AN INTERACTIVE IMPLEMENTATION OF CONTROL THEORY TECHNIQUES APPLIED TO PINDYCK'S MODEL OF THE U.S. ECONOMY

O. G. Johnson, X. Mangin, J. R. Rhyne
Computer Science Department
University of Houston
Houston, TX 77004/USA

I. INTRODUCTION

Pindyck and others introduced the use of control theory in econometric models in the 1960's, [14], [15]. These models, although the most advanced of all econometric techniques, involve considerable trial and error in the selection of nominal trajectories and weighting factors.

Thus, only with an interactive implementation can these techniques be efficient as tools for evaluating alternative scenerios.

This paper reports the design and implementation of a general state regulator control model which utilizes interactive techniques. This model is then applied to Pindyck's model of the U.S. Economy. The interactive functions have been selected to minimize both original computation time as well as re-computations due to parameter and data changes.

II. THE STATE REGULATOR MODEL AND ALGORITHMS FOR ITS SOLUTION

The state regulator model is called, variously, the linear regulator model (or problem), the linear quadratic tracking model or the linear tracking model with quadratic cost criterion. The model appears in the literature in both continuous and discrete form. In econometric models, the discrete form of the problem is implied since most data consist of monthly or quarterly time series.

The linear quadratic tracking problem is that of minimizing

$$J=1/2 \sum_{i=0}^{N} (x_i-\hat{x}_i)'Q(x_i-\hat{x}_i)+1/2 \sum_{i=0}^{N-1} (u_i-\hat{u}_i)'R(u_i-\hat{u}_i). \qquad (1)$$

Each x_i is an n vector which represents the value of n state variables and each u_i represents r control variables at time i.

Q is an n×n matrix which is either positive semi-definite or positive definite. Often in tracking models Q is diagonal and positive semi-definite. The matrix R is r×r and positive definite. The vectors

$\hat{x}_i$ $i=0,\ldots,N$ and $\hat{u}_i$ $i=0,\ldots,N-1$ are the desired or nominal states and controls.

The solution sequences u_i^* $i=0,\ldots,N-1$ and x_i^* $i=0,\ldots,N$ must satisfy the equations

$$x_0=\xi \tag{2}$$

$$x_{i+1}-x_i=Ax_i+Bu_i+Cz_i \quad i=0,\ldots,N-1 \tag{3}$$

The matrices A, B, C, have dimension $n\times n$, $n\times r$ and $n\times k$ respectively. The vectors z_i are known exogeneous variables of dimension k.

The standard indirect solution to this problem is to form the Hamiltonian of the system or, alternatively, the Lagrangian

$$L\equiv J+\sum_{i=0}^{N-1} p'_{i+1}(-x_{i+1}+x_i+Ax_i+Bu_i+Cz_i) \tag{4}$$

and observe that, if a solution, x_i^*,u_i^*,p_i^* exists which extremizes the Lagrangian then, necessarily

$$p_{i+1}^*-p_i^*=-Q(x_i^*-\hat{x}_i)-A'p_{i+1}^* \quad i=1,\ldots,N-1 \tag{5}$$

$$u_i^*=-R^{-1}B'p_{i+1}^*+\hat{u}_i \quad i=0,\ldots,N-1 \tag{6}$$

$$p_N^*=Q(x_N^*-\hat{x}_N) \tag{7}$$

In addition, x_i^* and u_i^* satisfy equations (2) and (3).

One then makes the assumption (and later proves) that

$$P_i^*=K_ix_i^*+g_i \quad i=1,\ldots,N \tag{8}$$

It can then be shown that the $n\times n$ matrices K_i $i=1,\ldots,N$ satisfy the initial condition

$$K_N=Q$$

and the Riccati difference equations

$$K_i=Q+(I+A)'(K_{i+1}-K_{i+1}B(R+B'K_{i+1}B)^{-1}B'K_{i+1})(I+A) \tag{10}$$
$$i=N-1,N-2,\ldots,1$$

The vectors g_i $i=1,\ldots,N$ satisfy

$$g_N=-Q\hat{x}_N \tag{11}$$

and

$$\begin{aligned} g_i=&-(I+A)'(K_{i+1}-K_{i+1}B(R+B'K_{i+1})B^{-1}B'K_{i+1})BR^{-1}B'g_{i+1} \\ &+(I+A)'g_{i+1}+(I+A)'(K_{i+1}-K_{i+1}B(R+B'K_{i+1}B)^{-1}B'K_{i+1})(B\hat{u}_i \\ &+Cz_i)-Q\hat{x}_i \quad i=N-1,N-2,\ldots,1 \end{aligned} \tag{12}$$

From the knowledge of K_i and g_i $i=1,\ldots,N$

u_i^* $i=0,\ldots,N-1$ can be computed from the equation

$$u_i^* = -(R+B'K_{i+1}B)^{-1}B'K_{i+1}[(I+A)x_i^* - BR^{-1}B'g_{i+1} + B\hat{u}_i + Cz_i] - R^{-1}B'g_{i+1} + \hat{u}_i \tag{13}$$

The solution state sequence x_i^* $i=0,\ldots,N$ can then be computed from the original difference equation. Details of the derivation of this algorithm are available in [13] and [14]. These derivations are the discrete analogues of the derivations in [1].

The modified Lagrange direct algorithm is to differentiate the Lagrangian with respect to each of the variables x_i, p_i and u_{i-1} $i=1,\ldots,N$. The resulting linear system is that given in Figure 1. The matrix in

$$\left[\begin{array}{cccc|cccc|cccc}
Q & & & & & & & & I & D' & & \\
 & Q & & & & & & & & I & D' & \\
 & & \ddots & & & & & & & & \ddots & \\
 & & & Q & & & & & & & & I \\
\hline
 & & & & R & & & & -B' & & & \\
 & & & & & R & & & & -B' & & \\
 & & & & & & \ddots & & & & \ddots & \\
 & & & & & & & R & & & & -B' \\
\hline
I & & & & -B & & & & & & & \\
D & I & & & & -B & & & & & & \\
 & & \ddots & & & & \ddots & & & & & \\
 & & D & I & & & & -B & & & &
\end{array}\right]
\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \\ u_0 \\ u_1 \\ \vdots \\ u_{N-1} \\ p_1 \\ p_2 \\ \vdots \\ p_N \end{bmatrix}
\begin{bmatrix} Q\hat{x}_1 \\ Q\hat{x}_2 \\ \vdots \\ Q\hat{x}_N \\ R\hat{u}_1 \\ R\hat{u}_2 \\ \vdots \\ R\hat{u}_{N-1} \\ -D\xi+Cz_0 \\ Cz_1 \\ \vdots \\ Cz_{N-1} \end{bmatrix}$$

$D \equiv -(I+A)$

The Lagrangian Matrix
Figure 1

Figure 1, although sparse, is quite large, $(2n+r)N\times(2n+r)N$. Examination of Figure 1 reveals that band structure can be obtained by eliminating some of the variables. The rows of the Lagrangian in Figure 1 are partitioned into three sections. The first section corresponds to the necessary conditions (5) and (7); the second section to condition (6) and the third section corresponds to equations (2) and (3). Conditions (5) and

(7) can be used to explicitly solve x_i in terms of p_i and p_{i+1}. Condition (6) can be used to solve u_i in terms of p_i. (If Q is semidefinite, a slight perturbation will make it positive definite).

$$x_i=Q^{-1}(p_{i+1}-p_i+A'p_{i+1})+\hat{x}_i\equiv x_i(p_i,p_{i+1}) \quad i+1,\ldots,N-1 \tag{14}$$

$$x_N=Q^{-1}p_N+\hat{x}_N\equiv x_N(p_N) \tag{15}$$

$$u_i=-R^{-1}B'p_{i+1}+\hat{u}_i\equiv u_i(p_{i+1}) \quad i=0,\ldots,N-1 \tag{16}$$

By the Courant extremum principle [9], each of these necessary relationships can be inserted into the Lagrangian without altering the solution in any way. Thus inserting (3), (14), (15) and (16) in (4) the Lagrangian becomes

$$\begin{aligned} L=&-1/2\sum_{i=1}^{N-1}[x_i(p_i,p_{i+1})-\hat{x}_i]'Q[x_i(p_i,p_{i+1})-\hat{x}_i] \\ &-1/2\sum_{i=0}^{N-1}[u_i(p_{i+1})-\hat{u}_i]'R[u_i(p_{i+1})-\hat{u}_i] \\ &-(x_N(p_N)-\hat{x}_N)'Q(x_N(p_N)-\hat{x}_N) \\ &-(\xi-\hat{x}_0)'Q(\xi-\hat{x}_0) \end{aligned} \tag{17}$$

From equation (17) the following can be deduced:

1. Since $x_i(p_i,p_{i+1})$ $i=1,\ldots,N-1$, $u_i(p_{i+1})$ $i=0,\ldots,N-1$ and $x_N(p_N)$ are all linear functions of the vector $P=(p_i,\ldots p_N)'$, L is a negative definite form in p.
2. Since p_j only occurs in L in terms with p_{j-1} and p_{j+1}, L is block tridiagonal, and thus band structured.

Hence, if L, thus modified, is differentiated with respect to each of p_i $k=1,\ldots,N$, the resulting matrix is that which is displayed in Figure 2.

Solution of this system of linear equations is faster computationally than the Ricatti algorithm. Also recomputations due to changes in $\hat{x}$, $\hat{u}$, C, or Z are minimal. See [13] for details. The above remarks are summerized in Figure 3.

III. THE INTERACTIVE PROCESS

Figure 4 is a general schematic of the interactive system in which the model is embedded. In the following X=x, U=u, Z=z, XH=$\hat{x}$, UH=$\hat{u}$

The available functions are:

$$
\begin{bmatrix}
M_0 & M_1 & & & \\
M_1' & M_2 & M_1 & & \\
 & M_1' & M_2 & M_1 & \\
 & & & \ddots & \\
 & & & M_1' & M_2
\end{bmatrix}
\begin{bmatrix}
P_1 \\ P_2 \\ P_3 \\ \vdots \\ P_N
\end{bmatrix}
=
\begin{bmatrix}
\hat{x}_1-\xi-A\xi-B\hat{u}_0-Cz_0 \\
\hat{x}_2-\hat{x}_1-A\hat{x}_1-B\hat{u}_1-Cz_1 \\
\hat{x}_3-\hat{x}_2-A\hat{x}_2-B\hat{u}_2-Cz_2 \\
\vdots \\
\hat{x}_N-\hat{x}_{N-1}-A\hat{x}_{N-1}-B\hat{u}_{N-1}-Cz_{N-1}
\end{bmatrix}
$$

$$M_0=-Q^{-1}-BR^{-1}B'$$

$$M_1=Q^{-1}+Q^{-1}A$$

$$M_2=-M_1-M_1'-BR^{-1}B'-AQ^{-1}A'$$

The Modified Langrangian Matrix
Figure 2

CREATE: Retrieve a set of data from a file or input it directly. Initialize local variables.

COMPUTE: Perform the Cholesky decomposition of M (i.e., compute M^{-1} essentially) and the values of variables X and U.

DISPLAY: Display the values of any variable (of the set of data or results).

UPDATE: Update any variable or any element of a variable of the set of data.

SAVE: Save a set of results in a file for future comparative plotting.

RESET: Erase the set of previously saved results.

PLOT: Display a comparative plot of an optimized state control variable and the corresponding nominal trajectory or display a comparative plot of the various sets of results already computed and saved.

INIT: Initialize the file in which different sets of data are to be saved.

PUT: Save the present set of data on file with given name.

GET: Retrieve the set of data with given name from file.

DELETE: Delete the set of data under a given name in file.

MENU: List the available functions.

STOP: Terminate the program.

The word "FUNCTION?" is typed whenever the system requests a new function. A succession of line feeds or carriage returns will always lead to a request for a "FUNCTION?".

PROBLEM I

GIVEN: $\hat{x}_t$, $\hat{u}_t$ and z_t for t=0 to N -1 and $\hat{x}_N$

COMPUTE: x_t for t= 1 to N

u_t for t=0 to N - 1

such that

$$x_{t+1}=x_t+Ax_t + Bu_t + Cz_t \quad t=0,\dots,N-1$$

and minimizing

$$\text{Cost}= \sum_{t=0}^{N} (x_t-\hat{x}_t)'Q(x_t-\hat{x}_t)+ \sum_{t=0}^{N-1} (u_t-\hat{u}_t)'R(u_t-\hat{u}_t)$$

$\Rightarrow$

PROBLEM II

GIVEN : M matrix tri-diagonal block symetric

P vector

COMPUTE: Y such that

M Y = P

x : State variables

$\hat{x}$: Nominal state trajectories

u : Control variables

$\hat{u}$: Nominal control trajectories

z : Exogeneous variables

A :

B : Transformations

C :

M : Tri-diagonal block symetrix matrix of figure 2

M : is only function of A,B,Q,R

P : Right hand side of figure 2

Figure 3

INTERACTIVE PROCESS

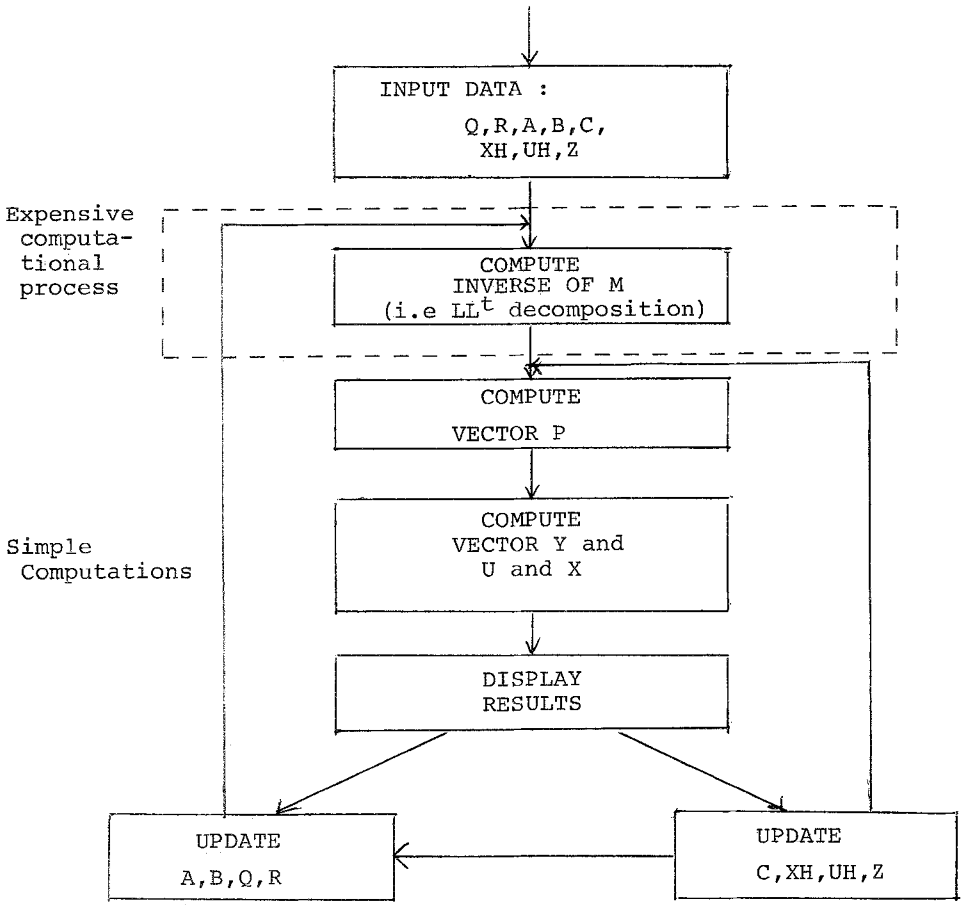

Figure 4

Three files are associated with the program. A file for the creation of data (Fortran File I/O Unit 7), a file to save and retrieve sets of data, corresponding to different problems (File I/O Unit 8), and a file to save and retrieve sets of results, corresponding to different problems (File I/O Unit 9).

The following section details the use of the different functions.

1) CREATE

Type "CREATE".

Create allows the creation of a completely new set of data (for a new problem). At this time and only at this time, the dimensions n, r, k, and N can be modified.

The system responds: "IS THE DATA TO BE FETCHED FROM FILE?"

a) If the user answers "YES" the data will be retrieved from the file associated with I/O Unit 7. The file, for keeping previous sets of results, will be initiated and an "END OF DATA TRANSFER" will be printed before the next "FUNCTION?" to indicate that the transfer was successful.

b) If the user answers "NO" the data must be entered through the terminal console. The system responds: "GIVE THE NUMBER OF STATE VARIABLES, CONTROL VARIABLES, EXOGENEOUS VARIABLES, AND TIME PERIOD RESPECTIVELY LESS THAN (maximum range for the dimensions in format 4I4)". The scalars n, r, k and N are to be entered in format 4I4. Then for each variable A, B, C, Q, R, Z, UH, XH, the following procedure will be repeated. (Remember UH=$\hat{u}$, XH=$\hat{x}$).

α) "IS MATRIX (VECTOR) variable name (first dim., second dim.) TO BE INPUT NON-ZERO ELEMENTS ONLY (IF SO TYPE (ELE)) OR IS THE ENTIRE ARRAY TO BE INPUT (IF SO TYPE ALL)".

 * If the answer is ALL the user will get: "GIVE FORMAT OF INPUT (Type NO for default format)".
 * If the response is NO, the input will be read in under Format control 5E12.4.
 * Otherwise the format will be read and afterwards the variables(s) will be read in the format. (Only the part of the format between the two external parenthesis is entered). The system will now return to α), if there are still variables to read; otherwise it will proceed to β).
 * If the answer is ELE the system will respond: "GIVE THE FORMAT OF INPUT OR TYPE 'NO' FOR DEFAULT I2, 1X, I2, 1X, E12.4 A NEGATIVE SUBSCRIPT WILL TERMINATE".
   - If the answer is NO each non-zero element of the variable must be given in the form:

     ddb̸ddb̸±ddd.dddd±dd (d stands for digit)

     row number

     column number

     A negative row number terminates input and the system returns to α) if there is still a variable to read; otherwise the system proceeds to β).

β) For each variable XNAM, UNAM, ZNAM the system prints "GIVE THE NAME OF THE ROWS OF variable name (Format 10(A6,1X))". Then the user enters the names of the different state variables, if variable name = X; control variables, if variable name = U; exogeneous variables, if variable name = Z in the form <u>ccccccbccccccb--------bcccccc</u>, where b is ignored.

The file for saving sets of results will be initiated and upon completion of β) a "FUNCTION?" will appear.

2) <u>COMPUTE</u>

From a set of data in memory (A, B, C, Q, R, Z, UH, XH) the system computes the optimal state variables and control variables X and U. If A, B, Q, and R are unchanged between two COMPUTER functions, only the part depending on the other variables is recomputed (right hand side of the linear system.) When "FUNCTION?" is printed, the computation is completed.

3) <u>DISPLAY</u>

This function displays any variable in the model. First appears: "COMPARATIVE DISPLAY OF X, XH OR U, UH?"; there are 4 possible answers:

α) "X" will give the names of the different state variables with their associated number. The response is a two digit number corresponding to the variable chosen. (If the number is not valid, [i.e., too big or zero or negative], the program returns to the state: "COMPARATIVE DISPLAY OF X, XH, OR U, UH?"); otherwise, the state variable corresponding to the number is printed; first the optimal result; then the nominal state trajectory. One may then enter a new number corresponding to another state variable.

β) "U" - The process is the same a in α) but for the control variables.

γ) "NO" - Go to the display of one variable.

δ) Any other response will cause the program to print "FUNCTION?"

* Display one variable.

The system displays "PRINT THE NAME OF THE VARIABLE TO BE DISPLAYED X, U, XH, UH, Z, A, B, C, Q, R, OR OUT IF DISPLAY IS TO BE TERMINATED".

α) If X, U, XH, UH, or Z is entered, the different names associated with the variables X or U or XH or UH or Z will be printed with their associated numbers. The response is the 2 digit number corresponding to the variable to be printed. If the number is not valid (too big or zero or negative) the program comes back to the step: Display one variable; otherwise, the corresponding variable is printed and the next variable number is given.

α) If A, B, C, Q, or R is entered, the entire matrix is printed row by row and the program comes back to the step: Display one variable.

β) If "OUT" is entered (or any other response) the program prints "FUNCTION?" and waits for the next function.

4) UPDATE

This function allows the updating of any variable of the model. The program prints: "GIVE THE VARIABLE OR THE MATRIX TO BE UPDATED XH, UH, Z, A, B, C, Q, OR ANY OTHER RESPONSE CAUSES "FUNCTION?" to appear". After the name of the variable to be updated has been entered, the program prints: "DISPLAY OF variable name (# of rows, # of columns)?". If the answer is "YES" the variable will be first displayed and the following will be printed: "ANSWER: NO CORRECTION (NO); CORRECTION OF ALL THE ELEMENTS (ALL); CORRECTION ELEMENT BY ELEMENT (ELE)".

α) If the answer is "ALL", the system will print: "GIVE THE FORMAT OF INPUT OR TYPE "NO" FOR DEFAULT FORMAT OF 5E12.4. The user enters the entire new variable. The program returns to "GIVE THE VARIABLE OR THE MATRIX TO BE UP-DATED..."

β) If the answer is "ELE", it will print: "GIVE THE FORMAT OF INPUT OR TYPE FOR DEFAULT FORMAT OF I2, 1X, I2, 1X, E12.4 A NEGATIVE SUBSCRIPT WILL TERMINATE". The user enters "NO" or the part of the format (between the two extreme parentheses) in which the input is to be read. This format must read, by line, 2 integers and a real number. The first integer is the row number; the second is the column number, and the third is the value updated. A subscript out of range will cause the system to branch to "DISPLAY OF____ ____ ____?"

γ) If the answer is "NO" the system branches to: "GIVE THE VARIABLE OR THE MATRIX TO BE UPDATED XH, UH, _ _ _"

5) SAVE

This function saves on the file corresponding to I/0 Unit 9, up to 5 sets of results catalogued under a given number (from

1 to 5). The following is printed: "NUMBER BY WHICH THE RESULT IS TO BE REFERENCED (1-5)". The user gives a digit by which the set of results, in memory (X and U), is to be referenced in this file. If the digit is not in the range 1 to 5, "INVALID REFERENCE NUMBER number" is printed and the program asks for the next function. If the number is in the range 1 to 5, * and a set of results have been catalogued under this number, the following is printed: "LAST SET OF RESULTS number TO BE DESTROYED?". If user responds "YES" the new set of results is written over the last one. If the response is "NO" the system branches to the step: "NUMBER ON WHICH THE RESULTS IS TO - - - - -" and there is no set of results catalogued under this number. The set of results is transferred to file. At the end of transfer, "END OF DATA TRANSFER" is printed.

6) <u>RESET</u>
All the sets of results catalogued in the file corresponding to I/0 Unit 9 are decatalogued.

7) <u>PLOT</u>
This function allows the plotting of data and computed results. For instance, one can displya a given state variable, compare the optimized solution variable with the nominal trajectory, or display sets of results in file unit 9 simultaneously. The first time PLOT is referenced "NUMBER OF LINE ON THE TTY (EX:15)" is printed. The response is a two digit number, giving the number of lines on which the plot must extend (a page of the device). At every reference, "LAST RESULT (L) OR ALL SAVED RESULTS (G)" is printed. If the user responds with L, the set of result in memory will be plotted, with the nominal trajectory in comparison. If the answer is G, all the sets of results in the file unit 9 will be plotted on the same plotting. After the response "L" or "G", "GIVE THE VARIABLE TO BE PLOTTED, X OR U" will be printed. If a response other than X or U is typed then "FUNCTION?" will appear, ready to receive the next function. If the response is X or U, the corresponding names of the variables will appear, with their reference numbers. In order to choose a variable, one types the two digits number corresponding to the variable. If the number is invalid, the system returns to the message "GIVE THE VARIABLE TO BE PLOTTED, X OR U" else the plotting will occur and the system will wait for the next

two digit variable designator.

8) INIT

This function initializes the file corresponding to I/0 unit 8. This function must be given when a new file is to be used or to reset an existing file. This file corresponds to a file containing different sets of data catalogued under given name. A "FUNCTION?" is typed at the completing of the process.

9) PUT

This function puts a set of data A, B, C, Q, R, Z, UH, XH (the one in memory) in the file I/0 Unit 8 and references it under a given name (6 characters). * If there are already 9 sets of data in the file, a message "ALREADY 9 SETS OF DATA, FILE FULL, A DELETION MUST BE MADE FIRST" appears and "FUNCTION?" is typed. * Otherwise, "GIVE THE NAME BY WHICH THE SET OF DATA IS TO BE CATALOGUED" appears and the user must answer by a name of 6 characters. Then at the end of transfer: "TRANSER COMPLETED" and "FUNCTION?" appears.

10) GET

This function retrieves a set of data under a given name from file Unit 8. First a list of the names under which sets of data have been referenced appears: "WHICH OF THE FOLLOWING DATA DO YOU WANT *** name 1 *** name 2 . . . *** name N". The response is one of the names listed. If the user does not give a proper name, "FUNCTION?" will appear and no set of data will have been retrieved. Otherwise, the proper set of data is retrieved and "TRANSFER COMPLETED" and "FUNCTION?" appears.

11) DELETE

This function decatalogues a set of data and packs the rest of the file. The system responds "GIVE THE SET OF DATA TO BE DELETED" *** name 1 *** name 2 . . . *** name N. The user gives the name of the set of data which is to be deleted. * If the name is incorrect "FUNCTION?" will appear asking for the next function. No deletion occurs. * If the name is correct, it is deleted from the dictionary, the file is packed and a "DELETION COMPLETED" and "FUNCTION?" appears.

12) MENU

This function lists the available functions with a short description of each one.

13) STOP

"RUN STOPPED BY USER. PROCESSING TERMINATED" is printed and the program stops.

IV. CONVERSION TO FULL SCALE GRAPHICS TERMINAL

As implemented now, the system is interactive with limited graphics capabilities. In the next few months the central programming effort will be directed toward software development for interfacing an IMLAC graphics minicomputer as a terminal to the central processor, a UNIVAC 1108.

The IMLAC PDS-1D Graphics system consists of two processors which share 8192 words (16 bits) of memory. One of the processors drives the deflection amplifiers of the CRT and is responsible for creating a display. The other processor can perform arithmetic, logical and branching operations, transfer data to and from peripheral devices, and can start and stop the display processor. This processor constructs display programs, refreshes the display by starting and stopping the display processor and communicates with the external devices.

Because of the dual processor construction of the IMLAC PDS-1D, it can function as a stand alone display system. In this mode of operation, there is an assembler which assembles programs for both processors, a test editor, and a debugging program. The IMLAC PDS-1D was used used in this fashion in conjunction with a course in interactive computer graphics which was taught during the summer of 1974.

The IMLAC PDS-1D is most useful when it is connected to a large scale digital computer, as its computational abilities are quite limited. The University of Houston Computing Center has a UNIVAC 1108 system and the IMLAC PDS-1D is being linked to the 1108. There were three methods by which the data transfer link could be established: (1) A direct channel connection, (2) connection via a synchronous serial data communications channel, or (3) connection via an asynchronous serial data communications channel. Alternative (2) was chosen because alternative (1) would have involved desinging and building a special interface at a very great cost, and alternative (3) would have resulted in a data transfer rate of approximately 15 words per second. The synchronous serial data communication channel offers a transfer rate of 1250 words per second in both directions simultaneously at a cost of under $400 for the necessary cable and interface circuitry. The hardware for this connection was designed and built by an M.S. student in Computer Science and is currently being tested.

Preliminary drive programs have been written for the 1108 and for the IMLAC PDS-1D. There are a number of unsatisfactory aspects of the 1108 program in its present form; it requires that a program using the IMLAC PDS-1D as a peripheral be locked into memory and it contributes a substantial amount of overhead in the communications processing routines of the operating system.

A new version of the EXEC-8 operating system will make it possible to swap a program that uses the IMLAC PDS-1D in and out of memory exactly as is done with other interactive programs. This revision of the operating system implements input and output queues for communications devices on a high speed (4 msec) drum. If the program using the IMLAC PDS-1D is swapped out, any incoming messages from the IMLAC PDS-1D are queued on the drum until the program is swapped back in again.

A series of local revisions to the communications routines of EXEC-8 will substantially reduce the overhead of communication with the IMLAC PDS-1D. These revisions will implement a "buddy system" method of allocating communications buffers that will allow variable length messages to be received without frequent interrupts to check on the status of the input. This overhead will be further reduced when a communication "front end" processor is made a part of the 1108 system.

Univac is contributing a complete graphics processing package for the 1108 which was developed in Japan for the use with Adage graphics systems. After the communications link is established and stablized between the 1108 and the IMLAC PDS-1D, work will begin on converting this program to handle the IMLAC PDS-1D.

In the meantime, a special purpose program has been developed and is being debugged to allow the IMLAC PDS-1D and the 1108 to be used in this research project.

REFERENCES

[1] M. Athans and P. Falb, Optimal Control--An Introduction To The Theory And It's Applications, McGraw-Hill, New York, 1966.

[2] James R. Bunch, "Equilibration of Symmetric Matrices in the Max-norm," Journal of the ACM, vol. 18, no. 4, October 1971, pp. 566-572.

[3] James R. Bunch, "Analysis of the Diagonal Pivoting Method, "SIAM J. Numerical Analysis, vol. 8, no. 4, December 1971, pp. 656-680.

[4] James R. Bunch, and B. N. Parlett, "Direct Methods for Solving Symmetric Indefinite Systems of Linear Equations, SIAM J. Numerical Analysis, vol. 8, no. 4, December 1971, pp. 639-655.

[5] W. E. Bosarge, Jr., and O. G. Johnson, "Direct Method Approximation To The State Regulator Control Problem Using A Ritz-Trefftz Suboptimal Control," IEEE Transactions on Automatic Control, December 1970, pp. 627-631.

[6] W. E. Bosarge, Jr., and O. G. Johnson, "Error Bounds of High Order Accuracy for the State Regulator Problem via Piecewise Polynomial Approximations," SIAM J. Control, vol. 9, no. 1, February 1971, pp. 15-28.

[7] W. E. Bosarge, Jr., and O. G. Johnson, "Numerical Properties of the Ritz-Treffrz Algorithm for Optimal Control," Communications of the ACM, vol. 14, no. 6, June 1971, pp. 402-406.

[8] W. E. Bosarge, Jr., and O. G. Johnson, R. S. McKnight, and W. P. Timlake, "The Ritz-Galerkin Procedure For Nonlinear Control Problems," SIAM J. Numerical Analysis, vol. 10, no. 1, March 1973, pp. 94-111.

[9] R. Courant and D. Hilbert, Methods of Mathematical Physics, vol. 1, Interscience, New York, 1953.

[10] E. H. Cuthill and R. S. Barga, "A Method of Normalized Block Iteration," Journal of the ACM, vol. 6, pp. 236-244.

[11] International Mathematical and Statistical Libraries Newsletter, no. 5, October, 1973.

[12] O. G. Johnson, "Convergence, Error Bounds, Sensitivity, and Numerical Comparisons of Certain Absolutely Continuous Raleigh-Ritz Methods For Sturm-Liouville Eigenvalue Problem," Technical Report No. 23, Computer Center, University of California, Berkeley, 1968.

[13] O. G. Johnson, B. C. McInnis, J. W. Nowakowski, "An Assessment of Discrete Linear Quadratic Tracking Problem Algorithms," IEEE Auto. Cont. (Submitted.)

[14] R. S. Pindyck, "An Application of the Linear Quadratic Tracking Problem To Economic Stabilization Policy," IEEE Transactions On Automatic Control, vol. AC-17, no. 3, June 1972, pp. 387-400.

[15] R. S. Pindyck, Optimal Planning for Economic Stabilization, North-Holland Publishing Company, Amsterdam, 1973.