SOFTWARE PACKAGE FOR ECONOMIC MODELING

Morris Norman

RR-77-21
November 1977

International Institute for Applied Systems Analysis
A-2361 Laxenburg, Austria

Preface

Several major activities at IIASA--including the Energy Systems Program, the Food and Agriculture Program, and the Water Tasks--concern the choice among economic policy alternatives for technology, conservation, and supply. One of the Tasks of the System and Decision Sciences Area is to prepare macroeconomic models that can be used to evaluate long-term options in industrialized economies. Their implementation on the computer requires programs that can manage data, estimate parameters statistically, and simulate nonlinear econometric models. This paper develops such programs and applies them to a small potential GNP model for the USA.

Acknowledgments

The author is grateful to Diane Wells, who prepared this manual with the NROFF text processor; to Jim Curry, who helped with the NROFF macros; and to Maria Sachs, who edited the manual. Without their help this manual would never have been. The author is responsible for all its shortcomings.

Summary

Several major world problems--including energy and food problems--concern the choice among economic policy alternatives. The desired production of energy, food and other commodities may not be feasible. The aggregate input requirements may exceed the available materials, manpower, capital, and so forth. In the light of these needs it is necessary to develop macroeconomic models for aggregated demand, which incorporate the constraints of the economic system. These models should be flexible so that the economic impact of various technological possibilities can be considered. Questions such as capital shortage, the ability of an economy to produce the necessary capital equipment to change over to new technologies, etc., can be studied with the help of macroeconomic models. These models are useful for economic forecasting, but are also a sophisticated device for observing--in summarized equation form--the past operation of an economy.

To develop a macroeconomic model we need a computer system to handle the various computation problems. The approach used is to break the tasks of simulating an econometric model into four distinct areas. For each area a separate program was written. Basically these areas are data management, parameter estimation, simulation input specifications, and model simulation.

A software package is developed to take advantage of the PDP 11* interactive mode of operation. The user is provided with appropriate prompts at his terminal. These prompts contain mnemonics for the input information required by the programs. The package includes the necessary programs to manage a random time series data file, BANK; to estimate regression parameters, AUTO; to create an input file for the simulations, SIMDAT; and to simulate linear or nonlinear econometric models, SIM.

Moreover, a small potential GNP model for the USA, with annual observations from 1947 to 1975, is developed.

---

*The program package is almost computer independent. With slight modification the package will run in batch mode on a CDC or IBM computer.

The following study includes the necessary write-up, descriptions and manuals for developing econometric models. The programs comprising the software package are briefly described in the following.

A DATA MANAGEMENT PROGRAM: BANK

BANK is a computer program designed to create and maintain a data bank of time series data for econometric research. The data series are stored on a random file with one series per record. For every series we have to specify the starting year, the number of observations per year and the starting period. This enables the user to store simultaneously series with different starting dates or different numbers of observations per year. The program in standard form is designed to handle up to 400 data series with a maximum of 100 of observations per series, but the user could adjust the DIMENSION statements to suit his particular needs.

The program provides the following facilities:

- Several alternate procedures can be used for creating and/or updating data series. Basically any existing data set can be placed automatically on the data bank if it can be read into a matrix. Once a data series is on the data bank a simple procedure is provided to extend or update the elements of the series.

- The entire data bank or any subset of the variables specified by the user can be listed. Graphs of the variables can also be obtained. A table of variable names and logical record numbers accompanies the list, which can be printed in both numerical an alphabetical order.

- A Fortran direct-access input/output system is used to obtain any series of the data bank.

- The user can maintain on the random file the documentation for the data series. The documentation file should contain definitions of all the variables and their sources.

A PARAMETER ESTIMATION PROGRAM: AUTO

AUTO is an extremely efficient ordinary least squares (OLS) regression program capable of dealing with large data files and with entire systems of equations. AUTO provides estimation of an equation in terms of moments around the origin and can be used to estimate the first and second order autoregressive scheme of error terms. Also, it calculates finite polymonial distributed lags, and allows the user to estimate distributed lags using one variable for the current values and a different one for the lagged values. Many options for data manipulation make the program very flexible; for example, AUTO has an option to store label and equation lines to allow the user to run the same regression on several sets of data without duplicating these lines, or to store and use the data with new equation lines and a different sample period. Moreover, the input lines needed to run a set of simple regressions are minimal.

A SIMULATION PROGRAM FOR ECONOMETRIC MODELS: SIM

SIM is designed to simulate linear or nonlinear econometric models. SIM was originally constructed to solve the models of Project LINK where speed of solution and minimum core usage were essential. The program is completely flexible for specification of lag lengths, time periods per year, order of endogenous variables and exogenous variables, and solution time horizon. The necessary Fortran code to represent the equations is supplied by the user. The equations are divided into a predetermined part and a simultaneous part. A Gauss-Seidel iteration procedure is used to evaluate the system of equations. SIM uses an input file produced by SIMDAT.

THE SIMULATION INPUT FILE PROGRAM: SIMDAT

SIMDAT produces the necessary input file, and SIM performs the actual simulations. The reason for dividing the simulation program into two modules is to minimize core requirements for SIM. SIMDAT handles the creation of a file containing the regression coefficients, the labels for the variables, and the data series arranged by observations. It can also be used to update and change the basic data requirements for SIM.

THE RR ROUTINES

The RR routines are a package of six subroutines designed to simulate CDC MASS STORAGE INPUT/OUTPUT. They allow the user to write and read random records without keeping track of their exact location: he can therefore write his records in any order. The six subroutines are the following: OPENMS to open an RR file, WRITMS to transmit data from central memory to mass storage, READMS to transmit data from mass storage to central memory, CLOSEMS to write the index from the central memory to the file, OPENIN to read a subindex from mass storage into an area specified by the program, and CLOSEIN to write a subindex from central memory to the file.

A POTENTIAL GNP MODEL FOR THE USA

A small national potential GNP macroeconomic model is developed that can be used in the analysis of long-term alternatives for industrialized economies. There are three main components of aggregate demand: consumption, investment, and government purchase of goods and services. The purpose of the potential GNP model is to produce a consistent set of estimates for these variables under different assumptions about the exogenous variables (scenarios). The government's direct effect on total demand through taxes and transfer payments is estimated. These taxes and transfer functions provide a means to change the proportion of consumption to GNP. The limiting factors in an economy are systematically examined by the use of production theory. The model was estimated with annual observations from 1947 to 1975.

MANUAL DOCUMENTATION FOR THE DIFFERENT PROGRAMS

In this section we describe how to run off new copies of the manuals for the programs BANK, AUTO, SIM, and SIMDAT, and the RR ROUTINES using the NROFF text processor.

Contents

BANK

I	Introduction	1
II	Formulation	1
III	Program	1
IV	Size Specification	4
V	Input Lines	5
VI	Files used by BANK	16
VII	Examples	16

RR ROUTINES

I	Introduction	29
II	Formulation	29
III	Subroutine OPENMS	30
IV	Subroutine WRITMS	30
V	Subroutine READMS	31
VI	Subroutine CLOSEMS	31
VII	Subroutine OPENIN	31
VIII	Subroutine CLOSEIN	32
IX	USING Subindices	32
X	Obtaining the RR Routine	34

AUTO

I	Introduction	35
II	Formulation	35
III	Subroutine TRAN	42
IV	Files Used by AUTO	46
V	Input Lines	46
VI	Examples	56

SIMDAT

I	Introduction	78
II	Formulation	78
III	The Lag Vector (d Vector)	79
IV	Files Used by SIMDAT	79
V	Input Lines	79
VI	Example	82

SIM

I	Introduction	87
II	Formulation	87
III	Program	88
IV	Coding a Model	90
V	Checking the Equations for Errors	107
VI	Arrangement of the Equations	107
VII	Files Used by SIM	108
VIII	Input Lines	108
IX	Examples	112

MANUAL DOCUMENTATION FOR AUTO, BANK, SIM,
SIMDAT, AND RR ROUTINES (by Diane Wells) 115

POTENTIAL GNP MODEL FOR THE USA 118

BANK
October 1977 Version for IIASA's pdp11

I INTRODUCTION

BANK* was designed to create and maintain a data bank for econometric research. The data bank system is compatible with the regression programs, AUTO and ECON. For the simulation program, SIM, a small interface program SIMDAT, is necessary. This program reads the data bank and generates a file to meet the input requirements of SIM.

II FORMULATION

The data series are stored on a random file with one series per record. Each series is identified by its logical record number (num) and its name (name). If either is specified incorrectly the series cannot be updated. With every series the number of observations (no) the starting year (iy), the number of observations per year (ip), and the starting period (ib) are saved on its logical record. This feature allows series with different starting dates and with a different number of observations per year to be stored simultaneously on the same data bank. A Fortran direct-access input/output system** is used so that any series can be obtained without reading the entire data bank. The user should adjust the DIMENSION statements to suit his particular needs. The program in standard form was designed to handle up to 400 data series with a maximum of 100 observations per series. These parameters can easily be changed (see below).

The user can also maintain on the random file the documentation for the data series. The program manages two indices, one for the data and one for the documentation. All information is saved on one random file.

III PROGRAM

The computer program consists of a main program which has eight subroutines: UPLIST, UPOUT, UPTRAN, DATRAN, CHAR, VAR, WRITRR, UPFIND and the RR ROUTINES. The purpose of the main program and the subroutines will be described in order.

---

*A batch version for Control Data 6000 series and for IBM 370 series is also available.
**BANK uses a random record library written for the pdp11 to simulate CDC's RR calls (see section RR ROUTINES).

A. main.f

The main program handles normal creations and updates for the series on the data bank. There are three alternate forms for reading data into the computer. One method uses the update options. It consists of reading an update line and then the data is typed with the default option of four observations per line for quarterly data and five observations per line for annual data. The second method uses SUBROUTINE UPTRAN to read a matrix of data. It consists of reading an UPTRAN PARAMETER LINE, a FORMAT LINE and then the data from a trandata file. The third method uses a card image file produced from a previous data bank.

Documentation is allowed for any series on the data bank. It is not necessary to have documentation for all series or it can be entirely omitted. The documentation can be placed on the random file by using the update options. It is very important to document your data series before you forget definitions, units, sources, etc. Use the general heading of documation to define abbreviations for codes used in the documentation.

B. uplist.f

SUBROUTINE UPLIST lists the data bank. The entire data bank or any subset up to 15 (maxc) variables which the user specifies can be listed. Only 80 (maxl) observations can be listed in one run. All observation can be listed in a series of passes. A table of variable names and logical record numbers accompanies the list. This list can be printed in both numerical and alphabetical order. The documentation is listed in the order of the final table of variable names.

C. upout.f

SUBROUTINE UPOUT prints matrices in readable form.

D. uptran.f

SUBROUTINE UPTRAN allows the user to read the data from a trandata file into the Z matrix. Any column of the Z matrix can be used for update or for the creation of new series. A series is placed on the data bank with a TRANSFER LINE. Data from the trandata file can be arranged by observation or by variable. Data can also be read from the data bank. The data are stored in the Z matrix which has 36 rows with 30 columns. A series is stored in a column. Transformations may be performed on the data if the user writes the necessary Fortran statements in SUBROUTINE DATRAN. It is advisable for the user to make a dummy run and check the listing of the Z matrix before the series are actually placed on the data bank. With the correct format statement, SUBROUTINE UPTRAN can read any of the data input files that are used with the old versions of AUTO or SIM. Using SUBROUTINE UPTRAN is the easiest way to place large quantities of data on the data bank.

E. datran.f

SUBROUTINE DATRAN is written by the user to transform the variables in the Z matrix defined by SUBROUTINE UPTRAN. SUBROUTINE DATRAN allows the user to transform original data into new variables. SUBROUTINE UPTRAN can then place the transformed data in the data bank. It is useful to store transformations which are used repeatedly in regression runs in the data bank. Transformations should be checked before they are placed in the data bank.

SUBROUTINE DATRAN can also be called during the listing of the data bank. The listing options placed the data in the Z matrix. SUBROUTINE DATRAN can then be used to transform the orginal data into new variables before they are listed (or graphed).

The user must insert the necessary Fortran statements into SUBROUTINE DATRAN to calculate the desired transformed variables. The following example illustrates one transformation. Variable 3 is defined equal to variable 1 divided by variable 2. no will take on the maximum value given it on the UPTRAN PARAMETER LINE(S).

```
      subroutine datran*
      common ns,nd,nextr,maxs,it,lis,nol,iy2,ip2,ib2,maxc,
     1       nl,no,mn,ipa,maxns,n6,ldo,ml,m2,maxl,db,tr
     2       /data/z(36,30),label(30),loc(30),lgr
     3       /index/ims(801)
     4       /subin/ixs(400),ixd(401)
      double precision name2,iabel,name,label
      integer db,tr
      mn=3
      do 1 i=nl,no
1     z(i,3)=z(i,1)/z(i,2)
      return
      end
```

If the user changes the number of variables in the Z

*The size of the Z matrix in this example is 36 by 30 (maxl by maxc). The label common /data/ must be the same size in all routines. The values for maxl and maxc are printed on the output file.

matrix, mn must be set equal to the new number. One new label will be read for the variable in column 3. The variables in the Z matrix can be listed before and after SUBROUTINE DATRAN. The user can place any variable from the Z matrix on the data bank file by using the appropriate TRANSFER LINE.

When the user writes his own transformation the SUBROUTINE DATRAN must be compiled and new bank.o file constructed.

F. char.f

SUBROUTINE CHAR is used to rearrange the bytes of a double precision word. SUBROUTINE CHAR is used by SUBROUTINE UPLIST to rearrange the labels of the variables for the alphabetical sort. After the sort the labels must be rearranged to their original position. The pdp11 does not store alphabetical information (letters) in a sequence.

G. var.f

SUBROUTINE VAR is used by SUBROUTINE UPLIST to print the numerical and alphabetical lists of logical numbers and names for the variables.

H. writrr.f

SUBROUTINE WRITRR is used to assign the random records of the subindices to the main index. A new record or an updated record larger than the original one is assigned to the next available record, which is at the end of the random file. On an IBM computer a logical record may span more than one record.

I. upfind.f

SUBROUTINE UPFIND retrieves series from the data bank and places them in the correct position in the Z matrix for SUBROUTINE UPTRAN.

IV SIZE SPECIFICATION

The standard size of the data bank is 400 variables with a maximum length of 100 observations. The size of the Z matrix used in SUBROUTINES DATRAN, UPFIND, UPLIST, UPOUT, and UPTRAN is 36 by 30. The program size is very close to the limit of the pdp11. The number of series can be increased to 800 by changing

```
     1       /subindex/ims(400),ixd(401)
```

to

```
     1       /subindex/ims(800),ixd(801)
```

and maxns to 800. The label common must be changed in every subroutine and if the user wishes to increase the number of series above 800 then the RR routines must be recompiled.

The size of the Z matrix can be changed provided the total number of elements remains about the same. The Z matrix can be modified to 100 by 10 by changing

```
      1       /data/z(36,30),label(30),loc(30),lgr
```

to

```
      1       /data/z(100,10),label(10),loc(10),lgr
```

in the above mentioned subroutines. Label common /data/ must also be changed in the main program from

```
      1       /data/num3,nam3,nn,z(410),space(664),lgr
```

to

```
      1       /data/num3,nam3,nn,z(410),space(404),lgr
```

The parameters maxs and maxc in the main program must be changed to maxs=100 and maxc=10.

V <u>INPUT LINES</u>

The program writes prompts to the user. The user types in the appropriate information under the mnemonics, right adjusted. This section gives the prompts and their meaning.

A. <u>PARAMETER LINE</u>

up lis nol iy2 ip2 ib2 ldo it ire ipa npu nad ncl ltr lgr

up	=	0	Normal.
	=	1	To update the data bank using UPDATE LINE(S).
	=	2	To update the data bank using DATA CORRECTION LINE(S).
lis	=	0	Normal.
	=		Number of series to be listed (up to maxc, the number of columms in the Z matrix). LIST LOCATION LINE(S) must be included. The program will continue to read location lines until "control d" is typed. (See LIST LOCATION LINE(S) below for more details.)

	=	101	To list all the series in the data bank with ip2 observations per year. If the data bank contains series which have other than ip2 observations per year ipa must be set equal to 1, or these series will not be listed. The program will then use ip=1, 4, and 12.
	=	102	To list the names of the series in the data bank. Numerical and alphabetical lists are given.
	=	103	Same as 101 except the names of the series are listed only numerically. The alphabetical listing causes a sort to be performed, and for large data systems (>2000) this takes a lot of computer time.
	=	104	To list the names of series numerically.
nol	=		Number of observations to be listed.
	=	0	The no from the first series in the Z matrix will be used.
iy2	=		Starting year for list, e.g., 47.
	=	0	The iy from the first series in the Z matrix will be used.
	=	-1	To ignore the starting year. All series start in the first row of the Z matrix (adjusted for their ib).
ip2	=		Number of observations per year. Default value is 1.
ib2	=		Starting period of list. Default value is 1.
ldo	=	0	No documentation is listed.
	≠	0	Documentation is listed after data.
it	=	0	Normal.
	≠	0	To call SUBROUTINE UPTRAN. (Reads an UPTRAN PARAMETER LINE.)
ire	=	0	Normal.
	=	1	Only documentation is written with the punch option.

	=	2	To recreate a data bank from bcd card images from the npu option (can also add more card images from another bank, use nad to change num).
	=	3	Same as 2 except num is not used and series are placed on the data bank in the order of their appearance on file.
ipa	=	Ø	The list will include only those series with ip2 observations per year. Any series with its ip≠ip2 will not be listed.
	=	1	To list all series. Any series with an ip≠ip2 will be listed separately at the end.
npu	=	Ø	Normal.
	=		Number of series to be written as bcd card images. PUNCH LOCATION LINES must be included. (See PUNCH LOCATION LINE(S) below for more details.) npu<21 (always).
	=	999	To write the entire data bank on bcd card images.
nad	=	Ø	Normal.
	=		Correction factor for logical numbers on a databank file when data from another file is added to the end of an existing file using the ire option.
ncl	=	Ø	Normal.
	=		Number of columns to be used in the Z matrix (<maxc).
ltr	=	Ø	Normal.
	>	Ø	To call SUBROUTINE DATRAN before listing the data in the Z matrix.
lgr	=	Ø	Normal.
	=	1	To graph series with listing.
		Note:	Program reads databank file if ire>1.

B. UPDATE LINE(S) (IF up = 1)

num name iy ib new no ip

num	=		Logical record number of the variable in the data bank.
name	=		Name of the series, left adjusted. Default value is name of the last series that was updated.
	=	$$$	To transfer program to the UPTRAN PARAMETER LINE.
iy	=		Starting year for data read, e.g., 52. Default value is iy2 of PARAMETER LINE.
ib	=		Starting period. Default value is ib2 of PARAMETER LINE.
new	=	0	To update an old series or create one on an empty record; requires DATA LINE(S).
	=	444	To transfer program to DATA CORRECTION LINE(S).
	=	555	To update or create a heading for documentation; requires HEADING LINE(S).
	=	666	To update or create documentation; requires DOCUMENTATION LINE(S). (An alternative method is for the user to edit the documentation part of the punch file. With npu=999 and ire=1 only documentation is written to the punch file. With ire=2 the edited file can placed back on the data bank.)
	=	777	To change the name of a series requires a NEW NAME LINE. It will also change the name on the documentation if any.
	=	888	To create a new series over an old series; requires DATA LINE(S).
	=	999	To change the beginning date at a series to an earlier period. A series can not be updated before its beginning date. No additional lines are read for this option. It will require no= -1 if series is longer than 100. Last periods will then be dropped.

no = Number of observations to be read. Default values are as follows:

= 5* When ip=1 and iy≠Ø.

= ip* When ip>1 and iy≠Ø.

= no1* When iy=Ø and it is a new series.

*Note: With a single input record or on the last input record of a group all trailing zeros are dropped and no is reduced. To place a zero over a number with a single input record no must specified.

= 9 For documentation or heading.

ip = Number of observations per year. Default value is ip2 of PARAMETER LINE(S).

C. INPUT LINE(S) FOR UPDATE OPTIONS

1. DATA LINE(S) (IF new = Ø OR 888)

Type data separated by commas with:

5 numbers per line for yearly data,
4 numbers per line for quarterly,
6 numbers per line if ip>6 and two or more lines are needed.

Note: If a read error occurs retype the same line. "Control d" (eof) will transfer the program back to the UPDATE LINE(S) if you realize you are doing the wrong thing.

2. HEADING LINE(S) (IF new = 555)

The program will read 9 words per line and information is stored in a vector. Part of the vector can be changed if iy is given a value. The (z(i),i=iy,no) is updated. "Control d" (eof) will transfer the program to UPDATE LINE(S), without changing the information. This option allows the user to look at the information and then reset the iy and no values. The name of the series does not have to be retyped.

3. DOCUMENTATION (IF new = 666)

Heading and documentation are handled in the same manner except that the series must exist before documentation will be accepted. A "control d" will transfer the program to the UPDATE LINE(S).

4. NEW NAME LINE (IF new = 777)

NAME = New name for series.

Note: The last line following all the UPDATE LINES must contain $$$ for the name to transfer the UPTRAN PARAMETER LINE.

D. DATA CORRECTION LINE(S) (IF up > 1)

num name iy ib data

num = Logical record number of the variable in the data bank.

name = Name of the series, left adjusted. Default value is name of the last series to be updated.

= $$$ To transfer program to UPTRAN PARAMETER LINE.

iy = Starting year for data read.

ib = Starting period for data read. Default value is ib2 of the PARAMETER LINE.

data = Data points to be added to an existing series. Up to six numbers placed in fieids of 10 with format(6f10.1)

E. UPTRAN PARAMETER LINE (IF it ≠ 0)

nl no ml mn db pa nsp aps cf tr iy ip ib li

nl	=		Starting row in the Z matrix for data. Default value is 1.
no	=		Ending row in the Z matrix for data. no-nl+1 equals number of observations. Default value is nol.
ml	=		Starting column in the Z matrix for data.
mn	=		Last column of the Z matrix (mn-ml+1 equals number of variables).
db	=	0	To pull data from the data bank with KSET GRABBER LINE(S).
	=	1	To pull data from an existing series on the data bank with GRABBER LINE(S).
	=	2	To read data by observation with a variable format. Program transfers to the FORMAT LINE(S).
	=	3	To read data by variable with a variable format. Program transfers to the FORMAT LINE(S).
	=	4	Same as 3 except the format is read only with the first set. The label is read as the first word of each series. For example: (a8/(6f10.1)).
pa	=	0	Normal.
	=	1	To read another UPTRAN PARAMETER LINE after data is stored in the Z matrix.
	=	2	To read another UPTRAN PARAMETER LINE after calling DATRAN and after finishing the TRANSFER LINE(S).
nsp	=	0	Normal.
	=		Number of series to be placed on the data bank. nsp TRANSFER LINE(S) will be read.
aps	=	0	Normal.
	=	1	To automatically place the first nsp columns of the Z matrix on the data bank with labels of the columns used as the names.
	=	n	Same as 1 except starts with the n^{th} column.

cf	=		Correction factor when aps option is used. The series will be numbered 1+cf to nspy+cf. Default value is ns+1: the present number of series on the datal file plus 1.
tr	=	Ø	Normal.
	≠	Ø	To call SUBROUTINE DATRAN.
iy	=		Starting year for row one of the Z matrix. Default value is iy2. (For weekly data the default value is increased by one year after each series. It is assumed that each year will be a separate series.)
ip	=		Number of observations per year for the Z matrix. Default value is ip2.
ib	=		Starting period for row one of the Z matrix. Default value is ib2.
li	=	Ø	Normal.
	=	1	To list orginal data before SUBROUTINE DATRAN is called.

F. INPUT LINES FOR UPTRAN OPTIONS

1. KSET GRABBER LINES(S) (IF db = Ø)

The logical record numbers for the variables to be placed in the Z matrix are typed in fields of 4. The numbers can be typed 2Ø per line separated by a comma. If the numbers are in a sequence only the first number should be typed. If however, you realize the line was incorrect, "control d" will transfer the program back to the UPTRAN PARAMETER LINE.

2. GRABBER LINE(S) (IF db = 1)

num name (there are mn-ml+1 lines).

num	=	Logical record number for the variables.
name	=	Name of the variables as recorded in the data bank.

Note: The program will go to the UPTRAN PARAMETER LINE if pa=1 or to the ADDITIONAL LABEL LINE(S) if pa=Ø or 2.

3. FORMAT LINE(S) (IF db = 2, 3 OR 4)

FORMAT = For example: (6F1Ø.3). The program will read the data from unit 9 which is assigned to trandata file. The trandata file must have the data organized by observation if db=2 or by variable if db=3 or 4 (either a row or a column of the Z matrix is read and there are mn-ml+1 of them). Following the data lines are the labels for the data with one label typed per line in a left adjusted field.

With db=4 the format is read only with the first data set. Also labels are read as the first word of each series.

(Maximum size of the Z matrix is maxl observation with maxc variables.)

4. ADDITIONAL LABEL LINE(S)

The labels are typed one per line, left adjusted. If the number of variables in the Z matrix is increased in SUBROUTINE DATRAN, additional labels must be read. The program will automatically read labels for the additional variables. However, the user can override the number to be read. The program reads labels for the variables ml through mn; ml equals the number of variables before SUBROUTINE DATRAN plus 1, and mn, which the user must set in SUBROUTINE DATRAN, equals the total number of variables. If ml is set equal to 1 in SUBROUTINE DATRAN an entire new set of labels will be read. If ml is set greater than mn, the labels will not be read.

5. TRANSFER LINE(S) (IF nsp > Ø AND asp = Ø)

ic num name nl no new iy ip ib (there are nsp lines)

ic	=		Column in the Z matrix where the variable is stored.
	=		"Control d" (eof) to end TRANSFER LINE(S) prematurely.
num	=		Logical record number of the variable. Default value is ns+1.
name	=		Name of the variable, left adjusted. Default value is the label from the ic column.
nl	=		Row of observation of year iy and period ib. Default value is 1.
no	=		Last row of data. no-nl+1 is the number of observations taken. Default value is no from the UPTRAN PARAMETER LINE.
new	=	Ø	If the variable is already on the data bank, or series is empty.
	=	888	To create a new series over an old series.
iy	=		Starting year of the series. Default value is iy2.
ip	=		Number of periods per year. Default value is ip2.
ib	=		Starting period of the series. Default value is ib2.
	Note:		If pa=2 on the UPTRAN PARAMETER LINE the program will now read another UPTRAN PARAMETER LINE.

G. LIST LOCATION LINE(S) (IF Ø < lis < 1Ø1)

The user has the option to list part of the data bank by typing lis<1Ø1 on PARAMETER LINE. The user must type logical record numbers for the series that are to be listed. A maximum of 2Ø can be typed at one time. After the first lis are listed on the output file, the program will require another LIST LOCATION LINE(S) until a "Control d" (eof) is typed. If the numbers are in a sequence, then only the first number should be typed. Up to maxc can be listed in this case.

H. PUNCH LOCATION LINE(S)

This option is the same as the list option except that the series are written to the punch file as bcd card images. All leading and trailing zeros are removed and the parameters are corrected accordingly. This option can be used, and should be used, to compress a file. First the punch file is moved to databank file and datal file deleted. Second, the data bank (datal file) is recreated. This option can be used to reorder a data bank if necessary. However, if the user simply wants to compress the data file, npu=999 is used and the PUNCH LOCATION LINE(S) are not required to provide a punch file.

VI FILES USED BY BANK

FILE	NAME	USE
1	datal	As the random record file (RR) for the data bank.
4	databank	To recreate an RR file from the bcd card image file produced by the punch option.
5	terminal	To type information for the program.
6	terminal	To receive prompts and other information from the program.
7	punch	To create a bcd card image file of the RR file with all leading and trailing zeros suppressed. This option is useful for taking your data to other computer centers.
8	output	For listing of the data and documentation. After the program is completed this file can be listed on the line printer.
9	trandata	For input data for SUBROUTINE UPTRAN.
	Warning:	Since it is possible to destroy an RR file the user should use the punch option and always have a bcd copy of his data bank on his "fs" tape. The punch file must be moved to the databank file before the program can read the file to recreate the data bank. This procedure will also remove any unused space from the RR file. The user should always have a copy of datal on his "fs" tape. Once a file has been recreated and checked, the old files should be removed.

VII EXAMPLES

<u>The user must keep a current copy of his datal file on his "fs" tape</u>. It is very easy to destroy an RR file so the computer can no longer read it. This can happen when an error occurs and the random indices are not written to the file. The user can protect himself against such an error by having a spare copy of the datal file on his "fs" tape or by making a temporary duplicate copy of datal. This is done by typing (after a %) cp datal exdatal. Be sure to remove exdatal after you are certain that the updates work. If the updates failed and you can no longer open your RR file, type

(after the %) cp exdatal datal. This will put you back at the beginning and you can try again.

To check a data bank you must first have the %. You then type bank.o and list the last few series. If you have destroyed the datal file you will not be able to list your data series.

In the following examples the user types only the lines marked with a T. The program echoes back the lines marked with an E. The echo lines have the default values filled in. In this set of examples, a data bank is created and updated several times.

When you are finished and have thoroughly checked your data bank, type (after a %) rm exdatal.

A. EXAMPLE 1: ADDING DATA FROM A TRANDATA FILE

This is the easiest way to add large quantities of data. The data file can be typed by anyone who knows how to produce a file, however, care should be taken in keeping the file formats consistent. This will greatly facilitate placing the data series on the data bank.

1. LISTING OF TERMINAL INPUT AND OUTPUT LINES

In this example two series are read from the trandata file. The labels for the series are typed at the bottom of the trandata file, one per line. The user has used the automatic option to place the first two series from the Z matrix to the data bank. Only four lines were typed: 1T, 9T, 12T, and 15T. Line 2 says a datal file was created. Line 82 says the file had 3 random records: two records for data series and one record for the data bank parameters.

```
 1T   % bank.o
 2     file named datal   was created
 3      new index for    1 with file name  datal
 4     record number     1 for unit     1 was empty
 5     bank by morris norman
 6     iiasa version february 1977
 7     maxs=   36 maxns=  400 maxc=   30 maxl=  100
 8      up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
 9T               30  46              1
10E      0   0  30  46   1   1   0   1   0   0   0   0
11     nl no ml mn db pa nsp aps cf tr iy ip ib
12T                2  2      2   1
13E      1 30  1  2  2  0   2   1  1  0 46  1  1
14    type in format
15T   (10x,2f11.3)
16E    variable format was(10x,2f11.3)
```

```
reads from trandata file
  1    1 gnp$72      1  30    0  46    1    1
 series was updated
 1  1946- 1        477.600
 2  1947- 1        468.300
 3  1948- 1        487.700
 4  1949- 1        490.700
 5  1950- 1        533.500
 6  1951- 1        576.500
 7  1952- 1        598.500
 8  1953- 1        621.800
 9  1954- 1        613.700
10  1955- 1        654.800
11  1956- 1        668.800
12  1957- 1        680.900
13  1958- 1        679.500
14  1959- 1        720.400
15  1960- 1        736.800
16  1961- 1        755.300
17  1962- 1        799.100
18  1963- 1        830.700
19  1964- 1        874.400
20  1965- 1        925.900
21  1966- 1        981.000
22  1967- 1       1007.700
23  1968- 1       1051.800
24  1969- 1       1078.800
25  1970- 1       1075.300
26  1971- 1       1107.500
27  1972- 1       1171.100
28  1973- 1       1233.400
29  1974- 1       1210.700
30  1975- 1       1186.400
  2    2 c$72        1  30    0  46    1    1
series was updated
 1  1946- 1        301.400
 2  1947- 1        306.200
 3  1948- 1        312.800
 4  1949- 1        320.000
 5  1950- 1        338.100
 6  1951- 1        342.300
 7  1952- 1        350.900
 8  1953- 1        364.200
 9  1954- 1        370.900
10  1955- 1        395.100
11  1956- 1        406.300
12  1957- 1        414.700
13  1958- 1        419.000
14  1959- 1        441.500
15  1960- 1        453.000
16  1961- 1        462.200
17  1962- 1        482.900
```

```
69     18  1963- 1      501.400
70     19  1964- 1      528.700
71     20  1965- 1      558.100
72     21  1966- 1      586.100
73     22  1967- 1      603.200
74     23  1968- 1      633.400
75     24  1969- 1      655.400
76     25  1970- 1      668.900
77     26  1971- 1      691.000
78     27  1972- 1      733.000
79     28  1973- 1      766.300
80     29  1974- 1      759.800
81     30  1975- 1      766.600
82     closed index for  1 with    3 words
83
84     closed sub index on lun    1 with      2 words
85T   % bye
```

2. <u>LISTING OF TRANDATA FILE</u>

This file must be prepared by the user before the example can be run. The fields can be separated by commas, and the first two columns are not necessary; however, these columns are helpful for checking the input data. The trailing zeros after the decimal are also not necessary.

Keep in mind: the format statement is difficult to make from a carelessly thrown together trandata file; therefore, much effort should be taken in providing a neat trandata file from the start. Line 15T was the format for this data. For any particular trandata file the series must start with the same data. For series with different starting data, add leading zeros or place in separate trandata files.

```
  1  1946     477.600     301.400
  2  1947     468.300     306.200
  3  1948     487.700     312.800
  4  1949     490.700     320.000
  5  1950     533.500     338.100
  6  1951     576.500     342.300
  7  1952     598.500     350.900
  8  1953     621.800     364.200
  9  1954     613.700     370.900
 10  1955     654.800     395.100
 11  1956     668.800     406.300
 12  1957     680.900     414.700
 13  1958     679.500     419.000
 14  1959     720.400     441.500
 15  1960     736.800     453.000
 16  1961     755.300     462.200
```

```
  17  1962      799.100      482.900
  18  1963      830.700      501.400
  19  1964      874.400      528.700
  20  1965      925.900      558.100
  21  1966      981.000      586.100
  22  1967     1007.700      603.200
  23  1968     1051.800      633.400
  24  1969     1078.800      655.400
  25  1970     1075.300      668.900
  26  1971     1107.500      691.000
  27  1972     1171.100      733.000
  28  1973     1233.400      766.300
  29  1974     1210.700      759.800
  30  1975     1186.400      766.600
gnp$72
c$72
```

B. ADDING DOCUMENTATION FOR THE DATA SERIES

A data file should always have documentation. A user will save himself a lot of grief and trouble by always placing documentation on the data file immediately after a series is added. SIM can read the documentation file. This allows definition of a variable to be pulled directly from the data file. The user should always include units and the source with table number or page number.

1. LISTING OF TERMINAL INPUT AND OUTPUT LINES

In this example, the user adds documentation for series 1 and 2. Note that the documentation for c$72 was not completed in line 26T. The user finished the documentation starting with word 3 by typing line 31T and then line 36T. On line 31T, the name is not required because c$72 was the last series updated. Line 41T is used to add a heading for the documentation. The actual heading was typed in line 46T. On line 51T $$$ is typed to end the reading of UPDATE LINE(S).

```
 1T   % bank.o
 2     opened index for    1 with     3 words file name  datal
 3     open sub index for   1 with     2 words
 4     bank by morris norman
 5     iiasa version february 1977
 6     maxs=   36 maxns=  400 maxc=   30 maxl=  100
 7      up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
 8T      1       30  46
 9E      1   0  30  46   1   1   0   0   0   0   0   0
10     num name       iy ib new no ip
11T     1 gnp$72            666
12E     1 gnp$72    46  1 666  0  1
13     type in documentation from word   1 to   9
```

```
14
15    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
16T  gross national product,billions of $72 scbl
17    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
18E  gross national product,billions of $72 scbl
19    series was updated
20   num name      iy ib new no ip
21T    2 c$72          666
22E    2 c$72     46  1 666  0  1
23    type in documentation from word   1 to   9
24
25    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
26T  personal consumption
27    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
28E  personal consumption
29    series was updated
30   num name      iy ib new no ip
31T    2          3    666
32E    2 c$72     3   1 666  0  1
33    type in documentation from word   3 to   9
34
35    3     $ 4     $ 5     $ 6     $ 7     $ 8     $ 9     $
36T  tion ,billions of $72 scbl
37    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
38E  personal consumption ,billions of $72 scbl
39    series was updated
40   num name      iy ib new no ip
41T                    555
42E    0          0  0 555  0  0
43    type in documentation from word   1 to   9
44
45    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
46T  scbl = survey of current business jan 1976 pt 1
47    1     $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
48E  scbl = survey of current business jan 1976 pt 1
49    series was updated
50   num name      iy ib new no ip
51T     $$$
52E    0 $$$      0  0   0  0  0
53    closed index for  1 with    6 words
54
55    closed sub index on lun    1 with     2 words
56    closed sub index on lun    1 with     3 words
57T  % bye
```

C. ADDING A SERIES CREATED IN SUBROUTINE DATRAN

In this example SUBROUTINE DATRAN was taken from section datran.f. The user must prepare SUBROUTINE DATRAN with the editor. Line 1T compiles SUBROUTINE DATRAN, and line 2T loads the object files of BANK with the RR routines. A load module bank.o is produced and datran.obj is removed. After the series is placed on the data file, and bank.o terminates, line 63T is typed to execute bank.o again. This time the documentation for the third series is added.

1. LISTING OF TERMINAL INPUT AND OUTPUT LINES

On line 74T the user types the wrong name. You cannot update a series if you use the wrong name. On line 78T, the user retypes the update requests with the correct name.

```
 1T   % ftn -c datran.f
 2T   % sh ldobj
 3    simdat.obj: non existent
 4T   % bank.o
 5        opened index for   1 with     6 words file name  datal
 6      open sub index for   1 with     2 words
 7      open sub index for   1 with     3 words
 8      bank by morris norman
 9      iiasa version february 1977
10     maxs=   36 maxns=  400 maxc=   30 maxl=  100
11      up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
12T             30  46              1
13E      0   0  30  46   1   1   0   1   0   0   0   0
14     nl no ml mn db pa nsp aps cf tr iy ip ib
15T               2         1         1
16E     1 30  1  2  0  0   1   0  3  1 46  1  1
17     type in kset card locations  1 to  2
18T   2,1
19E      2   1
20    ldata read from bank  no    iy   ip   ib
21        1    2 c$72        30   46    1    1
22        2    1 gnp$72      30   46    1    1
23    type in labels for columns   3 to   3 one / line
24T   c/gnp
25      ic num name       nl  no new  iy  ip  ib
26T      3   3
27E      3   3 c/gnp       1  30   0  46   1   1
28     series was updated
29      1  1946- 1        0.631
30      2  1947- 1        0.654
31      3  1948- 1        0.641
32      4  1949- 1        0.652
33      5  1950- 1        0.634
34      6  1951- 1        0.594
35      7  1952- 1        0.586
36      8  1953- 1        0.586
37      9  1954- 1        0.604
38     10  1955- 1        0.603
39     11  1956- 1        0.608
40     12  1957- 1        0.609
41     13  1958- 1        0.617
42     14  1959- 1        0.613
```

```
43     15  1960- 1       0.615
44     16  1961- 1       0.612
45     17  1962- 1       0.604
46     18  1963- 1       0.604
47     19  1964- 1       0.605
48     20  1965- 1       0.603
49     21  1966- 1       0.597
50     22  1967- 1       0.599
51     23  1968- 1       0.602
52     24  1969- 1       0.608
53     25  1970- 1       0.622
54     26  1971- 1       0.624
55     27  1972- 1       0.626
56     28  1973- 1       0.621
57     29  1974- 1       0.628
58     30  1975- 1       0.646
59     closed index for  1 with    7 words
60
61     closed sub index on lun    1 with     3 words
62     closed sub index on lun    1 with     3 words
63T   % bank.o
64     opened index for   1 with     7 words file name  datal
65     open sub index for   1 with     3 words
66     open sub index for   1 with     3 words
67     bank by morris norman
68     iiasa version february 1977
69     maxs=   36 maxns=  400 maxc=   30 maxl=  100
70      up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
71T      1
72E      1   0   0   0   1   1   0   0   0   0   0   0
73    num name      iy ib new no ip
74T     3 c/gno          666
75E     3 c/gno      0  1 666  0  1
76     series name was   c/gnp    must have correct name to update
77    num name      iy ib new no ip
78T     3 c/gnp          666
79E     3 c/gnp      0  1 666  0  1
80     type in documentation from word   1 to   9
81
82     1      $ 2      $ 3      $ 4      $ 5      $ 6      $ 7      $
83T   ratio of consumption to gnp in $72
84     1      $ 2      $ 3      $ 4      $ 5      $ 6      $ 7      $
85E   ratio of consumption to gnp in $72
86     series was updated
87    num name      iy ib new no ip
88T       $$$
89E     0 $$$        0  0   0  0  0
90     closed index for  1 with    8 words
91
92     closed sub index on lun    1 with     3 words
93     closed sub index on lun    1 with     4 words
94T   % bye
```

D. ADDING A SERIES AND DOCUMENTATION WITH UPDATE OPTIONS

1. LISTING OF TERMINAL INPUT AND OUTPUT LINES

This is the normal way to add one or two series. The update options are extremely useful for correcting errors in data. By appropriately choosing iy and ib the user can change any number without the rest. no will take its default values explained under UPDATE LINE(S).

In this example one series is added by typing the lines between 12 and 30 marked with a T. The series is written to the data file and then listed on the terminal. The user then corrects the mistake by typing lines 64T and 67T. The value for 1966 (line 53) should have been 161.3. Note that it is not necessary to type the name on line 64T since i$72 was the last series updated. The last zeros of a single update line are ignored. If the user wants a zero over a number, the no would need a value (no=1). In line 101T, the user calls for documentation. In line 106T, the documentation is added.

```
 1T   % bank.o
 2      opened index for   1 with     8 words file name  data1
 3     open sub index for   1 with     3 words
 4     open sub index for   1 with     4 words
 5     bank by morris norman
 6     iiasa version february 1977
 7     maxs=   36 maxns=  400 maxc=   30 maxl=  100
 8      up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
 9T      1 101  30  46           1
10E      1 101  30  46   1   1   1   0   0   0   0   0
11    num name      iy ib new no ip
12T     4 i$72
13E     4 i$72      46  1   0 30  1
14     type data begining iy= 46  1 with   5 words,6/ line
15T   71.,70.1,82.3,65.6,93.7
16E       71.000    70.100    82.300    65.600    93.700
17     type data begining iy= 51  1 with   5 words,6/ line
18T   94.1,83.2,85.6,83.4,104.1
19E       94.100    83.200    85.600    83.400   104.100
20     type data begining iy= 56  1 with   5 words,6/ line
21T   102.9,97.2,87.7,107.4,105.4
22E      102.900    97.200    87.700   107.400   105.400
23     type data begining iy= 61  1 with   5 words,6/ line
24T   103.6,117.4,124.5,132.1,150.1
25E      103.600   117.400   124.500   132.100   150.100
26     type data begining iy= 66  1 with   5 words,6/ line
27T   166.3,152.7,159.5,168.,154.7
28E      166.300   152.700   159.500   168.000   154.700
29     type data begining iy= 71  1 with   5 words,6/ line
30T   166.8,188.3,207.4,180.,138.9
```

```
31E     166.800   188.300   207.400   180.000   138.900
32     series was updated
33      1  1946- 1       71.000
34      2  1947- 1       70.100
35      3  1948- 1       82.300
36      4  1949- 1       65.600
37      5  1950- 1       93.700
38      6  1951- 1       94.100
39      7  1952- 1       83.200
40      8  1953- 1       85.600
41      9  1954- 1       83.400
42     10  1955- 1      104.100
43     11  1956- 1      102.900
44     12  1957- 1       97.200
45     13  1958- 1       87.700
46     14  1959- 1      107.400
47     15  1960- 1      105.400
48     16  1961- 1      103.600
49     17  1962- 1      117.400
50     18  1963- 1      124.500
51     19  1964- 1      132.100
52     20  1965- 1      150.100
53     21  1966- 1      166.300 <- error
54     22  1967- 1      152.700
55     23  1968- 1      159.500
56     24  1969- 1      168.000
57     25  1970- 1      154.700
58     26  1971- 1      166.800
59     27  1972- 1      188.300
60     28  1973- 1      207.400
61     29  1974- 1      180.000
62     30  1975- 1      138.900
63    num name      iy ib new no ip
64T     4           66
65E     4 i$72      66  1   0  0  1
66     type data begining iy= 66  1 with   5 words,6/ line
67T   161.3
68E     161.300     0.000     0.000     0.000     0.000
69     series was updated
70      1  1946- 1       71.000
71      2  1947- 1       70.100
72      3  1948- 1       82.300
73      4  1949- 1       65.600
74      5  1950- 1       93.700
75      6  1951- 1       94.100
76      7  1952- 1       83.200
77      8  1953- 1       85.600
78      9  1954- 1       83.400
79     10  1955- 1      104.100
80     11  1956- 1      102.900
81     12  1957- 1       97.200
82     13  1958- 1       87.700
```

```
83     14  1959- 1      107.400
84     15  1960- 1      105.400
85     16  1961- 1      103.600
86     17  1962- 1      117.400
87     18  1963- 1      124.500
88     19  1964- 1      132.100
89     20  1965- 1      150.100
90     21  1966- 1      161.300
91     22  1967- 1      152.700
92     23  1968- 1      159.500
93     24  1969- 1      168.000
94     25  1970- 1      154.700
95     26  1971- 1      166.800
96     27  1972- 1      188.300
97     28  1973- 1      207.400
98     29  1974- 1      180.000
99     30  1975- 1      138.900
100   num name iy ib new no ip
101T    4             666
102E    4 i$72 46  1 666  0  1
103    type in documentation from word   1 to   9
104
105    1      $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
106T  gross private investment,billions of $72,scbl
107    1      $ 2     $ 3     $ 4     $ 5     $ 6     $ 7     $
108E  gross private investment,billions of $72,scbl
109    series was updated
110   num name iy ib new no ip
111T      $$$
112E    0 $$$   0  0    0  0  0
113    closed index for  1 with   10 words
114
115    closed sub index on lunl with     4 words
116    closed sub index on lunl with     5 words
117   % bye
```

2. <u>LIST OF OUTPUT FILE PRODUCED WITH LAST EXAMPLE</u>

This file should be listed on the line printer with carriage control. The user should type (after a %) asa output ^lpr.

```
1                  1          2           3           4
                gnp$72     c$72       c/gnp       i$72
   1  1946- 1    477.600    301.400       0.631      71.000
   2  1947- 1    468.300    306.200       0.654      70.100
   3  1948- 1    487.700    312.800       0.641      82.300
   4  1949- 1    490.700    320.000       0.652      65.600
   5  1950- 1    533.500    338.100       0.634      93.700
   6  1951- 1    576.500    342.300       0.594      94.100
   7  1952- 1    598.500    350.900       0.586      83.200
   8  1953- 1    621.800    364.200       0.586      85.600
```

```
  9  1954- 1     613.700    370.900      0.604     83.400
 10  1955- 1     654.800    395.100      0.603    104.100
 11  1956- 1     668.800    406.300      0.608    102.900
 12  1957- 1     680.900    414.700      0.609     97.200
 13  1958- 1     679.500    419.000      0.617     87.700
 14  1959- 1     720.400    441.500      0.613    107.400
 15  1960- 1     736.800    453.000      0.615    105.400
 16  1961- 1     755.300    462.200      0.612    103.600
 17  1962- 1     799.100    482.900      0.604    117.400
 18  1963- 1     830.700    501.400      0.604    124.500
 19  1964- 1     874.400    528.700      0.605    132.100
 20  1965- 1     925.900    558.100      0.603    150.100
 21  1966- 1     981.000    586.100      0.597    161.300
 22  1967- 1    1007.700    603.200      0.599    152.700
 23  1968- 1    1051.800    633.400      0.602    159.500
 24  1969- 1    1078.800    655.400      0.608    168.000
 25  1970- 1    1075.300    668.900      0.622    154.700
 26  1971- 1    1107.500    691.000      0.624    166.800
 27  1972- 1    1171.100    733.000      0.626    188.300
 28  1973- 1    1233.400    766.300      0.621    207.400
 29  1974- 1    1210.700    759.800      0.628    180.000
 30  1975- 1    1186.400    766.600      0.646    138.900
1variables in data bank

       1  gnp$72
       2  c$72
       3  c/gnp
       4  i$72
1variables in data bank

       2  c$72
       3  c/gnp
       1  gnp$72
       4  i$72
1 general description of data
         scbl = survey of current business jan 1976 tab-1.2
1 num  label               definition of variable
    2  c$72         personal consumption,billions of $72 scbl
    3  c/gnp        ratio of consumption to gnp in $72
    1  gnp$72       gross national product,billions of $72 scbl
    4  i$72         gross private investment,billions of $72,scbl
```

E. CONDENSING THE DATA1 FILE

If a user is continually updating a data bank, and the updated series are longer than the original one, the data file will grow in size. Longer records are written at the end of a data file. Using the punch option and then creating the data bank will remove all unused space. All leading and trailing zeros will be removed. The punch file is in bcd card images. This file can be listed or edited if necessary.

1. LISTING OF TERMINAL INPUT AND OUTPUT LINES

The user types line 1T and 9T to produce a punch file. Line 15T moves the punch file to databank file. BANK reads only the databank file for recreating a data file. Line 16T moves the original datal file to an exdatal file. This is necessary for two reasons: first, the old version of datal file must be removed before a new one can be created, and second, the new databank file may have been made wrong (user error). By having an extra copy you get a second chance to make another mistake.

```
 1T  % bank.o
 2    opened index for   1 with    10 words file name  datal
 3    open sub index for   1 with     4 words
 4    open sub index for   1 with     5 words
 5    bank by morris norman
 6    iiasa version february 1977
 7    maxs=   36 maxns=  400 maxc=   30 maxl=  100
 8     up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
 9T                                           999
10E     0   0   0   0   1   1   0   0   0   0 999   0
11    closed index for  1 with   10 words
12
13    closed sub index on lun    1 with     4 words
14    closed sub index on lun    1 with     5 words
15T  % mv punch databank
16T  % mv datal exdatal
17T  % bank.o
18    file named datal   was created
19     new index for   1 with file name  datal
20    record number    1 for unit    1 was empty
21    bank by morris norman
22    iiasa version february 1977
23    maxs=   36 maxns=  400 maxc=   30 maxl=  100
24     up lis nol iy2 ip2 ib2 ldo  it ire ipa npu nad
25T                                     2
26E     0   0   0   0   1   1   0   0   2   0   0   0
27    recreates a data bank from cards from databank file
28    reads documentations from databank file
29    closed index for  1 with   10 words
30
31    closed sub index on lun    1 with     4 words
32    closed sub index on lun    1 with     5 words
33T  % rm databank
34T  % bye
```

RR ROUTINES
Random Records for the pdp11

I INTRODUCTION

The RR routines are a package of six subroutines designed to simulate CDC MASS STORAGE INPUT/OUTPUT. There are several differences between the RR routine calls and CDC mass storage calls. A user who is familiar with the CDC calls should pay close attention to the number of words that are to be written or read.

The pdp11 has integer words consisting of 2 bytes, real words consisting of 4 bytes, and double precision words consisting of 8 bytes. The user is required to convert the number of words into the number of integer equivalents. (A real word is considered 2 words for the input/output calls.)

The user must close the file with a call to closems or the index will not be written. An RR file without an index at the end of the RR records cannot be opened. The following calls are available:

```
call openms (lun,ims,lngth,t)
call writms (lun,fwa,nwds,rnorn)
call readms (lun,fwa,nwds,rnorn)
call closems(lun)
call openin (lun,ixs,nwds)
call closein(lun,ixs,nwds)
```

The use of these six routines will be individually described below. The last two calls are a substitute for CDC call stindx.

II FORMULATION

The RR routines allow the user to write and read random records without keeping track of their exact location on a file. With the use of these random records, the user can write his records in any order. For example, the RR records could be written in the order of 10, 8, 50, 2. The actual file has only four records; however, the index for the file will now have 50 words, because the 50th record was written.

The RR routines keep track of where the records are written and how many words (consisting of 2 bytes) are in each record. Each record can have a different number of words. If a record is extended, it is written at the next available space at the end of the file. If a record is changed, and remains the same length or is shortened, it is written back to its original position. The RR routines only keep track of the present length of a record. If, however, the record is later extended back to its original length, it is then written at the end of the file. Because of space limitations, the third vector that kept track of the original length was deleted from the RR routines. If a user is continually changing the size of the random records, the RR file will grow in size. The user must condense his file by reading all the records and writing them to a new file.

At present, the file size is limited to 2^{15} -1 bytes. Any writms call will be ignored if it exceeds the maximum size. The program saves 800 bytes for the main index.

III <u>SUBROUTINE OPENMS</u>

call openms (lun,ims,lngth,t)

If an existing file is opened, the main index is read into common/index/ims(801). These calls open an RR file with lun as the unit designator, ims as the first word address of the index in central memory, and lngth which is the length of the index +1. The +1 is to be consistent with CDC. The length must be greater than or equal to the number of records in the file +1. A file can be opened with a longer index than when it was closed. These RR routines do not allow referencing through a name index. t=0 for a file is referenced by a numbered index. t=3 will reinitialize an existing file. For example:

```
common/index/ims(801)
call openms (3,ims,11,0)
```

Unit 3 is opened for a maximum of 10 random records. It is not necessary to include the label common unless you want to test the index.

IV <u>SUBROUTINE WRITMS</u>

call writms (lun,fwa,nwds,rnorn)

This call transmits data from central memory to mass storage. lun is the unit designator, fwa is the central memory address of the first word of the data area. nwds is the number of transferred central memory 2 byte words. If nwds=0 the RR routines set nwds equal to the number of words in the record. If the new record length exceeds the old one, the record is written at the end of the information. rnorn is the random record number. The record number has the limits 1 $\leq$ rnorn < lngth. An example:

```
 common/zz/num,name,x(100)
 real*8 name
 ...
 num=50
 call writms (3,num,num*2+5,10)
```

In the above example words are transmitted from label common starting with the first word, num. The user must calculate the number of integer equivalents (number of bytes divided by 2), that is, 1 integer, 1 real*8, and 50 real equals 105 interger equivalents.

V SUBROUTINE READMS

call readms (lun,fwa,nwds,rnorn)

This call transmits data from mass storage to central memory. lun, fwa, nwds, rnorn are the same as for writms. The user can always do a readms with nwds=0, and the entire record will be transmitted. For example:

```
       common/index/ims(801)
      1      /zz/num,name,z(100)
       ...
       if (ims(10).eq.0) go to 1
       call readms (3,num,0,10)
 1     continue
```

If the record is empty a message will be written to the terminal (unit 6) and the program will continue. A user can test for an empty record with an if statement. The 10th element of ims will be zero if the 10th record is empty (has never been written).

VI SUBROUTINE CLOSEMS

call closems (lun)

This call writes the index from the central memory to the file. closems must be called (before the job terminates) if a files has been changed. For example:

```
 call closems (3)
```

VII SUBROUTINE OPENIN

call openin (lun,ixs,lngth)

This call reads a subindex into an area specified by the program. lun is the unit designator, ixs is the first word address of the subindex in central memory, and lngth is the length of the subindex. The user must know the size of the subindex before this call can be executed.

A subindex cannot be opened before its main index is opened. If a program has more than one RR file the subindices must be opened after their main index, and before another main index is opened. Subindices must be opened in the same order they were closed on a file. For example:

```
 common ns,nd,nextr
1       /index/ims(801)
2       /subindex/ixs(400),ixd(401)
 call openms (1,ims,801,0)
 call readms (1,ns,0,1)
 call openin (1,ixs,ns)
 if (nd.gt.0) call openin (1,ixd,nd)
```

In this example, subindex ixs is opened with ns words. If nd is greater than zero, subindex ixd is opened with nd words. An explanation of subindices will be provided below in the section USING SUBINDICES.

VIII <u>SUBROUTINE CLOSEIN</u>

call closein (lun,ixs,lngth)

This call writes a subindex from central memory to the file. lun, ixs, and lngth are the same as for call openin. This call must be made immediately after the closems for this unit. The subindex will be written at the end of the information. For example:

```
 common ns,nd,nextr
1        /index/ims(801)
2        /subindex/ixs(400),ixd(401)
 ...
 ...
 call writms  (1,ns,3,1)
 call closems (1)
 call closein (1,ixs,ns)
 if (nd.gt.0) call closein (1,ixd,nd)
```

In the above example, record 1 of the main index contains ns (the number of words in subindex ixs), nd (the number of words in subindex ixd), and nextr (the next empty record in the main index).

IX <u>USING SUBINDICES</u>

The purpose of a subindex is to provide a mapping from the random records of the subindex to the records of the main index. This eliminates the need for many files when the user needs a set of RR files. Each random record in the subindex is assigned a random record number in the main index. The user simply refers to the 10^{th} record of a subindex, and the subindex is used to map this record number into a random record in the main index. The user must provide a subroutine to assign the random records of an index to the main index.

The proper use of the subindex can greatly simplify the logic in managing the RR files. Below is an example using two subindices. Only record 1 of the main index is accessed directly. This record contains the size of the subindices and nextr, which is the next empty record on the main index. nextr is initialized as 2 and incremented each time a new record is written. The total number of non-zero entries in the subindices equals 1 less than the number of random records on the main index. There is a direct mapping from the random record of the subindices into the main index. An easier way to impliment the use of subindices is to use the following subroutines:

```
subroutines writrr(lun,fwa,nwds,ir)
common ns,nd,nextr
dimension fwa(100)
if(ir.eq.0) ir=nextr
call writms(lun,fwa,nwds,ir)
if(ir.eq.nextr) nextr=nextr+1
return
end
```

This subroutine makes the calls to writms. Each new record is assigned a random record number on the main index. The record is new if ir equals zero. ir is then given the value of nextr and nextr is incremented by 1. The normal call to writms are now replaced by:

```
call writrr (lun,fwa,nwds,ixs(rnorn))
```

lun, fwa, nwds, and rnorn are the same as for writms. ixs is a subindex, and its elements are assigned values as the random records of the subindex are written. The user only has to remember the record numbers of the subindex, not their assignment to the main index. For example:

```
dimension  z(50)
...
call writrr (3,z(1),100,ixs(10))
```

In the above example, 50 real words are written as the 10^{th} random record of subindex ixs. The element ixs(10) will be the random record number of the main index corresponding to the 10^{th} record of this subindex.

X OBTAINING THE RR ROUTINE

When the RR routines are required in a program the user must include the l.ra library on his % ftn statement. At present the following procedure will work:

```
1)  %ln /mnt/morris/random/l.ra l.ra
2)  %ftn *.obj l.ra -ls
```

Step 1 will link the l.ra library to the user's files. It is not necessary to repeat step one. In step two the last two entries are required to have a load module with the the RR routines.

AUTO
February 1977 Version for IIASA's pdp11

I INTRODUCTION

AUTO has evolved over the years into an extremely usable and efficient regression program capable of dealing with large data files (in conjunction with BANK) and with entire systems of equations. The input lines needed by the program to run a set of simple regressions are minimal and easily understood. The more experienced user will find that the many options available for data manipulation make this program extremely flexible. For example, AUTO has an option to store label and equation lines, which allows the user to run the same regression on several sets of data without duplicating these lines. Or, the data can be stored and used with new equation lines and a different sample period.

In addition to providing OLS estimates of an equation, AUTO can be used to estimate the first and second order autoregressive scheme of the error terms. Two methods of performing the autoregressive transformation are available, i.e., the Cochrane-Orcutt Iterative Technique for first order, and the Scanning Technique for first and second order. Also, AUTO calculates finite polynomial distributed lags, and has an option which allows the user to estimate a distributed lag using one variable for the current values and a different variable for the lagged values. Using the predicted values, computed from the reduced form regression for the current values and the actual values or the lagged values, produces a TSLS result.

II FORMULATION

A. OLS

Equation y = xa + e is estimated in terms of moments around the origin by OLS. The a vector of the coefficients would be estimated as follows (the example given is for two independent variables):

$$\begin{vmatrix} \hat{a}_1 \\ \\ \hat{a}_2 \\ \\ \hat{a}_3 \end{vmatrix} = \begin{vmatrix} \Sigma x_i^2 & \Sigma x_1 x_2 & \Sigma x_1 \\ \\ \Sigma x_1 x_2 & \Sigma x_2^2 & \Sigma x_2 \\ \\ \Sigma x_1 & \Sigma x_2 & T \end{vmatrix}^{-1} \begin{vmatrix} \Sigma x_1 y \\ \\ \Sigma x_2 y \\ \\ \Sigma y \end{vmatrix} \qquad \text{(II.1)}$$

where $\hat{a}_i$ is the estimate of a_i and T is the number of observations. The variance of the estimated residuals, VAR, is given by

$$\text{VAR} = \frac{1}{T-K}\left(\Sigma y^2 - \sum_{i=1}^{K-1} a_i \Sigma x_i y - a_K \Sigma y \right) \qquad \text{(II.2)}$$

which is equivalent to

$$\text{VAR} = \frac{1}{T-K} \Sigma e^2. \qquad \text{(II.3)}$$

R^2, adjusted for degrees of freedom, is given by

$$R^2 = 1 - \frac{\text{VAR}}{s_y^2} \qquad \text{(II.4)}$$

where

$$s_y^2 = \frac{1}{T-1} \left(\Sigma y^2 - \frac{(\Sigma y)^2}{T}\right).$$

B. AUTOREGRESSIVE OLS

The equation $y = xa + u$ is estimated by AOLS in terms of moments around the origin which have been transformed by the autoregressive scheme. For the first order autoregressive scheme, it is assumed that $u_t = pu_{t-1} + e_t$ and the equation is estimated as

$$y_t = py_{t-1} + \sum_{j=1}^{K-1} a_j (x_{jt} - px_{jt-1}) + a_K(1-p) + e_t.$$

The variance is defined in terms of the transformed residuals and calculated by equation (II.2) where

Σy^2, $\Sigma x_i y$, and Σy

are replaced by the appropriate transformed moments. R^2 is given by equation (II.4), where the variance of y is not replaced by the variance of the transformed y.

For the second order autoregressive scheme it is assumed that $u_t = p_1 u_{t-1} + p_2 u_{t-2} + e_t$ and the equation is estimated as

$$y_t = p_1 y_{t-1} + p_2 y_{t-2} + \sum_{j=1}^{K-1} a_j (x_{jt} - p_1 x_{jt-1} - p_2 x_{jt-2})$$

$$+ a_K (1 - p_1 - p_2) + e_t.$$

The second order scan will be performed only over the values of the parameters that imply a stationary process. That is, the parameters p_1 and p_2 must lie in the triangular region*

$$p_1 + p_2 < 1$$

$$p_2 - p_1 < 1$$

$$-1 < p_2 < 1.$$

The variance and R^2 are calculated by the same equation as the first order regression with the exception that the moments are transformed by a second order scheme.**

A transformed moment, i.e.,

$$(Y\ X)_T = \Sigma (y_0 - p_1 y_{-1} - p_2 y_{-2})\ (x_0 - p_1 x_{-1} - p_2 x_{-2}),$$

can be calculated from the original moment matrix if the data set is extended to include all the variables lagged one and two periods. The transformed moment can then be written as

$$(Y\ X)_T = \sum_{i=0}^{n} \sum_{j=0}^{n} p_i p_j \ y_i x_j,$$

and where $p_0 = -1$, $y_i x_j$ represents the cross product of y lagged i periods with x lagged j periods, and n is the order

*See G.E.P. Box and G. M. Jenkins, Time Series Analysis: Forecasting and Control, Holden Day, San Francisco, 1970.

**The standard error and R^2 of the untransformed residuals (i.e., the u_t) are also calculated and printed under the heading R/SU.

of the transformation, i.e., n=1 or n=2.

The p's used in the autoregressive transformation can be specified in the three ways that are described in the section below.

1. COCHRANE-ORCUTT ITERATIVE FIRST ORDER SCHEME

An initial estimate of p is obtained from the residuals of the equation y = xa + u, then the moments are transformed using p, and the new estimates of the coefficients are obtained. A second estimate of p is calculated from the residuals of the equation using the revised estimates of the coefficients. This procedure is continued until two successive estimates of p differ by less than .001 or the number of iterations exceeds 20. When p is greater than .975 the constant term is approximately zero and will be suppressed.

p is estimated by OLS for the equation $u_t = pu_{t-1} + e_t$ by the formula

$$\hat{p} = \frac{\Sigma u_t u_{t-1}}{\Sigma u_{t-1}^2}.$$

The sums in this formula can be expressed in terms of the original moments and coefficients of y = ax + u as

$$\begin{aligned}
\Sigma u_t u_{t-1} &= (Y - Xa)'(Y_1 - X_1 a) \\
&= YY_1 - YX_1 a - Y_1 Xa + aX_1 Xa \\
\Sigma u_{t-1}^2 &= (Y_1 - aX_1)'(Y_1 - aX_1) \\
&= Y_1 Y_1 - 2aX_1 Y_1 + aX_1 X_1 a.
\end{aligned}$$

The Cochrane-Orcutt Iterative Technique yields a local minimum for the transformed residual sum of squares which may or may not be the global minimum. A user must do a first order scan to be certain that the global minimum was found.

2. SCANNING TECHNIQUE FOR FIRST AND SECOND ORDER SCHEMES

A range of values for p, i.e., initial value, ending value, and step size, is specified by the user and an equation is estimated for each p. For a second order, a range of values for p's must also be specified.

3. <u>SPECIFIED p TECHNIQUE (SPECIAL CASE FOR SCANNING)</u>

A particular value of p (and p_2) is specified and the equation is estimated.

If the graph option is used with AOLS, the transformation residuals are calculated, i.e.,

$$e_t = u_t - pu_{t-1}$$

for first order and

$$e_t = u_t - p_1 u_{t-1} - p_2 u_{t-2}$$

for second order. The e's are calculated by performing the appropriate transformations on the u's. The Durbin-Watson "d" statistic is based on the transformed residuals.

C. <u>OLS FINITE POLYNOMIAL DISTRIBUTED LAGS</u>

The distributed lag equation

$$y_t = \sum_{i=0}^{n} w_i x_{t-1} + e_t$$

can be estimated by assuming that the weights are produced by an n^{th} degree polynomial. AUTO allows the user to specify the weights as:

1st	$w_i = a + bi$
2nd	$w_i = a + bi + ci^2$
3rd	$w_i = a + bi + ci^2 + di^3.$

To estimate the lags the data are transformed as:

1st	$\Sigma\ (a + bi)x$	$= a\Sigma x + b\Sigma ix$
2nd	$\Sigma\ (a + bi + ci^2)x$	$= a\Sigma x + b\Sigma ix + c\Sigma i^2 x$
3rd	$\Sigma\ (a + bi + ci^2 + di^3)x$	$= a\Sigma x + b\Sigma ix + c\Sigma i^2 x + d\Sigma i^3 x$

where x is subscripted with t-1 and the summations run from i=0 to n. The four scale factors are

$$\Sigma,\ \Sigma i,\ \Sigma i^2,\ \text{and}\ \Sigma i^3.$$

An end restriction can be placed on the polynomial, i.e., $w_n=0$:

1st $w_n = a + bn = 0,$ -> $a = -bn$

2nd $w_n = a + bn + cn^2 = 0,$ -> $a = -bn - cn^2$

3rd $w_n = a + bn + cn^2 + dn^3 = 0,$ -> $a = -bn - cn^2 - dn^3.$

To estimate the lags with this end restriction the data are transformed as follows:

1st $b(\Sigma ix - n\Sigma x)$

2nd $b(\Sigma ix - n\Sigma x) + c(\Sigma i^2x - n^2\Sigma x)$

3rd $b(\Sigma ix - n\Sigma x) + c(\Sigma i^2x - n^2\Sigma x) + d(\Sigma i^3x - n^3\Sigma x)$

where x is subscripted with t-i and the summations run from i=0 to n-1. The three scale factors are:

$$\Sigma i - n\Sigma,\ \Sigma i^2 - n^2\Sigma, \text{ and } \Sigma i^3 - n^3\Sigma.$$

Two end restrictions can be placed on the polynomials, $w_0=0$ and $w_n=0$, which implies $a=0$:

1st $w_n = bn = 0$ -> $b = 0$

2nd $w_n = bn + cn^2 = 0,$ -> $b = -nc$

3rd $w_n = bn + cn^2 + dn^3 = 0,$ -> $b = -nc - n^2d.$

To estimate the lags with these end restrictions the data are transformed as follows:

2nd $c(\Sigma i^2x - n\Sigma ix)$

3rd $c(\Sigma i^2x - n\Sigma ix) + d(\Sigma i^3x - n^3\Sigma ix)$

where x is subscripted with t-i and the summations run from i=1 to n-1. The two scale factors are

$$\Sigma i^2 - n\Sigma i \text{ and } \Sigma i^3 - n^2\Sigma i.$$

The transformed data, when divided by the scale factors, are approximately the same size as the original variables.

The actual distributed lag weights are computed as the dot product of the appropriate sub-vectors of the weight vector vector e_i and the coefficient vector a. For the standard case with no restrictions the vectors are:

$$e_i = \begin{vmatrix} 1 \\ i \\ i^2 \\ i^3 \end{vmatrix}, \quad a = \begin{vmatrix} a \\ b \\ c \\ d \end{vmatrix}, \text{ and } w_i = e_i' a.$$

When the end restriction $w_{n=\emptyset}$ is placed on the polynomial the vectors e_i and a are:

$$e_i = \begin{vmatrix} i-n \\ i^2-n^2 \\ i^3-n^3 \end{vmatrix}, \quad a = \begin{vmatrix} b \\ c \\ d \end{vmatrix}, \text{ and } w_i = e_i' a.$$

When the end restrictions $w_\emptyset = w_n = \emptyset$ are placed on the polynomial the vectors e_i and a are:

$$e_i = \begin{vmatrix} i^2-in \\ i^3-in^2 \end{vmatrix}, \quad a = \begin{vmatrix} c \\ d \end{vmatrix}, \text{ and } w_i = e_i' a.$$

The variance of the distributed lag coefficients can be calculated from Ω, the variance matrix of the coefficients a with the formula:

$$\text{var}(w_i) = e_i' \Omega e_i.$$

A user can place the distributed transformed variables anywhere in the data matrix or let the program automatically place them at the end of the data set. If a user wants to run a 2nd and a 3rd degree polynomial on the same variable only the 3rd degree transformed variable need be calculated. For the 2nd degree polynomial lag the user can specify the appropriate transformed variables from the set calculated for the 3rd degree polynomial lag. When running polynomial lags it is the user's responsibility to place the appropriate variables in the correct position on the equation line. The polynomial transformed lag variable must be placed in increasing order before the other variables of the regression. Up to nine polynomial lags can be in one equation.

D. <u>TSLS DISTRIBUTED LAGS</u>

The user can estimate a distributed lag using one variable for the current values and a different variable for the lagged values; e.g., the predicted values can be used for the current values and the actual values can be used for the lagged values. This procedure would produce a TSLS result. To obtain the first stage predicted values the user would run the reduced form equation and use SUBROUTINE RHO3's option (parameter lyh on the PART 3 OF THE EQUATION LINE) to place the predicted values in the data matrix. Then a new PARAMETER LINE would be read with nx=4. This option allows the user to respecify the size of the Z matrix and call SUBROUTINE TRAN, if required, using the old data.

III <u>SUBROUTINE</u> <u>TRAN</u>

SUBROUTINE TRAN enables the user to transform the original variables from the data bank into new variables. The user must insert into SUBROUTINE TRAN the necessary Fortran statements to calculate the desired transformed variables. A dummy SUBROUTINE TRAN can be used if no transformations are made.

LISTING 1

```
      subroutine tran
      common det,zero,code(8),xx(22,22),yy(22,22),label(22),
     1       coef(22),vv,ij(22),ik(22),nll,n6,ly,nao,n,m,no,
     2       xno,mn,mnl,nx,n8,jl,nl,dis,db,j4,n9,pc,ic,ml,m3,
     3       maxr,maxc
     d       /dat/z(30,21),tr,iy,ip,ib,ln,iabel(22)
     o       /opt/xk,ng,yvar,npage,iuse,nlx,nox,lnx
     i       /index/ ims(401),ixs(390),ixd(1),kset(60),nk
      real*8 label,iabel,lab,eqn,c,d,c4,name,re,code,b,ab
      integer pa,gr,grx,sc,ar,art,tr,cor,dis,db,pc,pcx
      do 1 i=nl,no
c     place transformation here
1     continue
      return
      end
```

Listing 1 contains the basic cards for SUBROUTINE TRAN. The user must add to these cards the necessary Fortran statements to perform his desired transformations.

Listing 2 is an example of how to compute the lagged values of three variables and the natural log of a fourth variable from the original Z matrix of ten variables. The new variables 11, 12, and 13 are the lagged values of the original variables 5, 3, and 10, respectively. Since the first observation is lost in these lagged variables, the first row of the Z matrix is ignored by defining nl=2. The default value for nl is 1. The fourth transformation variable, variable 14, is the natural log of the first variable in the Z matrix. Since the number of variables in the Z matrix has been increased from 10 to 14 variables, mn, the number of variables in the Z matrix, must be redefined, i.e., mn=14. If mn had not been set equal to 14, the program would have ignored the new variables. Only the Z matrix from 11 to 14 will be listed after transformation and only labels for the variables 11 through 14 will be required. If j4=1 labels will be required for all of the 14 variables. The original data plus the transformed data will be listed after transformation.

LISTING 2

```
      subroutine tran
      common det,zero,code(8),xx(22,22),yy(22,22),label(22),
     1       coef(22),vv,ij(22),ik(22),nll,n6,ly,nao,n,m,no,
     2       xno,mn,mnl,nx,n8,jl,nl,dis,db,j4,n9,pc,ic,ml,m3,
     3       maxr,maxc
     d       /dat/z(30,21),tr,iy,ip,ib,ln,iabel(22)
     o       /opt/xk,ng,yvar,npage,iuse,nlx,nox,lnx
     i       /index/ ims(401),ixs(390),ixd(1),kset(60),nk
      real*8 label,iabel,lab,eqn,c,d,c4,name,re,code,b,ab
      integer pa,gr,grx,sc,ar,art,tr,cor,dis,db,pc,pcx
      nl=2
      mn=14
      do 1 i=nl,no
c     place transformation here
      z(i,11)=z(i-1,5)
      z(i,12)=z(i-1,3)
      z(i,13)=z(i-1,10)
      z(i,14)=alog(z(i,1))
1     continue
      return
      end
```

LISTING 3

```
      subroutine tran
      common det,zero,code(8),xx(22,22),yy(22,22),label(22),
     1       coef(22),vv,ij(22),ik(22),nll,n6,ly,nao,n,m,no,
     2       xno,mn,mnl,nx,n8,jl,nl,dis,db,j4,n9,pc,ic,ml,m3,
     3       maxr,maxc
     d       /dat/z(30,21),tr,iy,ip,ib,ln,iabel(22)
     o       /opt/xk,ng,yvar,npage,iuse,nlx,nox,lnx
     i       /index/ ims(401),ixs(390),ixd(1),kset(60),nk
      real*8 label,iabel,lab,eqn,c,d,c4,name,re,code,b,ab
      integer pa,gr,grx,sc,ar,art,tr,cor,dis,db,pc,pcx
      mn=18
      do 1 i=nl,no
c     place transformation here
      z(i,16)=0.0
      z(i,17)=0.0
      z(i,18)=0.0
1     continue
      do 2 i=1,no,4
      z(i,16)=1.0
      z(i+1,17)=1.0
2     z(i+2,18)=1.0
      return
      end
```

Listing 3 is an example of how to construct seasonal dummy variables. First, variables 16, 17, and 18 must be set to zero. This TRAN assumes that the data are quarterly and the i^{th} seasonal dummy will be 1 for the i^{th} period and zero elsewhere. The second do loop increments i by 4 so i takes on the values 1, 5, 9, 13,.... Variables 16, 17, and 18 are the first, second, and third quarter seasonal dummies, respectively. mn is set to 1. Only labels for the three seasonal dummy variables will be read.

When a large model is estimated it is important to break the equations into related groups. Each group can be debugged separately, then all of the TRAN subroutines can be combined. The user should be careful to number the groups consecutively and then to use the group number for tr on part 2 of the PARAMETER LINE. SUBROUTINE TRAN is called if tr is greater than zero. tr can be used as a control word for a computed go to statement; e.g.,

```
go to (10,20,30,40), tr.
```

tr takes on the values 1, 2, 3, and 4. The user should also have pa=9 on the last equation card for each group. This procedure will accumulate the summary tables for each group into one comprehensive table. If the table is filled up before the last equation card, it will be printed out before the user has an opportunity to specify the number of copies desired. By setting m3=3 in SUBROUTINE TRAN the user would get three copies of the summary table. If the value given to pa on the last control card is greater than zero and less than 9 it will override the value given to m3, and pa copies will be printed. (See section SUBROUTINE TRAN FOR THE GNP MODEL that gives more details on estimating a complete model.)

When the absolute value of the determinant is less than .00001, the program assumes that the matrix being inverted is singular. This test value can be changed by setting zero equal to the desired value in SUBROUTINE TRAN, e.g., zero=1.0E-20. Changing the test value is sometimes necessary in cases where the data have been scaled down, for example, price equations where the base is 1.00.

In special cases the user may want to override the degrees of freedom used in the calculation of the variance. The program uses the following formula:

$$\text{Variance} = \frac{1}{xno-xn}\Sigma e^2 \text{ where } xno = \sum_{i=nl}^{no} Z(i,mnl) = no-nl+1.$$

xn=K is the number of coefficients estimated, including the constant term. If the user gives xk a value in SUBROUTINE TRAN then xno-xk will be used for the degrees of freedom instead of xno-xn.

If the user gives ng a value ≠ 0, the variance from the previous regression will be used in the calculation of the "t" statistic, etc., in this set of regressions. This assumes the user runs a regression, then with a new parameter card calls SUBROUTINE TRAN where ng is set ≠ 0. The second regression is then run using the variance from the previous regression.

In some cases the dependent variable is transformed but the user is interested in how well the equation explains the levels form of the dependent variable. If the user calculates the variance of the untransformed dependent variable and then sets YVAR equal to the variance, R^2 will be calculated using this value, i.e.,

$$R^2 = 1 - \frac{VAR}{YVAR},$$

where VAR equals the variance of the residuals adjusted for the degrees of freedom.

Another manipulation that can be carried out in SUBROUTINE TRAN is setting npage=0 which causes the output listing to start a new page only for the summary table or for a graph. npage is set back to 1 before SUBROUTINE INFISH.

The user may only want to list the data after SUBROUTINE TRAN. This can be accomplished by having ln=9 on PART TWO OF PARAMETER LINE and then setting ln=0 in SUBROUTINE TRAN.

In some cases the user may want to define labels in SUBROUTINE TRAN. Remember if j4 is not defined in SUBROUTINE TRAN, it is given the default value of the original mn+1. Labels will be read for only the new variables. The user can set j4 so labels are read for (label(i),i=j4,mn). The label for variables 1 to j4-1 comes from the original vector of labels (names of series on data bank). These labels are stored in vector iabel. If the user were using the save label options, the vector could be changed in SUBROUTINE TRAN by setting label(i)=iabel(i) for i=1,j4-1. In regression studies for many industries, countries, etc., it is very important that the data in the data bank be grouped so that they can be accessed in a sequence; e.g., manhours for the twenty 2 digit industries should be variables 1 through 20, value added the next 20, capital the next 20 and so on. This organization of the data will facilitate the running of the regressions so the user only has to specify the locations of the first set of variables for industry 1; then the data for the next industry can be found by adding 1 to the location numbers.

Remember the kset vector can be used to pull series from the data bank by leaving db blank on PART 1 OF PARAMETER LINE. The next set of data for industry 2 using the nx=6 would be found if the kset vector were incremented by 1 in SUBROUTINE TRAN. For example:

```
      do 1 i=1,15
1     kset(i)=kset(i)+1
```

IV FILES USED BY AUTO

FILE	NAME	USE
1	datal	Data bank file from BANK.
2	incards	For a permenant copy of all lines typed to terminal.
5	terminal	To type information to the program.
6	terminal	To receive prompts and other information from the program.
7	coef	Coefs file for SIMDAT.
8	output	All results are written to this file. After the program has terminated this file can be listed on the line printer.

V INPUT LINE(S)

The program writes prompts to the user. The user types in the appropriate information under the mnemonics, right adjusted. This section gives the prompts and their meaning.

A. OVERRIDE LINE

iuse nlx nox grx npx jcx lnx mlx kc

Note: Generally a blank line will be used for these options. The override options are useful when estimating a complete model where many groups of control lines are present and it would be infeasible to change the individual lines. If a user plans to change the sample period with nlx and nox, the first row of the Z matrix for each group should be the same date and the minimum nl should be calculated in TRAN; e.g., nl=maxo(nl,5) would be used for a four-period lag.

iuse	=	0	Override options are ignored.
	=	1	Override options are used.
nlx	=	0	Value specified for nl on PARAMETER LINE is used.
	=		First row of the Z matrix; i.e., nl on the PARAMETER LINE is set equal to nlx.
nox	=	0	Value specified for no on PARAMETER LINE is used.
	=		Last row of the Z matrix; i.e., no on PARAMETER LINE is set equal to nox.
grx	=	0	Graph option is deleted from all equations.
	>	0	Graph options for all equations; gr=grx.
npx	=	0	No new pages.
	=	1	New page for each equation.
pcx	=	0	No coefficients are punched.
	=	1	Coefficients are written to coef file in bcd card images.
lnx	=	0	All data are listed.
	=	1-8	The first lnx and last lnx observations are listed.
	=	9	Label only.
mlx	=		Maximum number of lines to be used in the summary table: 4<mlx<80. Useful in making summary tables print on whole pages.
kc	=		The number of parameter columns in summary tables. With kc=4, a summary table will fit on a normal page (66 characters). kc=7 is for computer output and the default value is 7.

B. PARAMETER LINE(S)

1. PART 1 OF PARAMETER LINE

no mn iy ip ib db nl ml noa

no	=		Last row of the Z matrix (number of observations equals no-nl+1). If no=0 and tr≠0 the program will transfer directly to SUBROUTINE TRAN without reading any data.
mn	=		Last column of the Z matrix. (Number of variables equals mn-ml+1.)
iy	=		Beginning year of data. The beginning data are placed in row 1 of the Z matrix.
ip	=		Number of observations per year. Default value equals 1.
ib	=		Starting period of series. Default value equals 1.
db	=	0	Normal. Reads data from the data bank. KSET GRABBER LINE(S) must follow PARAMETER LINE(S).
	=	1	To read data from data bank with GRABBER LINE(S).
nl	=		First row of the Z matrix. Default value equals 1.
ml	=		First column of the Z matrix. Default value equals 1.
noa	=		Last row to be used after the data is read. Default value is no. Useful if there is more than one set of PARAMETER LINE(S).

2. PART 2 OF PARAMETER LINE

tr nx ln dis art cor pc ns

tr	=	0	Normal.
	≠	0	To call SUBROUTINE TRAN.
nx	=	0	Normal.
	=	1	To read and save LABEL(S) and EQUATION LINE(S) for the next set of data. A maximum

of 2Ø EQUATION LINE(S) can be stored with no more than 14 variables for any equation.

	=	2	To use the previously stored LABEL(S) and EQUATION LINE(S) with new data. Requires nx=1 on a previous data set.
	=	3	To use the previously stored LABEL(S), EQUATION LINE(S), and Z matrix. (Allows the same regression to be run with a different sample period or different transformations on the same data.) Requires nx=1 on a previous PARAMETER LINE.
	=	4	To use the previously stored Z matrix (with new labels if TRAN or DIST is called) with new EQUATION LINE(S). (Allows different regressions to be run on same data after no and nl have been changed on PARAMETER LINE.)
	=	5	To use the previously stored LABEL(S) and EQUATION LINE(S) with new data. On first set of PARAMETER LINE(S), put nx=1, on second set, put nx=5, then for the remaining sets use only the appropriate GRABBER LINE(S).
	=	6	Same as 5 with the additional feature that new GRABBER LINE(S) are not read. This assumes the user will change the kset vector in SUBROUTINE TRAN. The first mn will be saved as nk; then mn is automatically set to the original mn before the next set of data is read. nk is in labeled common /INDEX/ and could be changed. (Note: nl is set to 1 in AUTO. The remaining parameters for the Z matrix will remain as set in TRAN or as originally read on PARAMETER LINE).
ln	=	Ø	All rows of the Z matrix are listed.
	=	1-8	The first ln and last ln observations are listed in rows of the Z matrix.
	=	9	Only the labels are listed. (The labels are not listed if SUBROUTINE TRAN is not called.)
dis	=	Ø	Normal.

	=	1	To calculate distributed lags.
	=	2	To call SUBROUTINE TRAN after calculating distributed lags.
art	=	0	Normal.
	=	1	First order of autoregressive scheme option (one observation lost). mn<11.
	=	2	Second order of autoregressive scheme option (two observations lost). mn<7.
cor	=	0	Normal.
	=	1	To calculate correlation matrix.
	=	2	To read another PARAMETER LINE after GRABBER LINE. No operations are performed on the data.
	=	3	To read another PARAMETER LINE after SUBROUTINE TRAN is called and before SUBROUTINE DIST is called.
pc	=	0	Normal.
	≠	0	Coefficients are written to coef file in bcd card images.
ns	=	0	Normal.
	≠	0	To print a summary table of previous regressions and start a new table with this equation. This option is used to override the stored equation lines if the last equation line had pa=9. One copy of the table will be printed unless m3 is set greater than 1 in SUBROUTINE TRAN.

3. CODE

code	=	Any message. (Can be defined in TRAN when using nx=5 option or any time.)
	=	$$$ or "control d" if there is an error on PARAMETER LINE(S) and a second try is needed.

C. INPUT LINE(S) TO RETRIEVE DATA FOR THE Z MATRIX

1. KSET GRABBER LINE(S)

The user must type the logical record numbers of the series that are to be placed in the Z matrix. The numbers are in fields of 4 or separated by commas, with a maximum of 20 per line. If the numbers are in a sequence only the first number should be typed. The numbers are stored in the kset vector.

2. GRABBER LINE(S)

num name (mn-ml+1 lines will be read).

num	=	Logical record number of variable in data bank to be placed in the Z matrix.
name	=	Name of variable as in data bank, left adjusted.

D. LAG DISTRIBUTION SPECIFIATIONS

1. LAG DISTRIBUTIONS CONTROL LINE

nld mn j4

nld	=		Number of lag distributions < 10.
mm	=	0	Uses value for mn that program computes.
	=		Number of variables to be used in data matrix after transformed variables are added. This value will override the present value of mn.
j4	=		1st column in the Z matrix for new labels; default value is mn+1 if tr=0.

2. LAG DISTRIBUTION PARAMETER LINE(S)

lvc lv dp n nr l1 l2 l3 l4 (nld lines are read)

lvc	=	Location of variable to be used for current period. Default value is lv.
lv	=	Location of variable to be used for the distributed lag.

dp	=		Degree of polynomial (1, 2, or 3).
n	=		Length of lag, < 4Ø.
nr	=	Ø	No restrictions.
		1	End restriction, w_n=Ø.
	=	2	End restrictions, $w_Ø$=w_n=Ø.
l1	=		Column in the Z matrix for the first transformed variable.
l2	=		Column in the Z matrix for the second transformed variable.
l3	=		Column in the Z matrix for the third transformed variable.
l4	=		Column in the Z matrix for the fourth transformed variable.
		Note:	If l's are left blank, transformed variables will be added at the end of the Z matrix.

E. LABEL LINE(S) (FOR THE Z MATRIX)

label	=		name of variable (mn-j4+1 labels are read with one per line).
			The labels are for the data in the Z matrix if any operations have been performed on the data, i.e., transformations or distributed lags. The user has the option of saving all or some of the original labels by specifying j4 in TRAN as the number of labels to be saved plus 1. After transformation and/or distributed lag variables are formed the data are listed starting with the j4th variable through the mnth variable. The program automatically sets j4 equal to the original mn+1 if the user does not specify a value for it in TRAN.
		Note:	Labels are not read if n2=j2=Ø or nx=2, 3, or 5, or if j4>mn. Data are not listed in this case.

F. EQUATION SPECIFICATION LINE(S)

1. PART 1 OF EQUATION LINE

eqn numc

eqn	=		Name of equation, e.g., eqn-6. This information will appear in the summary table, and on the coef file.
	=	$$$	To list variables in the Z matrix. This is handy if you forget the location of the variables.
	=		"Control d" to transfer to new PARAMETER LINE.
numc	=		Number assigned to first coefficient. This option is only required when a coef file is produced for SIMDAT.

2. PART 2 OF EQUATION LINE

m il i2 i3...dp

m	=		Number of variables in the equation (independents plus dependent).
il	=		Location of first independent variable in the Z matrix.
i2	=		Location of second independent variable (if there is one) in the Z matrix.
dp	=		Location of dependent variables in the Z matrix.

3. PART 3 OF EQUATION LINE

pa gr ar nd sc lyh nao

pa	=	Ø	To read another EQUATION LINE.
	=	1-8	To read a new PARAMETER LINE and give pa summary tables. The summary table will hold at most 26 regressions. If this limit is exceeded the summary table is printed out. In this case to get more than one copy, m3 must be set in SUBROUTINE TRAN to the number of copies desired.
	=	9	To accumulate the summary table between data

groups. To get more than one copy of the table, the last EQUATION LINE of the group must specify the number desired, or the user can set m3 equal to the number desired in SUBROUTINE TRAN, and when an eof is encountered on the input file summary tables will be printed.

gr	=	Ø	Normal.
	=	1	For a graph and Durbin-Watson "d" statistic.
	=	2	For a "d" statistic without graph.
	=	3	For a graph and for the ê (residuals) to be placed in column lyh of the Z matrix (see lyh option below).
ar	=	Ø	Normal.
	=	1	Cochrane-Orcutt autoregressive option.
	=	2	Scanning autoregressive option, first order (this requires one RHO LINE).
	=	3	Scanning autoregressive option, second order (this requires two RHO LINES).
nd	=	Ø	Normal.
	=		Number of distributed lags in the equation. This number must be specified or the weights will not be calculated.
sc	=	Ø	Normal.
	=	1	To suppress constant term.
lyh	=	Ø	Normal.
	=		Location in the Z matrix where ŷ will be stored if gr=1 or 2, or where ê will be stored if gr=3.
nad	=	Ø	Normal.
	=		Number of additional observations to be included in the graph. If actual values for these additional observations equal zero, the predicted values, ŷ , are placed in the Z matrix.

G. RHO LINE(S) (IF ar > 1)

rhol rho2 rho3 (ar-l lines are read)

rhol = Starting value of p_1 for scan.

rho2 = Ending value of p_1 for scan.

rho3 = Step size for scan.

For second order scan the second RHO LINE gives the scanning values for p_2 . If the ending values and step size are omitted from the RHO LINE(S) the regression will be computed from the starting values only. The second order scan is performed over the values that imply a stationary process. Only the equation with the minimum variance over the range is saved.

Note: The program reads another EQUATION LINE if pa=Ø on the previous EQUATION LINE or another PARAMETER LINE if pa≠Ø. The program will print out the summary table when an end of file ("control d" or eof) is encountered on the input file.

VI EXAMPLES

In these examples it is assumed that the user has prepared a datal file with BANK. Some of the computer output is designed for 132 columns. Whenever necessary the last 60 columns are printed below the first group.

The user types the lines marked with a T. The program echoes back the lines marked with an E.

A. MINIMUM INFORMATION TO RUN A SIMPLE REGRESSION

This section introduces the user to the input lines for AUTO without the confusion of the many possible options. Line 1T commands the computer to execute AUTO. For line 6T the user simply hits a CR (carriage return). This option can be omitted. Line 10T tells the computer the size of the Z matrix and the starting date to row 1. For line 13T a CR is given. For line 16T the user types any title to describe this set of regressions. Line 20T calls for variables 1 and 2 off the data bank. Line 25T allows the user to name the following equation. Line 27T says there are two variables in the equation with variable 1 (gnp$72) as the independent and variable 2 (c$72) as the dependent. Line 30 calls for the graph option and a new PARAMETER LINE. Line 33T is a "control d" to end the run.

1. LIST OF TERMINAL INPUT OUTPUT LINES

```
 1T  % auto.o
 2   1            auto by morris norman
 3                feb 1977 iiasa version
 4                 maxr  30 maxc  20
 5   iuse nlx nox grx npx pcx lnx mlx
 6T
 7E     0    0    0    0    0    0    0    0
 8   1
 9    no mn iy ip ib db nl ml noa
10T   30  2 46
11E   30  2 46  0  0  0  0  0   0
12    tr nx ln dis art cor pc ns
13T
14E    0  0  0   0   0   0  0  0
15    type in code one line
16T  example one
17    opened index for   1 with    10 words file name  datal
18    open sub index for   1 with    4 words
19    type in kset card location  1 to  2
20T  1,2
21    data read from data bank   no   iy   ip   ib
22E      1    1 gnp$72           30   46    1    1
23E      2    2 c$72             30   46    1    1
```

```
24    equ-name numc
25T   el.1
26      m il i2 i3 i4 i5 i6 i7 .. .. de
27T     2  1  2
28E     2  1  2
29     pa gr ar nd sc lyh nao
30T     1  1
31E     1  1  0  0  0   0   0
32     no mn iy ip ib db nl ml noa
33T    "control d"
34        end of data cards
```

2. OUTPUT FILE

This file is listed on the line printer by typing (after a %) asa output ^lpr

```
example one
 no mn iy ip ib db nl ml noa
 30  2 46  0  0  0  0  0   0
 tr nx ln dis art cor pc ns
  0  0  0   0   0   0  0  0
 data read from data bank   no   iy   ip   ib
    1    1 gnp$72           30   46    1    1
    2    2 c$72             30   46    1    1
-                     1         2
                    gnp$72     c$72
   1  1946- 1     477.600    301.400
   2  1947- 1     468.300    306.200
   3  1948- 1     487.700    312.800
   4  1949- 1     490.700    320.000
   5  1950- 1     533.500    338.100
   6  1951- 1     576.500    342.300
   7  1952- 1     598.500    350.900
   8  1953- 1     621.800    364.200
   9  1954- 1     613.700    370.900
  10  1955- 1     654.800    395.100
  11  1956- 1     668.800    406.300
  12  1957- 1     680.900    414.700
  13  1958- 1     679.500    419.000
  14  1959- 1     720.400    441.500
  15  1960- 1     736.800    453.000
  16  1961- 1     755.300    462.200
  17  1962- 1     799.100    482.900
  18  1963- 1     830.700    501.400
  19  1964- 1     874.400    528.700
  20  1965- 1     925.900    558.100
  21  1966- 1     981.000    586.100
  22  1967- 1    1007.700    603.200
  23  1968- 1    1051.800    633.400
  24  1969- 1    1078.800    655.400
  25  1970- 1    1075.300    668.900
```

26	1971- 1	1107.500	691.000
27	1972- 1	1171.100	733.000
28	1973- 1	1233.400	766.300
29	1974- 1	1210.700	759.800
30	1975- 1	1186.400	766.600

```
1
 nrl equ-name numc
   1 el.1       0
  m il  i2 i3 i4 i5 i6 i7 .. .. de
  2  1  2
 pa gr ar nd sc lyh nao
  1  1  0  0  0   0   0
```

example one

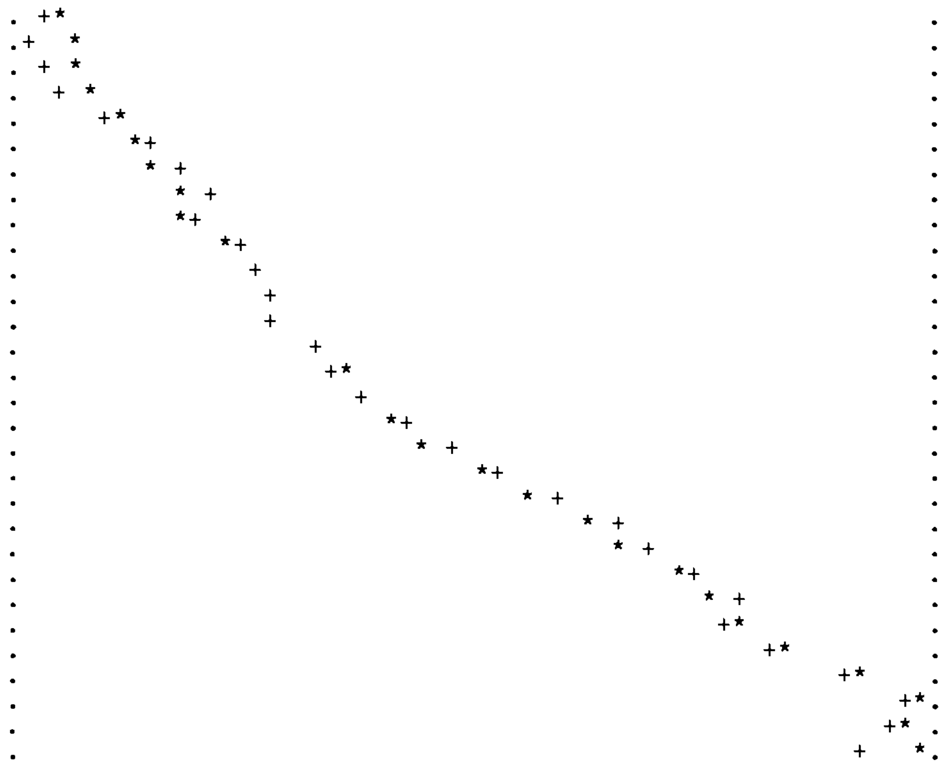
c$72	0 gnp$72	const	r/se	dw/df
ols	0.621502	-5.607956	0.992454	0.382560
el.1	61.764511	-0.659163	13.367540	28.000000

	actuals(*)	predicted(+)	residuals	% error
1946- 1	301.4000	291.2216	10.1784	3.3770
1947- 1	306.2000	285.4416	20.7584	6.7794
1948- 1	312.8000	297.4988	15.3012	4.8917
1949- 1	320.0000	299.3633	20.6367	6.4490
1950- 1	338.1000	325.9636	12.1364	3.5896
1951- 1	342.3000	352.6882	-10.3882	-3.0348
1952- 1	350.9000	366.3612	-15.4612	-4.4062
1953- 1	364.2000	380.8423	-16.6422	-4.5695
1954- 1	370.9000	375.8081	-4.9081	-1.3233
1955- 1	395.1000	401.3518	-6.2518	-1.5823
1956- 1	406.3000	410.0529	-3.7529	-0.9237
1957- 1	414.7000	417.5731	-2.8730	-0.6928
1958- 1	419.0000	416.7029	2.2971	0.5482
1959- 1	441.5000	442.1224	-0.6224	-0.1410
1960- 1	453.0000	452.3150	0.6850	0.1512
1961- 1	462.2000	463.8128	-1.6128	-0.3489
1962- 1	482.9000	491.0346	-8.1346	-1.6845
1963- 1	501.4000	510.6741	-9.2741	-1.8496
1964- 1	528.7000	537.8337	-9.1337	-1.7276
1965- 1	558.1000	569.8411	-11.7411	-2.1038
1966- 1	586.1000	604.0859	-17.9859	-3.0687
1967- 1	603.2000	620.6800	-17.4800	-2.8979
1968- 1	633.4000	648.0883	-14.6882	-2.3190
1969- 1	655.4000	664.8688	-9.4688	-1.4447
1970- 1	668.9000	662.6936	6.2064	0.9279
1971- 1	691.0000	682.7059	8.2941	1.2003
1972- 1	733.0000	722.2335	10.7665	1.4688
1973- 1	766.3000	760.9531	5.3469	0.6978
1974- 1	759.8000	746.8450	12.9550	1.7051
1975- 1	766.6000	731.7425	34.8575	4.5470

durbin-watson d statistic is 0.38256

(the following graph was moved from the above lines)

range 285.44 to 766.60



B. RERUNNING A SET OF REGRESSIONS

AUTO writes all the input lines to the incards file. This file must be moved to another file if the user wants to save his input lines. Often it is easier to edit the incards file than it is to correct the input lines. This file is very important when building a complete model. The incard files can be collected into one master file and then the whole model can be run together.

1. LIST OF TERMINAL INPUT AND OUTPUT LINE(S)

To rerun the previous example the user only has to type lines 1T and 2T. Line 27T prints the output file on the high speed printer.

```
 1T  % mv incards card
 2T  % auto.o 5=card
 3   1             auto by morris norman
 4                 feb 1977 iiasa version
 5                  maxr  30 maxc  20
 6   iuse nlx nox grx npx pcx lnx mlx
 7      0   0   0   0   0   0   0   0
 8   1
 9    no mn iy ip ib db nl ml noa
10    30  2 46  0  0  0  0  0   0
11    tr nx ln dis art cor pc ns
12     0  0  0   0   0   0  0  0
13    type in code one line
14    opened index for   1 with    10 words file name  datal
15    open sub index for   1 with     4 words
16    type in kset card location  1 to  2
17    data read from data bank   no   iy   ip   ib
18       1    1 gnp$72           30   46    1    1
19       2    2 c$72             30   46    1    1
20   equ-name numc
21     m il i2 i3 i4 i5 i6 i7 .. .. de
22     2  1  2
23    pa gr ar nd sc lyh nao
24     1  1  0  0  0   0   0
25    no mn iy ip ib db nl ml noa
26    end of data cards
27T  % asa output^lpr
28T  % bye
```

2. CARD FILE

The card files contain the bcd card images of the lines typed in the first example. Commas that were used to separate the integers are removed from the card file.

```
   0    0    0    0    0    0    0    0
 30   2 46   0   0   0   0   0    0
  0   0   0    0    0    0   0   0
example one
   1   2
el.1         0
  2  1  2
  1  1  0  0  0   0   0
```

C. RUNNING A COMPLETE MODEL

The potential GNP Model for the USA is estimated in one run. The card file used was built up from the individual incards file.

1. TERMINAL LISTING FOR GNP MODEL

```
% auto.o 5=card
1             auto by morris norman
              feb 1977 iiasa version
               maxr  30 maxc  20
iuse nlx nox grx npx pcx lnx mlx  kc
   1   0   0   2   0   1   9  72   4
0
 no mn iy ip ib db nl ml noa
 30  2 46  0  0  0  0  0   0
 tr nx ln dis art cor pc ns
  1  0  0   0   0   0  0  0
 type in code one line
 opened index for   1 with   255 words file name  datal
 open sub index for   1 with   128 words
 type in kset card location  1 to  2
 data read from data bank   no   iy   ip   ib
    1   32 dep$72           30   46    1    1
    2   79 k$72             30   46    1    1
 type labels for variables  3 to  3one / line
equ-name numc
  m il i2 i3 i4 i5 i6 i7 .. .. de
  2  3  1
 pa gr ar nd sc lyh nao
  9  1  0  0  0   0   0
 no mn iy ip ib db nl ml noa
 30  5 46  0  0  0  0  0   0
 tr nx ln dis art cor pc ns
  9  0  0   0   0   0  0  0
 type in code one line
 type in kset card location  1 to  5
 data read from data bank   no   iy   ip   ib
    1   12 gnp              30   46    1    1
    2  120 taxin$72         30   46    1    1
    3   63 ni$72            30   46    1    1
    4   51 taxes            30   46    1    1
    5   49 ipdgnp           30   46    1    1
 type labels for variables  6 to  7one / line
equ-name numc
  m il i2 i3 i4 i5 i6 i7 .. .. de
  2  1  7
 pa gr ar nd sc lyh nao
  0  1  0  0  0   0   0
equ-name numc
  m il i2 i3 i4 i5 i6 i7 .. .. de
  2  6  4
 pa gr ar nd sc lyh nao
  9  1  0  0  0   0   0
 no mn iy ip ib db nl ml noa
 30  5 46  0  0  0  6  0   0
 tr nx ln dis art cor pc ns
```

```
51     2  0  0    0    0    0  0  0
52    type in code one line
53    type in kset card location  1 to  5
54    data read from data bank   no    iy    ip    ib
55       1   19 gt               30    46     1     1
56       2   49 ipdgnp           30    46     1     1
57       3   76 pop65            30    46     1     1
58       4   74 unempl           29    47     1     1
59       5   77 time46=1         30    46     1     1
60    type labels for variables  6 to  6one / line
61   equ-name numc
62     m il i2 i3 i4 i5 i6 i7 .. .. de
63     2  5  6
64    pa gr ar nd sc lyh nao
65     9  1  0  0  0   0   0
66    no mn iy ip ib db nl ml noa
67    30  2 46  0  0  0  0  0   0
68    tr nx ln dis art cor pc ns
69     0  0  0   0   0   0  0  0
70    type in code one line
71    type in kset card location  1 to  2
72    data read from data bank   no    iy    ip    ib
73       1   12 gnp              30    46     1     1
74       2   25 bt               30    46     1     1
75   equ-name numc
76     m il i2 i3 i4 i5 i6 i7 .. .. de
77     2  1  2
78    pa gr ar nd sc lyh nao
79     9  1  0  0  0   0   0
80    no mn iy ip ib db nl ml noa
81    30  4 46  0  0  0  0  0   0
82    tr nx ln dis art cor pc ns
83     3  0  0   1   0   0  0  0
84    type in code one line
85    type in kset card location  1 to  4
86    data read from data bank   no    iy    ip    ib
87       1    3 i$72             30    46     1     1
88       2   49 ipdgnp           30    46     1     1
89       3   72 baa              30    46     1     1
90       4   60 is               30    46     1     1
91   nld mm j4
92     1  0  5
93   lvc lv dp  n nr l1 l2 l3 l4
94     0  5  2  8  1  0  0  0  0
95    type labels for variables  5 to  7one / line
96   equ-name numc
97     m il i2 i3 i4 i5 i6 i7 .. .. de
98     3  6  7  4
99    pa gr ar nd sc lyh nao
100    9  1  0  1  0   0   0
```

(lines 101 to 326 were removed)

```
327   pa gr ar nd sc lyh nao
328    2  1  0  0  0   0   0
329   no mn iy ip ib db nl ml noa
330   end of data cards
331T % asa output^lpr
332T % bye
```

2. SUMMARY TABLE FOR GNP MODEL

This is the complete table for all the equations estimated by OLS. kc equals 4 so the equation would fit on regular paper. mlx equals 72 so the tables will fit on 2 pages of computer output. The default value is 80. The user can adjust mlx so a series of regressions fit on 2 pages instead of 2 1/4 pages. This table is translated in the next section.

POTENTIAL GNP MODEL FOR THE USA

capital consumption allowances with cca

dep$72	24	k$72-1	const	r/se	dw/df
ols		0.054049	-8.754915	0.997292	0.684969
eq-3		101.556679	-10.676810	1.250174	27.000000

tax equations

taxi$*p	6	gnp	const	r/se	dw/df
ols		0.089793	1.837248	0.998744	0.427301
eq-4		151.835907	4.241861	1.163508	28.000000
taxes	8	p*ni$72	const	r/se	dw/df
ols		0.293797	-25.987469	0.993579	1.058154
eq-6		66.998161	-9.947808	6.993081	28.000000

government transfer payments to persons

l(gt$/n)	11	time46=1	const	r/se	dw/df
ols		0.052441	-0.105794	0.980874	0.292265
eq-7		35.097740	-3.651536	0.053872	23.000000

business transfer payments

bt	35	gnp	const	r/se	dw/df
ols		0.004439	-0.413032	0.995343	0.749465
eq-8		78.731148	-10.003003	0.110920	28.000000

net interest income

is	26	avin-a	avin-b	const	r/se
ols		4.267571	-2.478419	7.620531	0.953075
eq-9		0.721286	-0.400257	10.980065	1.955655
	26	dw/df	avin	avin-1	avin-2
		0.272124	0.521955	0.410073	0.311517
eq-9		20.000000	2.086602	3.975542	15.864024
	26	avin-3	avin-4	avin-5	avin-6
		0.226285	0.154379	0.095797	0.050540
eq-9		2.390518	1.087170	0.612697	0.367569
	26	avin-7	avin-8		
		0.018607	0.000000		
eq-9		0.217924	0.000000		

consumption function

c$72	1	ys-a	ys-b	const	r/se
ols		1.177289	-0.267998	-12.931071	0.996743
eq-10		1.138113	-0.258012	-2.042084	8.412811
	1	dw/df	ys$72	ys$7-1	ys$7-2
		0.713950	0.385156	0.272787	0.171138
eq-10		24.000000	4.727843	81.974411	4.030215
	1	ys$7-3	ys$7-4		
		0.080209	0.000000		
eq-10		1.910024	0.000000		

investment function

net-i	22	k*-k-1	const	r/se	dw/df
ols		0.851681	7.634630	0.878822	0.868080
eq-11		12.381376	1.826662	6.797543	20.000000

desired capital stock

k$72	76	yl	y*pl	y*p2	const
ols		1.628686	0.456863	-0.322602	9.249557
eq-12		28.854733	3.358184	-2.232763	1.083928
	76	r/se	dw/df	gnp$72	gnp$-1
		0.999508	0.724402	0.000000	0.135724
eq-12		8.886806	19.000000	0.000000	28.854731
	76	gnp$-2	gnp$-3	gnp$-4	gnp$-5
		0.232669	0.290837	0.310226	0.290837
eq-12		28.854733	28.854733	28.854733	28.854733
	76	gnp$-6	gnp$-7	gnp$-8	y*sq(w/c
		0.232669	0.135724	0.000000	0.079513
eq-12		28.854733	28.854731	0.000000	8.344512
	76	y*sq-1	y*sq-2	y*sq-3	y*sq-4
		0.040278	0.013948	0.000522	0.000000
eq-12		8.366484	1.733981	0.076573	0.000000

derived demand for labor - manhours index

manhours	37	gnp$72	y*sq(ck/	time46=1	const
ols		0.042406	0.068120	-1.044381	59.950798
eq-16		4.849672	3.395016	-6.822764	29.903814

	37 r/se	dw/df	
	0.977264	1.518881	
eq-16	1.020723	25.000000	

average weekly hours for private nonagricultural

hours	41 prop-un	time46=1	const	r/se
ols	-0.143988	-0.115710	41.163960	0.922064
eq-17	-3.238260	-15.739469	181.871490	0.310813

	41 dw/df
	0.634145
eq-17	26.000000

employment of workers

employ	44 manh/hr	const	r/se	dw/df
ols	36.325302	-21.690697	0.984270	0.454459
eq-18	41.869400	-9.996724	1.129044	27.000000

electric sales, residential & small light & power/capita

el-r+c	52 c$72	const	r/se	dw/df
ols	2.515014	-4.587585	0.991511	0.870848
eq-21	57.195202	-37.726295	0.116840	27.000000

electric sales, large light & power and misc.

el-la+ot	56 gnp$72	y*sq(w/p	y*sq(ck/	const
ols	0.848264	-0.031798	0.098287	-282.226776
eq-22	7.735893	-2.530367	2.530669	-8.231606

	56 r/se	dw/df
	0.991863	0.374521
eq-22	18.223612	25.000000

demand for fuel, household & commercial sector

fuel-h/c	70 c$72	c*sq(p/	const	r/se
ols	3.502822	1.274964	2601.113281	0.980612
eq-24	2.031298	6.927813	11.687482	328.281311

	70 dw/df
	1.403355
eq-24	26.000000

demand for fuel, transportation sector

fuel-trn	61	gnp$72	y*sq(w/p	y*sq(w/p	time46=1
ols		19.317823	-1.686795	5.624977	-173.549820
eq-25		6.781799	-6.776175	6.589031	-3.087632

	61	const	r/se	dw/df
		-1427.860962	0.988707	0.927722
eq-25		-1.576501	364.819580	24.000000

demand for fuel, industrial & misc. sectors

fu-in/ot	66	gnp$72	y*sq(ck/	time46=1	const
ols		6.995428	1.475639	-107.624176	7059.432617
eq-26		1.835218	2.730760	-1.433962	5.828745

	66	r/se	dw/df
		0.954201	1.265607
eq-26		488.395905	25.000000

demand for fuel electric utilities sector

fuel-eu	46	el-tot	el*sq(w/	el*sq(ck	time46=1
ols		22.273735	-0.821967	1.400685	-327.090820
eq-27		17.115053	-4.993039	2.286249	-9.391029

	46	const	r/se	dw/df
		2002.971191	0.998223	1.469907
eq-27		18.700449	218.772339	24.000000

3. POTENTIAL GNP MODEL FOR USA

1) Definition of real GNP

GNP\$72 = C\$72 + I\$72 + G\$72 + EX\$72

2) Definition of real national income

NI\$72 = GNP\$72 - DEP\$72 - TAXIN\$72 - BT/P - RES

3) Capital consumption allowance with CCA

$$DEP\$72 = \underset{(101.5)}{.054} * K\$72_{-1} - \underset{(10.7)}{8.75}$$

R^2 = .997, se = 1.25, d = .68

4) Indirect business tax function

$$P * TAXIN\$72 = \underset{(151.9)}{.09} * P * GNP\$72 + \underset{(4.2)}{1.84}$$

R^2 = .999, se = 1.16, d = .43

5) Definition of real spendable income

YS\$72 = NI\$72 - (TAXES - GT - BT - IS)/P

6) Income tax function

$$TAXES = \underset{(66.9)}{.29} * P * NI\$72 - \underset{(9.9)}{25.99}$$

R^2 = .994, se = 6.99, d = 1.06

7) Government transfer payments to persons

$$GT = P * (POP65 + UNEMP)\ EXP\ (\underset{(35.1)}{.052} * TIME - \underset{(3.6)}{.106})$$

R^2 = .981, se = .054, d = .29

8) Business transfer payments

$$BT = \underset{(78.7)}{.0044} * P * GNP\$72 - \underset{(10.0)}{.41}$$

$R^2 = .995$, se = .111, d = .75

9) Net interest income

$$IS = \sum_{i=0}^{7} w_i * P_{-i} * I\$72_{-i} * BAA_{-i} + \underset{(11.0)}{7.62}$$

$w_0 = .522$ $w_4 = .154$
$w_1 = .410$ $w_5 = .096$
$w_2 = .312$ $w_6 = .051$
$w_3 = .226$ $w_7 = .019$

$R^2 = .953$, se = 1.96, d = .27

10) Consumption function

$$C\$72 = \underset{(4.7)}{.39} * YS\$72 + \underset{(82.0)}{.27} * YS\$72_{-1} + \underset{(4.0)}{.17} * YS\$72_{-2}$$

$$+ \underset{(1.9)}{.08} * YS\$72_{-3} - \underset{(2.0)}{12.43}$$

$R^2 = .997$, se = 8.41, d = .71

11) Investment function

$$I\$72 = \underset{(12.4)}{.85} * (K\$72^* - K\$72_{-1}) + DEP\$72 + \underset{(1.8)}{7.6}$$

$R^2 = .879$, se = 6.80, d = .87

12) Desired capital stock

$$K\$72^* = \sum_{i=0}^{7} w_i GNP\$72_{-i} + \sum_{i=0}^{4} v_i GNP\$72_{-i} \left|\frac{w\text{-rate}}{\text{cost-k}}\right|^{1/2} + \underset{(1.08)}{9.25}$$

$w_0 = .000$ $w_4 = .310$ $v_0 = .080$
$w_1 = .136$ $w_5 = .291$ $v_1 = .040$
$w_2 = .233$ $w_6 = .233$ $v_2 = .014$
$w_3 = .291$ $w_7 = .136$ $v_3 = .0005$

$R^2 = .999$, se = 8.89, d = .72

13) Actual capital stock

$$K\$72 \quad K\$72_{-1} + I\$72 - DEP\$72$$

14) Capital output ratio

$$K/\emptyset = K\$72/GNP\$72$$

15) Cost of capital

$$COST\text{-}K = (BAA * .01 + .054) * P$$

16) Derived demand for labor - manhours index

$$MANHOURS = \underset{(4.8)}{.042} * GNP\$72 + \underset{(3.4)}{.068} * GNP\$72 \left|\frac{cost\text{-}k}{w\text{-}rate}\right|^{1/2}$$

$$- \underset{(6.8)}{1.04} * TIME + \underset{(29.9)}{59.9}$$

$R^2 = .977$, se = 1.02, d = 1.52

17) Average weekly hours for total private nonagricultural

$$HOURS = \underset{(3.2)}{-.144} * PROP\text{-}UN - \underset{(15.7)}{.166} * TIME + \underset{(182.0)}{41.2}$$

$R^2 = .922$, se = .311, d = .63

18) Employment of workers

$$EMPLOY = \underset{(41.9)}{36.33} * \frac{MANHOURS}{HOURS} - \underset{(10.0)}{21.69}$$

$R^2 = .984$, se = 1.13, d = .45

19) Total unemployment

$$UNEMP = LABOR - EMPLOY$$

20) Proportion of civilian labor force unemployed

$$PROP\text{-}UN = UNEMP/EMPLOY * 100$$

Demand for Electricity

21) Residential and small light & power per capita

$$\frac{EL\text{-}R{+}C}{POP} = \underset{(57.1)}{2.52}\,\frac{C\$72}{POP} - \underset{(37.7)}{4.59}$$

$R^2 = .992$, se = .117, d = .871

22) Electric sales to large light and power and misc.

$$EL\text{-}LA{+}OT = \underset{(7.7)}{.848} * GNP\$72 - \underset{(2.5)}{.032} * GNP\$72 \left|\frac{w\text{-}rate}{Pe}\right|^{1/2}$$

$$+\underset{(2.5)}{.098} * GNP\$72 \left|\frac{cost\text{-}k}{Pe}\right|^{1/2} - \underset{(8.2)}{282.2}$$

$R^2 = .992$, se = 18.2, d = .375

23) Total electric sales

EL-TOT = EL-LA+OT + EL+R+C

Demand for Fuel

24) Household and commercial sectors

$$FUEL\text{-}H/C = \underset{(2.0)}{3.50} * C\$72 + \underset{(6.9)}{1.275} * C\$72 \left|\frac{P}{PBF}\right|^{1/2}$$

$$+ \underset{(11.7)}{2601.1}$$

$R^2 = .981$, se = 328.3, d = 1.40

25) Transportation sector

$$FUEL\text{-}TRN = \underset{(6.8)}{19.32} * GNP\$72 - \underset{(6.8)}{1.69} * GNP\$72 \left|\frac{w\text{-}rate}{PBF}\right|^{1/2}$$

$$+\underset{(6.6)}{5.62} * GNP\$ \left|\frac{cost\text{-}k}{PBF}\right|^{1/2} - \underset{(3.1)}{173.5} * TIME$$

$$-\underset{(1.6)}{1428.9}$$

$R^2 = .989$, se = 364.8, d = .93

26) Industrial sector and miscellaneous sector

$$FU\text{-}IN/OT = \underset{(1.8)}{7.0} * GNP\$72 + \underset{(2.7)}{1.48} * GNP\$72 \left|\frac{cost\text{-}k}{PBF}\right|^{1/2}$$

$$\underset{(1.4)}{-107.6} * TIME + \underset{(5.8)}{7059.4}$$

R^2 = .954, se = 488.4, d = 1.27

27) Electrical utilities sector

$$FUEL\text{-}EU = \underset{(17.1)}{22.3} * EL - \underset{(5.0)}{.82} * EL \left|\frac{w\text{-}rate}{PBF}\right|^{1/2}$$

$$+ \underset{(2.3)}{1.4} * EL \left|\frac{cost\text{-}k}{PBF}\right|^{1/2} - \underset{(9.4)}{327.1} * TIME$$

$$+ \underset{(18.7)}{2003.0}$$

R^2 = .998, se = 218.8, d = 1.47

28) Electricity produced by fuel

EL = 1.094 * EL-TOT - EL-NUC - EL-HY

29) Total demand for fuel

FUEL-ALL = FUEL-H/C + FUEL-TRN + FU-IN/OT + FUEL-EU

4. CARD FILE FOR GNP MODEL

This card file is an accumulation of all the incards files combined into one file. The first card is removed from the incards files before it is added to the card file. The user can now set the OVERRIDE LINE for the desired option. The first line (OVERRIDE LINE) calls for: "d" statistics, the coefficient to be written to the coef file, only the labels to be listed, a maximum of 72 lines in the summary table, and only 4 columns for the parameters in the summary table.

```
1      1  0   0   2   0   1   9  72   4
2P   30  2 46  0   0   0   0   0   0
3      1  0  0    0   0    0  0  0
4    capital consumption allowances with cca
5      32  79
6    k$72-1
```

```
 7    eq-3        24
 8      2  3  1
 9      9  1  0  0  0   0   0
10P   30  5 46  0  0  0  0  0   0
11      9  0  0   0   0   0  0  0
12    tax equations
13      12 120  63  51  49
14    p*ni$72
15    taxi$*p
16    eq-4         6
17      2  1  7
18      0  1  0  0  0   0   0
19    eq-6         8
20      2  6  4
21      9  1  0  0  0   0   0
22P   30  5 46  0  0  0  6  0   0
23      2  0  0   0   0   0  0  0
24    government transfer payments to persons
25      19  49  76  74  77
26    l(gt$/n)
27    eq-7        11
28      2  5  6
29      9  1  0  0  0   0   0
30P   30  2 46  0  0  0  0  0   0
31      0  0  0   0   0   0  0  0
32    business transfer payments
33      12  25
34    eq-8        35
35      2  1  2
36      9  1  0  0  0   0   0
37P   30  4 46  0  0  0  0  0   0
38      3  0  0   1   0   0  0  0
39    net interest income
40       3  49  72  60
41      1  0  0
42      0  5  2  8  1  0  0  0  0
43    avin
44    avin-a
45    avin-b
46    eq-9        26
47      3  6  7  4
48      9  1  0  1  0   0   0
49P   30  2 46  0  0  0  0  0   0
50      0  0  0   1   0   0  0  0
51    consumption function
52       2 121
53      1  0  0
54      0  2  2  4  1  0  0  0  0
55    ys-a
56    ys-b
57    eq-10        1
58      3  3  4  1
```

```
59      9  1  0  1  0   0   0
60P   30  4 46  0  0  0  8  0   0
61      4  0  0   0   0   0  0  0
62    investment function
63       3  32  79  78
64    k*-k-1
65    net-i
66    eq-11        22
67      2  5  6
68      9  1  0  0  0   0   0
69P   30  6 46  0  0  0  1  0   0
70      5  0  0   1   0   0  0  0
71    desired capital stock
72      79   1  47 103 116  77
73      2  0  0
74      0  2  2  8  2  0  0  0  0
75      0  8  2  4  1 14 15  0  0
76    sq(w/ck
77    y*sq(w/c
78    sq(pf/ck
79    y*sq(pf/
80    k/y
81    y1
82    y2
83    y*p1
84    y*p2
85    eq-12        76
86      4 12 14 15  1
87      9  1     2
88P   30  6 46  0  0  0  2  0   0
89      5  0  0   0   0   0  0  0
90    derived demand for labor - manhours index
91      33   1 116  47 103  77
92    sq(pf/w)
93    y*sq(pf/
94    sq(ck/w)
95    y*sq(ck/
96    man/y
97    eq-16        37
98      4  2 10  6  1
99      9  1  0  0  0   0   0
100P  30  3 46  0  0  0  2  0   0
101     0  0  0   0   0   0  0  0
102   average weekly hours for total private nonagricultural
103     30 104  77
104   eq-17        41
105     3  2  3  1
106     9  1  0  0  0   0   0
107P  30  3 46  0  0  0  2  0   0
108     6  0  0   0   0   0  0  0
109   employment of workers
110     28  30  33
```

```
manh/hr
eq-18        44
  2  4  1
  9  1  0  0  0   0   0
 30  7 46  0  0  0  2  0   0
  7  0  0   0   0   0  0  0
electric sales, residential & small light & power  per capita
 113   2  49 102 103  77  26
sq(p/pe
c*sq(p/
sq(ck/pe
c*(ck/pe
elrc/c
eq-21        52
  2  2  1
  9  1  0  0  0   0   0
 30  6 46  0  0  0  2  0   0
  5  0  0   0   0   0  0  0
electric sales, large light & power and other classification
 114   1  47 101 103  77
sq(w/pe)
y*sq(w/p
sq(ck/pe
y*sq(ck/
el/y
eq-22        56
  4  2  8 10  1
  9  1  0  0  0   0   0
 30  6 46  0  0  0  2  0   0
  5  0  0   0   0   0  0  0
demand for fuel, household & commercial sector
  96   2  49 116 103  77
sq(pc/pf
c*sq(p/
sq(ck/pf
c*sq(ck/
fu/c
eq-24        70
  3  2  8  1
  9  1  0  0  0   0   0
 30  6 46  0  0  0  2  0   0
  5  0  0   0   0   0  0  0
demand for fuel, transportation sector
  98   1  47 116 103  77
sq(w/pf)
y*sq(w/p
sq(ck/pf
y*sq(w/p
fu/y
eq-25        61
  5  2  8 10  6  1
  9  1  0  0  0   0   0
```

```
163   30  6 46  0  0  0  2  0   0
164    5  0  0   0   0   0  0  0
165   demand for fuel, industrial & miscellaneous sectors
166    115   1  47 116 103  77
167   sq(w/pf)
168   y*sq(w/p
169   sq(ck/pf
170   y*sq(ck/
171   fu/y
172   eq-26        66
173     4  2 10  6  1
174     9  1  0  0  0   0   0
175P  30  8 46  0  0  0  2  0   0
176     8  0  0   0   0   0  0  0
177   demand for fuel, electric utilities sector
178     88 112  47 116 103  77 117 118
179   sq(w/pf)
180   el*sq(w/
181   sq(ck/pf
182   el*sq(ck
183   fu/el
184   eq-27        46
185     5  2  8 10  6  1
186     1  1  0  0  0   0   0
```

5. SUBROUTINE TRAN FOR THE GNP MODEL

This SUBROUTINE TRAN is a combination of all the SUBROUTINE TRAN cases used to estimate the separate blocks. A computed go to statement is used to transfer the program to the appropriate section. All the lines marked with P in the card file are part 1 of the PARAMETER LINE. The first number for part 2 of the PARAMETER LINE is tr. This parameter is used to call SUBROUTINE TRAN. Note that tr takes on a value from 0 to 9. When tr is greater than zero, SUBROUTINE TRAN is called. tr is then used in the computed go to statement to transfer the program to the appropriate set of transformations.

```
      subroutine tran
      common det,zero,code(8),xx(22,22),yy(22,22),label(22),
     1        coef(22),vv,ij(22),ik(22),nll,n6,ly,nao,n,m,no,
     2        xno,mn,mnl,nx,n8,jl,nl,dis,db,j4,n9,pc,ic,ml,m3
     3       ,maxr,maxc
     d        /dat/z(30,21),tr,iy,ip,ib,ln,iabel(22)
     o        /opt/xk,ng,yvar,npage,iuse,nlx,nox,lnx,xmo,last
     i        /index/ ims(401),ixs(390),ixd(1),kset(60),nk
      real*8 label,iabel,lab,eqn,c,d,c4,name,re,code,b,ab
      integer pa,gr,grx,sc,ar,art,tr,cor,dis,db,pc,pcx
      go to (10,20,30,40,50,60,70,80,90),tr
10    nl=nl+1
      do 1 i=nl,no
```

```
      z(i,3)=z(i-1,2)
1     continue
      mn=mn+1
      return
20    do 2 i=nl,no
      z(i,6)=alog(z(i,1)/z(i,2)*100.0/(z(i,3)+z(i,4)))
2     continue
      mn=mn+1
      return
30    continue
      do 3 i=nl,no
3     z(i,5)=z(i,1)*z(i,2)*z(i,3)*.0001
      mn=mn+1
      return
40    continue
      nl=nl+1
      do 4 i=nl,no
      z(i,6)=z(i,1)-z(i,2)
4     z(i,5)=z(i,4)-z(i-1,3)
      mn=mn+2
      return
70    continue
      do 7 i=nl,no
      z(i,1)=z(i,1)/z(i,7)
7     z(i,2)=z(i,2)/z(i,7)
50    continue
      do 5 i=nl,no
      z(i,7)=sqrt(z(i,3)/z(i,4))
      z(i,9)=sqrt(z(i,5)/z(i,4))
      z(i,8)=z(i,7)*z(i,2)
      z(i,10)=z(i,9)*z(i,2)
      z(i,11)=z(i,1)/z(i,2)
5     continue
      j4=7
      mn=11
      return
60    continue
      do 6 i=nl,no
6     z(i,4)=z(i,3)/z(i,2)
      mn=mn+1
      return
80    continue
      do 8 i=nl,no
8     z(i,2)=1.094*z(i,2)-z(i,7)-z(i,8)
      go to 50
90    continue
      do 9 i=nl,no
      z(i,7)=z(i,2)*z(i,5)*0.01
9     z(i,6)=z(i,3)*z(i,5)*0.01
      mn=7
      return
      end
```

SIMDAT
February 1977 Version for IIASA's pdp11

I INTRODUCTION

SIMDAT is used to produce a data file for SIM. This data file contains the regression coefficients, the labels for the variables, and the data series arranged by observations. SIMDAT can also be used to update the coefficients and data, and to change the basic data requirements for SIM. The reader should read the write-up for SIM before attempting to build a data file.

II FORMULATION

SIMDAT and SIM are the two modules required to simulate a nonlinear econometric model. SIMDAT produces the necessary input file, and SIM performs the actual simulations. The main reason for two modules is to minimize core requirements for SIM. All data specifications and creations are done in SIMDAT. The basic data file for the control (or baseline) solution is the original data2 file. SIMDAT labels this data file as actual values--that is, the actual values taken from the data1 file produced by BANK. SIM can write any solution to the data2 file. When this happens the data are labeled baseline.

SIMDAT consists of the main program and a SUBROUTINE FIND. The main program handles the creation and updating of the input (data2) file for SIM. SUBROUTINE FIND is used to read the data1 file produced by BANK. The data are read into the Z matrix by columns.

The user must specify the starting date for the longest lag, the length of the longest lag, and the number of lagged values needed for each variable used in the model. This information is used to construct a vector containing the necessary lagged values needed to start a simulation. The starting period for the simulation data file is nl periods after the starting date for the longest lag.

SIMDAT writes the lag vector as the 10th random record of the data2 file. The remaining records are in the rows of the Z matrix starting with the nl+1 row. The lag vector construction is explained in the next solution.

III THE LAG VECTOR (d vector)

The lag vector contains the current value plus the lag values (if any) for all the variables. The number of lags for each variable is specified with the inl vector. The total length of the lag vector is the only constraint on the number of lags. The ibx vector listed in lines 125 to 177 of example A has the starting locations for each variable in the d vector. For this model gnp$72 is 1, c$72 is 9, and i$72 is 10. The lag values for gnp$72 are in locations 2 to 8, c$72 hasn't any lag values, and the lag values for i$72 are in 11 to 17.

The lag vector is called the d vector in SIM.

IV FILES USED BY SIMDAT

FILE	NAME	USE
1	data1	Data bank file from BANK.
2	data2	Input data file for SIM.
3	coefs	Coef file from AUTO.
5	terminal	To type information to the program.
6	terminal	To receive prompts and other information from the program.
8	output	For permanent listing of information sent to the terminal. After the program has completed this file can be listed on the line printer.

V INPUT LINE(S)

The program saves all the information and parameters from previous runs. After a file has been created, only changes to the parameters are required.

A. COEFFICIENT OPTION LINE

up

up	=	0	Coefficient vector is not modified.
	=	1	For COEFFICIENT UPDATE LINE(S).

	=	2	To read the coefs file and then use COEFFICIENT UPDATE LINE(S).

1. COEFFICIENT UPDATE LINE(S)

j a(j)

j	=		The j^{th} element of the coefficient vector.
	=	0	To transfer to DATA OPTION LINE.
	=	-1	To list coefficient vector.
a(j)	=		The new value for the j^{th} coefficient.

B. DATA OPTION LINE

up	=	0	No updates.
	=	1	To read a PARAMETER LINE.

1. PARAMETER LINE

no ned nex nxs nl iy ip ib nod

no	=		Number of observations including longest lag.
ned	=		Number of endogenous variables.
nex	=		Number of exogenous variables.
nxs	=	0	If data are grouped together without separating the endogenous and exogenous variables.
	=	1	If data are separated into an endogenous group and an exogenous group.
nl	=		Length of the longest lag in the model.
iy	=		Starting year for longest lag.
ip	=		Number of observations per year with a default value of 1.
ib	=		Starting period for longest lag with a default value of 1.
nod	=		Number of observations for endogenous variables with a default value of no. The data

file may contain exogenous variables for forecasting future periods when the endogenous variables do not exist. In this case nod < no.

Note: If nxs=0 the variables are numbered consecutively. The 6th variable may be an endogenous or an exogenous variable. If nxs=1 then the endogenous variables are numbered consecutively, and the exogenous variables are also numbered consecutively, starting with 1.

2. LENGTH OF LAG LINE(S) (INL VECTOR)

num lag

num	=		Variable number with a lag.
	=	0	For next option.
	=	-1	To list the inl vector.
lag	=		Length of lag for this variable.

Note: If nxs=1 the data are divided into two groups: endogenous variables, and exogenous variables. The user must first specify all lags for the endogenous variables, then type zero for i. The program will now require an inl vector for the exogenous variables.

3. IK VECTOR LINES FOR ENDOGENOUS VARIABLES (IF nxs=0)

i ik(i)

i	=		The i^{th} element of the ik vector.
	=	0	For next option.
	=	-1	To list the ik vector.
ik(i)	=		The location of the endogenous variable. There are ned locations in the ik vector which must contain the variable numbers of the endogenous variables.

Note: The ik vector is not needed if nxs=1 since the first ned variables are endogenous and the next nex variables are exogenous.

4. IK VECTOR LINES FOR EXOGENOUS VARIABLES (IF nxs=0)

i ik(i)

i	=		The i^{th} element of the ik vector.
	=	0	For next option.
	=	-1	To list the ik vector.
ik(i)	=		The location of the exogenous variable. There are nex locations in the ik vector which must contain the variable numbers of the exogenous variables.

5. TEST VECTOR LINE(S)

i tes

i	=		The i endogenous variable.
	=	0	For next option.
	=	-1	To list test vector.
tes	=		Convergence test value for the i^{th} endogenous variable with a default value of .01.

6. KSET GRABBER LINE(S) (KSET VECTOR)

loc num

loc	=		i^{th} variable number.
	=	0	For next option.
	=	-1	To list the kset vector.
num	=		Random record number of the variables on the datal file.
		Note:	If nxs=1, the program will require a second set of KSET GRABBER LINES for the exogenous variables.

VI EXAMPLE

A. TERMINAL INPUT OUTPUT LINES FOR GNP MODEL

This listing represents the completed file for the GNP MODEL illustrated in AUTO and SIM. All parameters and vectors were defined in previous runs. In this run the complete coef file is read. Lines 8 to 46 list the file as read. The numbers in line 8 are 24 and 2. The program reads 2 coefficients for equation 3 and places them in locations 24 and 25 of the a vector. AUTO generated this coef file. The user must indicate the equation name and starting coefficient number on PART 1 OF EQUATION LINE. Lines 123 to 176 summarize the data specifications. The vector ibx contains the starting location for each variable in the d vector. This vector is used by SIM to locate the lag values of the variables.

```
 1T  sdat.o
 2   %    opened index for   1 with   258 words file name  data1
 3    open sub index for   1 with   130 words
 4      opened index for   2 with    33 words file name  data2
 5    up    to update coef vector
 6T  2
 7E    2
 8    eq-3         24     2
 9         0.0540    -8.7549
10    eq-4          6     2
11         0.0898     1.8372
12    eq-6          8     2
13         0.2938   -25.9875
14    eq-7         11     2
15         0.0524    -0.1058
16    eq-8         35     2
17         0.0044    -0.4130
18    eq-9         26     9
19         7.6205     0.5220      0.4101      0.3115      0.2263
20         0.1544     0.0958      0.0505      0.0186
21    eq-10         1     5
22       -12.9311     0.3852      0.2728      0.1711      0.0802
23    eq-11        22     2
24         0.8517     7.6346
25    eq-12        76    14
26         9.2496     0.0000      0.1357      0.2327      0.2908
27         0.3102     0.2908      0.2327      0.1357      0.0000
28         0.0795     0.0403      0.0139      0.0005
29    eq-16        37     4
30         0.0424     0.0681     -1.0444     59.9508
31    eq-17        41     3
32        -0.1440    -0.1157     41.1640
33    eq-18        44     2
34        36.3253   -21.6907
35    eq-21        52     2
36         2.5150    -4.5876
37    eq-22        56     4
38         0.8483    -0.0318      0.0983   -282.2268
39    eq-24        70     3
40         3.5028     1.2750   2601.1133
```

```
41     eq-25       61     5
42         19.3178      -1.6868      5.6250  -173.5498 -1427.8610
43     eq-26       66     4
44         6.9954      1.4756  -107.6242  7059.4326
45     eq-27       46     5
46        22.2737      -0.8220      1.4007  -327.0908  2002.9712
47    a vector
48      1  -12.9311  2     0.3852  3     0.2728  4     0.1711  5     0.0802
49      6     0.0898  7     1.8372  8     0.2938  9   -25.9875 10     3.5376
50     11     0.0524 12    -0.1058 13     0.0000 14     0.0000 15     0.0000
51     16     0.0000 17     0.0000 18     0.0000 19     0.0000 20     0.0000
52     21     0.0000 22     0.8517 23     7.6346 24     0.0540 25    -8.7549
53     26     7.6205 27     0.5220 28     0.4101 29     0.3115 30     0.2263
54     31     0.1544 32     0.0958 33     0.0505 34     0.0186 35     0.0044
55     36    -0.4130 37     0.0424 38     0.0681 39    -1.0444 40    59.9508
56     41    -0.1440 42    -0.1157 43    41.1640 44    36.3253 45   -21.6907
57     46    22.2737 47    -0.8220 48     1.4007 49  -327.0908 50  2002.9712
58     51     1.0940 52     2.5150 53    -4.5876 54     0.0194 55    -4.6009
59     56     0.8483 57    -0.0318 58     0.0983 59  -282.2268 60     0.0000
60     61    19.3178 62    -1.6868 63     5.6250 64  -173.5498 65-1427.8610
61     66     6.9954 67     1.4756 68  -107.6242 69  7059.4326 70     3.5028
62     71     1.2750 72  2601.1133 73     1.4210 74     1.0314 75    -4.4697
63     76     9.2496 77     0.0000 78     0.1357 79     0.2327 80     0.2908
64     81     0.3102 82     0.2908 83     0.2327 84     0.1357 85     0.0000
65     86     0.0795 87     0.0403 88     0.0139 89     0.0005 90     0.9650
66       j a(j)
67T
68     up   to update data file
69T   1
70     no ned nex nxs nl iy ip ib nod
71     30  37  16   0  7 46  1  1  30
72     no ned nex nxs nl iy ip ib nod
73T
74E    30  37  16   0  7 46   1  1  30
75    inl vector
76      1=  7     2=  0     3=  7     4=  0     5=  0     6=  0     7=  0
77      8=  0     9=  3    10=  0    11=  0    12=  0    13=  0    14=  0
78     15=  0    16=  0    17=  0    18=  7    19=  0    20=  0    21=  0
79     22=  7    23=  0    24=  0    25=  0    26=  7    27=  0    28=  3
80     29=  3    30=  0    31=  0    32=  0    33=  0    34=  0    35=  0
81     36=  0    37=  0    38=  0    39=  0    40=  0    41=  0    42=  0
82     43=  0    44=  0    45=  0    46=  0    47=  0    48=  0    49=  0
83     50=  0    51=  0    52=  0    53=  0
84     num lag
85T
86    ik vector for endogenous variables
87      1=  1     2=  2     3=  3     4=  6     5=  7     6=  8     7=  9
88      8= 11     9= 12    10= 13    11= 14    12= 15    13= 16    14= 17
89     15= 19    16= 21    17= 22    18= 25    19= 27    20= 28    21= 30
90     22= 31    23= 32    24= 33    25= 35    26= 38    27= 39    28= 40
91     29= 42    30= 44    31= 47    32= 48    33= 49    34= 50    35= 51
92     36= 52    37= 53
```

```
  i ik(i)

ik vector for exogenous variables
  1=  4     2=  5     3= 18     4= 23     5= 24     6= 26     7= 29
  8= 34     9= 36    10= 37    11= 41    12= 43    13= 45    14= 46
 15= 10    16= 20
  i ik(i)

 length of d vector is   97
test vector
  1= 0.01   2= 0.01   3= 0.01   4= 0.01   5= 0.01   6= 0.01   7= 0.01
  8= 0.01   9= 0.01  10= 0.01  11= 0.01  12= 0.01  13= 0.01  14= 0.01
 15= 0.01  16= 0.01  17= 0.01  18= 0.01  19= 0.01  20= 0.01  21= 0.01
 22= 0.01  23= 0.01  24= 0.01  25= 0.01  26= 0.01  27= 0.01  28= 0.01
 29= 0.01  30= 0.01  31= 0.01  32= 0.01  33= 0.01  34= 0.01  35= 0.01
 36= 0.01  37= 0.01  38= 0.01  39= 0.01  40= 0.01  41= 0.01  42= 0.01
 43= 0.01  44= 0.01  45= 0.01  46= 0.01  47= 0.01  48= 0.01  49= 0.01
 50= 0.01  51= 0.01  52= 0.01  53= 0.01
  i tes

kset vector
  1=  1     2=  2     3=  3     4=  6     5=  5     6= 32     7=120
  8= 63     9=121    10=129    11= 51    12= 19    13= 60    14= 25
 15= 42    16=122    17=123    18= 49    19=119    20=129    21= 74
 22= 79    23= 76    24= 77    25= 78    26= 72    27= 33    28=103
 29= 47    30= 80    31=104    32= 30    33= 28    34=105    35= 88
 36=116    37= 26    38= 98    39=115    40=113    41=102    42=114
 43=101    44=112    45=117    46=118    47= 96    48= 91    49=124
 50=125    51=127    52=128    53=126
 loc num

```

1 num		name	test	kset	lag	ibx	no	iy	ip	ib
1	e	gnp$72	0.010	1	7	1	30	46	1	1
2	e	c$72	0.010	2	0	9	30	46	1	1
3	e	i$72	0.010	3	7	10	30	46	1	1
4	x	g$72	0.010	6	0	18	30	46	1	1
5	x	ex$72	0.010	5	0	19	30	46	1	1
6	e	dep$72	0.010	32	0	20	30	46	1	1
7	e	taxin$72	0.010	120	0	21	30	46	1	1
8	e	ni$72	0.010	63	0	22	30	46	1	1
9	e	ys$72	0.010	121	3	23	30	46	1	1
10	x	empty	0.010	129	0	27	1	46	1	1
11	e	taxes	0.010	51	0	28	30	46	1	1
12	e	gt	0.010	19	0	29	30	46	1	1
13	e	is	0.010	60	0	30	30	46	1	1
14	e	bt	0.010	25	0	31	30	46	1	1
15	e	surplus	0.010	42	0	32	30	46	1	1
16	e	ele/y	0.010	122	0	33	29	47	1	1
17	e	fuel/y	0.010	123	0	34	29	47	1	1
18	x	ipdgnp	0.010	49	7	35	30	46	1	1
19	e	residual	0.010	119	0	43	30	46	1	1
20	x	empty	0.010	129	0	44	1	46	1	1

21	e	unempl	0.010	74	0	45	29	47	1	1
22	e	k$72	0.010	79	7	46	30	46	1	1
23	x	pop65	0.010	76	0	54	30	46	1	1
24	x	time46=1	0.010	77	0	55	30	46	1	1
25	e	k$72*	0.010	78	0	56	23	53	1	1
26	x	baa	0.010	72	7	57	30	46	1	1
27	e	manhours	0.010	33	0	65	29	47	1	1
28	e	cost-k	0.010	103	3	66	30	46	1	1
29	x	w-rate	0.010	47	3	70	29	47	1	1
30	e	k/o	0.010	80	0	74	29	47	1	1
31	e	prop-un	0.010	104	0	75	29	47	1	1
32	e	hours	0.010	30	0	76	29	47	1	1
33	e	employ	0.010	28	0	77	29	47	1	1
34	x	labor	0.010	105	0	78	29	47	1	1
35	e	fuel-eu	0.010	88	0	79	29	47	1	1
36	x	pbf	0.010	116	0	80	29	47	1	1
37	x	pop	0.010	26	0	81	80	46	1	1
38	e	fuel-trn	0.010	98	0	82	29	47	1	1
39	e	fu-in/ot	0.010	115	0	83	29	47	1	1
40	e	el-r+c	0.010	113	0	84	29	47	1	1
41	x	p-elc-h	0.010	102	0	85	30	46	1	1
42	e	el-la+ot	0.010	114	0	86	29	47	1	1
43	x	p-elc-in	0.010	101	0	87	30	46	1	1
44	e	el-tot	0.010	112	0	88	29	47	1	1
45	x	el-nuc	0.010	117	0	89	19	57	1	1
46	x	el-hy	0.010	118	0	90	29	47	1	1
47	e	fuel-h/c	0.010	96	0	91	29	47	1	1
48	e	fuel-all	0.010	91	0	92	29	47	1	1
49	e	c/gnp	0.010	124	0	93	30	46	1	1
50	e	k/n	0.010	125	0	94	29	47	1	1
51	e	i/gnp	0.010	127	0	95	30	46	1	1
52	e	g/gnp	0.010	128	0	96	30	46	1	1
53	e	c/pop	0.010	126	0	97	30	46	1	1

closed index for 2 with 33 words

% bye

SIM
February 1977 Version for IIASA's pdp11

I INTRODUCTION

SIM is designed to simulate nonlinear econometric models. The user must supply the necessary Fortran code to represent his equations. SIM is the fifth component of an interactive econometric model building system for the pdp11. SIM uses a data base produced by BANK, a coef file produced by estimating the model with AUTO or ECON, and an input data file produced by SIMDAT.

II FORMULATION

Each equation of the model is normalized for a different endogenous variable. The equations are divided into two parts: a constant part, and a simultaneous part.

The constant part (predetermined) can be evaluated with only predetermined variables and parameters.

The simultaneous parts contain current period endogenous variables and other constant terms which are not additive. A particular equation can be expressed as:

$$y_i = f_i(y,z) + g_i(z)$$

where y_i is the i^{th} endogenous variable,

y is the vector of endogenous variables,

z is the vector of predetermined variables consisting of exogenous and lagged endogenous,

f_i is the simultaneous part of the equation,

g_i is the constant part of the equation.

The user writes the necessary Fortran code in SUBROUTINE CONST to evaluate the term $g_i(z)$ as c_i. All recursive endogenous variables should be evaluated in SUBROUTINE CONST. The user then writes the code to evaluate the y_i in SUBROUTINE SOLVE. An equation would be represented as:

$$y_i = f_i(y,z) + c_i.$$

A Gauss-Seidel iteration procedure is used to evaluate the system of equations. The program starts with an initial vector y^0 and then uses the equations coded in SUBROUTINE SOLVE to update the solution vector. The elements of y are updated as the new values are computed. The i^{th} equation in the 6^{th} iteration would be revalued as:

$$y_i^6 = f_i(y_1^6, y_2^6, \ldots, y_{i-1}^6, y_{i+1}^5, \ldots, y_n^5, z) + c_i.$$

When using the Gauss-Seidel algorithm, the convergence of the solution vector is dependent on the order of the equations and the normalization rule. For more detail see the section on ARRANGEMENT OF THE EQUATIONS.

III <u>PROGRAM</u>

The computer program consists of a main program with six subroutines CONST, GRAPH, MATOUT, SOLN, SOLVE, and POST; and the two functions E and X. SIM also requires the RR routines. The purpose of each component will be described below in order.

A. <u>main.f</u>

The main program opens the data2 file and initializes the data set required for a simulation. This data file was prepared by SIMDAT. The user specifies the options and the starting date for the simulation. SIM reads the vector of lag values for the variables. These lag values are stored in the d vector. The main program then reads the data vectors and updates the d vector until the starting date is reached. A maximum of nl records are read.

After the lag vector is initialized and all necessary parameters are entered, the main program calls SUBROUTINE SOLN. Control is returned to the main program after a solution is found for the first period. Then the solution vector is recorded, a new data vector is read from the data2 file, and the lag vector is updated for the next period solution. This process continues until a period specified by the user is reached. At this time the user can change parameters and continue, or print results and quit.

B. <u>const.f</u>

SUBROUTINE CONST is used to evaluate the constant or predetermined parts of the model. The evaluation is done before SUBROUTINE SOLVE is used in the iterative scheme. All terms whose value remains constant during the iterative scheme should be placed in SUBROUTINE CONST. This procedure will greatly reduce the number of operations per iteration. The actual Fortran coding of the model will be explained in the section TUTORIAL EXAMPLE USING A GNP MODEL.

C. matout.f

SUBROUTINE MATOUT prints the solution matrix in a readable form. The solution is saved in the R matrix. For a large model it may be necessary to eliminate this routine. The solution vector could be written to a random file and analyzed later with another program.

D. soln.f

SUBROUTINE SOLN manages the solution of the model for a particular period. If the model is solved for 20 periods then SUBROUTINE SOLN is called 20 times. SUBROUTINE SOLN first calls SUBROUTINE CONST to evaluate the predetermined parts. It then calls SUBROUTINE SOLVE repeatedly until the solution vector converges. After a solution is found SUBROUTINE POST is called to evaluate the post recursive equations of the model.

SUBROUTINE SOLN stores the solution vector in the R matrix and calls SUBROUTINE MATOUT when the matrix is full. If the user is running a residual check SUBROUTINE SOLN computes the residuals and places them in the R matrix.

E. solve.f

SUBROUTINE SOLVE evaluates the endogenous variables for each iteration. The user must supply the necessary Fortran code for the equations.

F. post.f

SUBROUTINE POST evaluates the post recursive equation after a solution is obtained. The user must supply the necessary Fortran code for these equations.

G. fune.f

FUNCTION E is used by SUBROUTINE CONST to locate the lag values of the exogenous variables in the d vector. This function is only used if the variables are divided into two groups (nxs=1).

H. funx.f

FUNCTION X is used by SUBROUTINE CONST to locate the lag values of the variables in the d vector. This function is not used for exogenous variables if nxs=1.

IV CODING A MODEL

The user has to provide the necessary Fortran statements to evaluate the equations of his model. A small macro model of the USA will be used to illustrate the steps necessary to simulate a model. The coding for each equation will be given along with a brief explanation. Then the coding of each equation will be placed in the appropriate subroutine (CONST, SOLVE, and POST). Listings of these subroutines are in sections F, G, and H.

A. DEFINITION OF SYMBOLS USED TO CODE THE MODEL

a(i) = i^{th} coefficient of the model.

c(i) = Constant part of the equation for the i^{th} endogenous variable.

y(i) = The i^{th} endogenous variable on the LHS (left hand side) of an equation.

z(i) = The i^{th} variable on the RHS (right hand side) of an equation. Not used for the exogenous variables if nxs=1 on the PARAMETER LINE.

Note: The user must use this convention, as y(i) cannot appear on the RHS or the program will be unable to calculate residuals.

x(t,i) = The t^{th} lag of the i^{th} variable. Used to retrieve a lag value from the d vector (not used for exogenous variables if nxs=1).

xl(i) = The 1^{st} lag of the i^{th} variable (not used for exogenous variables if nxs=1). Value will be -1.0e30 if a lag was not declared for this variable in SIMDAT.

Note: There are two methods for numbering the variables of a model. One method simply numbers them 1 through nv (nxs=0). The second method divides the variables into two groups: endogenous and exogenous variables (nxs=1). The endogenous variables are numbered 1 through ned, and the exogenous variables are numbered 1 through nex. ned plus nex equals nv. When the second method is used only the endogenous variables are referenced with z(i), x(t,i), or xl(i). The exogenous variables are referenced with ex(i), e(t,i) and el(t).

ex(i) = The i^{th} exogenous variable. Used when the variables are divided into two groups; i.e. nxs=1 on the PARAMETER LINE.

e(t,i) = The t^{th} lag of the i^{th} exogenous variable. Used when the variables are divided into two groups.

el(i) = The 1^{st} lag of the i^{th} exogenous variable. Used when the variables are divided into two groups. The value will be -1.0e30 if a lag was not declared for this variable.

Note: The functions x(t,i) and e(t,i) can be used in SUBROUTINE SOLVE but it is a much better idea to evaluate a b(j)=x(t,i) in SUBROUTINE CONST. The b(j) is used in SUBROUTINE SOLVE instead of x(t,i). This procedure will reduce the number of calculations necessary for an iteration.

b(i) = A constant term for an equation in SUBROUTINE SOLVE. b(i) is used for multiplicative constant terms. They are evaluated in SUBROUTINE CONST.

d(150) = The d vector for the lagged values. SIMDAT calculates the length requirement for the d vector. The user may have to adjust the length in SIM and SIMDAT for a model with many lag values.

xnor(i) = The normalization value for the i^{th} endogenous variable with a default value of 1.

logic(i) = i^{th} value of logic vector. This vector is used when alternative forms of an equation are required. The default values are true and with the LOGIC VECTOR MODIFICATION LINE(S) the user can change any value to false.

nrr = Number of solutions. nrr is incremented by 1 in the main program after a solution is found.

i1,i2 = Date for current solution: i1 is the year, and i2 is the period.

B. TUTORIAL EXAMPLE USING A GNP MODEL

This example is a continuation of the examples used for BANK, AUTO, and SIMDAT. The basic data file was prepared with BANK. The coefficients were estimated with AUTO. The data file for SIM was prepared with SIMDAT.

The user must now provide the Fortran coding for his model. The following table for the GNP Model has been carefully labeled to facilitate the actual coding. The numbers above the variable names are assigned by SIMDAT (see SIMDAT write-up). The data specification table from SIMDAT appears in section D. The numbers in brackets above the coefficients are assigned by AUTO. The user can edit the card file to change the coefficient assignment (see AUTO write-up). The summary table from AUTO can be used instead of the following table.

The Fortran code used in SIM appears below each equation. The last line (only line) is placed in SUBROUTINE SOLVE unless labeled {post} or {const}. The lines that are labeled are placed in SUBROUTINE {post} or {const}. All the beginning lines are placed in SUBROUTINE CONST.

The d vector in the common statements of SIM has a dimension of 150. The output file from SIMDAT gives the required length of the d vector. Line 101 of example A in the write-up for SIMDAT gives the length as 97. For models with many lag distributions it may be necessary to increase the size of the d vector.

C. POTENTIAL GNP MODEL FOR THE USA

1) Definition of real GNP

```
   1        2        3        4        5
GNP$72 = C$72 + I$72 + G$72 + EX$72

  y(1)=z(2)+z(3)+z(4)+z(5)
```

2) Definition of real national income

```
  8         1         6           7         14/18   19
NI$72 = GNP$72 - DEP$72 - TAXIN$72 - BT/P - RES

  y(8)=z(1)-z(6)-z(7)-z(14)/z(18)*100.0-z(19)
```

3) Capital consumption allowance with CCA

```
   6       [24]  22        [25]
DEP$72 = .054* K$72_-1 - 8.75

  y(6)=a(24)*x1(22)+a(25)  {const}
```

4) Indirect business tax function

```
18     7          [6]   18      1       [7]
P * TAXIN$72 = .09 * P * GNP$72 + 1.84

  xnor(7)=z(18)*0.01

  y(7)=(a(6)*z(18)*z(1)*0.01+a(7))/(z(18)*0.01)
```

5) Definition of real spendable income

```
  9       8          11     12    14    13 /18
YS$72 = NI$72 - (TAXES - GT - BT - IS)/P

  y(9)=z(8)-(z(11)-z(12)-z(13)-z(14))/z(18)*100.0
```

6) Income tax function

```
  11     [8]    18     8       [9]
TAXES = .29 * P * NI$72 - 25.99

  y(11)=a(8)*z(18)*z(8)*0.01+a(9)
```

7) Government transfer payments to persons

```
12   18     23          21           [11]     24      [12]
GT = P * (POP65 + UNEMP) EXP (.052 * TIME - .106)
```

```
  xnor(12)=100.0/(z(18)*(z(23)+z(21)))
  b(9)=z(18)*a(18)*.01
  c(12)=exp(a(11)*z(24)+a(12))
  z(12)=alog(z(12)*xnor(12))/xnor(12)   {post}

  y(12)=z(18)*(z(23)+z(21))*c(12)*0.01+b(9)
```

8) Business transfer payments

```
14     [35]    18     1       [36]
BT = .0044 * P * GNP$72 - .41
```

```
  y(14)=a(35)*z(1)*z(18)*0.01+a(36)
```

9) Net interest income

```
13     7      18        3        26       [26]
IS =  Σ wi  * P-i  * I$72-i*BAA-i  + 7.62
     i=0
 w0 = .522[27]           w4 = .154[31]
 w1 = .410[28]           w5 = .096[32]
 w2 = .312[29]           w6 = .051[33]
 w3 = .226[30]           w7 = .019[34]
```

$$IS = \sum_{i=0}^{7} w_i * P_{-i} * I\$72_{-i} * BAA_{-i} + 7.62$$

```
      r=0.0
      do 5 i=1,7
5     r=r+a(i+27)*x(i,3)*x(i,18)*x(i,26)*0.0001
      c(13)=a(26)+r

      y(13)=a(27)*z(3)*z(18)*z(26)*0.0001+c(13)
```

10) Consumption function

```
 2       [2]     9      [3]    9         [4]     9
C$72 = .39 *YS$72 + .27 *YS$72-1 + .17 *YS$72-2

        [5]    9         [1]
      + .08*YS$72-3 - 12.43
```

$$C\$72 = .39 * YS\$72 + .27 * YS\$72_{-1} + .17 * YS\$72_{-2} + .08 * YS\$72_{-3} - 12.43$$

```
  c(2)=a(1)+a(3)*x1(9)+a(4)*x(2,9)+a(5)*x(3,9)

  y(2)=a(2)*z(9)+c(2)
```

11) Investment function

```
 3      [22]     25      22               6      [23]
I$72 = .85 *  (K$72*-  K$72-1  )  + DEP$72 + 7.6
```

```
 c(3)=a(23)-a(22)*x1(22)

 y(3)=a(22)*z(25)+z(6)+c(3)
```

12) Desired capital stock

```
                                                     29/28
 25        7        1          4        1        |w-rate-i|         [76]
K$72*  =   Σ  wiGNP$72-i  +    Σ  viGNP$72-i  |---------|1/2   + 9.25
          i=Ø                 i=Ø                |cost-k-i|

   wØ  =   .ØØØ[77]      w4  = .31Ø[81]     vØ  = .Ø8Ø [86]
   w1  =   .136[78]      w5  = .291[82]     v1  = .Ø4Ø [87]
   w2  =   .233[79]      w6  = .233[83]     v2  = .Ø14 [88]
   w3  =   .291[8Ø]      w7  = .136[84]     v3  = .ØØØ5[89]
```

```
      ss=Ø.Ø
      do 5 i=1,7
5     ss=ss+a(i+77)*x(i,1)
      do 6 i=1,3
6     ss=ss+a(i+86)*x(i,1)*sqrt(x(i,29)/x(i,28))
      c(25)=ss+a(76)
      b(8)=a(77)+a(86)*sqrt(z(29)/z(28))

      y(25)=b(8)*z(1)+c(25)
```

13) Actual capital stock

```
 22      22        3        6
K$72  = K$72-1  + I$72  - DEP$72
```

```
 c(22)=x1(22)

 y(22)=z(3)-z(6)+c(22)
```

14) Capital output ratio

```
 3Ø     22       1
K/Ø  = K$72/GNP$72
```

```
 y(3Ø)=z(22)/z(1)   {post}
```

15) Cost of capital

```
  28          26            [24]  18
COST-K = (BAA * .01 + .054)  *P

 y(28)=(z(26)*0.01+0.054)*z(18)   {const}
```

16) Derived demand for labor - manhours index

```
                                                      28/29
   27         [37]     1       [38]        1      |cost-k|
MANHOURS = .042 *GNP$72 + .068  * GNP$72  |------|1/2
                                                  |w-rate|

              [39]    24     [40]
            -1.04 * TIME + 59.9

 b(1)=a(37)+a(38)*sqrt(z(28)/z(29))
 c(27)=a(39)*z(24)+a(40)
 y(27)=b(1)*z(1)+c(27)
```

17) Average weekly hours for total private nonagricultural

```
 32       [41]      31        [42]      24     [43]
HOURS = -.144 * PROP-UN - .166  * TIME + 41.2

 c(32)=a(42)*z(24)+a(43)
 y(32)=a(41)*z(31)+c(32)
```

18) Employment of workers

```
                        27/32
  33       [44]       MANHOURS      [45]
EMPLOY = 36.33 *  --------  - 21.69
                       HOURS

 y(33)=a(44)*z(27)/z(32)+a(45)
```

19) Total unemployment

```
  21       34       33
UNEMP = LABOR - EMPLOY

 y(21)=z(34)-z(33)
```

20) Proportion of civilian labor force unemployed

```
  31          21     33
PROP-UN = UNEMP/EMPLOY * 100

 y(31)=z(21)/z(33)*100.0
```

Demand for Electricity

21) Residential and small light & power per capita

$$\overset{40/37}{\frac{\text{EL-R+C}}{\text{POP}}} = \overset{[52]}{2.52}\ \overset{2/37}{\frac{\text{C\$72}}{\text{POP}}} - \overset{[53]}{4.59}$$

```
xnor(40)=1.0/z(37)
c(40)=a(53)*z(37)

y(40)=a(52)*z(2)+c(40)   {post}
```

22) Electric sales to large light and power and misc.

$$\overset{42}{\text{EL-LA+OT}} = \overset{[56]}{.848} * \overset{1}{\text{GNP\$72}} - \overset{[57]}{.032} * \overset{1}{\text{GNP\$72}} \overset{29/43}{\left|\frac{\text{w-rate}}{\text{Pe}}\right|^{1/2}}$$

$$+ \overset{[58]}{.098} * \overset{1}{\text{GNP\$72}} \overset{28/43}{\left|\frac{\text{cost-k}}{\text{Pe}}\right|^{1/2}} - \overset{[59]}{282.2}$$

```
b(5)=a(57)*sqrt(z(29)/z(43))+a(58)*sqrt(z(28)/z(43))+a(56)
c(42)=a(59)

y(42)=z(1)*b(5)+c(42)   {post}
```

23) Total electric sales

$$\overset{44}{\text{EL-TOT}} = \overset{42}{\text{EL-LA+OT}} + \overset{40}{\text{EL+R+C}}$$

```
y(44)=z(42)+z(40)   {post}
```

Demand for Fuel

24) Household and commercial sectors

$$\overset{47}{\text{FUEL-H/C}} = \overset{[70]}{3.50} * \overset{2}{\text{C\$72}} + \overset{[71]}{1.275} * \overset{2}{\text{C\$72}} \overset{18/36}{\left|\frac{\text{P}}{\text{PBF}}\right|^{1/2}}$$

$$+ \overset{[72]}{2601.1}$$

```
b(7)=a(70)+a(71)*sqrt(z(18)/z(36))
c(47)=a(72)

y(47)=z(2)*b(7)+c(47)   {post}
```

25) Transportation sector

$$\overset{38}{\text{FUEL-TRN}} = \overset{[61]}{19.32} * \overset{1}{\text{GNP\$72}} - \overset{[62]}{1.69} * \overset{1}{\text{GNP\$72}} \overset{29/36}{\left|\frac{\text{w-rate}}{\text{PBF}}\right|} 1/2$$

$$+\overset{[63]}{5.62} * \overset{1}{\text{GNP\$}} \overset{28/36}{\left|\frac{\text{cost-k}}{\text{PBF}}\right|} 1/2 - \overset{[64]}{173.5} * \overset{24}{\text{TIME}}$$

$$\overset{[65]}{-1428.9}$$

```
b(3)=a(61)+a(62)*sqrt(z(29)/z(36))+a(63)*sqrt(z(28)/z(36))
c(38)=a(64)*z(24)+a(65)

y(38)=z(1)*b(3)+c(38)   {post}
```

26) Industrial sector and miscellaneous sector

$$\overset{39}{\text{FU-IN/OT}} = \overset{[66]}{7.0} * \overset{1}{\text{GNP\$72}} + \overset{[67]}{1.48} * \overset{1}{\text{GNP\$72}} \overset{28/36}{\left|\frac{\text{cost-k}}{\text{PBF}}\right|} 1/2$$

$$\overset{[68]}{-107.6} * \overset{24}{\text{TIME}} + \overset{[69]}{7059.4}$$

```
b(6)=a(66)+a(67)*sqrt(z(28)/z(36))
c(39)=a(68)*z(24)+a(69)

y(39)=z(1)*b(6)+c(39)   {post}
```

27) Electrical utilities sector

$$\overset{35}{\text{FUEL-EU}} = \overset{[46]}{22.3} * \text{EL} - \overset{[47]}{.82} * \text{EL} \overset{29/36}{\left|\frac{\text{w-rate}}{\text{PBF}}\right|} 1/2$$

$$+ \overset{[48]}{1.4} * \text{EL} \overset{28/36}{\left|\frac{\text{cost-k}}{\text{PBF}}\right|} 1/2 - \overset{[49]}{327.1} * \overset{24}{\text{TIME}}$$

$$+ \overset{[50]}{2003.0}$$

```
b(2)=a(46)+a(47)*sqrt(z(29)/z(36))+a(48)*sqrt(z(28)/z(36))
c(35)=a(49)*z(24)+a(50)

y(35)=(a(51)*z(44)-z(45)-z(46))*b(2)+c(35)   {post}
```

28) Electricity produced by fuel

$$\text{EL} = \overset{[51]}{1.094} * \overset{44}{\text{EL-TOT}} - \overset{45}{\text{EL-NUC}} - \overset{46}{\text{EL-HY}}$$

29) Total demand for fuel

```
    48          47            38           39           35
FUEL-ALL = FUEL-H/C + FUEL-TRN + FU-IN/OT + FUEL-EU
```

y(48)=z(47)+z(39)+z(38)+z(35) {post}

D. DATA SPECIFICATIONS FROM SIMDAT

num		name	test	kset	lag	ibx	no	iy	ip	ib
1	e	gnp$72	0.010	1	7	1	30	46	1	1
2	e	c$72	0.010	2	0	9	30	46	1	1
3	e	i$72	0.010	3	7	10	30	46	1	1
4	x	g$72	0.010	6	0	18	30	46	1	1
5	x	ex$72	0.010	5	0	19	30	46	1	1
6	e	dep$72	0.010	32	0	20	30	46	1	1
7	e	taxin$72	0.010	120	0	21	30	46	1	1
8	e	ni$72	0.010	63	0	22	30	46	1	1
9	e	ys$72	0.010	121	3	23	30	46	1	1
10	x	empty	0.010	129	0	27	1	46	1	1
11	e	taxes	0.010	51	0	28	30	46	1	1
12	e	gt	0.010	19	0	29	30	46	1	1
13	e	is	0.010	60	0	30	30	46	1	1
14	e	bt	0.010	25	0	31	30	46	1	1
15	e	surplus	0.010	42	0	32	30	46	1	1
16	e	ele/y	0.010	122	0	33	29	47	1	1
17	e	fuel/y	0.010	123	0	34	29	47	1	1
18	x	ipdgnp	0.010	49	7	35	30	46	1	1
19	e	residual	0.010	119	0	43	30	46	1	1
20	x	empty	0.010	129	0	44	1	46	1	1
21	e	unempl	0.010	74	0	45	29	47	1	1
22	e	k$72	0.010	79	7	46	30	46	1	1
23	x	pop65	0.010	76	0	54	30	46	1	1
24	x	time46=1	0.010	77	0	55	30	46	1	1
25	e	k$72*	0.010	78	0	56	23	53	1	1
26	x	baa	0.010	72	7	57	30	46	1	1
27	e	manhours	0.010	33	0	65	29	47	1	1
28	e	cost-k	0.010	103	3	66	30	46	1	1
29	x	w-rate	0.010	47	3	70	29	47	1	1
30	e	k/o	0.010	80	0	74	29	47	1	1
31	e	prop-un	0.010	104	0	75	29	47	1	1
32	e	hours	0.010	30	0	76	29	47	1	1
33	e	employ	0.010	28	0	77	29	47	1	1
34	x	labor	0.010	105	0	78	29	47	1	1
35	e	fuel-eu	0.010	88	0	79	29	47	1	1
36	x	pbf	0.010	116	0	80	29	47	1	1
37	x	pop	0.010	26	0	81	80	46	1	1
38	e	fuel-trn	0.010	98	0	82	29	47	1	1
39	e	fu-in/ot	0.010	115	0	83	29	47	1	1
40	e	el-r+c	0.010	113	0	84	29	47	1	1
41	x	p-elc-h	0.010	102	0	85	30	46	1	1

42	e	el-la+ot	0.010	114	0	86	29	47	1	1
43	x	p-elc-in	0.010	101	0	87	30	46	1	1
44	e	el-tot	0.010	112	0	88	29	47	1	1
45	x	el-nuc	0.010	117	0	89	19	57	1	1
46	x	el-hy	0.010	118	0	90	29	47	1	1
47	e	fuel-h/c	0.010	96	0	91	29	47	1	1
48	e	fuel-all	0.010	91	0	92	29	47	1	1
49	e	c/gnp	0.010	124	0	93	30	46	1	1
50	e	k/n	0.010	125	0	94	29	47	1	1
51	e	i/gnp	0.010	127	0	95	30	46	1	1
52	e	g/gnp	0.010	128	0	96	30	46	1	1
53	e	c/pop	0.010	126	0	97	30	46	1	1

E. EXPLANATION OF THE FORTRAN STATEMENTS

The actual Fortran statements used in SIM appear in section B under the equations. The Fortran statements were taken from SUBROUTINE CONST, SOLVE, and POST (Section F, G, and H contain the listngs of these subroutines).

The explanation of the Fortran statement assumes the user is familiar with Fortran. Only the equations with special features will be explained.

eqn-2 The factor *100.0 or *0.01 is used with the price index p to convert p=100.0 in 1972 to p=1.0

eqn-3 This equation has no simulation part and should be evaluated in SUBROUTINE CONST. The term xl(22) is variable 22 lagged one period. The user must define all lags in SIMDAT. Line 22 of section D indicates that variable k$72 has 7 lags.

eqn-4 This equation requires a normalization factor, xnor(7), for the residual calculations. The equation was estimated with AUTO as

$$py = xb + e$$

and used in SIM as

$$y = xb/p + e/p.$$

When checking the coding of the equations, the user specifies sim=2 on the SOLUTION CONTROL LINE and SIM calculates the normalized residuals. That is, residual e_i is multiplied by xnor(i). The default value for xnor(i) is 1. The residuals from SIM must check exactly with the residuals from AUTO before the coding of an equation can be considered correct.

eqn-7 The term b(9) was added to allow the user to specify a change to government transfer in constant \$72. The normal value for a(18) is 0. The user can run a simulation with more or less GT by changing a(18).

To check the residuals in this equation requires special attention. The equation was estimated as

ln(y/pz) = a + bt + e

and used in SIM as

y = pz*exp(a + bt).

This equation requires special attention because the normalization of the equation for y requires a nonmultiplicative transformation. With xnor(12)=1/pz, and exp(zzx)=zzx in SUBROUTINE CONST, SIM will compute the $\hat{y}$ estimated by AUTO. This equation also requires the actual z(12) to be transformed in SUBROUTINE POST. The transformation

z(12)=ln(y/pz)*pz

can be written as

z(12)=alog(z(12)*xnor(12))/xnor(12).

SUBROUTINE SOLN computes the residuals as

e(12)=(z(12)-y(12))*xnor(12).

These residuals will now compare with the residuals from AUTO. The exp(zzx)=zzx is only used when checking log transformed residuals and <u>must</u> be removed after coding is checked.

eqn-9 This equation has a distributed lag variable. The weights are stored in the a vector from a(27) to a(34). The FUNCTION X is used to locate the lag values in the d vector. The term .0001 is to change the unit of P and BAA.

eqn-10 A do loop could have been used for this distributed lag but it is simpler to add the variables together. xl(9) is used instead of x(1,9) to save on operations.

eqn-12 This equation has two distributed lags of different lengths. GNP$72 appears in both terms and the computation in SUBROUTINE SOLVE is reduced by using b(8).

eqn-13 The c(22) is not necessary, as xl(22) could have been used directly in SUBROUTINE SOLVE; however, it makes the coding more consistent and easier to modify at a later date.

eqn-14 This equation is placed in SUBROUTINE POST. K/O is not used in the simulation solution.

eqn-15 This equation consists of parameters and exogenous variables. It is evaluated in SUBROUTINE CONST. This equation must be placed before the equations and constant terms using z(28).

eqn-16 b(1) is used to reduce calculation in SUBROUTINE SOLVE. In this way, b(1) is evaluated only once instead of every iteration.

eqn-21 A normalization factor is required for this equation to compute the residuals.

eqn-22 The term b(5) is not necessary since this equation is evaluated in SUBROUTINE POST; however, the model may be changed so that this variable becomes a simultaneous endogenous variable.

eqn-28 This equation was substituted directly into equation 27.

F. LISTING OF SUBROUTINE CONST

There are a few special options for the model that are coded in SUBROUTINE CONST. The line marked g$72 allows the user to add a given amount to g$72. The user can specify a value for a_{15} with a COEFFICIENT VECTOR MODIFICATION LINE.

A more complicated option is coded with the lines marked newk-Ø to newk-1Ø. This option allows the user to multiply the desired capital stock k$72* by a constant factor, a_{14} (an energy production scenario might require 25% more capital stock). The user must specify how the transition to the new capital requirements will be accomplished. a_{13} is the proportional step made each time period towards the new capital requirement. For example, with a_{13} =.Ø4 it would take 25 (1/.Ø4) years to switch to the new k*/Ø ratio. Besides a straight line transition the user can change the capital requirement with a 4 year step function. For example with logic(4)=f and a_{13}=.2 it would take 5 steps (the first one starting in year 1 and last one starting in year

16 of the simulation) to reach to the new K*/O ratio. With logic(4)=t a straight line transition period is used, and with logic(4)=f a step function with 4 year intervals is used.

G. LISTING OF SUBROUTINE SOLVE

SUBROUTINE SOLVE has two versions of the model. With logic(1)=t the original version of the model as represented in section B is used. With logic(1)=f the model becomes a potential GNP model. In this version, g$72 is endogenous and employment is set equal to a a_{90} times the labor force. The user can specify a_{90}=.96 for 4% unemployment. g$72 will be adjusted so aggregate demand equals aggregate supply.

There are two versions of the equations for employ, manhours, gnp$72, and g$72. The program will switch to version two if the employment rate is greater than a_{90}. The user can set logic(3)=f for version 1 with no ceiling on employment. At the bottom of SUBROUTINE CONST logic(2)=t is used to set logic(1)=t for the beginning of each solution. With logic(1)=t the model will start each period as version one and then switch to version two if employment is above the ceiling and logic(3)=t.

H. LISTING OF SUBROUTINE POST

All of the equations in SUBROUTINE POST are post recursive. Their values do not affect the solution for the simultaneous endogenous variables. At the bottom of SUBROUTINE POST is the transformation of z_{12} for the residuals calculation. Equations estimated in log form generally require a transformation of the actual dependent variable so that the residuals from SIM will match the residuals from the graph option of AUTO.

```
      subroutine const(y,ex,el)
      common i4,i5,i6,d(150),ia,a(200),i1,i2,pa,z(100),c(55),xnor(55)
     1      ,ibx(55),ca(55),inl(55),b(55),nv1,iy1,ip1,ib1,lab(56),ngr
     2      ,ik(55),test(55),logic(100),xl(55),sim,nvc,ned,nex,nxs,nl
     3      ,max,nt,ned1,nr,date1,date2,lis,title(12),ncol,nit,nvc1
     4      ,maxr,it(55),kset(55),nrr
      real*8 lab,ld,label
      integer date1,date2,error,sim,pa
      logical*1 logic,ltu,lfa
      dimension y(100),ex(100),el(100)
c     exp(zzx)=zzx
      do 2 i=1,ned1
      xl(i)=-1.0e30
      if(inl(i).gt.0) xl(i)=x(1,i)
2     xnor(i)=1.0
      if(nxs.eq.1) go to 4
      do 3 i=1,nex
      el(i)=-1.0e30
      if(inl(i+ned).gt.0) el(i)=e(1,i)
3     ex(i)=z(i+ned)
4     continue
      y(28)=(z(26)*0.01+0.054)*z(18)                               cost-k
      y(4)=z(4)+a(15)                                              g$72
      xnor(7)=z(18)*0.01                                           taxin$72
      c(2)=a(1)+a(3)*xl(9)+a(4)*x(2,9)+a(5)*x(3,9)                 c$72
      b(9)=z(18)*a(18)*.01                                         gt-0
      xnor(12)=100.0/(z(18)*(z(23)+z(21)))                         gt-1
      c(12)=exp(a(11)*z(24)+a(12))                                 gt-2
      r=0.0                                                        is-0
      ss=0.0                                                       k$72*-0
      do 5 i=1,7                                                   i&k-00
      r=r+a(i+27)*x(i,3)*x(i,18)*x(i,26)*0.0001                    is-1
5     ss=ss+a(i+77)*x(i,1)                                         k$72*-1
      do 6 i=1,3                                                   k$72*-2
6     ss=ss+a(i+86)*x(i,1)*sqrt(x(i,29)/x(i,28))                   k$72*-3
      b(8)=a(77)+a(86)*sqrt(z(29)/z(28))                           k$72*-4
      c(25)=ss+a(76)                                               k$72*-5
      if(nrr.eq.1.or.sim.gt.0) pr=1.0                              newk-0
      if(nrr.eq.1) ics=0                                           newk-1
      if(.not.logic(4).and.mod(ics,4).gt.0) go to 7                newk-2
      pr=pr-a(13)                                                  newk-3
      if(pr.lt.0.0) pr=0.0                                         newk-4
      cf=pr+(1.0-pr)*a(14)                                         newk-5
      if(cf.ne.1.0) write(8,1) i1,i2,cf                            newk-6
1     format(i5,i2,f5.2)                                           newk-7
7     b(8)=b(8)*cf                                                 newk-8
      ics=ics+1                                                    newk-9
      c(25)=c(25)*cf                                               newk-10
      c(13)=a(26)+r                                                is-2
      c(3)=a(23)-a(22)*xl(22)                                      i$72
      c(22)=xl(22)                                                 k$72
      y(6)=a(24)*xl(22)+a(25)                                      dep$72
```

```
      b(1)=a(37)+a(38)*sqrt(z(28)/z(29))                                manh-0
      c(27)=a(39)*z(24)+a(40)                                           manh-1
      c(32)=a(42)*z(24)+a(43)                                           hours
      xnor(40)=1.0/z(37)                                                el-hc-0
      c(40)=a(53)*z(37)                                                 el-hc-1
      b(5)=a(57)*sqrt(z(29)/z(43))+a(58)*sqrt(z(28)/z(43))+a(56)        el-la-0
      c(42)=a(59)                                                       el-la-1
      b(2)=a(46)+a(47)*sqrt(z(29)/z(36))+a(48)*sqrt(z(28)/z(36))        fuel-e-0
      c(35)=a(49)*z(24)+a(50)                                           fuel-e-1
      b(6)=a(66)+a(67)*sqrt(z(28)/z(36))                                fu-ind-0
      c(39)=a(68)*z(24)+a(69)                                           fu-ind-1
      b(7)=a(70)+a(71)*sqrt(z(18)/z(36))                                fu-hc-0
      c(47)=a(72)                                                       fu-hc-1
      b(3)=a(61)+a(62)*sqrt(z(29)/z(36))+a(63)*sqrt(z(28)/z(36))        fu-trn-0
      c(38)=a(64)*z(24)+a(65)                                           fu-trn-1
      if(logic(2)) logic(1)=.true.
      return
      end



      subroutine solve(y,ex,el)
      common i4,i5,i6,d(150),ia,a(200),i1,i2,pa,z(100),c(55),xnor(55)
     1      ,ibx(55),ca(55),inl(55),b(55),nv1,iy1,ip1,ib1,lab(56),ngr
     2      ,ik(55),test(55),logic(100),xl(55),sim,nvc,ned,nex,nxs,nl
     3      ,max,nt,ned1,nr,date1,date2,lis,title(12),ncol,nit,nvc1
      real*8 lab,ld,label
       integer date1,date2,error,sim,pa
      logical*1 logic,ltu,lfa
      dimension y(100),ex(100),el(100)
1     nt=nt+1
      if(.not.logic(1)) y(33)=a(90)*z(34)                               employ-a
      y(21)=z(34)-z(33)                                                 unemp
      y(31)=z(21)/z(33)*100.0                                           prop-un
      y(32)=a(41)*z(31)+c(32)                                           hours
      if(logic(1)) go to 2
      y(27)=z(32)*(z(33)-a(45))/a(44)                                   manh-a
      y(1)=(z(27)-c(27))/b(1)                                           gnp$72-a
      y(4)=z(1)-z(2)-z(3)-z(5)                                          g$72-a
      go to 3
2     y(1)=z(2)+z(3)+z(4)+z(5)                                          gnp$72
3     y(7)=(a(6)*z(18)*z(1)*0.01+a(7))/(z(18)*0.01)                     taxin$72
      y(14)=a(35)*z(1)*z(18)*0.01+a(36)                                 bt
      y(12)=z(18)*(z(23)+z(21))*c(12)*0.01+b(9)                         gt
      y(13)=a(27)*z(3)*z(18)*z(26)*0.0001+c(13)                         is
      y(8)=z(1)-z(6)-z(7)-z(14)/z(18)*100.0-z(19)                       ni$72
      y(11)=a(8)*z(18)*z(8)*0.01+a(9)                                   taxes
      y(9)=z(8)-(z(11)-z(12)-z(13)-z(14))/z(18)*100.0                   ys$72
      y(2)=a(2)*z(9)+c(2)                                               c$72
      y(25)=b(8)*z(1)+c(25)                                             k$72*
      y(3)=a(22)*z(25)+z(6)+c(3)                                        i$72
```

```
      y(22)=z(3)-z(6)+c(22)                                            k$72
      if(logic(1)) y(27)=b(1)*z(1)+c(27)                               manhours
      if(logic(1)) y(33)=a(44)*z(27)/z(32)+a(45)                       employ
      if(logic(3).and.nt.gt.4.and.z(33)/z(34).gt.a(90)) logic(1)=.false.
      if (nt.lt.nit) go to 1
      return
      end



      subroutine post(y,ex,el)
      common i4,i5,i6,d(150),ia,a(200),i1,i2,pa,z(100),c(55),xnor(55)
     1      ,ibx(55),ca(55),inl(55),b(55),nv1,iy1,ip1,ib1,lab(56),ngr
     2      ,ik(55),test(55),logic(100),xl(55),sim,nvc,ned,nex,nxs,nl
     3      ,max,nt,ned1,nr,date1,date2,lis,title(12),ncol,nit,nvc1
      real*8 lab,ld,label
       integer date1,date2,error,sim,pa
      logical*1 logic,ltu,lfa
      dimension y(100),ex(100),el(100)
      y(15)=z(11)+(z(7)+z(19)-z(4))*z(18)*0.01-z(12)                   surplus
      y(30)=z(22)/z(1)                                                 k/o
      y(40)=a(52)*z(2)+c(40)                                           el-hc
      y(42)=z(1)*b(5)+c(42)                                            el-la
      y(44)=z(42)+z(40)                                                el-tot
      y(35)=(a(51)*z(44)-z(45)-z(46))*b(2)+c(35)                       fuel-eu
      y(38)=z(1)*b(3)+c(38)                                            fuel-trn
      y(39)=z(1)*b(6)+c(39)                                            fu-ind
      y(47)=z(2)*b(7)+c(47)                                            fu-hc
      y(48)=z(47)+z(39)+z(38)+z(35)                                    fuel-all
      y(16)=z(44)/z(1)                                                 ele/y
      y(17)=z(48)/z(1)                                                 fuel/y
      y(49)=z(2)/z(1)                                                  c/gnp
      y(50)=z(22)/z(33)                                                k/n
      y(51)=z(3)/z(1)                                                  i/gnp
      y(52)=z(4)/z(1)                                                  g/gnp
      y(53)=z(2)/z(37)                                                 c/pop
      if(sim.lt.2) go to 1
      z(12)=alog(z(12)*xnor(12))/xnor(12)                              gt-resd
1     return
      end
```

V CHECKING THE EQUATIONS FOR ERRORS

The user must check for coding errors before the model can be used. SIM has an option to calculate the residuals for each endogenous variable using only the actual values on the RHS of the equations. These residuals will correspond to the residuals estimated with the graph option in AUTO, if the equations were not renormalized.

The above example explains how to handle equations that were renormalized. It is very important for a user to estimate his model in one computer run and save all the graphs. The residuals should agree in at least 3 places.

VI ARRANGEMENT OF THE EQUATIONS

The convergence of the Gauss-Seidel algorithm is dependent on the arrangement of the equations in the iterative scheme. An arrangement is defined by the order of the equations and the normalization rule for each equation. By changing the order of the equations in SUBROUTINE SOLVE a convergence arrangement can be converted into a non-convergence arrangement, or vice versa.

Unfortunately, there is no straightforward rule that finds the best arrangement (or simply an arrangement) to minimize the number of iterations. However, the method has been used successfully on all the models in Project LINK, the Wharton EFA Model, the Brookings Model, and many others. Most models that are well defined have an arrangement which yields convergence in 6 to 10 iterations.

The author has had great success using the following general rules. The rules may apply only to the macroeconomic models.

1. Choose the normalized rule that makes the most economic sense.

2. Order the equations so they are as close to a recursive system as possible. The order generally should make economic causal sense: GNP before taxes, spendable income before consumption, manhours before employment.

3. Sometimes identities or critical equations should be repeated after their main explanatory endogenous variables are evaluated.

4. For demand and supply equations it may be necessary to renormalize them before a converging arrangement can be found.

5. A damping factor can be used on variables whose solution values oscillate from iteration to iteration; however, damping factors are not a good thing because the number of calculations are doubled in most models if a damping factor is placed on all the variables.

6. Some experimenting is necessary to find a good arrangement.

7. Choose the correct test vector for the convergence check.

VII FILES USED BY SIM

FILE	NAME	USE
1	data1	Data bank file from BANK, used for the definition of variables with the graph option.
2	data2	Input file for SIM produced by SIMDAT.
5	terminal	To type information for the program.
6	terminal	To receive prompts and other information from the program.
8	output	All results are written to this file. After the program terminates this file can be listed on the line printer.

VIII INPUT LINES

A. PART 1 OF PARAMETER LINE

ned nex nxs nl iy ip ib type of data

ned	=		Number of endogenous variables.
nex	=		Number of exogenous variables.
nxs	=	0	Data grouped together.
	=	1	Data separated into two groups.
nl	=		Length of longest lag.
iy	=		Earliest starting year.
ip	=		Number of observations per year.
ib	=		Earliest starting period.

type of data — Actual for historical data.

Baseline for simulated data from a previous solution.

Note: All of these parameters are read from the data2 file. The user must use SIMDAT to redefine the data base.

B. PART II PARAMETER LINE

max

max = Maximum number of iterations allowed for a solution with a default value of 5Ø.

C. TITLE LINE

Type in any title to identify this simulation run.

D. STARTING PERIOD LINE

iy ib

iy = Starting year of simulation.

ib = Starting period of simulation.

E. SOLUTION CONTROL LINE

sim lis ngr err nit ncol ws ns

sim	=	Ø	For a dynamic simulation.
	=	1	For a one-period simulation.
	=	2	For a residual check.
	=	3	For actual values in output.
lis	=	Ø	Normal.
	=	1	To list the solution values.
ngr	=	Ø	Normal.
	=		Number of variables to be graphed; ngr<16.
err	=	Ø	Normal.

= 1 For a summary table of the differences between actual values and solution values. err can also be used to compare a baseline solution with a new solution.

nit = Number of iterations before the testing for convergence begins.

ncol = Number of columns used in printing the solution matrix, with a default value of 12; $7<ncol<13$.

ws = 0 Normal.

= 1 To write the solution values to data2 file. The type of data parameter will be set to baseline. This option writes both the endogenous and the exogenous values.

ns = 0 Normal.

= 1 To read the coefficient vector from the data2 file. This is useful if the coefficients were changed in a previous simulation run.

F. LOC VECTOR LINE (For Graph Option)

Type the location of variable to be graphed. There must be ngr numbers separated by commas (or in fields of three). The user can define a set of default values with a data statement in the main program.

G. SOLUTION OPTION LINE

pa nlog nc ive nee iu nsim

pa = 0 To continue.

= 2 Last line of this simulation; program transfers to new TITLE LINE.

= 3 Last line of simulation run; program terminates.

nlog = 0 Normal.

= Number of elements in logic vector to be printed. This requires LOGIC VECTOR MODIFICATION LINE(S).

nc	=	Ø	Normal.
	>	Ø	To change coefficient vector.
ive	=	Ø	Normal.
	=		The location of an exogenous variable to be multiplied by 1.1. This is useful for an elasticity calculation.
nee	=	Ø	Normal.
	=	1	To accumulate the summary table of differences. Used for a simulation with a split sample period.
in	=	Ø	To use nsim to change the value assigned to SIM on the SOLUTION CONTROL LINE.
	=	1	Normal.
nsim	=		New value for SIM.

H. SOLUTION DATE LINES

iy ib (Next solution option line is to be read before this date.)

iy	=		year, e.g. 76.
ib	=		Period with a default value of 1.

I. LOGIC VECTOR MODIFICATION LINE(S)

j log

j	=		The element to be changed.
	=	Ø	For next option.
	=	-1	To list logic vector.
log	=	t	For true.
	=	f	For false.

J. COEFFICIENT VECTOR MODIFICATION LINE(S)

i a(i)

i	=		The element in the coefficient vector to be changed.
	=	0	For next option.
	=	-1	To list the coefficient vector.
a(i)	=		The new value for i^{th} coefficient. It is not to be used as a permanent change to the coefficient vector.

IX EXAMPLES

A. TERMINAL INPUT OUTPUT LINES FOR GNP MODEL

In this example, a control solution over the sample period was calculated. The user types only the lines marked with a T. The program echoes back the lines marked with an E. Lines that are marked with a T, and are blank, require only a carriage return (CR). Line 17T calls for 14 graphs and then the user in line 20T uses the default values. Line 26T calls for LOGIC MODIFICATION LINE(S) and the first three elements of the logic vector to be listed. The user sets logic(3)=f with line 34T. This option means there is no ceiling on employment.

```
 1T  % sim.o
 2      opened index for    2 with    33 words file name  data2
 3   1           sim by morris norman
 4                 feb 1977 version
 5    simdat file was opened with following parameters
 6    ned nex nxs  nl  iy  ip  ib type of data
 7      37  16   0   7  53   1   1    actual
 8    max
 9T
10E    50
11   type in title for simulation run              $
12T  control solution over sample period 1955-1975
13    iy ib  starting period for sim
14T  55
15E   55  1
16   sim lis ngr err nit ncol ws ns
17T        1  14   1
18E    0   1  14   1   8   0  0  0
19   type in locations of variables to be graphed 10 / line
20T
21E    1 53  2 49  3 51 22 30 33 50  4 52 16 17
22    1952 1 0
23    1953 1 0
24    1954 1 0
25    pa nlog nc ive nee
26T         3
```

```
27E     0    3  0    0   0
28     iy ib  next solution card read before this date
29T   76
30     1955 1 0
31    logic vector
32      1=  t    2=  t    3=  t
33      j log
34T     3 f
35E     3   f
36      j log
37T
38     solution for  1955 1  required   10 iterations
39    logic vector
40      1=  t    2=  t    3=  f
41     1956 1 0
42     solution for  1956 1  required   10 iterations
43    logic vector
44      1=  t    2=  t    3=  f
45     1957 1 0
46     solution for  1957 1  required   10 iterations
47    logic vector
48      1=  t    2=  t    3=  f
49     1958 1 0
50     solution for  1958 1  required   11 iterations
51    logic vector
52      1=  t    2=  t    3=  f
53     1959 1 0
54     solution for  1959 1  required    9 iterations
55    logic vector
56      1=  t    2=  t    3=  f
57     1960 1 0
58     solution for  1960 1  required   10 iterations
59    logic vector
60      1=  t    2=  t    3=  f
61     1961 1 0
62     solution for  1961 1  required   10 iterations
63    logic vector
64      1=  t    2=  t    3=  f
65     1962 1 0
66     solution for  1962 1  required   10 iterations
67    logic vector
68      1=  t    2=  t    3=  f
69     1963 1 0
70     solution for  1963 1  required    9 iterations
71    logic vector
72      1=  t    2=  t    3=  f
73     1964 1 0
74     solution for  1964 1  required    9 iterations
75    logic vector
76      1=  t    2=  t    3=  f
77     1965 1 0
78     solution for  1965 1  required   11 iterations
```

```
79    logic vector
80      l=  t    2=  t    3=  f
81     1966 l 0
82     solution for  1966 l  required   11 iterations
83    logic vector
84      l=  t    2=  t    3=  f
85     1967 l 0
86     solution for  1967 l  required    9 iterations
87    logic vector
88      l=  t    2=  t    3=  f
89     1968 l 0
90     solution for  1968 l  required    9 iterations
91    logic vector
92      l=  t    2=  t    3=  f
93     1969 l 0
94     solution for  1969 l  required    9 iterations
95    logic vector
96      l=  t    2=  t    3=  f
97     1970 l 0
98     solution for  1970 l  required    9 iterations
99    logic vector
100     l=  t    2=  t    3=  f
101    1971 l 0
102    solution for  1971 l  required    9 iterations
103   logic vector
104     l=  t    2=  t    3=  f
105    1972 l 0
106    solution for  1972 l  required   11 iterations
107   logic vector
108     l=  t    2=  t    3=  f
109    1973 l 0
110    solution for  1973 l  required   11 iterations
111   logic vector
112     l=  t    2=  t    3=  f
113    1974 l 0
114    solution for  1974 l  required   10 iterations
115   logic vector
116     l=  t    2=  t    3=  f
117    1975 l-l
118    solution for  1975 l  required   10 iterations
119   logic vector
120     l=  t    2=  t    3=  f
121    pa nlog nc ive nee
122T 3
123E   3    0  0   0   0
124      opened index for   l with   258 words file name  datal
125    open sub index for   l with   130 words
126    open sub index for   l with   131 words
127T % asa output ^lpr
128T % bye
```

MANUAL DOCUMENTATION for AUTO, BANK, SIM, SIMDAT, and RR ROUTINES

Diane Wells

This paper describes how to run off new copies of the manuals (on the Diablo terminal) for the programs: AUTO, BANK, RR ROUTINES, SIM, and SIMDAT. It assumes the user is familiar with the NROFF text processor and the procedure for extracting information from an FS tape. This procedure allows the user to modify the manuals as needed. The directories and files described in this paper can be regenerated from the FS tape labeled fs.diane April 1, 1977.

The file macs is a slightly modified version of the canned programs for NROFF. Unfortunately, this version still has bugs of an indescribable nature. This file should be linked to all the directories.

The program AUTO is in the directory "auto". It consists of the 5 files: macs auto, tran, inauto, and aexample.

FILE	CONTENTS
macs	nroff macros
auto	Sections I through II D
tran	Section III
inauto	Section IV
aexample	Section V

The manual is run off with the following command:

```
%neqn macs auto tran inauto aexample ^nroff -n35 ^gsi
```

The program BANK is in the directory "bank". It consists of the 4 files: macs, bank, inbank, bexample.

FILE	CONTENTS
macs	nroff macros
bank	Sections I through IV
inbank	Sections V through VI
bexample	Section VII

The manual is run off with the following command:

```
%nroff macs bank inbank bexample^sl
```

The program RR ROUTINES is in the directory "rand". It consists of the files macs and random.

FILE	CONTENTS
macs	nroff macros
random	entire manual

The manual is run off with the following command:

```
%neqn macs random ^nroff -n29 ^sl
```

The program SIM is in the directory "sim". It consists of the 5 files: macs, sim, code, insim, and sexample.

NAME	CONTENTS
macs	nroff macros
sim	Sections I through III B
code	Sections IV C through VII
insim	Section VIII
sexample	Section IX

The manual is run off with the following command:

```
%neqn macs sim code insim sexample ^nroff -n87 ^gsi
```

The program SIMDAT is also in the directory "sim". It consists of the 3 files: macs, simdat, and sdexample.

NAME	CONTENTS
macs	nroff macros
simdat	Sections I through V
sdexample	Section VI

The manual is run off with the following command:

```
%neqn macs simdat sdexample ^nroff -n78^sl
```

POTENTIAL GNP MODEL FOR THE USA
(Annual Observations 1947-1975)

1) Definition of real GNP

GNP$72 = C$72 + I$72 + G$72 + EX$72

2) Definition of real national income

NI$72 = GNP$72 - DEP$72 - TAXIN$72 - BT/P - RES

3) Capital consumption allowance with CCA

$$\underset{(101.5)}{DEP\$72 = .054^{*}}\ K\$72_{-1} - \underset{(10.7)}{8.75}$$

$R^2 = .997$, se = 1.25, d = .68

4) Indirect business tax function

$$P * TAXIN\$72 = \underset{(151.9)}{.09} * P * GNP\$72 + \underset{(4.2)}{1.84}$$

$R^2 = .999$, se = 1.16, d = .43

5) Definition of real spendable income

YS$72 = NI$72 - (TAXES - GT - BT - IS)/P

6) Income tax function

$$TAXES = \underset{(66.9)}{.29} * P * NI\$72 - \underset{(9.9)}{25.99}$$

$R^2 = .994$, se = 6.99, d = 1.06

7) Government transfer payments to persons

$$GT = P * (POP65 + UNEMP)\ EXP\,(\underset{(35.1)}{.052} * TIME - \underset{(3.6)}{.106})$$

$R^2 = .981$, se = .054, d = .29

8) Business transfer payments

$$BT = \underset{(78.7)}{.0044} * P * GNP\$72 - \underset{(10.0)}{.41}$$

$R^2 = .995$, se = .111, d = .75

9) Net interest income

$$IS = \sum_{i=0}^{7} w_i * P_{-i} * I\$72_{-i} * BAA_{-i} + \underset{(11.0)}{7.62}$$

$$\begin{array}{ll} w_0 = .522 & w_4 = .154 \\ w_1 = .410 & w_5 = .096 \\ w_2 = .312 & w_6 = .051 \\ w_3 = .226 & w_7 = .019 \end{array}$$

$R^2 = .953$, se = 1.96, d = .27

10) Consumption function

$$C\$72 = \underset{(4.7)}{.39} * YS\$72 + \underset{(82.0)}{.27} * YS\$72_{-1} + \underset{(4.0)}{.17} * YS\$72_{-2}$$

$$+ \underset{(1.9)}{.08} * YS\$72_{-3} - \underset{(2.0)}{12.43}$$

$R^2 = .997$, se = 8.41, d = .71

11) Investment function

$$I\$72 = \underset{(12.4)}{.85} * (K\$72^* - K\$72_{-1}) + DEP\$72 + \underset{(1.8)}{7.6}$$

$R^2 = .879$, se = 6.80, d = .87

12) Desired capital stock

$$K\$72^* = \sum_{i=0}^{7} w_i GNP\$72_{-i} + \sum_{i=0}^{4} v_i GNP\$72_{-i} \left|\frac{w\text{-}rate_{-i}}{cost\text{-}k_{-i}}\right|^{1/2} \underset{(1.08)}{+ 9.25}$$

w_0 = .000 w_4 = .310 v_0 = .080
w_1 = .136 w_5 = .291 v_1 = .040
w_2 = .233 w_6 = .233 v_2 = .014
w_3 = .291 w_7 = .136 v_3 = .0005

R^2 = .999, se = 8.89, d = .72

13) Actual capital stock

$K\$72 \quad K\$72_{-1} + I\$72 - DEP\$72$

14) Capital output ratio

K/0 = K\$72/GNP\$72

15) Cost of capital

COST-K = (BAA * .01 + .054) *P

16) Derived demand for labor - manhours index

$$MANHOURS = \underset{(4.8)}{.042} * GNP\$72 + \underset{(3.4)}{.068} * GNP\$72 \left|\frac{cost\text{-}k}{w\text{-}rate}\right|^{1/2}$$

$$\underset{(6.8)}{-1.04} * TIME + \underset{(29.9)}{59.9}$$

R^2 = .977, se = 1.02, d = 1.52

17) Average weekly hours for total private nonagricultural

$$HOURS = \underset{(3.2)}{-.144} * PROP\text{-}UN - \underset{(15.7)}{.166} * TIME + \underset{(182.0)}{41.2}$$

R^2 = .922, se = .311, d = .63

18) Employment of workers

$$EMPLOY = \underset{(41.9)}{36.33} * \frac{MANHOURS}{HOURS} - \underset{(10.0)}{21.69}$$

$R^2 = .984$, se = 1.13, d = .45

19) Total unemployment

UNEMP = LABOR - EMPLOY

20) Proportion of civilian labor force unemployed

PROP-UN = UNEMP/EMPLOY * 100

21) Residential and small light & power per capita

$$\frac{EL\text{-}R+C}{POP} = \underset{(57.1)}{2.52} \frac{C\$72}{POP} - \underset{(37.7)}{4.59}$$

$R^2 = .992$, se = .117, d = .871

22) Electric sales to large light and power and misc.

$$EL\text{-}LA+OT = \underset{(7.7)}{.848} * GNP\$72 - \underset{(2.5)}{.032} * GNP\$72 \left|\frac{w\text{-}rate}{Pe}\right|^{1/2}$$

$$+ \underset{(2.5)}{.098} * GNP\$72 \left|\frac{cost\text{-}k}{Pe}\right|^{1/2} - \underset{(8.2)}{282.2}$$

$R^2 = .992$, se = 18.2, d = .375

23) Total electric sales

EL-TOT = EL-LA+OT + EL+R+C

<u>Demand for Fuel</u>

24) Household and commercial sectors

$$FUEL\text{-}H/C = \underset{(2.0)}{3.50} * C\$72 + \underset{(6.9)}{1.275} * C\$72 \left|\frac{P}{PBF}\right|^{1/2}$$

$$+ \underset{(11.7)}{2601.1}$$

$R^2 = .981$, se = 328.3, d = 1.40

25) Transportation sector

$$\text{FUEL-TRN} = \underset{(6.8)}{19.32} * \text{GNP\$72} - \underset{(6.8)}{1.69} * \text{GNP\$72} \left|\frac{\text{w-rate}}{\text{PBF}}\right|^{1/2}$$

$$+\underset{(6.6)}{5.62} * \text{GNP\$} \left|\frac{\text{cost-k}}{\text{PBF}}\right|^{1/2} - \underset{(3.1)}{173.5} * \text{TIME}$$

$$\underset{(1.6)}{-1428.9}$$

R^2 = .989, se = 364.8, d = .93

26) Industrial sector and miscellaneous sector

$$\text{FU-IN/OT} = \underset{(1.8)}{7.0} * \text{GNP\$72} + \underset{(2.7)}{1.48} * \text{GNP\$72} \left|\frac{\text{cost-k}}{\text{PBF}}\right|^{1/2}$$

$$\underset{(1.4)}{-107.6} * \text{TIME} + \underset{(5.8)}{7059.4}$$

R^2 = .954, se = 488.4, d = 1.27

27) Electrical utilities sector

$$\text{FUEL-EU} = \underset{(17.1)}{22.3} * \text{EL} - \underset{(5.0)}{.82} * \text{EL} \left|\frac{\text{w-rate}}{\text{PBF}}\right|^{1/2}$$

$$+ \underset{(2.3)}{1.4} * \text{EL} \left|\frac{\text{cost-k}}{\text{PBF}}\right|^{1/2} - \underset{(9.4)}{327.1} * \text{TIME}$$

$$+ \underset{(18.7)}{2003.0}$$

R^2 = .998, se = 218.8, d = 1.47

28) Electricity produced by fuel

EL = 1.094 * EL-TOT - EL-NUC - EL-HY

29) Total demand for fuel

FUEL-ALL = FUEL-H/C + FUEL-TRN + FU-IN/OT + FUEL-EU

DEFINITION OF VARIABLES

label	definition of variable
baa	bond yields and interest rates, percent per annum
bt	business transfer payments, billions of $
c$72	personal consumption expenditures, billions of $72
cost-k	cost of capital (baa + .054)*ipdgnp
dep$72	capital consumption allowance with cca, billions of $72
el-hy	elec. produced by hydropower, billions of kwh
el-la+ot	elec. sales large + other, billions of kwh
el-nuc	elec. produced by nuclear power, billions of kwh
el-r+c	elec. sales res. + small light, billions of kwh
el-tot	elec. sales total, billions of kwh
employ	employment of persons over 16 years, millions
ex$72	net exports of goods and services, billions of $72
fu-in/ot	cons. all fuels in industrial + misc., trillions of btu
fuel-all	cons. all fuels in all sectors, trillions of btu
fuel-eu	cons. all fuels in electrical utilites, trillions of btu
fuel-h/c	cons. all fuels in res. & com. sectors, trillions of btu
fuel-trn	cons. all fuels in transportation sector, trillions of btu
g$72	govt. purchases of goods and services, billions of $72
gnp$72	gross national product, billions of $72
gt	government transfer payments to persons, billions of $
hours	average weekly hours total private nonagricultural
i$72	gross private domestic investment, billions of $72
ipdgnp	implicit price deflators for gnp, 1972=100
is	spendable interest, billions of $
k$72	estimated capital stock at end of period, billions of $72
k$72*	desired capital stock predicted values from eqn-12
k/o	capital output ratio, k$72/gnp$72
labor	civilian labor force, millions of persons
manhours	hours of all persons total private, 1967=100
ni$72	national income, billions of $72
p-elc-h	cents/kwh all consumption residential service
p-elc-in	cents/kwh large light and power, average price
pbf	wholesale price index: fuels, related products, 1967=1.0
pop	population, millions of persons
pop65	population over 65, millions of persons
prop-un	proportion of civilian labor force unemployed
residual	residual from gnp identity , billions of $72
taxes	personal + corp. taxes + social insurance, billions of $
taxin$72	indirect tax + govt. surplus, billions of $72
time46=1	time trend 1946 equals 1
unempl	total unemployment, millions of persons
w-rate	compensation per hour, 1967=100
ys$72	spendable income, billions of $72

RELATED IIASA PUBLICATIONS

Stochastic Control for Linear Discrete-Time Distributed-Lag Models. W.B. Arthur. (rr77-018) $1.00/AS20.

Dual Systems of Dynamic Linear Programming. A.I. Propoi. (rr77-009) $1.50/AS30.

Proceedings of a Conference on Pest Management, 25-29 October, 1976. G.A. Norton, C.S. Holling, editors. (cp77-006) $13.40/AS240.

Health System Modeling and the Information System for the Coordination of Research in Oncology. D.D. Venedictov, editor. (cp77-004) $20.80/AS375.

MOIRA: Food and Agriculture Model, Proceedings of the Third IIASA Symposium on Global Modeling, September 22-25, 1975. G. Bruckmann, editor. (cp77-001) $10.00/AS200. (MICROFICHE ONLY)

A Computer Assisted Approach to Regional Development: Report of Research Activities on the Kinki Region of Japan. H. Knop, editor. (cp76-010) $5.20/AS95.

Latin American World Model. Proceedings of the Second IIASA Symposium on Global Modelling. G. Bruckmann, editor. (cp76-008) $12.60/AS225.

First IIASA Conference on Energy Resources. M. Grenon, editor. (cp76-004) $24.60/AS440.

Proceedings of the Workshop on Energy Demand. W.D. Nordhaus, editor. (cp76-001) $10.00/AS200. (MICROFICHE ONLY)

Health Care Resource Allocation Models - a Critical Review. R.J. Gibbs. (rm77-053) $2.50/AS45.

International Trade Policies in Models of Barter Exchange. M.A. Keyzer. (rm77-051) $1.50/AS30.

Planning Long Range Agricultural Investment Projects: a Dynamic Linear Programming Approach. H. Carter, C. Csåki, A.I. Propoi. (rm77-038) $1.50/AS30.

Dynamic Linear Programming Model for Agricultural Investment and Resources Utilization Policies. C. Csåki. (rm77-036) $1.50/AS30.

Adaptivity and Stability of Time Series Models. J. Ledolter. (rm77-035) $1.50/AS30.

A Multivariate Time Series Approach to Modelling Macroeconomic Sequences. J. Ledolter. (rm77-033) $1.50/AS30.

Dynamic Linear Programming Models for Livestock Farms. A.I. Propoi. (rm77-029) $1.50/AS30.

The State of the Art in Modelling of Food and Agriculture Systems. M. Neunteufel. (rm77-027) $5.00/AS90.

On the Use of Matrix Factorization Techniques in Penalty Function Methods for Structured Linear Programs. S. Chebotarev. (rm77-021) $1.50/AS30.

Analysis of a National Model with Domestic Price Policies and Quota on International Trade. M.A. Keyzer. (rm77-019) $4.50/AS80.

Assessment of Existing and Prospective World Economic and Food Trends. S.C Schmidt. (rm77-014) $4.50/AS80.

A Dynamic Linear Programming Approach to National Settlement System Planning. A.I. Propoi, F. Willekens. (rm77-008) $2.00/AS40. (MICROFICHE ONLY)

An Approach to Building a Universal Health Care Model: Morbidity Model of Degenerative Diseases. S. Kaihara, I. Fujimasa, K. Atsumi, A.A. Klementiev. (rm77-006) $3.50/AS60.

Linking National Models of Food and Agriculture: An Introduction. M.A. Keyzer. (rm77-002) $2.50/AS45.

Computer Programs for Spatial Demographic Analysis. F. Willekens, A. Rogers. (rm76-058) $6.00/AS110.

A Computer Method for Projecting a Population Sex-Age Structure. A.A. Klementiev. (rm76-036) $1.50/AS30.

An Attempt of Long-Range Macroeconomic Modelling in View of Structural and Technological Change. W. Häfele, R. Bürk. (rm76-032) $1.50/AS30.

WISSIM: An Interactive Simulation Control Language. J.S. Buehring. (rm76-024) $4.50/AS80.

Energy/Environ. Models and their Relationship to Planning in Wisc. the GDR and Rhone Alpes. S. Born, C. Cicchetti, R. Cudahy, J. Pappas, P. Hedrich, K. Lindner, D. Ufer, J. Martin, D. Finon. (rm76-021) $4.50/AS80.

Please cite publication number when making an order. See inside back cover for order information.