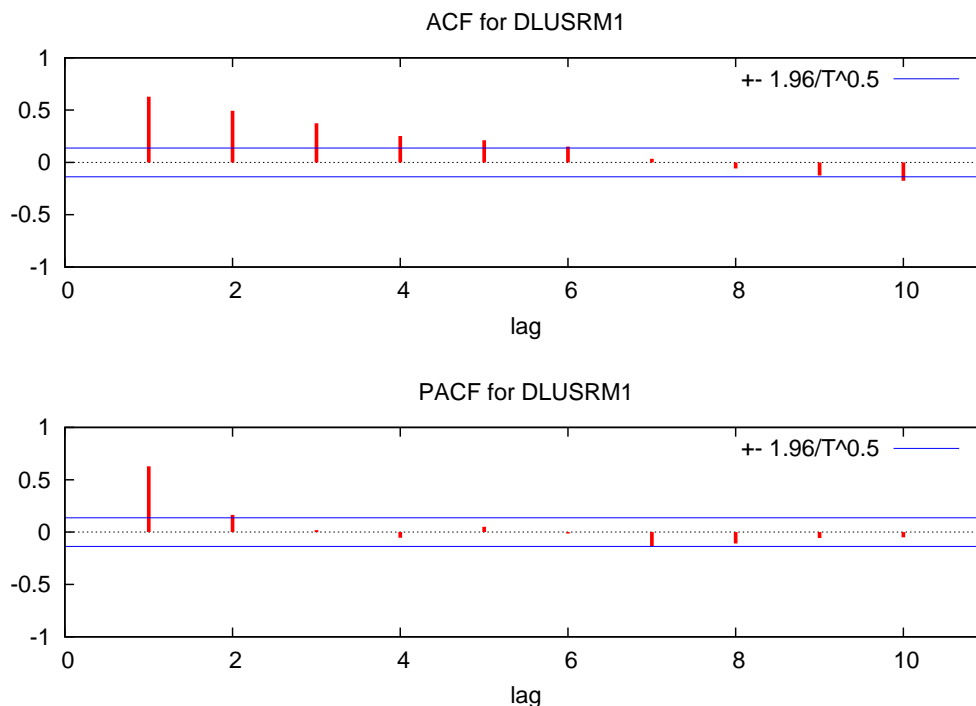
```
DLUSRM1 = LUSRM1 - LUSRM1_1
```

And prior to this, we must have first created the one-period lag of LUSRM1, which appears in the right-most position in the code-line above, by highlighting the variable in the Data Window and then clicking on **add** and **Lags of selected variable**, and setting the lag at 1. The procedure that creates the correlogram also requires that the selected variable be highlighted when the command is given. The result of all this is a window containing the following autocorrelation and partial autocorrelation functions (i.e., the correlogram),

Autocorrelation function for DLUSRM1

LAG	ACF		PACF		Q-stat.	[p-value]
1	0.6279	***	0.6279	***	81.6187	[0.000]
2	0.4943	***	0.1651	**	132.4395	[0.000]
3	0.3741	***	0.0193		161.7020	[0.000]
4	0.2521	***	-0.0535		175.0592	[0.000]
5	0.2125	***	0.0511		184.5913	[0.000]
6	0.1511	**	-0.0135		189.4391	[0.000]
7	0.0356		-0.1346	*	189.7091	[0.000]
8	-0.0573		-0.1089		190.4125	[0.000]
9	-0.1247	*	-0.0574		193.7612	[0.000]
10	-0.1750	**	-0.0496		200.3947	[0.000]

together with the figure on the next page.



The essence of the above results can also be obtained by adding the following lines to our input script file.

```
lags 1; LUSRM1
DLUSRM1 = LUSRM1 - LUSRM1_1
corrgm DLUSRM1 12
```

Using the script approach, however, produces only a crude text graph containing the autocorrelations only and no delineation of the critical values. The Q-statistics in the results above test whether the current and previous autocorrelations are zero—if that statistic is larger than some critical value we can reject the null hypothesis of no significant autocorrelation. The P-Values above clearly indicate that the series is not white-noise. The suggested optimum number of lags, based on the autocorrelation function, is six—this is two less than the eight lags selected by Gretl. As a matter of interest, we can try six lags below using the code line

```
adf 6 LUSRM1 --nc --c --ct
```

which produces the a result below that is pretty much the same as that produced with the 8 lags selected by Gretl.

Augmented Dickey-Fuller test for LUSRM1
including 6 lags of (1-L)LUSRM1
sample size 198
unit-root null hypothesis: $a = 1$

test without constant
model: $(1-L)y = (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: 0.004
lagged differences: $F(6, 191) = 22.901 [0.0000]$
estimated value of $(a - 1)$: 0.000156681
test statistic: $\tau_{nc}(1) = 1.45716$
asymptotic p-value 0.9645

test with constant
model: $(1-L)y = b_0 + (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: 0.004
lagged differences: $F(6, 190) = 22.895 [0.0000]$
estimated value of $(a - 1)$: -0.00210785
test statistic: $\tau_c(1) = -0.643705$
asymptotic p-value 0.8585

with constant and trend
model: $(1-L)y = b_0 + b_1*t + (a-1)y(-1) + \dots + e$
1st-order autocorrelation coeff. for e: 0.009
lagged differences: $F(6, 189) = 24.644 [0.0000]$
estimated value of $(a - 1)$: -0.025681
test statistic: $\tau_{ct}(1) = -2.88989$
asymptotic p-value 0.1657

Since it is clear that all three of the variables in our regression are non-stationary it is necessary to perform a **Johansen Cointegration Test**. To do this we click in the data window on **Model**, then on **Time series**, then on **Cointegration test** and then on **Johansen**. At that point we have to choose whether or not we will restrict the constant and or trend in the test procedure and we have to choose the number of lags. An alternative is to perform the test via our script in which case we can run the test with several alternative lags and restriction options. It turns out that when we run the test with a restricted constant and 6 lags we obtain the result below. The coding for all the options is presented with all the codes except that for the restricted constant case commented out.

```
#coint2 6 LUSRM1 LUSRGDP US3MTBR --nc #no constant
coint2 6 LUSRM1 LUSRGDP US3MTBR --rc #restricted constant
#coint2 6 LUSRM1 LUSRGDP US3MTBR #unrestricted constant
#coint2 6 LUSRM1 LUSRGDP US3MTBR --crt #constant and restricted trend
#coint2 6 LUSRM1 LUSRGDP US3MTBR --ct #constant and unrestricted trend
```

? coint2 6 LUSRM1 LUSRGDP US3MTBR --rc #restricted constant

Johansen test:

Number of equations = 3

Lag order = 6

Estimation period: 1960:3 - 2010:1 (T = 199)

Case 2: Restricted constant

Rank	Eigenvalue	Trace test	p-value	Lmax test	p-value
0	0.15638	49.381	[0.0006]	33.842	[0.0004]
1	0.041198	15.540	[0.2009]	8.3720	[0.5153]
2	0.035377	7.1676	[0.1207]	7.1676	[0.1206]

eigenvalue	0.15638	0.041198	0.035377
------------	---------	----------	----------

beta (cointegrating vectors)

LUSRM1	4.0720	14.919	-2.7323
LUSRGDP	-2.4488	-5.9701	-0.30060
US3MTBR	0.098356	0.20665	-0.41052
const	-5.3596	-52.438	24.157

alpha (adjustment vectors)

LUSRM1	-0.00040147	-0.0016629	9.0419e-005
LUSRGDP	0.0022339	-0.00014355	0.00095356
US3MTBR	-0.12617	0.024650	0.10178

renormalized beta

LUSRM1	1.0000	-2.4989	6.6557
LUSRGDP	-0.60138	1.0000	0.73225
US3MTBR	0.024154	-0.034614	1.0000
const	-1.3162	8.7835	-58.845

renormalized alpha

LUSRM1	-0.0016348	0.0099279	-3.7119e-005
LUSRGDP	0.0090966	0.00085702	-0.00039145
US3MTBR	-0.51377	-0.14716	-0.041781

long-run matrix (alpha * beta')

	LUSRM1	LUSRGDP	US3MTBR	const
LUSRM1	-0.026691	0.010884	-0.00042025	0.091538
LUSRGDP	0.0043496	-0.0049001	-0.00020140	0.018590
US3MTBR	-0.42410	0.13121	-0.049097	1.8422

The null-hypothesis of no cointegrating vectors can be rejected at better than the 1% level and there is a single cointegrating vector which, as can be seen from the left-most column under **renormalized beta**, represents an equation with the coefficient signs obtained in our regression. That regression is therefore not spurious.

Another frequently encountered problem with regression analysis is **simultaneity bias**. As an example, consider the estimation of demand and supply curves, where the quantities demanded and supplied are expressed as functions of the price. The problem is that the error terms in the demand and supply curves are correlated with the price variables in the respective equations, biasing our estimates. The solution we will consider here is to find variables, called **instrumental variables**, that are correlated with the endogenous independent variable (here, the price) but not with the error terms in the respective equations. The fundamental approach, **two-stage least squares**, proceeds in two stages. In the first stage we regress the price variable on the instrument or instruments for that equation as well as the other exogenous independent variables and save the fitted values. These fitted values of the price variable are uncorrelated with the error term in this first-stage regression due to the properties of regression analysis. Then we run a second stage regression of the quantity on the fitted values of the price variable, cleansed of correlation with the errors by the first-stage regression, plus the other independent variables affecting the demand or the supply according to which of the two equations we are estimating. The resulting coefficient of the fitted value of the price level will, if the instruments are good ones, be an unbiased measure of the true coefficient of the price variable in the underlying demand or supply curve being estimated.

Let us proceed using an example presented by G. S. Maddala in his introductory econometrics textbook.¹ He estimates the demand and supply curves of commercial banks' loans to business firms in the United States using monthly data for the years 1979 through 1984. The variables are in the Excel worksheet file `loandata.xls` and the Gretl data file `loandata.gdt`. These variables are described and named for estimation purposes as follows.

QLOANS	—Quantity of Commercial Loans Made by Banks
PRIMRATE	—Bank's Prime Rate on Commercial Loans
CBNDRATE	—Interest Rate on Corporate Bonds
TBRATE	—30-Day Treasury Bill Rate
INDPROD	—Industrial Production
TOTDEP	—Total Bank Deposits

Appropriate structural equations representing the demand and supply of commercial bank loans are:

Demand

$$\text{QLOANS} = \beta_0 + \beta_1 \text{PRIMRATE} + \beta_2 \text{CBNDRATE} + \beta_3 \text{INDPROD}$$

Supply

$$\text{QLOANS} = \delta_0 + \delta_1 \text{PRIMRATE} + \delta_2 \text{TBRATE} + \delta_3 \text{TOTDEP}$$

where β_1 and δ_2 are negative and all the other coefficients are positive. This says that banks will expand their supply of loans in response to a higher prime rate, greater deposits, and a lower rate of return on alternative investments in treasury bills, and that commercial enterprises will increase their demand for loans in response to a fall in the prime rate, a rise in the cost of funding through corporate bond issues and an increase in their output. Maddala notes that it

¹G. S. Maddala, *Introduction to Econometrics*, Macmillan, 1988, pages 313-317.

would seem reasonable to expect that the two exogenous variables in the demand equation will be independent of the error term in the supply equation since they would be unlikely to shift the supply curve and the two exogenous variables in the supply equation will be independent of the error term in the demand equation in that they are unlikely to shift the demand curve. We thus use the exogenous variables in the demand equation as instruments for the supply equation, and the exogenous variables in the supply equation as instruments for the demand equation.

We construct and run the following script file. In applying Gretl's two-stage-least-squares `tsls` function, the constant and independent variables in the final regression are entered first, followed by a semi-colon, and the constant and independent variables are then listed again followed by the instrumental variables with the code `--robust` added if desired. In the script we first run the demand and supply curve regressions by OLS without the instruments. And at the very bottom we run the first-stage regression, which turns out to be the same for both curves. Alternatively, we could run each two-stage-least-squares procedure by loading the data into Gretl and then clicking on **Modules**, then on **Instrumental variables** and then on **Two-Stage Least Squares** and follow the instructions in the window that emerges.

```
open "C:\DIRECTORY PATH\loandata.gdt"
#
# DEMAND CURVE
# *****
#
ols QLOANS const PRIMRATE CBNDRATE INDPROD --robust
#
tsls QLOANS const PRIMRATE CBNDRATE INDPROD ; const CBNDRATE INDPROD TBRATE TOTDEP
tsls QLOANS const PRIMRATE CBNDRATE INDPROD ; const CBNDRATE INDPROD TBRATE \
TOTDEP --robust
#
# SUPPLY CURVE
# *****
#
ols QLOANS const PRIMRATE TBRATE TOTDEP --robust
#
tsls QLOANS const PRIMRATE TBRATE TOTDEP ; const TBRATE TOTDEP CBNDRATE INDPROD
tsls QLOANS const PRIMRATE TBRATE TOTDEP ; const TBRATE TOTDEP CBNDRATE \
INDPROD --robust
#
# FIRST-STAGE REGRESSIONS
#
ols PRIMRATE const TBRATE TOTDEP CBNDRATE INDPROD
```

The results are as follows.

```

gretl version 1.9.1
Current session: 2011-09-19 16:23
? open "C:\DIRECTORY PATH\loandata.gdt"

```

```

Read datafile E:\DSLMHTML\MANUALS\loandata.gdt
periodicity: 12, maxobs: 72
observations range: 1979:01-1984:12

```

Listing 7 variables:

```

0) const      1) QLOANS      2) PRIMRATE      3) CBNDRATE      4) INDPROD
5) TBRATE     6) TOTDEP

```

```

#
# DEMAND CURVE
# *****
#
? ols QLOANS const PRIMRATE CBNDRATE INDPROD --robust

```

```

Model 1: OLS, using observations 1979:01-1984:12 (T = 72)
Dependent variable: QLOANS
HAC standard errors, bandwidth 3 (Bartlett kernel)

```

	coefficient	std. error	t-ratio	p-value
const	-214.062	129.874	-1.648	0.1039
PRIMRATE	-16.5616	1.86620	-8.875	5.69e-013 ***
CBNDRATE	36.6761	3.83334	9.568	3.22e-014 ***
INDPROD	2.36876	0.786843	3.010	0.0037 ***

```

Mean dependent var    361.5556    S.D. dependent var    62.31587
Sum squared resid    63305.98    S.E. of regression    30.51181
R-squared              0.770391    Adjusted R-squared    0.760261
F(3, 68)              45.27016    P-value(F)            3.38e-16
Log-likelihood         -346.2101    Akaike criterion      700.4201
Schwarz criterion     709.5268    Hannan-Quinn          704.0455
rho                   0.664409    Durbin-Watson         0.658519

```

```
? tsls QLOANS const PRIMRATE CBNDRATE INDPROD ; const CBNDRATE INDPROD TBRATE \
TOTDEP
```

Model 2: TSLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: QLOANS

Instrumented: PRIMRATE

Instruments: const CBNDRATE INDPROD TBRATE TOTDEP

	coefficient	std. error	z	p-value	
const	-220.874	77.9247	-2.834	0.0046	***
PRIMRATE	-20.8050	1.67702	-12.41	2.43e-035	***
CBNDRATE	41.4223	2.97754	13.91	5.39e-044	***
INDPROD	2.42085	0.474417	5.103	3.35e-07	***

Mean dependent var	361.5556	S.D. dependent var	62.31587
Sum squared resid	71759.49	S.E. of regression	32.48517
R-squared	0.761451	Adjusted R-squared	0.750927
F(3, 68)	77.72069	P-value(F)	6.24e-22
rho	0.626987	Durbin-Watson	0.735020

Hausman test -

Null hypothesis: OLS estimates are consistent

Asymptotic test statistic: Chi-square(1) = 81.7343

with p-value = 1.55665e-019

Sargan over-identification test -

Null hypothesis: all instruments are valid

Test statistic: LM = 19.6841

with p-value = P(Chi-Square(1) > 19.6841) = 9.13602e-006

Weak instrument test -

First-stage F-statistic (2, 67) = 133.379

Critical values for desired TSLS maximal size, when running tests at a nominal 5% significance level:

size	10%	15%	20%	25%
value	19.93	11.59	8.75	7.25

Maximal size is probably less than 10%

```
? tsls QLOANS const PRIMRATE CBNDRATE INDPROD ; const CBNDRATE INDPROD TBRATE \
TOTDEP --robust
```

Model 3: TSLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: QLOANS

Instrumented: PRIMRATE

Instruments: const CBNDRATE INDPROD TBRATE TOTDEP

HAC standard errors, bandwidth 3 (Bartlett kernel)

	coefficient	std. error	z	p-value	
const	-220.874	111.602	-1.979	0.0478	**
PRIMRATE	-20.8050	2.28982	-9.086	1.03e-019	***
CBNDRATE	41.4223	3.80076	10.90	1.17e-027	***
INDPROD	2.42085	0.674790	3.588	0.0003	***

Mean dependent var	361.5556	S.D. dependent var	62.31587
Sum squared resid	71759.49	S.E. of regression	32.48517
R-squared	0.761451	Adjusted R-squared	0.750927
F(3, 68)	51.46455	P-value(F)	1.78e-17
rho	0.626987	Durbin-Watson	0.735020

Hausman test -

Null hypothesis: OLS estimates are consistent

Asymptotic test statistic: Chi-square(1) = 81.7343

with p-value = 1.55665e-019

Sargan over-identification test -

Null hypothesis: all instruments are valid

Test statistic: LM = 19.6841

with p-value = P(Chi-Square(1) > 19.6841) = 9.13602e-006

Weak instrument test -

First-stage F-statistic (2, 67) = 89.2323

A value < 10 may indicate weak instruments

```

#
# SUPPLY CURVE
# *****
#
? ols QLOANS const PRIMRATE TBRATE TOTDEP --robust

```

Model 4: OLS, using observations 1979:01-1984:12 (T = 72)
Dependent variable: QLOANS
HAC standard errors, bandwidth 3 (Bartlett kernel)

	coefficient	std. error	t-ratio	p-value	
const	-78.9066	13.1195	-6.014	8.02e-08	***
PRIMRATE	2.05005	1.17351	1.747	0.0852	*
TBRATE	-1.78460	1.50184	-1.188	0.2389	
TOTDEP	0.336810	0.00766580	43.94	1.18e-051	***
Mean dependent var	361.5556	S.D. dependent var	62.31587		
Sum squared resid	11833.13	S.E. of regression	13.19154		
R-squared	0.957082	Adjusted R-squared	0.955188		
F(3, 68)	656.8475	P-value(F)	4.02e-50		
Log-likelihood	-285.8353	Akaike criterion	579.6706		
Schwarz criterion	588.7773	Hannan-Quinn	583.2960		
rho	0.135962	Durbin-Watson	1.716490		

Excluding the constant, p-value was highest for variable 5 (TBRATE)

```
#
? tsls QLOANS const PRIMRATE TBRATE TOTDEP ; const TBRATE TOTDEP CBNDRATE \
  INDPROD
```

Model 5: TSLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: QLOANS

Instrumented: PRIMRATE

Instruments: const TBRATE TOTDEP CBNDRATE INDPROD

	coefficient	std. error	z	p-value	
const	-89.6899	18.1217	-4.949	7.45e-07	***
PRIMRATE	6.63045	2.46133	2.694	0.0071	***
TBRATE	-7.08268	2.94262	-2.407	0.0161	**
TOTDEP	0.339387	0.0100972	33.61	1.13e-247	***

Mean dependent var	361.5556	S.D. dependent var	62.31587
Sum squared resid	14551.46	S.E. of regression	14.62847
R-squared	0.947431	Adjusted R-squared	0.945112
F(3, 68)	412.6124	P-value(F)	1.50e-43
rho	0.209809	Durbin-Watson	1.575911

Hausman test -

Null hypothesis: OLS estimates are consistent

Asymptotic test statistic: Chi-square(1) = 6.78328

with p-value = 0.00920157

Sargan over-identification test -

Null hypothesis: all instruments are valid

Test statistic: LM = 3.87846

with p-value = P(Chi-Square(1) > 3.87846) = 0.0489094

Weak instrument test -

First-stage F-statistic (2, 67) = 12.5559

Critical values for desired TSLS maximal size, when running tests at a nominal 5% significance level:

size	10%	15%	20%	25%
value	19.93	11.59	8.75	7.25

Maximal size may exceed 10%

```
? tsls QLOANS const PRIMRATE TBRATE TOTDEP ; const TBRATE TOTDEP CBNDRATE \
  INDPROD --robust
```

Model 6: TSLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: QLOANS

Instrumented: PRIMRATE

Instruments: const TBRATE TOTDEP CBNDRATE INDPROD

HAC standard errors, bandwidth 3 (Bartlett kernel)

	coefficient	std. error	z	p-value	
const	-89.6899	18.7285	-4.789	1.68e-06	***
PRIMRATE	6.63045	2.36423	2.804	0.0050	***
TBRATE	-7.08268	3.28435	-2.156	0.0310	**
TOTDEP	0.339387	0.0104725	32.41	2.14e-230	***
Mean dependent var	361.5556	S.D. dependent var	62.31587		
Sum squared resid	14551.46	S.E. of regression	14.62847		
R-squared	0.947431	Adjusted R-squared	0.945112		
F(3, 68)	374.2501	P-value(F)	3.45e-42		
rho	0.209809	Durbin-Watson	1.575911		

Hausman test -

Null hypothesis: OLS estimates are consistent

Asymptotic test statistic: Chi-square(1) = 6.78328

with p-value = 0.00920157

Sargan over-identification test -

Null hypothesis: all instruments are valid

Test statistic: LM = 3.87846

with p-value = P(Chi-Square(1) > 3.87846) = 0.0489094

Weak instrument test -

First-stage F-statistic (2, 67) = 8.08769

A value < 10 may indicate weak instruments

```
#
# FIRST-STAGE REGRESSIONS
#
? ols PRIMRATE const TBRATE TOTDEP CBNDRATE INDPROD
```

Model 7: OLS, using observations 1979:01-1984:12 (T = 72)
 Dependent variable: PRIMRATE

	coefficient	std. error	t-ratio	p-value
const	1.95945	3.03128	0.6464	0.5202
TBRATE	0.772759	0.116738	6.620	7.22e-09 ***
TOTDEP	-0.00521408	0.00151897	-3.433	0.0010 ***
CBNDRATE	0.760620	0.178192	4.269	6.34e-05 ***
INDPROD	0.00705941	0.0224958	0.3138	0.7546

Mean dependent var	14.08333	S.D. dependent var	3.176402
Sum squared resid	94.24382	S.E. of regression	1.186012
R-squared	0.868440	Adjusted R-squared	0.860586
F(4, 67)	110.5684	P-value(F)	9.31e-29
Log-likelihood	-111.8555	Akaike criterion	233.7109
Schwarz criterion	245.0943	Hannan-Quinn	238.2427
rho	0.650325	Durbin-Watson	0.701157

Excluding the constant, p-value was highest for variable 4 (INDPROD)

The results are very good for the Demand Curve but in the case of the Supply Curve we have a **problem of weak instruments**. Interpreting the results is easy when we use **robust** estimation. The relevant code lines in Model 6 say

```
Weak instrument test -
  First-stage F-statistic (2, 67) = 8.08769
  A value < 10 may indicate weak instruments
```

which can be clearly interpreted as meaning that one or both of our two instruments for the supply curve are weak. In Model 5, the case where we do not use **robust** estimation, the relevant code lines are

```
Weak instrument test -
  First-stage F-statistic (2, 67) = 12.5559
  Critical values for desired TSLS maximal size, when running
  tests at a nominal 5% significance level:
```

size	10%	15%	20%	25%
value	19.93	11.59	8.75	7.25

Maximal size may exceed 10%

which is telling us that our interpretation of our two-stage-least-squares results as statistically significant at the 5% level may, given the F-statistic of 12.5559, be really mean statistically significance at only a bit less than the 15% level.²

Our final regression of the group above, which is in fact the first-stage regression, indicates clearly that the INDPROD variable is not statistically significant and therefore a weak instrument. When we re-run that first-stage regression excluding that variable we obtain the following result.

```
? ols PRIMRATE const TBRATE TOTDEP CBNDRATE
```

Model 10: OLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: PRIMRATE

	coefficient	std. error	t-ratio	p-value	
const	2.80398	1.38572	2.023	0.0470	**
TBRATE	0.794364	0.0936483	8.482	2.92e-012	***
TOTDEP	-0.00491780	0.00118198	-4.161	9.12e-05	***
CBNDRATE	0.728401	0.144673	5.035	3.74e-06	***

Mean dependent var	14.08333	S.D. dependent var	3.176402
Sum squared resid	94.38234	S.E. of regression	1.178124
R-squared	0.868247	Adjusted R-squared	0.862434
F(3, 68)	149.3721	P-value(F)	7.33e-30
Log-likelihood	-111.9083	Akaike criterion	231.8167
Schwarz criterion	240.9233	Hannan-Quinn	235.4421
rho	0.649538	Durbin-Watson	0.703203

which indicates that all the remaining instruments are statistically significant. When we then rerun the two-stage-least-squares calculation for the supply curve excluding that instrument we obtain the result below.

```
? tsls QLOANS const PRIMRATE TBRATE TOTDEP ; const TBRATE TOTDEP CBNDRATE
```

Model 8: TSLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: QLOANS

Instrumented: PRIMRATE

Instruments: const TBRATE TOTDEP CBNDRATE

	coefficient	std. error	z	p-value	
const	-88.9941	17.9127	-4.968	6.76e-07	***
PRIMRATE	6.33488	2.43723	2.599	0.0093	***
TBRATE	-6.74079	2.91344	-2.314	0.0207	**
TOTDEP	0.339221	0.00997913	33.99	2.82e-253	***

²I must thank colleague Angelo Melino for helping me understand what the code here means.

Mean dependent var	361.5556	S.D. dependent var	62.31587
Sum squared resid	14211.95	S.E. of regression	14.45680
R-squared	0.948621	Adjusted R-squared	0.946354
F(3, 68)	422.2446	P-value(F)	7.14e-44
Log-likelihood	-564.6957	Akaike criterion	1137.391
Schwarz criterion	1146.498	Hannan-Quinn	1141.017
rho	0.199762	Durbin-Watson	1.595630

Hausman test -

Null hypothesis: OLS estimates are consistent
Asymptotic test statistic: Chi-square(1) = 5.83289
with p-value = 0.0157293

Weak instrument test -

First-stage F-statistic (1, 68) = 25.3494
Critical values for desired TSLS maximal size, when running
tests at a nominal 5% significance level:

size	10%	15%	20%	25%
value	16.38	8.96	6.66	5.53

Maximal size is probably less than 10%

#

? tsls QLOANS const PRIMRATE TBRATE TOTDEP ; const TBRATE TOTDEP CBNDRATE --robust

Model 9: TSLS, using observations 1979:01-1984:12 (T = 72)

Dependent variable: QLOANS

Instrumented: PRIMRATE

Instruments: const TBRATE TOTDEP CBNDRATE

HAC standard errors, bandwidth 3 (Bartlett kernel)

	coefficient	std. error	z	p-value	
const	-88.9941	18.3089	-4.861	1.17e-06	***
PRIMRATE	6.33488	2.28663	2.770	0.0056	***
TBRATE	-6.74079	3.16223	-2.132	0.0330	**
TOTDEP	0.339221	0.0102229	33.18	1.93e-241	***

Mean dependent var	361.5556	S.D. dependent var	62.31587
Sum squared resid	14211.95	S.E. of regression	14.45680
R-squared	0.948621	Adjusted R-squared	0.946354
F(3, 68)	390.9100	P-value(F)	8.54e-43
Log-likelihood	-564.6957	Akaike criterion	1137.391
Schwarz criterion	1146.498	Hannan-Quinn	1141.017

rho 0.199762 Durbin-Watson 1.595630

Hausman test -

Null hypothesis: OLS estimates are consistent
Asymptotic test statistic: Chi-square(1) = 5.83289
with p-value = 0.0157293

Weak instrument test -

First-stage F-statistic (1, 68) = 16.3892
A value < 10 may indicate weak instruments

We can conclude that the single remaining instrument is not weak. In this case, Gretl does not run the **Sargan over-identification test** for instrument validity because we only have one instrument. We can nevertheless conclude from our earlier results that this single instrument is valid.

Upon looking at our two-stage-least-squares results as compared to the simple application of OLS to the demand and supply curves we can conclude that the procedure produces statistically significant coefficients with the expected signs.

Many problems can be best analyzed using **Panel Data** organized either in **stacked time-series** or **stacked cross-section** form. To illustrate, consider the data in the Excel spreadsheet file `paneldat.xls` which contains the variables noted below for eleven countries stacked in time-series form with the times series running from 2000Q1 through 2010Q4 and thereby comprising 44 quarters of data. The countries are Euro-Area countries—Austria, Belgium, Finland, France, Germany, Greece, Ireland, Italy, Netherlands, Portugal and Spain—which are numbered respectively from 1 through 11. These data are used to analyze the determination of the real exchange rates of each of the eleven countries with respect to the remaining ten.³ The variables can be described as follows.

REXCPI —CPI-Based Real Exchange Rate vs. the Remaining Ten Countries
TOT —Terms of Trade vs. the Rest of the World
GCONY —Government Consumption Expenditure as Percent of GDP
UEMPR —Unemployment Rate minus the Remaining Ten-Country Average
UEMPRL4 —Unemployment Rate Variable Lagged 4 Quarters

One would expect that a rise in a country's terms of trade would increase the prices of its traded output components relative to those of the other countries and thereby increase the domestic real exchange rate. An increase in government consumption expenditure would be expected to raise the domestic real exchange rate because of the political pressure to use domestic resources almost entirely as inputs. And a rise in the domestic unemployment rate relative to the average unemployment rate of the remaining ten countries would be expected to reduce domestic relative to foreign output, raising its price. Also, it would seem likely that a high domestic unemployment rate four quarters in the past would tend to cause the domestic price level to fall relative to the price levels in the other ten Euro-Area countries, reducing the domestic real exchange rate.

³The analysis here is a small segment of a broader analysis contained in John E. Floyd, 'Real Exchange Rates and Unemployment Within the Euro Area,' Working Paper, University of Toronto, 2011, available from my web-site.

The Excel data file can be loaded into Gretl and given the appropriate panel interpretation using the code

```
open C:\DIRECTORY PATH\paneldat.xls
setobs 44 1.1 --stacked-time-series
```

and then saved as the Gretl data file `paneldat.gdt` by clicking in the Data Window on **File** and then on **Save data as**. Descriptions of the variables are added by clicking on **Variable** and then **Edit attributes** and typing in descriptive lines. Had the data been organized in stacked cross-section form the `setobs` code line would be

```
setobs 11 1.1 --stacked-cross-section
```

where 11 is the number of countries and the 44 in the previous code line is the number of quarters of data. To create **entity fixed-effects** and **time dummies** we execute the code lines

```
genr unitdum
genr timedum
```

or, alternatively, we can click on **Add** in the Data Window and then on **Unit dummies** and **Time dummies**. To illustrate **Panel-Data-Analysis** we feed Gretl the following script file which uses the `ols` function with and without entity fixed effects and then the `panel` function, which automatically includes entity fixed effects.

```
open C:\DIRECTORY PATH\paneldat.gdt
genr unitdum
genr timedum
#
ols REXCPI const TOT GCONY UEMPR UEMPRL4 # Do a pooled analysis -- no fixed effects
RRESID = $uhat # Save the residuals
RSSE = sum(RRESID^2) # Obtain the (restricted) sum of squared errors
#
# Run the regression including all unit dummy variables but the first one
#
ols REXCPI const TOT GCONY UEMPR UEMPRL4 du_2 du_3 du_4 du_5 du_6 du_7 du_8 \
du_9 du_10 du_11
#
URESID = $uhat # Save the residuals
USSE = sum(URESID^2) # Obtain the (unrestricted) sum of squared errors
UDF = $df # Save the degrees of freedom
NUMRES = 10 # Note that the number of restrictions equals
# 10, the number of unit dummies dropped
FSTAT = ((RSSE - USSE)/NUMRES)/(USSE/UDF) # Calculate the F-Statistic resulting from
# imposing the restrictions
PVAL = pvalue(F, NUMRES, UDF, FSTAT) # Calculate the P-Value of null-hypothesis
# that restrictions have no effect
```

```

Now do a panel analysis with entity fixed effects
#
panel REXCPI const TOT GCONY UEMPR UEMPRL4
#
# Then add time fixed effects
#
panel REXCPI const TOT GCONY UEMPR UEMPRL4 --time-dummies # Add time dummies

```

The results of running this script are as follows.

```
? open C:\DIRECTORY PATH\paneldat.gdt
```

```

Read datafile C:\DIRECTORY PATH\paneldat.gdt
periodicity: 44, maxobs: 484
observations range: 1:01-11:44

```

Listing 9 variables:

```

0) const      1) QUARTER    2) ENTITY      3) REXCPI      4) TOT
5) GCONY      6) NCIY          7) UEMPR       8) UEMPRL4

```

```
? genr unitdum
```

Panel dummy variables generated.

```
? genr timedum
```

Panel dummy variables generated.

```
#
```

```
? ols REXCPI const TOT GCONY UEMPR UEMPRL4
```

```

Model 1: Pooled OLS, using 476 observations
Included 11 cross-sectional units
Time-series length: minimum 40, maximum 44
Dependent variable: REXCPI

```

	coefficient	std. error	t-ratio	p-value
const	91.0400	2.53457	35.92	5.51e-137 ***
TOT	7.37696	2.40226	3.071	0.0023 ***
GCONY	0.0825546	0.0280500	2.943	0.0034 ***
UEMPR	0.609811	0.0793481	7.685	8.95e-014 ***
UEMPRL4	-0.539056	0.0811813	-6.640	8.66e-011 ***

```

Mean dependent var    100.0851    S.D. dependent var    2.267198
Sum squared resid     2111.190    S.E. of regression    2.117158
R-squared              0.135322    Adjusted R-squared    0.127978
F(4, 471)             18.42779    P-value(F)            4.43e-14
Log-likelihood         -1029.937    Akaike criterion      2069.874
Schwarz criterion      2090.701    Hannan-Quinn          2078.063

```

rho 0.906017 Durbin-Watson 0.118344

? RRESID = \$uhat

Generated series RRESID (ID 64)

? RSSE = sum(RRESID^2)

Generated scalar RSSE = 2111.19

#

? ols REXCPI const TOT GCONY UEMPR UEMPRL4 du_2 du_3 du_4 du_5 du_6 du_7 du_8 \ du_9 du_10 du_11

Model 2: Pooled OLS, using 476 observations

Included 11 cross-sectional units

Time-series length: minimum 40, maximum 44

Dependent variable: REXCPI

	coefficient	std. error	t-ratio	p-value	
const	88.8492	2.70208	32.88	5.79e-123	***
TOT	8.82768	2.44544	3.610	0.0003	***
GCONY	0.129681	0.0420786	3.082	0.0022	***
UEMPR	0.688570	0.0782894	8.795	2.87e-017	***
UEMPRL4	-0.733142	0.0852215	-8.603	1.23e-016	***
du_2	-0.538608	0.462652	-1.164	0.2450	
du_3	1.51965	0.469366	3.238	0.0013	***
du_4	-0.714875	0.488007	-1.465	0.1436	
du_5	0.740513	0.439577	1.685	0.0927	*
du_6	0.437498	0.462672	0.9456	0.3449	
du_7	-0.438676	0.443116	-0.9900	0.3227	
du_8	-0.331305	0.431047	-0.7686	0.4425	
du_9	-1.69261	0.516560	-3.277	0.0011	***
du_10	-1.71162	0.448171	-3.819	0.0002	***
du_11	-0.717120	0.512431	-1.399	0.1624	

Mean dependent var	100.0851	S.D. dependent var	2.267198
Sum squared resid	1780.100	S.E. of regression	1.965042
R-squared	0.270926	Adjusted R-squared	0.248785
F(14, 461)	12.23633	P-value(F)	2.26e-24
Log-likelihood	-989.3384	Akaike criterion	2008.677
Schwarz criterion	2071.158	Hannan-Quinn	2033.246
rho	0.876141	Durbin-Watson	0.181214

Excluding the constant, p-value was highest for variable 16 (du_8)

```

? URESID = $uhat
Generated series URESID (ID 65)
? USSE = sum(URESID^2)
Generated scalar USSE = 1780.1
? UDF = $df
Generated scalar UDF = 461
? NUMRES = 10
Generated scalar NUMRES = 10
? FSTAT = ((RSSE - USSE)/NUMRES)/(USSE/UDF)
Generated scalar FSTAT = 8.57437
? PVAL = pvalue(F, NUMRES, UDF, FSTAT)
Generated scalar PVAL = 6.85527e-013
#
? panel REXCPI const TOT GCONY UEMPR UEMPRL4

```

```

Model 3: Fixed-effects, using 476 observations
Included 11 cross-sectional units
Time-series length: minimum 40, maximum 44
Dependent variable: REXCPI

```

	coefficient	std. error	t-ratio	p-value	
const	88.5412	2.74447	32.26	2.64e-120	***
TOT	8.82768	2.44544	3.610	0.0003	***
GCONY	0.129681	0.0420786	3.082	0.0022	***
UEMPR	0.688570	0.0782894	8.795	2.87e-017	***
UEMPRL4	-0.733142	0.0852215	-8.603	1.23e-016	***

```

Mean dependent var    100.0851    S.D. dependent var    2.267198
Sum squared resid    1780.100    S.E. of regression    1.965042
R-squared              0.270926    Adjusted R-squared    0.248785
F(14, 461)           12.23633    P-value(F)            2.26e-24
Log-likelihood        -989.3384    Akaike criterion      2008.677
Schwarz criterion     2071.158    Hannan-Quinn          2033.246
rho                   0.876141    Durbin-Watson         0.181214

```

```

Test for differing group intercepts -
Null hypothesis: The groups have a common intercept
Test statistic: F(10, 461) = 8.57437
with p-value = P(F(10, 461) > 8.57437) = 6.85527e-013

```

? panel REXCPI const TOT GCONY UEMPR UEMPRL4 --time-dummies

Model 4: Fixed-effects, using 476 observations

Included 11 cross-sectional units

Time-series length: minimum 40, maximum 44

Dependent variable: REXCPI

	coefficient	std. error	t-ratio	p-value	
const	80.0832	3.14189	25.49	3.80e-087	***
TOT	17.8077	2.87264	6.199	1.36e-09	***
GCONY	0.0515733	0.0479192	1.076	0.2824	
UEMPR	0.659915	0.0792324	8.329	1.18e-015	***
UEMPRL4	-0.762678	0.0859304	-8.876	2.06e-017	***
dt_2	0.218562	0.874504	0.2499	0.8028	
dt_3	0.572026	0.875016	0.6537	0.5136	
dt_4	0.641683	0.876426	0.7322	0.4645	
dt_5	-0.141758	0.855416	-0.1657	0.8685	
dt_6	0.0399050	0.855768	0.04663	0.9628	
dt_7	-0.0606123	0.855731	-0.07083	0.9436	
dt_8	-0.109481	0.858075	-0.1276	0.8985	
dt_9	0.261103	0.857685	0.3044	0.7610	
dt_10	0.450619	0.856935	0.5258	0.5993	
dt_11	0.386723	0.857766	0.4508	0.6523	
dt_12	0.787089	0.857563	0.9178	0.3592	
dt_13	0.516916	0.857281	0.6030	0.5469	
dt_14	0.693138	0.857354	0.8085	0.4193	
dt_15	0.404097	0.857584	0.4712	0.6377	
dt_16	0.521695	0.857569	0.6083	0.5433	
dt_17	0.290969	0.858383	0.3390	0.7348	
dt_18	0.553318	0.857295	0.6454	0.5190	
dt_19	0.648936	0.856404	0.7577	0.4490	
dt_20	0.776737	0.857324	0.9060	0.3655	
dt_21	0.924230	0.857420	1.078	0.2817	
dt_22	1.09060	0.856923	1.273	0.2038	
dt_23	1.17026	0.857355	1.365	0.1730	
dt_24	1.21605	0.858459	1.417	0.1574	
dt_25	1.29950	0.860560	1.510	0.1318	
dt_26	1.45252	0.861146	1.687	0.0924	*
dt_27	1.41487	0.858248	1.649	0.1000	*
dt_28	1.50966	0.858780	1.758	0.0795	*
dt_29	1.26526	0.859329	1.472	0.1417	
dt_30	1.68112	0.859409	1.956	0.0511	*
dt_31	1.48919	0.858728	1.734	0.0836	*
dt_32	1.94941	0.863625	2.257	0.0245	**

dt_33	2.28293	0.865946	2.636	0.0087	***
dt_34	2.82201	0.873497	3.231	0.0013	***
dt_35	2.43253	0.898574	2.707	0.0071	***
dt_36	2.23438	0.867778	2.575	0.0104	**
dt_37	1.54334	0.870888	1.772	0.0771	*
dt_38	1.25940	0.872964	1.443	0.1499	
dt_39	1.05228	0.869343	1.210	0.2268	
dt_40	1.13348	0.878790	1.290	0.1978	
dt_41	1.45688	0.895835	1.626	0.1046	
dt_42	2.03449	0.896414	2.270	0.0237	**
dt_43	1.96224	0.892941	2.198	0.0285	**
dt_44	2.20247	0.905931	2.431	0.0155	**
Mean dependent var	100.0851	S.D. dependent var	2.267198		
Sum squared resid	1597.787	S.E. of regression	1.955110		
R-squared	0.345596	Adjusted R-squared	0.256359		
F(57, 418)	3.872785	P-value(F)	5.89e-16		
Log-likelihood	-963.6225	Akaike criterion	2043.245		
Schwarz criterion	2284.839	Hannan-Quinn	2138.244		
rho	0.878787	Durbin-Watson	0.181553		

Test for differing group intercepts -

Null hypothesis: The groups have a common intercept

Test statistic: $F(10, 418) = 8.59085$

with p-value = $P(F(10, 418) > 8.59085) = 8.37999e-013$

Wald test for joint significance of time dummies

Asymptotic test statistic: $\text{Chi-square}(43) = 47.6953$

with p-value = 0.287634

Note that the `panel` function in Model 3 yields the same Test-statistic and P-Value for the null-hypothesis that the groups have a common intercept as the F-Statistic and corresponding P-Value obtained when we estimated using OLS with the entity dummies added in Model 2 and compared the result to a restricted OLS pooled estimate with no dummies added in Model 1 and then tested the null-hypothesis that the entity dummies contribute nothing in explaining the movements of REXCPI. The entity dummies allow the constant term to be different for each of eleven countries. The `panel` function estimate with time dummies added in Model 4 indicates clearly that the time dummies contribute nothing in explaining the real exchange rate movements. Normally we would only use straight OLS if we cannot reject, in our analysis using the `panel` function, the null-hypothesis that the entity dummies and time dummies together contribute nothing in explaining the dependent variable. The `ols` function must be used to obtain a pooled estimate. Of course, we could go the long-route and add both the time and entity dummies in our `ols` analysis and then use F-tests of their statistical significance.

Notice that in the above panel analysis we loaded a superficially balanced panel data set containing 484 data observations but the results produced by Gretl were based on only 476

observations. This is because some of the year 2000 data for Greece was missing and represented by NA's as was some of the year 2010 data for Portugal. Gretl automatically adjusted for the actual imbalance in panel data. Always load into Gretl a balanced panel containing NA values for any missing observations.

Gretl Sessions

An alternative way to work in Gretl is to use a Session Approach in which you load your data and transform it and then plot charts and do statistical analysis, saving your results as icons in a session file. You can return to this file to review or extend your results at any time, saving a complete record of all you have done.

Demand for Money Session

We begin by loading Gretl and then the data file `usmond.dat.gdt`. Then we click on **File** and then on **Session files...** and on **Save session as...** giving it the name `usdemon.gretl` and then exit Gretl.

Then we reload Gretl and reopen the session just saved. By clicking on the **Data info** icon we can remind ourselves of the Excel file from which we loaded the data into Gretl. We can edit our data by clicking on the **Data set** icon. Clicking on the **Summary Statistics** icon produces a summary of all the variables in our data set and by clicking on the **Correlations** icon we can view the correlations of each variable with each of the others.

To obtain the two alternative U.S. inflation rates we first highlight the **USCPI** and **USIPD** variables in the Data Window and then click on **Add**, then **Lags of selected variables**, setting the number of lags at four. We then click twice more on **Add** in the Data Window and then on **Define new variable...** and insert the following formulas in the respective windows that appear.

```
YYGCPI = 100*(USCPI - USCPI_4)/USCPI_4  
YYGIPD = 100*(USIPD - USIPD_4)/USIPD_4
```

To view summary-statistics for these new variables we again click on the **Summary** icon and for correlations we click again on the **Correlations** icon. We then plot the above inflation rates by clicking on **View** and then **Graph specified variables**, choosing **YYGCPI** and **YYGIPD**. We then save this graph as an icon and rename it as **Inflation Graph**.

Next we calculate real M1, real M2 and real GDP using the procedure above. The price level used is the implicit GDP deflator and the three code lines are the following.

```
USRM1 = 100*(USM1/USIPD)  
USRM2 = 100*(USM2/USIPD)  
USRGDP = 100*(USGDP/USIPD)
```

Then, clicking on **Add** along the top of the Data Window, we add the logarithms of the above three variables to our data set.

Now we run the U.S. M1 regression in logarithms by clicking on **Model** and then **Ordinary Least Squares** and follow the prompts without using robust estimation. We save the Model Window that appears as Model 1. We then click on **Tests** along the top of that Window and then on **Heteroskedasticity** and choose **Breusch-Pagan**, saving the result as an icon. Following the same procedure, we then choose **autocorrelation** and set the lag at 4, again saving the result as an icon. We then click on **Graphs** along the top of that Model Window and chose **Fitted, actual plot** and **Residual plot**, saving the two graphs to the session as icons. Recognizing the presence of heteroskedasticity and autocorrelation in the residuals, we now click on **Edit** along the top of the Model Window and then on **Modify model** and using the same variables choose robust estimation, saving the result as Model 2.

Finally, we follow the same procedures for U.S. M2 as we did for U.S. M1 above. And we then click on the Notes icon and write a description of what we have done sufficient to remind us of the contents of the four Models and the five Graphs that that appear as session icons.

Needless to say, throughout the entire process above we must frequently save the session to ensure that we do not lose any of our work.

Two-Stage Least Squares Session

Here we begin by loading Gretl and accessing the data file `loandata.gdt` and then saving the session as `loandata.gretl`. Then after exiting and restarting Gretl and loading the new session file we estimate demand and supply curves using standard OLS by clicking on **Model** and then **Ordinary Least Squares** and saving the two Models as icons.

Now we estimate the demand and supply functions using two-stage least squares by clicking on **Model** and then on **Instrumental Variables** and **Two-Stage Least Squares** and entering the appropriate variables in the windows that appear, again saving the results as icons. We fit each curve twice, once without robust estimation of the coefficient standard errors and once with. Given the weakness of one an the instruments in the estimation of the supply function we then run the first-stage regression using **Ordinary Least Squares**. Finally, given that `IPROD` is the weak instrument, we fit the supply curve again using **Two Stage Least Squares** using `CBNDRATE` as the only instrument, doing so twice, once with robust coefficient standard errors and once without and saving both results as icons.

We end the session by entering appropriate material in the Notes icon to remind ourselves later of what we did in this session.

Panel Estimation Session

The first step is to start Gretl and load the data file `paneldat.gdt` and save the session as `panel.gretl`. Then we exit and restart and load the session file. Then we do a pooled analysis using straight OLS and save the result as the icon Model 1, apply the `panel` function with entity fixed effects, saving the result as icon Model 2, and apply the panel function again with both entity fixed effects and time fixed effects, saving the result as icon Model 3. This information is then written in the notes icon.

Index

- Adding Data Descriptions — 1
- Autocorrelations, calculating — 15, 16
- Bootstrapping — 10
- Correlogram — 15, 16
- Data, Summary Statistics — 3
- Dickey-Fuller Tests — 12, 13, 14, 15, 16, 17
- Instrumental Variables:
 - Modules --> Two-Stage-Least Squares — 20
 - Weak-Instruments — 27
- Johansen Cointegration Test — 17, 18
- Kernel Densities — 10, 11
- Lags, Setting — 2
- Loading Data — 1
- Ljung-Box Q-Statistic — 7
- Ordinary Least Squares Regressions:
 - Calculating using `ols` function — 3, 10
 - Extracting Fitted Values and Residuals — 3, 37
 - HAC Coefficient Standard-Errors — 12
 - Testing Residuals for Heteroskedasticity and Autocorrelation — 3, 6, 7, 37
 - Robust Estimation — 12
 - Spurious Regression — 12
- Panel Data Analysis:
 - Balanced and Unbalanced Panels — 36, 37
 - Entity Fixed-Effects and Time Dummies — 31, 32, 33
 - `panel` Function — 32, 34, 35, 36, 37
 - Pooled Estimation using OLS — 32
 - Stacked-Cross-Section — 30, 31
 - Stacked-Time-Series — 30, 31
 - Session Approach — 39
- Partial Autocorrelations, calculating — 15, 16
- Plotting Variables — 1, 7, 8, 9, 10, 37
- Quantiles — 3
- Saving Data — 1
- Scripts, Writing and Editing — 2
- Sessions Approach — 37, 38, 39
- Two-Stage Least Squares — 19, 20, 22, 23, 25, 26, 27, 28, 29, 30, 38
 - See Instrumental Variables
- Variables:
 - Creating — 2, 3
 - Summary Statistics — 3, 37