Buckler

Rm #8

DETAILED DESCRIPTION OF THE MARYLAND INTERINDUSTRY FORECASTING PROGRAM

The Maryland Interindustry Forecasting Program consists of thirteen symbolic decks written in Fortran V but kept as close as possible to Fortran IV. The Program reads the data, computes the forecasts, and prepares three types of reports: (1) the general report, which is printed as the forecast is made and amounts to a disaggregation of the national accounts, (2) the matrix listing showing the sales, in selected future years, of each seller to each buyer, and (3) plots of the forecasts related to historical series of the same items.

In what follows, we will trace the development of the forecasts through the statements of the Program. We will go through the program twice, once in general terms to see the whole structure quickly and then a second time to observe the details and instruct the reader in operating the Forecasting Program.

The main or starting deck is named RUN. It performs no computations. Its function is to call the subroutines in the proper order. RUN divides the program into three distinct parts: the reading routine, the forecasting program, and the editing routines. The first part calls the READER subroutine and thereby introduces into the active memory of the computer most of the data necessary for forecasting.

The second forecasting part (the DO 20 loop.*) calls the subroutines necessary to calculate the forecast for one year. The program passes through these calls once for each forecast year. Each pass calls seven subroutines, in this order:

POLYLG - Lagrangian polynomial interpolation

EXOG - exogenous demand

CONDEM - consumer demand

INVEST - equipment investment demand.

CONSTR - construction demand.

IOCOMP - input-output computations

EMPLOY - employment computations

* (DO loops will be referred to repeatedly in this way; this reference indicates the series of FORTRAN statements beginning with "DO 20 I=1, NP" and ending with the statement numbered 20.)

POLYLG--pronounced Polly Lagrange--computes five vectors of interpolation weights. EXOG, CONDEM, INVEST, and CONSTR (read "Construct") compute, print, and store on tapes the vectors of the components of GNP. IOCOMP then converts the total final demand vector into a vector of industry outputs and EMPLOY transforms the outputs into a vector of employment by industry; outputs and employment are printed. In addition to the printed output, these subroutines create two files (tapes on a 7094) which are used by the editing routines. To forecast years 1965 to 1980 requires executing the entire sequence sixteen times.

The last part of RUN calls two editing and plotting subroutines if they are desired. To get a listing of the interindustry sales matrix for five forecast years, we call MATLIS (matrix list). MATLIS reads one of the files created in the forecasting part and calls SCRIVO subroutine to write the desired matrices. Back in RUN we look to see whether plots are desired; if so, we call TRIGRA (pronounced tri-graph, so named because it can plot the output, employment, and investment graphs on the same set of axes). TRIGRA reads the other file created by the forecast program and also the cards carrying the historical data; it links the history to the forecast in each series and calls PLOTER to plot the graphs. The program is then completed.

READER

Let us now return to the beginning of RUN to analyze the program in more detail. RUN immediately calls READER, which then rewinds units 10 and 11. Unit 10 must be a tape drive; unit 11 may be a tape (as on the 7094) or may be a file on a drum (as on the 1108). With this business accomplished, we come to three READ statements, which read in five cards. From the first we take the TITLE for that particular run, e.g. "JUNE 12, 1968 VERSION". From the next three we take the values of seventeen control variables, among them NP (number of periods), NS, the number sectors, IPLOT, which is 0 if no plots desired and otherwise i, and LIRTAP (meaning to read tape; it is 1 if we want to take data for forecasting from tape 10; 0 if we will read all data from cards). From the fifth card we read the values of ITAPE. ITAPE(I) is 1 if we want to store the complete forecast for the I-th year on unit 11; otherwise it is 0. The purpose of unit 11 is to provide the data necessary for listing the matrices; subroutine MATLIS will read this unit. However MATLIS is limited to a maximum of five years' matrices; therefore, if more than five elements of ITAPE are set to 1, all of the corresponding forecasts will be saved on 11, but only the first five years' data on that tape will be read and printed by MATLIS.

Just below statement 4 we find a test on LIRTAP. We shall assume that LIRTAP is 0, as it is in the decks as supplied. What happens

when LIRTAP is 1 will be explained later. We go therefore to statement 6. Here we find a short loop which may appear puzzling at first. It sets thousands of values of IEQ equal to zero. If we look in the third card of the Common statement above, we see that IEQ has only 34 elements, but stands near the beginning of Common. The effect of this loop, therefore, is to set most of Common equal to zero. The items at the end of Common are not data but working space, and so need not be zeroed.

Beginning with statement 8, we read a number of arrangement codes whose significance need not detain us. But note that these codes must be in the proper order.

Now we come to the heart of READER, the loop that goes from statement 9 to the statement below 70, which says GO to 9. This loop reads all of the basic data and puts them in their right places in Common. The order in which the data occurs does not matter. (There is one exception to be mentioned below). This release from strict order is an enormous convenience in working with the program. The matrix or vector to which each card pertains is indicated by a two-digit number in columns 1 and 2. For example, a code 12 means that the card has on it elements of the A matrix; code 15 indicates that it relates to exports. Statement 9 reads this code number into NCODE and then reads five fields, each consisting of two integers (a row number and a column number) and a real number (the coefficient in that row and column of the matrix indicated by NCODE). Suppose NCODE is 15; how do we get the information where it belongs?

First we test to see if NCOPE is 99; if it is, we have read all the data and skip down to 100 to attend to other business. With NCODE = 16, the next IF sends us to 10 where we subtract 10 from NCODE and initiate a DO 70 K=1,5. The 5 is for the 5 fields on the data card. First we move the row number into I and test for a zero; if we find it, we assume the field is blank and go on to the next field. If I is positive, we put the row number into J and the coefficient into C and execute a long computed GO TO statement on NCODE. Since NCODE was read as 15, after subtraction of 10 it is 5. Notice that the fifth prong of this GO TO is 15. Therefore we go to statement 15 and put out coefficient, C, into row I, column J of the EXPORT matrix. Then we go on to the next field. Note that the statement number to which we go on the computed GO TO is the same as NCODE on the card. This identity holds for all values of NCODE less than 31. It makes READER an index of the data codes. When we have put in place all elements on one card, we return to 9 to read another.

Most of the NCODE's are handled just that simply. A few, however, get special treatment. They are:

Code	Treatment
11	These cards define the conversion from consumer categories to I-O sectors. For each element, I is the number of a consumer category, J is the number of an I-O sector, and C is the fraction of category I assigned to sector J. The program counts, in I11, the number of such Conversion factors and stores them in CONV in the order in which they are read. At the same time, it packs the I and J together in one word and stores it in ICONV. These vectors are used in the DO85 loop of Subroutine CONDEM.
13	These cards introduce coefficient changes in the inter-industry flow matrices. First we look to see whether there is already a change recorded for the element specified. If so, we replace it; if not, we add an element. I13 counts the number of such changes. On the cards, the yearly change is expressed in percent of the base year coefficient. READER converts these into absolute numbers. Columns 1-200 refer to the A matrix, columns 201-300 refer to the construction matrix, and columns 401-511 refer to the capital equipment matrix. The absolute change is stored in ADEL, and the row and column numbers to which the change refers are packed into the corresponding element of IADEL.
30-39	All of these codes refer to construction equations. The value of NCODE indicates the type--e.g. 30 is an exponential trend, 31 is a stock adjustment equation. Seven lines below statement 9, the value of NCODE is recorded in KSTYP(I) (read--construction type) and NCODE is changed to 30.
40-49	The codes designate different types of equipment investment and are treated essentially like the codes 30-39. One wrinkle is added: instead of having to make our cards with a special set of numbers for the equipment investment equations, we use the I-O number of any sector in the equipment-buying group, and the program makes the conversion.
50	These are cards for handling special final demands. Comment cards below statement 501 describe their functioning.

When all of the Common data have been read, the all-9 card is found and the IF just below statement 9 sends us to statement 100. Here we write the contents of Common (except for the data on the first five cards) onto the tape on 10. Next time we run the program, we change LIRTAP (on the 3rd data card) to 1. Then the IF statement below statement 4 will fail and, instead of clearing Common to zero, we read back in what has been written on tape 10 on the first run. We have to submit only as many of the cards between ITAPE and all-9 as we want to change.

When tape 10 is written, the I11 and I13 count is printed. This count must be transferred into the appropriate columns of the 3rd data card for running without the 11 and 13 data. On the other hand, when all of the 11 cards are present, I11 must be 0 in the 3rd card. Control now returns to RUN and RUN calls POLYLG.

POLYLG

As soon as RUN has defined T, the number of the year being forecast (beginning with T=1 in 1965), POLYLG is called. This modest subroutine produces no printed output of its own, but prepares the way for others by computing Lagrangian interpolation weights. These weights provide an extremely convenient method of specifying exogenous variables. We simple specify the value of the function for a few future years, and the program interpolates on a smooth curve between them. Suppose we specify three values, v_1, v_2, and v_3, of some variable V for, let us say 1966, 1970, and 1975. We call these years the interpolation points, t_1, t_2, and t_3. Lagrangian interpolation requires us to find three second degree polynomials, $w_1(t)$, $w_2(t)$, and $w_3(t)$, such

that the polynomial $p(t) = w_1(t)v_1 + w_2(t)v_2 + w_3(t)v_3$ will have the values v_1, v_2, and v_3 for t equal to t_1, t_2, and t_3, respectively.

The $w_i(T)$ are called Lagrangian polynomials. The theory of these polynomials is simply explained in a three-point example. Suppose we know v(t) at t_1, t_2, and t_3. Let us define

$$w_1(t) = \frac{(t-t_2)(t-t_3)}{(t_1-t_2)(t_1-t_3)} = \begin{cases} 1 \text{ for } t=t_1 \\ 0 \text{ for } t=t_2 \\ 0 \text{ for } t=t_3 \end{cases}$$

$$w_2(t) = \frac{(t-t_1)(t-t_3)}{(t_2-t_1)(t_2-t_3)} = \begin{cases} 0 \text{ for } t=t_1 \\ 1 \text{ for } t=t_2 \\ 0 \text{ for } t=t_3 \end{cases}$$

and

$$w_3(t) = \frac{(t-t_1)(t-t_2)}{(t_3-t_1)(t_3-t_2)} = \begin{cases} 0 \text{ for } t=t_1 \\ 0 \text{ for } t=t_2 \\ 1 \text{ for } t=t_3 \end{cases}$$

By virtue of the equalities shown on the right, we see that

$$p(t) = w_1(t)v_1 = w_2(t)v_2 + w_3(t)v_3$$

is a second degree polynomial with the required values v_1, v_2, and v_3 at t_1, t_2, and t_3. The generalization to more than three points is immediate and is embodied in the Fortran of POLYLG.

POLYLG calculates the values of these $w_i(T)$ for the given value of T. Naturally, the values of the $w_i(T)$ depend on the interpolation points: the $w_i(T)$ for interpolating from points in 1966, 1969, and 1975 differ from those for interpolating from points in 1967, 1972, 1975, and 1980. Both the number and the spacing of points may vary between different patterns. Each column of TINTER contains a different interpolation pattern. The first element in the column is the number of points in the pattern. Then there follow from 2 to 4 year numbers of the

interpolation points. In these numbers, 1965 is year 1. Thus the pattern 1966, 1970, 1975 would appear in a column of TINTER as

3 2 6 11.

The weights for each pattern are calculated for the current value of T and stored in POLY for use by other subroutines.

EXOG

Before this subroutine is called, RUN prints the heading for the first page of output and sets ITR equal to 1 if we are in one of the five years which we want to use in the matrix listing.

EXOG firsts clears the space, FD, in which the final demand will be accumulated. First, exports are calculated from exponential trends and put into FD while their sum goes into the appropriate spot in the GNP account. Then the transferred (= competitive not sold directly to final demand) imports are treated similarly. Note that both are written onto unit 11 if the current year is desired for the matrix listing.

Between the comment "Calculate Defense Spending" and statement 38, we find provision to read in the defense vector in the first four years. Beyond that time, it is interpolated from the data found in the first three columns of the GOV matrix using the interpolation pattern found in the fourth column of TINTER as reflected in the current weights in the fourth column of POLY.

Regardless of where the vector came from, 38 sums it for the GNP account and adds it to FD.

Next we calculate special project final demands. This section of the program was added to be able to give, within limited core space,

considerable flexibility in specifying the time path of the demands of special projects or assumptions which affect directly only a few (less than 15) products.

Statement 137 moves the present value of Disposable Income Per Capita, DISPC, into DISPCL, its lagged value.

In the first three years, we read in the values of DISPC, POPULation, the labor force, WORKRS, state and local expenditures except for employees and construction (into GNP(7)) and then the same for Federal non-defense (into GNP(9)) and then employment of domestic servants and government workers, including military. Next, we read in equipmemt investment by purchases (V), construction by type (S), and consumer purchases per capita. In the first year, we read the industry outputs in the previous year. These are used only in the inventory equations.

If the total material purchases of State and Local governments and Federal Non-Defense were not read in, they are interpolated in Statement 44-45. Similarly with exogenous employment, Statements 80-90 distribute these totals to the supplying industries in fixed proportions. Then the statements following 70 add the contributions of exogenous employment to appropriate parts of the GNP accounts.

Next comes the calculation of a number of exogenous variables used elsewhere in the program. In the first years, these were read in on Format 42 above, or are not needed. POPUL was read in; HONEY, defined by

$$\text{HONEY (T)} = \frac{\text{Cash flow (T)}}{\text{Sales (T)}} \Big/ \frac{\text{Cash flow (1)}}{\text{Sales (1)}}$$

is used only in the investment equations. Since V is read in for the first years, HONEY is not needed.

After these variables come the changes in the coefficient matrix. They are made in all years except the first.

Control then returns to RUN and RUN calls CONDEM.

CONDEM

This subroutine has two functions, first to calculate per capita consumer expenditures by consumer categories (CONPUR) and second to multiply them by population and allocate the categories to the I-O industries. In the first years, CONPUR was read in EXOG, so we go directly to the second function. If the first function must be performed, we go through the NCS consumer categories and, for each, first calculate by interpolation the relative price. Then we test to see whether the income coefficient of the consumption equation is zero. If it is, we know that we have simply an exponential trend in that item. Otherwise we go to statement 30 and apply the standard formula. All the variables in it are familiar except perhaps DELDPC, the first difference of DISPC, which was calculated in EXOG.

We then come to the second half. The DO 75 loop multiplies by population. DO 80 clears X to receive the converted demands, and DO 85 performs the classification conversion. Compare the unpacking of ICONV with its packing in READER; recall that integer division truncates the answer. The number 512 is used in packing because it is 1000 in octal. After conversion, consumption is added into FD

and summed in the appropriate GNP account. Finally, CONPUR is converted back to a per capita basis for the benefit of the categories generated by exponential trends. Control then returns to RUN, and RUN calls CONSTR.