Chapter 3
TSP FUNDAMENTALS

This chapter introduces some basic TSP concepts and describes commands for setting up the sample of observations, reading data, making transformations, and setting options. It concludes with a fairly large example illustrating these concepts and some estimation methods from OLS to FIML.

3.1. Describing the sample of observations: FREQ, SMPL

Before beginning a TSP program, you need to specify the frequency of your data and the number of observations with the FREQ and SMPL commands. The **FREQ** command tells TSP how often a year each series is observed. For example,

```
FREQ A ;
```

specifies annual data, observed once a year. Other frequencies TSP understands are

Q	Quarterly (four times per year)
M	Monthly (12 times per year)
W	Weekly (52 times per year)
numeric	Any number less than 32,768 (for example, 26 times per year)
N	Undated (usually cross-sectional or survey data)

If no frequency is specified, TSP assumes the frequency **N** (none).

A special FREQ statement is available for panel data (time series - cross section data). For details, see Chapter 15.

The **SMPL** command defines the set of observations to be used. For example, in the simple run in the last chapter,

```
SMPL 76,85;
```

informed TSP that observations dated from 1976 through 1985 were to be used. You only need a SMPL command when you want to change the observations you are using. There must be a sample in effect before the first operation on a time series, but if you read data from a file without specifying a SMPL, the sample is defined for you.

Formats for dates (the SMPL endpoints) are:

Annual: the full year or the last digits of the year (for the twentieth and twenty-first centuries). Examples: 1981, 1895, 82 (1982), 101 (2001). 200 would normally denote the year 2100 (using the default OPTIONS BASEYEAR=1900), while 201 would be just the year 201. Negative years are valid; they are "B.C.".

> **Note:** TSP is year 2000 compliant.

Quarterly, Monthly, Weekly, and *numeric*: the full year or the last two digits of the year, colon, and the quarter, month, week, or period number. Examples: 1972:1, 65:4, 80:11. Obviously, quarters beyond 4 and months beyond 12 are invalid, as are negative or absent values.

No Frequency: use the observation numbers. Examples: 1, 2000.

More than one pair of dates may appear in a SMPL command. For example,

```
SMPL 59,70 72,83;
```

omits the year 1971. To omit several scattered observations, use SELECT or SMPLIF (see Section 3.3).

The dates in a SMPL statement must be in ascending order -- SMPL 70,59; would be invalid.

3.2. Reading data into TSP: READ

There are several ways to input your data into TSP. You can include it with the program or put it in a separate file. The separate file can be in a variety of formats, including spreadsheet and *MicroTSP/EViews*. This chapter describes the simplest and easiest methods, reading data within the TSP program and reading data from an external file. See Chapter 16 for more complex examples.

3.2.1. Reading data in free format within the program

If you have only a few short series, you will probably find it convenient to read the data in free format within the program. To read data in free format, use the READ command followed by the names of one or more series (names can be up to eight characters long). The data follows immediately in free format, observation by observation, with one (rough) column per series. The numbers are separated from each other by blanks or commas. Here is the example from Chapter 2:

```
FREQ A;  SMPL 76,85;
READ SALES,GNP;
11.7 1706
13.7 1901
11.4 2151
12.3 2391
19.4 2608
20.4 2956
18.2 3051
25.3 3261
24.3 3639
28.3 3843
```

The data for SALES and GNP could also be read separately (by rows) if it was more convenient. Semicolons after the data are optional:

```
FREQ A;  SMPL 76,85;
LOAD SALES;
11.7 13.7 11.4 12.3 19.4 20.4 18.2 25.3 24.3 28.3 ;
LOAD GNP;
1706 1901 2151 2391 2608 2956 3051 3261 3639 3843 ;
```

If the number of data points does not correspond to the current sample, an error message will note the discrepancy. Missing values can be entered as a period (.), or the SMPL can be set to read in only those observations that are available. For example, if you had data on imports (IMPT) only for 1977 and later in the above example, use

```
SMPL 77,85;
LOAD IMPT;
35 42 67 85 90 92 97 120 126 ;
```

If you have a large volume of data, the commands and data can be moved to the bottom of the program, after an END; statement which terminates the data analysis. A NOPRINT; command after the END; statement will suppress printing of the commands and data in the output file. This is recommended after an initial run has checked for any error messages in loading the data. See Section 3.6 for more details.

Matrices may also be loaded -- see READ in the *Reference Manual* or Section 12.2.1 in this manual for more information on loading and using matrices. The matrix type and dimensions are required.

3.2.2. Reading data from an external file

For large datasets, you will probably read the data from a separate file created with a text editor, written by another program, or copied from disk or tape. To input the file, use the READ command with the FILE='filename' *option.* (Note: TSP *Options* are always enclosed in parentheses following the command name.) The following example would read a file containing just the data from the example in Chapter 2:

```
READ(FILE='SALESGNP.DAT') SALES,GNP;
```

The file SALESGNP.DAT could look like the following (free format allows more than one observation per line, or one observation split across more than one line):

```
11.7 1706
13.7 1901
11.4
2151
12.3 2391  19.4 2608  20.4 2956
18.2 3051 25.3 3261 24.3 3639  28.3 3843
```

The next example would read the Excel file, SML.XLS. For information on reading spreadsheet files, see Chapter 16.

```
READ(FILE='SML.XLS') ;
```

3.3. Selection of observation subsets: SELECT, SMPLIF

In addition to the SMPL command, the sample can also be selected by rules based on the values of series using SELECT and SMPLIF commands. The difference between SELECT and SMPLIF is that successive SMPLIFs nest (define smaller and smaller subsets of observations) while successive SELECT statements do not. For example, suppose you want to run a regression on only the positive values of the variable X, but these values are scattered throughout a dataset with 200 observations. It would be tedious to specify all the gaps for omitted observations in a SMPL statement. Instead, use SELECT:

```
SMPL 1,200;
SELECT X>0;
OLSQ Y C,X;
```

SELECT always refers to the most recent SMPL statement to define the overall sample of observations -- any intervening SELECT or SMPLIF statements are ignored. As a result, successive SELECT statements do not "nest" within each other. Nesting is provided with the SMPLIF command, which includes only those observations from the *current sample* which make the selection criteria true. For example,

```
SMPL 1,200;          is equivalent to     SMPL 1,200;
SMPLIF X>0;                               SELECT X>0;
  OLSQ Y C,X;                               OLSQ Y C,X;
  SMPLIF X<=10;                           SELECT X>0 & X<=10;
    OLSQ Y C,X;                             OLSQ Y C,X;
```

Note that the samples defined by SELECT and SMPLIF remain in effect until the next SMPL, SELECT, or SMPLIF statement. The "full sample" in these examples could be restored in one of two ways:

```
SMPL 1,200;                or              SELECT 1;
```

(because the value 1 makes the selection criterion true for all observations in the previous SMPL statement).

In either a SMPLIF or a SELECT statement, the selection criterion can be any expression like those used in GENR (see

Section 3.5). When the expression is evaluated for each observation, values that are positive will be treated as true and values that are zero or negative as false.

3.4. Missing Values

TSP supports missing values in data series. In free format input files you should indicate missing data by a period (.); in fixed format files you will have to use another value and then recode that value to the missing value code (see Section 3.5.1 for an example). TSP names the missing value code @MISS so that you can refer to it in your program.

The WRITE statement also uses a period (.) to indicate missing values when printing or writing output files. Procedures that transform data such as GENR automatically propagate missing values for most operation when they are present (missing values are also generated if there are arithmetic errors in the transformations).

Many statistical procedures (MSD, OLSQ, INST, AR1) automatically delete observations with missing values before executing. Others (the time series procedures such as ARCH) print an error message if you attempt estimation with missing data. A complete list of these procedures that cannot estimate gaps in the data are given in the *Reference Manual*. You can use the MISS() function to get around this problem:

```
SELECT .NOT. MISS(SALES) .AND. .NOT. MISS(GNP) ;
```

will choose only observations for which SALES and GNP are not missing for the estimation procedure.

3.5. Creating new series with transformations: GENR

The GENR command creates a new series based on a formula you supply. For example,

```
GENR Z=LOG(REV);
```

forms a new series called Z, the natural logarithm of REV. Series Z is defined only for those observations in the current sample; any other observations of Z that are used later will have a missing value code as their value. A missing value code is also given to any observation that has an arithmetic error when computed. For example, if one of the observations of REV were zero, Z would be missing for that observation since the log of zero is undefined. If Z is used later in another GENR, the missing value will propagate, that is, the new series will also have a missing value for that observation.

You can use parentheses without limit to indicate the order in which the elements of the formula should be evaluated. The word GENR is not required; you can omit it and just give the equation you want computed:

```
Z=LOG(REV) ;
```

A complete set of rules and functions (like CNORM(), GAMMA(), etc.) for composing GENR formulas appears in Appendix A. Some common operations are:

+	addition	=	equal	LOG	natural log	&	and
-	subtraction	^=	not equal	EXP	exponential	^	not, or power
*	multiplication	>	greater than	SQRT	square root	|	or
/	division	>=	greater than or equal	LOG10	log base 10		
**	exponentation	<	less than	ABS	absolute value		
		<=	less than or equal	MISS	missing flag value		

Typical examples:

```
SUM = A+B;
AVEQ = Q1*ALPHA + Q2*(1-ALPHA);
```

If you are unsure about the effect of a GENR command and you don't understand the precedence rules in Appendix A, use the PRINT, PLOT, GRAPH, HIST, or MSD commands to check your transformations.

3.5.1. Dummy variables and recoding

The logical operators such as = ^= > < >= <= & | ^ yield true/false (one/zero) values. This is useful for creating dummy variables and recoding categorical variables. The following example creates a dummy variable that is 1 when X is positive, and zero otherwise:

```
XPOS = X>0;
```

This is much faster than the alternative with SELECT:

```
XPOS = 0;  SELECT X>0;  XPOS = 1;  SELECT 1;
```

See also the DUMMY command (Section 5.6), which is useful for creating large sets of dummy variables and seasonal dummies.

To recode a variable, include logical expressions (0/1) which are multiplied by the value for each case. If the 0/1 expressions are mutually exclusive, only one value will be selected (for the case that is true). For example, say EDUC has 3 categories, and we want to assign them the values 9, 12 and 16 respectively:

```
SCHOOL = (EDUC=1)*9 + (EDUC=2)*12 + (EDUC=3)*16;
```

To recode all negative values to zero, but leave positive values unchanged:

```
X = (X<0)*0 + (X>=0)*X;     or     X = (X>0)*X;     or     X = POS(X) ;
```

Operations with missing values are a little more tricky, because most operations on missing values yield a missing value. For example, to recode missing to -99:

```
SELECT MISS(X);  X=-99;  SELECT 1;
```

To recode missing values to zero, GENR can be used directly because 0*X is zero, even when X is missing:

```
X = MISS(X)*0 + (.NOT.MISS(X))*X;     or     X = (.NOT.MISS(X))*X;
```

The operations which can handle missing values without generating a missing answer are:

```
MISS(@MISS)  [true]     0*@MISS  [0]     true .OR. @MISS   [true]      false .AND. @MISS  [false]
```

To recode zeros to missing:

```
SELECT X=0;  X=@MISS;  SELECT 1;
```

In some computer packages, IF/THEN/ELSE statements are used for these types of transformations, but TSP uses IF/THEN/ELSE for program control. Be careful not to use IF/THEN/ELSE for recoding, since they operate on scalars and not on series.

3.5.2. Lags and leads

A lagged series is indicated by putting the lag in parentheses with a minus sign. For example,

```
FREQ M;
SMPL 72:1,85:12;
```

```
DA = AUTOS - AUTOS(-12) ;
```

computes the 12-month change in the monthly series AUTOS. Note that AUTOS must be defined from 1971 through 1985 -- otherwise missing values will be stored in DA. Leads (future values) are defined similarly to lags (past values), except the number in parentheses is positive, e.g. AUTOS(24) is a two-year lead. Lags and leads with series can be used in any command where a plain series may appear. For example,

```
OLSQ Y C,Y(-1);  performs a regression of Y on Y lagged once.
```

When there is a SMPL gap, lagged values of series still contain the true lagged observations. For example,

```
SMPL 59,70 72,83;
Y=X(-1) ;
```

puts the value of X for the 1971 observation (not the 1970 observation) into the observation of Y for 1972.

Here are some more GENR examples with lags and leads:

```
DP = LOG(P/P(-1)) ;
```

computes the rate of change of P in logs.

```
MOVA = (X(-2) + 2*X(-1) + 3*X + 2*X(1) + X(2))/9 ;
```

computes a moving average of the series X with variable weights, using both lags and leads.

> **Note for Power Users:** Entire equations can also be lagged. See Chapter 9 for information.

3.5.3. Dynamic GENR

If the right-hand side of the GENR equation contains explicit lags (or leads) of the left-hand side series, the series will be updated dynamically. Note that the series must already be defined in the observation(s) preceding the current sample (or following the current sample, in the case of leads). For example,

```
SMPL 1,10;  SUM = X;  SMPL 2,10;  SUM = SUM(-1) + X;
```

creates SUM as the accumulated sum of variable X. This feature can also be used to create autoregressive disturbance terms, dynamic simulations, capital stocks (see also the CAPITL command), and net present values. The dynamic GENR is much faster than using a DO loop to create sums. TSP always prints a note if a dynamic GENR is computed. However, care should be taken to avoid an unintentional dynamic GENR when redefining a series.

For example, the correct commands

```
SMPL 70,85;  MNEW = M(-1);  M = MNEW;      or      GENR(STATIC) M = M(-1);
```

replace M with its lag, while the innocent-looking commands

```
SMPL 70,85;  M = M(-1);
```

put the 1969 value of M into all observations 1970 through 1985 (probably not what the user intended).

Dynamic GENRs can be computed backwards as well as forwards. For example, a net present value can be computed by using a lead on the right hand side:

```
SMPL 95,2020;  NPV = 0;
SMPL 95,2019;  NPV = {NPV(+1) + CASH(+1)}/[1+R(+1)];   ? the + sign is not required but emphasizes the lead
```

When NPV is computed, a message warns that the dynamic recursion was done in reverse (starting in the year 2019).

3.6. Useful statements at the beginning of a TSP job

The OPTIONS command is handy for customizing various TSP options. If you intend to use some of these options in most of your program, it is easiest to put them in a LOGIN.TSP file (see Chapter 17). The most common use is to customize the output file for viewing on an 80-column terminal or similar printer (it also suppresses page headers):

```
OPTIONS CRT;
```

This must be the very first TSP command if it is to take effect for the listing of the program in the output file.

If you have a large dataset/model which requires more than the default 4MB of memory, use a command such as:

```
OPTIONS MEMORY=12;
```

Plots of actual and fitted values and residuals are available for many statistical procedures in TSP (ACTFIT, OLSQ, 2SLS, LIML, AR1, LSQ, and FIML). Normally they are not printed. The PLOTS option tells TSP to produce plots for subsequent procedures, while NOPLOT stops the plots. Example:

```
OPTIONS PLOTS;
REGOPT(PVPRINT,STARS) T;    ? this is another common customization, for regression output and diagnostics
OLSQ Y,C,X;
OPTIONS NOPLOT;
```

If OPTIONS CRT; is not used, a header will be printed at the top of each page in the output file. The header contains space for a user name and a 60-character title, which may be varied throughout the run. Example:

```
NAME KMARX 'FINAL RESULTS FOR DAS KAPITAL' ;
```

The title can be changed during the job by the TITLE command containing the new title. Example:

```
TITLE 'Results for Transformations' ;
```

TITLE works even with OPTIONS CRT -- the title is just printed directly to the output file, centered and underlined with asterisks. Titles cannot contain the ; or $ characters.

More than one option may be specified on an OPTIONS statement. For example:

```
OPTIONS CRT, PLOTS;
```

The ? character begins a comment -- TSP will ignore everything that follows it until the end of the line. This is useful for reminding yourself what the program is doing at the top and at any critical sections. For example:

```
OPTIONS CRT;  ? This program tests all models with dataset 4.
```

3.7. The order of statements in a TSP job

Although the ordering of a TSP program is very flexible, statements in a job are generally executed in the order in which they appear. Thus, information about inputs must be provided before they can be used. For example, GENR statements for the variables in a particular regression must appear before the OLSQ for that regression. Most users tend to group GENR statements in a job together, but this is not required.

A TSP job often has two main parts: a (data analysis) program and a data loading section, separated by an END; statement. In processing the job, TSP starts by reading the program and checking some aspects of it, such as unbalanced parentheses in equations or unterminated DO loops. When it reaches an END; statement it stops reading and begins execution. At this point it has not yet read the data section. When it executes a LOAD; statement in the program, it reads and checks the data until it finds another END; statement or an end of file that marks the end of the data. Then it continues executing the program.

Placing the data in a separate load section is not required. Data can be included with a READ statement anywhere in the program. However, using a data section can be more efficient in terms of memory usage. It is also very useful for suppressing the automatic listing of the data and commands in the output file (with the NOPRINT; command).

3.8. The next step

How you proceed from this point (if you made it this far) depends on your background. If this is your first statistical package, you should read the rest of Chapter 3 and Chapter 4 carefully and then start on your own projects. If you are familiar with statistical packages, you will probably plunge into running TSP without much study, and consult the manual whenever you reach a dead end. The *Reference Manual* is designed to make it easy to look up a command and find out about its options, features, method used and some references for further reading. We recommend that you browse through the manual occasionally; you may be unaware of some of the things TSP can do.

3.9. An extended example

The example introduced here and used throughout the manual is built around a simple illustrative macro model of the U.S. economy. The illustrative model contains equations for five endogenous variables:

CONS:	Consumption in real terms
I:	Investment in real terms
GNP:	Real GNP
R:	Interest rate
P:	Price level--implicit GNP deflator

There are four behavioral equations:

1. A consumption function where consumption depends only on GNP.
2. An investment function where investment depends on its own lagged value, GNP, and the interest rate.
3. A money demand function relating the interest rate to the velocity of money.
4. A Phillips curve making the price level rate of change depend on its own lagged value, GNP, and a time trend.

The fifth equation is the GNP identity, stating that GNP equals the sum of consumption, investment, and an exogenous variable called G that measures government expenditures and net exports. The other exogenous variables of the model are the money supply M, and the time trend TIME.

The first step in the development of the model is to assemble the data. The data section for the illustrative model is typical:

```
FREQ A;  SMPL 1946,1975;
READ GNP CONS I ;
475.7 301.4 71
... etc. ...
1186.4 766.6 138.9
;
READ EXPORTS ;
11.6 16.6 8.5  8.8 4.0 7.4 4.9 2 4.5 4.7 7.3 8.9 3.5 .9
... etc. ...
END ;
```

Now for some preliminary transformations of the data: first we want to create a trend variable that is one in the first year and increments by one in each succeeding year. To do this

```
TREND TIME ;
```

creates a series called TIME. Next we perform some GENRs:

```
P = P/100 ;  LP = LOG(P) ;
```

The price index with 1972 = 100 is converted to one with 1972 = 1.0 by dividing by 100. The second GENR statement creates a series LP as the natural log of the price level.

```
G = GOVEXP+EXPORTS ;
```

This creates the variable G out of the standard variables GOVEXP and EXPORTS from the National Income Accounts.

```
R = RS ;
```

At the time the TSP input file was prepared, it was not clear whether a short-term or a long-term interest rate was appropriate. Both were included in the data (RS and RL). In later commands, R is used to stand for the interest rate, whether short-term or long-term. This statement sets R equal to RS. A complete set of results with R equal to RL could be obtained just by changing this statement to GENR R = RL ;

```
PRINT GNP,CONS,I,RS,RL,P ;
```

This statement simply prints these series in a table. It is a good idea to check the transformed data to see that the manipulations were correctly specified.

```
SMPL 1948,1975;
MNEW = (M+M(-1))/2 ;  M = MNEW;
```

Now we change the sample to begin in 1948 rather than 1946 because we are going to use some lagged data in the computations. When we define the lagged price change (DP1) below, we will need observations lagged twice, so we start in 1948 rather than 1947. In the published data, M is the money stock at the end of the year. Money in this model is the average during the year. This is approximated by averaging the year-end value and the value at the end of the previous year.

```
DP = LOG(P/P(-1)) ;
DP1 = DP(-1) ;
```

DP1 is the lagged rate of inflation.

```
SMPL 1949,1975 ;  DM = LOG(M/M(-1)) ;
```

DM is the rate of change of the money stock. Multiplying DM by 100 would give the percent rate of change. Because the first observation of the money stock was zero, we need to start in 1949 to obtain good values of DM -- remember that M was redefined as a two year moving average earlier.

```
LGNPCUR = LOG(GNP*P) ;
```

LGNPCUR is the log of GNP in current prices.

With the data prepared, we can begin estimation of the equations of the illustrative model by ordinary least squares and two-stage least squares. For the consumption equation, the TSP command for regression is the following:

```
OLSQ CONS C,GNP ;
```

OLSQ computes the simple regression of CONS on GNP with an intercept. Note how the intercept is specified: C is a built-in series in TSP consisting entirely of 1's. C is used mainly to specify a constant or intercept in regressions, but may be used anywhere that a series is valid in any TSP statement. If C does not appear in the list of variables for a regression, the regression will be estimated without an intercept; that is, the regression line will be forced through the origin. In this respect, TSP is different from most regression programs.

The commands

```
FRML MPC GNP -1;
ANALYZ MPC;
```

define a hypothesis to be tested (that the coefficient of GNP is equal to one) and then test it with a t-test.

To compute two-stage least squares estimates for the simple consumption function, use:

```
LIST IVS C,G,LM,TIME;
2SLS(INST=IVS) CONS C,GNP;
```

The LIST command defines a TSP variable list (in this case the list of instruments for the model, which will be used repeatedly in 2SLS and LSQ). The INST= option in 2SLS specifies the list of instrumental variables for this equation -- note that the constant, C, must be specified as an instrument explicitly. Any exogenous variable that appears in the equation must also be in the list of instruments; in this case, C is the only such exogenous variable.

We estimate this equation with LIML (Limited Information Maximum Likelihood), which is specified like 2SLS:

```
LIML(INST=IVS) CONS C,GNP;
```

Estimation of the investment equation is next. We hypothesize that investment depends on the demand for capital services, which is the ratio of real GNP to the rental price of capital. The rental price of capital, in turn, is the sum of the rate of depreciation and the interest rate. The following statements form the appropriate variables and carry out the estimation.

```
CONST DELTA 15 ;
```

defines a constant, DELTA, interpreted as the rate of depreciation and assigned the value 15 (% per year).

```
GENR GNPN = GNP/(DELTA+R) ;
```

creates the capital demand variable, GNPN. Note that constants may appear in GENR statements (see also Section 6.1).

```
OLSQ I I(-1),GNPN ;
```

estimates the investment equation. It is followed by a nonlinear least squares estimation of the same equation as an illustrative example. However, since the equation in this version is really linear, there will be no difference in the estimates. If DELTA was specified as a parameter rather than a constant, the estimation would be truly nonlinear.

The next OLSQ and 2SLS commands estimate a constrained version of the interest-rate equation where the coefficients of LGNPCUR and LM are required to be the same in magnitude but opposite in sign. The constraint is imposed by combining them in LVELOC, the log of monetary velocity.

```
LVELOC = LGNPCUR-LM ;
OLSQ R C,LVELOC ;  2SLS(INST=IVS) R C,LVELOC ;
```

The final OLSQ and 2SLS commands estimate the inflation equation.

```
OLSQ DP DP1,LGNP,TIME,C ;
2SLS(INST=IVS) DP DP1,LGNP,TIME,C ;
```

OLSQ and 2SLS statements can appear in any order, not just in pairs as in this example. It is usually more convenient to group the estimation statements for a particular equation so that the output from the estimations appear adjacently.

Here is the TSP job for the illustrative model, which includes several commands like LSQ, FIML, ANALYZ, and SIML that will be introduced in later chapters. We have not included the full data set, but you can find the complete program and data on the web site.

```
OPTIONS CRT;
NAME ILLUS44 'Sample Run for TSP 4.4: Illustrative Model from Manual' ;
?
FREQ A;                 ? Annual frequency.
SMPL 46 75 ;            ? Sample is 1946 to 1975.
LOAD ;                  ? Read in data.
TREND TIME ;
?
?    Data Transformations.
?
P = P/100 ;  LP = LOG(P) ;
G = GOVEXP+EXPORTS ;
R = RS ;
PRINT GNP CONS I RS RL P ;
SMPL 48,75 ;
MNEW = (M+M(-1))/2 ;  M = MNEW;
DP = LOG(P/P(-1)) ;  DP1 = DP(-1) ;
SMPL 49,75 ;  DM = LOG(M/M(-1)) ;
PRINT G M DM TIME DP ;
LGNPCUR = LOG(GNP*P) ;  LM = LOG(M) ;
LVELOC = LGNPCUR-LM ;  LGNP = LOG(GNP) ;
?
? Data is now generated; Try estimation on single equations.
?
SMPL 49,75 ;
OPTIONS PLOTS ;
TITLE 'OLSQ - Ordinary Least Squares' ; PAGE ;
OLSQ CONS C,GNP ;
?
? Test the hypothesis that the marginal propensity to consume is one.
FRML MPC GNP-1 ;
ANALYZ MPC ;
?
? Form a consumption equation for model estimation later.
FORM(COEFPREF=B) CONSEQ;
TITLE 'PRINT Example from Users Guide' ; PAGE ;
PRINT B1 CONSEQ @VCOV @RES;
?
TITLE 'Engle-Granger Test for Cointegration' ; PAGE ;
SMPL 50,75 ;
DU = @RES-@RES(-1) ;
OLSQ(SILENT) DU C TIME @RES(-1) ;                  ? Regress diff. residuals on trend and lag residual.
SET TSTAT = @T(3);
CDF(DICKEYF,TREND,DF=@NOB,PRINT) TSTAT ;    ? Test with Dickey-Fuller distribution.
?
?    Plot the residuals from the consumption equation.
?
SMPL 49,75 ;
GENR CONSEQ CONSFIT ;  RES = CONS-CONSFIT ;
PLOT (MIN=-25,MAX=25,VALUES,HEADER,BAND=STANDARD,INTEGER,BMEAN)RES * ;
```

```
?
?     Estimation with correction for first order serial correlation.
?
AR1 CONS C,GNP ;
FORCST(PRINT) CONSP ;
OPTIONS NOPLOT ;
?
?     Two stage least squares.
?
LIST IVS C G LM TIME ;                                  ? Define a list of instrumental variables
?
TITLE 'INST - Instrumental Variable Estimation' ; PAGE ;
2SLS(INST=IVS) CONS C,GNP;
?
TITLE 'LIML - Limited Information Maximum Likelihood Estimation' ; PAGE ;
LIML(INST=IVS) CONS C,GNP;
PAGE;
?
?     Linear Estimation of Investment Equation.
CONST DELTA 15 ;  GNPN = GNP/(DELTA+R) ;
OLSQ I I(-1),GNPN ;
FORCST(PRINT,DYNAM,DEPVAR = I) IFIT ;
?
?     'Nonlinear' Estimation of the same equation by OLS and 2SLS.
FRML INVEQ I = LAMBDA*I(-1) + ALPHA*GNP/(DELTA+R) ;
PARAM LAMBDA .05 ALPHA .2 ;
LSQ(PRINT) INVEQ ;
LSQ(INST=IVS) INVEQ ;
?
?      Linear Estimation of Interest Rate Equation.
OLSQ R C,LVELOC ;
2SLS(INST=IVS) R C,LVELOC ;
PARAM D F;
UNMAKE @COEF D F;                                       ? Starting values for INTRSTEQ below.
?
?       Linear Estimation of Price Change Equation.
OLSQ DP DP1 LGNP TIME C ;
2SLS(INST=IVS) DP DP1,LGNP,TIME,C ;
PARAM PSI PHI TREND P0 ;                                ? Parameters of price change eqn.
UNMAKE @COEF PSI PHI TREND P0 ;                         ? Set starting values for PRICEQ below.
?
?     These are the equations of the simultaneous equations model.
?
IDENT GNPID GNP-CONS-I-G ;
?      The next equation was made by FORM earlier in the run.
? FRML CONSEQ CONS = B0 + B1*GNP ;
FRML INTRSTEQ R = D + F*(LOG(GNP)+LP-LM) ;
FRML PRICEQ LP = LP(-1) + PSI*(LP(-1)-LP(-2)) + PHI*LOG(GNP) + TREND*TIME +P0 ;
?
?     Estimate by multivariate regression, assuming contemporaneous correlation of the residuals.
?
LSQ(STEP = CEAB) CONSEQ INVEQ INTRSTEQ PRICEQ ;
PRINT @FIT ;
?
?     Estimate by nonlinear three stage least squares.
?
TITLE '3SLS - Three Stage Least Squares' ; PAGE ;
```

```
3SLS(PRINT,INST=IVS) CONSEQ INVEQ INTRSTEQ PRICEQ ;
?
?    Estimate by Full Information Maximum Likelihood.
?
FIML(ENDOG=(GNP,CONS,I,R,LP)) GNPID CONSEQ INVEQ INTRSTEQ PRICEQ ;
?
?    Test some nonlinear hypotheses about Longrun coefficients.
?
FRML LR1 ALPHALR = ALPHA/(1.-LAMBDA) ;
FRML LR2 PHILR = PHI/(1.-PSI) ;
PAGE;
ANALYZ LR1 LR2 ;
PAGE;
?
?   Simulate the model over the second half of the sample and plot the results.
?
SMPL 66 75 ;
SIML (PRNSIM,PRNDAT,TAG = S,ENDOG = (GNP,CONS,I,R,LP))
        CONSEQ INVEQ INTRSTEQ GNPID PRICEQ ;
PLOT (RESTORE, MAX=1500, MIN=500, LINES=(1000)) GNP G GNPS H CONS C CONSS D ;
PLOTS ;
ACTFIT R RS ;
END ;
?
?   START OF THE DATA SECTION
?
NOPRINT;
SMPL 46 75 ;
LOAD GNP CONS I ;                                   ? GNP, CONSUMPTION, INVESTMENT
475.7 301.4 71
468.3 306.2 70.1
487.7 312.8 82.3
... data continues...
1186.4 766.6 138.9
;
LOAD EXPORTS ;                                      ? EXPORTS
11.6 16.6 8.5 8.8 4.0 7.4 4.9 2 4.5 4.7 7.3 8.9 3.5 .9
5.5 6.7 5.8 7.3 10.9 8.2 4.3 3.5 -.4 -1.3 1.4 -.6 -3.3
7.2 16.6 23.5  ;

LOAD GOVEXP ;                                       ? GOVERNMENT EXPENDITURES
91.8 75.4 84.1 96.2 97.7 132.7 159.5 170.0 154.9 150.9
152.4 160.1 169.3 170.7 172.9 182.8 193.1 197.6 202.7 209.6
229.3 248.3 259.2 256.7 250.2 249.4 253.1 252.5 254.3 257.4
 ;

...reads in M, P, RS -- full data set available on the web site...

LOAD RL ;                                           ? LONG TERM INTEREST RATE.
2.53 2.61 2.82 2.66 2.62 2.86 2.96 3.2 2.9
3.06 3.36 3.89 3.79 4.38 4.41 4.35 4.33 4.26 4.4
4.49 5.13 5.51 6.18 7.03 8.04 7.39 7.21 7.44 8.57 8.83
 ;
END ;
```