Optimum digital control synthesis.

Ogden, Don J.

Monterey, California: U.S. Naval Postgraduate School

http://hdl.handle.net/10945/12778
OPTIMUM DIGITAL CONTROL SYNTHESIS

DON J. OGDEN
OPTIMUM DIGITAL CONTROL SYNTHESIS

* * * * * *

Captain Don J. Ogden USMC
OPTIMUM DIGITAL CONTROL SYNTHESIS

By

Don J. Ogden

Captain, United States Marine Corps

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
IN
ENGINEERING ELECTRONICS

United States Naval Postgraduate School
Monterey, California

1965
OPTIMUM DIGITAL CONTROL SYNTHESIS

by

Captain Don J. Ogden USMC

This work is accepted as fulfilling the thesis requirements for the degree of

MASTER OF SCIENCE

IN

ENGINEERING ELECTRONICS

from the

United States Naval Postgraduate School
ABSTRACT

Sampled-data control systems are coming of age. Many methods have been formulated by which both continuous and sampled-data control systems may be optimized. Here one method of optimum control design is presented using the approach of dynamic programming. A program to generate the state transition matrices from the system dynamics was developed. A program for minimizing rather general cost functions was written and examples are presented utilizing the results.
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>PROGRAM OPCON1</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>PROGRAM OPCON2</td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>Example Problem</td>
<td>45</td>
</tr>
<tr>
<td>5</td>
<td>Bibliography</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Appendix I</td>
<td>51</td>
</tr>
<tr>
<td>7</td>
<td>Appendix II</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>Appendix III</td>
<td>62</td>
</tr>
<tr>
<td>9</td>
<td>Appendix IV</td>
<td>69</td>
</tr>
<tr>
<td>Figure</td>
<td>Illustration</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>I-1</td>
<td>Block Diagram of an Open Loop Control System</td>
<td>2</td>
</tr>
<tr>
<td>I-2</td>
<td>Flow Chart for PROGRAM PHIDEL</td>
<td>4</td>
</tr>
<tr>
<td>I-3</td>
<td>Multivariable Digital Control System</td>
<td>5</td>
</tr>
<tr>
<td>II-1</td>
<td>Phase Plane for Example Problem</td>
<td>11</td>
</tr>
<tr>
<td>II-2</td>
<td>Flow Chart of PROGRAM OPCON1</td>
<td>13-16</td>
</tr>
<tr>
<td>II-3</td>
<td>Flow Chart for Subroutine PROD(A,B,C,L,M,N)</td>
<td>17</td>
</tr>
<tr>
<td>II-4</td>
<td>Flow Chart for Subroutine SUM(A,B,C,N)</td>
<td>18</td>
</tr>
<tr>
<td>II-5</td>
<td>Flow Chart for Subroutine TRANSQ(A,B,N)</td>
<td>19</td>
</tr>
<tr>
<td>II-6</td>
<td>Flow Chart for Subroutine TRANCOL(A,B,N)</td>
<td>20</td>
</tr>
<tr>
<td>II-7</td>
<td>Flow Chart for Subroutine Money (MON,E,Q,N)</td>
<td>20</td>
</tr>
<tr>
<td>II-8</td>
<td>Flow Chart for Subroutine PPSI(B,C,D,E,F,N)</td>
<td>21</td>
</tr>
<tr>
<td>II-9</td>
<td>Flow Chart for Subroutine PP(A,B,C,N)</td>
<td>22</td>
</tr>
<tr>
<td>II-10</td>
<td>Flow Chart for Subroutine FF(A,B,C,D,N)</td>
<td>22</td>
</tr>
<tr>
<td>III-1</td>
<td>Flow Chart for PROGRAM OPCON2</td>
<td>29-35</td>
</tr>
<tr>
<td>III-2</td>
<td>Flow Chart for Subroutine ATRAN(AT,P PHI,DEL,R,N)</td>
<td>36</td>
</tr>
<tr>
<td>III-3</td>
<td>Flow Chart for Subroutine PPSI(PSI PHI,DEL,AT,N)</td>
<td>37</td>
</tr>
<tr>
<td>III-4</td>
<td>Flow Chart for Subroutine COST(DOL,Y,Q,R,Z,N)</td>
<td>37</td>
</tr>
<tr>
<td>III-5</td>
<td>Flow Chart for Subroutine PP(P,PSI,Pl,Q,AT,R,N)</td>
<td>38-39</td>
</tr>
<tr>
<td>III-6</td>
<td>Phase Plane for Example Problem</td>
<td>43</td>
</tr>
<tr>
<td>IV-1</td>
<td>Block Diagram of the Open Loop Control System</td>
<td>45</td>
</tr>
<tr>
<td>IV-2</td>
<td>Plot of Control versus Time for the Four Cases</td>
<td>48</td>
</tr>
<tr>
<td>IV-3</td>
<td>Phase Plane Plot of the Four Cases</td>
<td>49</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

In general, any linear control system may be expressed by the following state equation:

\[ \dot{x} = Fx + Du \]  

(1)

where \( x \) represents the system variables and \( u \) are the control variables. \( F \) and \( D \) are \( nxn \) and \( nxm \) matrices defining the dynamical relationship of the states and controls. We will sample the states at discrete intervals \( t_k \) and at a sampling rate with period \( T \). A recursive relationship from the solution of the above differential equation may be established:

\[ x(k+1) = e^{FT}x(k) + \int_0^T e^{F(T-\tau)}Du(k)d\tau \]  

(2)

where

\[ e^{FT} = I + FT + \frac{F^2T^2}{2!} + \cdots + \frac{F^nT^n}{n!} \]  

(3)

Equation (2) may be written:

\[ x(k+1) = \phi x(k) + \Delta u(k) \]  

(4)

where

\[ \phi = e^{FT} \]  

(5)

and

\[ \Delta = \int_0^T e^{F(T-\tau)}Dd\tau \]  

(6)

As an example of the above equations, the open-loop system of Figure I-1 will be considered.
By inspection, the following state equation defines the example system:

\[
\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u
\]  

The \( \phi \) matrix as defined by (5) has only two non-zero terms in the Taylor expansion for the exponential function:

\[
\phi = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} T
\]  
or

\[
\phi = \begin{bmatrix} 1 \\ 0 \end{bmatrix} T
\]

In a like manner, the convolution integral of (6) may be solved for \( \Delta \):

\[
\Delta = \int_{0}^{T} \begin{bmatrix} 1 & T-t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \, dt
\]

giving

\[
\Delta = \begin{bmatrix} T^2/2 \\ T \end{bmatrix}
\]

The system difference equation may now be written as a function of the sampling period \( T \):

\[
x(k+1) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} u(k)
\]  

The calculation of the transition matrix, \( \phi \), and the external forcing function's impulse response matrix, \( \Delta \), from the matrices \( F \) and \( D \),
of the process dynamics was accomplished with the digital computer. This program required an input of $F$ and the sampling rate $T$. The amount of accuracy desired may be varied by the user by inputing the least significant figure desired. This test point for accuracy results from the fact that the sum of all terms in a Taylor series following the test term is less in magnitude than the test term. Figure I-2 is the flow chart of the program. The program has been incorporated into other programs written and presented in this paper.
START

READ:
N, DT, TEST, F, and D

TM = 0.0
TERM = I
WORM = 0.0
φ = TERM

TELM = TERM*DT
DELP = TELM
DEL = 0.0

TERM = WORM
WORM = 0.0
TELM = DELM

TM = TM + 1.0
DELM = DELM - TELM*F*DT/(TM+1.0)
WORM = WORM + TERM*F*DT/TM

TERM = WORM
WORM = 0.0
TELM = DELM

φ = φ + TERM

ABC = |TERM(IR, IC)| max

TEST: ABC

Δ = Δ + φ*DELP*D

PRINT: φ and Δ

STOP

Figure I-2: Flow Chart for PROGRAM PHIDEL
The optimal control of a system depends upon a valid determination of an optimal control policy which will maximize or minimize a performance criteria. This control policy may be generated from criteria involving time, system state variables, noise, control effort, etc. In some cases, this control policy may be generated by active networks or filters incorporated within the system, or a digital computer may be used to generate the control. To accomplish this, all dynamics of the system must be fed back to the digital computer as indicated in Figure I-3.

![Figure I-3: Multivariable Digital Control System](image)

A general approach to the optimization of any dynamic system was developed through the efforts of R. E. Bellman and has been called "dynamic programming". As an example of this method, in a missile problem, one might choose a proper cost function (terminal CEP) and minimize this cost throughout the trajectory. The optimum trajectory, according to the chosen cost function, would be computed.

Dynamic programming appears to be an excellent method for determining an optimal control process because of its adaptability to use on a digital computer. Throughout the remainder of this paper, this technique will be utilized in formulating general programs to synthesize optimum controllers.
CHAPTER II

PROGRAM OPCON1

Again let us consider the system:

\[ \dot{x} = Fx + D u \] (1')

whose discrete solution is

\[ x(k+1) = \phi x(k) + \Delta u(k) \] (4')

where we now take the case where \( u(k) \) is a scalar control and \( \Delta \) is a column vector of system impulse responses due to control inputs.

The recursion formulas will be derived for this system minimizing the sum of the quadratic terms as described by the matrix \( Q \) over an \( N \) stage process. The \( Q \) matrix is specified by the requirements of the user and the application of solid engineering judgment. This cost function may be written:

\[
J(N) = \min \sum_{k=1}^{N} x^T(k)Qx(k) \quad (13)
\]

By starting with the last stage first, equation (7) becomes:

\[
J(N) = \min \left\{ x^T(N)Qx(N) + \sum_{k=1}^{N-1} x^T(k)Qx(k) \right\}
\]

or

\[
J(N) = \min \left\{ x^T(N)Qx(N) \right\} + J(N-1) \quad (14)
\]

It will now be noted that only the \( x^T(N)Qx(N) \) term contains the control policy \( u(N-1) \); therefore, to minimize this function over the last stage, we need only to find:
\[
\frac{\partial (x^T(N)Qx(N))}{\partial u(N-1)} = 0
\]  

By substitution of equation (4) into \( x^T(N)Qx(N) \):

\[
x^T(N)Qx(N) = \left[ \phi x(N-1) + \Delta u(N-1) \right]^T \cdot Q \cdot \left[ \phi x(N-1) + \Delta u(N-1) \right]
\]

and

\[
\frac{\partial (x^T(N)Qx(N))}{\partial u(N-1)} = 2 \Delta^T Q \phi x(N-1) + 2 \Delta^T QA \cdot u(N-1) = 0
\]

or

\[
\therefore u(N-1) = -\frac{\Delta^T Q \phi}{\Delta^T QA} x(N-1)
\]

Equation (18) now represents the optimum control policy \( u(N-1) \). It will be noted that we are dividing by a multiplication of three matrices; this is a valid operation because a row matrix times a square matrix times a column matrix will always yield a scalar number if the multiplication is valid. The multiplication is valid if the dimensions of the respective matrices are proper; and in this case, it will always be valid.

By proceeding back one more stage, the control policy \( u(N-2) \) may be optimized.

\[
J(N) = \text{minimum} \left\{ x^T(N)Qx(N) + x^T(N-1)Qx(N-1) \right\} + J(N-2)
\]

Again by direct substitution, equation (19) becomes:

\[
J(N) = \text{minimum} \left\{ \left[ \phi - \frac{\Delta^T Q \phi}{\Delta^T QA} \right] \cdot \left[ \phi x(N-2) + \Delta u(N-2) \right] \right\}^T Q
\]

\[
\left[ \phi - \frac{\Delta^T Q \phi}{\Delta^T QA} \right] \cdot \left[ \phi x(N-2) + \Delta u(N-2) \right]
\]

\[
+ \left[ \phi x(N-2) + \Delta u(N-2) \right]^T Q \left[ \phi x(N-2) + \Delta u(N-2) \right]
\]

\[
+ J(N-2)
\]
Minimizing with respect to $u(N-2)$, remembering that $J(N-2)$ is not a function of $u(N-2)$; and $\frac{\partial J(N-2)}{\partial u(N-2)} = 0$:

$$\frac{\partial J(N)}{\partial u(N-2)} = 2\Delta^T Q_\phi \Delta x(N-2) + 2\Delta^T Q \Delta u(N-2)$$

$$+ 2\Delta^T \{ \phi - \frac{\Delta^T Q_\phi}{\Delta^T Q \Delta} \} \Delta^T \{ \phi - \frac{\Delta^T Q_\phi}{\Delta^T Q \Delta} \} \Delta u(N-2) = 0 \quad (21)$$

Let $\psi_1 = \left[I - \frac{\Delta^T (Q + P_o)}{\Delta^T Q \Delta} \right] \phi$, where $P_o = 0$.

Therefore

$$\frac{\partial J(N)}{\partial u(N-2)} = \Delta^T Q_\phi \Delta x(N-2) + \Delta^T \psi_1^T Q \psi_1 \Delta x(N-2)$$

$$+ \Delta^T Q \Delta u(N-2) + \Delta^T \psi_1^T Q \psi_1 \Delta u(N-2) = 0 \quad (23)$$

or

$$u(N-2) = -\frac{\Delta^T (Q + \psi_1^T Q \psi_1) \phi}{\Delta^T (Q + \psi_1^T Q \psi_1) \Delta} \Delta x(N-2) \quad (24)$$

Let $P_1 = \psi_1^T Q \psi_1 \quad (25)$

$$u(N-2) = -\frac{\Delta^T (Q + P_1) \phi}{\Delta^T (Q + P_1) \Delta} \Delta x(N-2) \quad (26)$$

The optimum control policy for the $(N-2)$ stage has been found represented by equation (26).

To further the establishment of recursion formulas, the next stage
back must be considered and minimized with respect to \( u(N-3) \). The following symbology will be utilized in this stage of the development:

\[
A = \Delta \left[ \phi \mathbf{x}(N-3) + \Delta u(N-3) \right] \quad (27)
\]

and

\[
B = \Delta \left[ \psi_1 \phi A + \psi_1 \Delta \left( \frac{\Delta^T(Q+P_1) \phi}{\Delta^T(Q+P_1) \Delta} \right) A \right] \quad (28)
\]

The cost function then becomes:

\[
J(N) = \text{minimum} \{ A^T Q A + B^T Q B \} + J(N-3) \quad (29)
\]

Let

\[
\psi_2 = \left[ I - \frac{\Delta A^T(Q+P_1)}{\Delta^T(Q+P_1) \Delta} \right] \phi \quad (30)
\]

Therefore

\[
J(N) = \text{minimum} \left( A^T Q A + \left[ \psi_1 \psi_2 A \right]^T Q \left[ \psi_1 \psi_2 A \right] \right) + J(N-3) \quad (31)
\]

Minimizing the cost function with respect to \( u(N-3) \):

\[
\frac{\partial J(N)}{\partial u(N-3)} = 2 \Delta^T Q \phi \mathbf{x}(N-3) + 2 \Delta^T Q A u(N-3)
\]

\[
+ 2 \Delta^T Q \psi_2^T \psi_1^T Q \psi_1 \psi_2 \phi \mathbf{x}(N-3)
\]

\[
+ 2 \Delta^T Q \phi_2^T \psi_1^T Q \psi_1 \psi_2 \Delta u(N-3) = 0 \quad (32)
\]

Solving for \( u(N-3) \):

\[
u(N-3) = - \frac{\Delta^T Q \phi + \Delta^T \psi_2^T \psi_1^T Q \psi_1 \psi_2 \phi}{\Delta^T Q A + \Delta^T Q \phi_2^T \psi_1 \psi_2 \Delta} \mathbf{x}(N-3) \quad (33)
\]
giving

\[ u(N-3) = - \frac{\Delta^T(Q+\psi T\psi^TQ\psi^T)\phi}{\Delta^T(Q+\psi T\psi^TQ\psi^T)\Delta} \times (N-3) \]  

(34)

Let

\[ P_2 = \psi T\psi^TQ\psi^T \]  

(35)

Therefore

\[ u(N-3) = - \frac{\Delta^T(Q+P_2)\phi}{\Delta^T(Q+P_2)\Delta} \times (N-3) \]  

(36)

It is believed that an iterative relationship may now be shown without regressing to another stage. The following equations may readily be shown to be true by an inductive proof.

\[ \psi_{k+1} = \left[ I - \frac{\Delta \Delta^T(Q+P_k)}{\Delta^T(Q+P_k)\Delta} \right] \phi \]  

(37)

where

\[ P_k = \psi T\psi^T \psi_{k-1} \psi_{k-2} \ldots \psi_{2} \psi_{1} \psi_2 \ldots \psi_{k-1} \psi_k \]  

(38)

or

\[ P_k = \psi T\psi_{k-1}^T \psi_k, \quad P_0 = 0 \]  

(39)

and

\[ u(N-k) = - \frac{\Delta^T(Q+P_{k-1})\phi}{\Delta^T(Q+P_{k-1})\Delta} \times (N-k) \]  

(40)

\[ x(k+1) = \phi x(k) + \Delta u(k) \]  

(41)

To illustrate the use of the above equations, the system presented in Chapter I will be considered with the sampling period equal to one second:

\[ \phi = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad \Delta = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \]
To find $u(k)$, equations 37, 39, and 40 must be solved:

$$\psi_1 = \phi - \frac{\Delta T_Q\phi}{\Delta T_Q\Delta} = \begin{bmatrix} 0 \\ 0 \\ -2 \\ -1 \end{bmatrix}$$

(42)

and

$$p_1 = \psi_1^T \psi_1 = 0$$

(43)

and

$$u(k) = -\frac{\Delta T_Q \phi}{\Delta T_Q \Delta} \phi(k) = -2x_1(k) - 2x_2(k)$$

(44)

By substitution into equation (44) and solving for $u(0)$, we find $x_1(1) = 0$ and $x_2(1) = -2$. The following phase plane (Figure II-1) results:

![Figure II-1: Phase Plane For Example Problem](image)

This clearly indicates that a limit cycle results and that the cost function, as determined with the use of this weighting matrix, yields a relative minimum cost of one unit.

Using the derived recursion formulas 37, 39, 40 and 41, PROGRAM OPCON1 was written. Figures II-2 through II-10 represent abbreviated flow charts of the program and the subroutines used in the program. It
must be remembered that the arithmetic operations represented in the
flow charts are all matrix operations. As can be seen from Figure II-2,
the required inputs to the program must be read from data cards. The
following table (Table II-1) indicates the order that the data cards are
read, the format required on these cards, and the definition of the vari-
able concerned. The primary output of OPCON1 is the value of each state
variable, the control signal, and the cost of each stage. A complete pro-
gram listing may be found in the appendix.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
<th>Format of Data Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Number of stages plus one</td>
<td>I5</td>
</tr>
<tr>
<td>N</td>
<td>Order of system</td>
<td>I5</td>
</tr>
<tr>
<td>Δ</td>
<td>Column vector of system impulse responses due to control inputs</td>
<td>8F10.6 (read from top of column to bottom)</td>
</tr>
<tr>
<td>φ</td>
<td>State transition matrix</td>
<td>8F10.6 (read in order by columns)</td>
</tr>
<tr>
<td>Q</td>
<td>Weighting matrix of the cost function</td>
<td>8F10.6 (read in order by columns)</td>
</tr>
<tr>
<td>x (0)</td>
<td>Initial conditions on the state vector</td>
<td>8F10.6 (read from top of column to bottom)</td>
</tr>
</tbody>
</table>

Table II-1: Definition and Format of Data Cards for Program OPCON1.
READ:
M, N, Δ, ϕ,
Q, x(0)

PRINT:
DATA CARDS

Z(l, I, J) = Q(I, J)
ZZ(I, J) = Z(1, I, J)

Figure II-2: Flow Chart Of Program OPCON1
FIND:
PSI,P,and (Q+P)

\[ P(I,J,K) = P(J,K) \]
\[ \psi(I,J,K) = RR(J,K) \]
\[ Z(I+1,J,K) = ZZ(J,K) \]

FIGURE II-2: (continued)
Figure II-2: (continued)
Figure II-2: (concluded)
Figure II-3: Flow Chart For Subroutine PROD(A,B,C,L,M,N)
Figure II-4: Flow Chart For Subroutine SUM(A,B,C,N)
Figure II-5: Flow Chart For Subroutine TRANSQ(A,B,N)
\[ B(I,1) = A(I,1) \]

Figure II-6: Flow Chart For Subroutine TRANCOL(A,B,N)

Figure II-7: Flow Chart For Subroutine Money(MON,E,Q,N)
Figure II-8: Flow Chart For Subroutine PPSI(B,C,D,E,F,N)
\[ D = Q^*t \]

Figure II-9: Flow Chart For Subroutine PP(A,B,C,N)

\[ E = (Q+P)^*\phi \]

\[ F \]nd:

\[ \Delta T \]

\[ H = \Delta T^*E \]

\[ G = (Q+P)^*\Delta \]

\[ A = \Delta T^*G \]

\[ H = H/A \]

Figure II-10: Flow Chart For Subroutine FF(A,B,C,D,N)
CHAPTER III
PROGRAM OPCON2

Program OPCON2 was written as the final phase for a versatile simulator to determine a constant gain matrix for optimization of a control system according to the following cost function:

\[ J(N) = \min_{k=1}^{N} \sum_{k=1}^{N} (x^T(k)Qx(k) + ru^2(k-1)) \]  

Again u(k) is a scalar control and the Q matrix is determined in the same manner as outlined in the previous chapter. The variable, r, is a weighting constant, (Lagrange multiplier) determined by user needs and acts as a quadratic integral constraint on the control effort; such as a limit on energy necessary to provide the desired control. By letting r equal zero, we have the cost function of OPCON1 and identical results are obtained from the two programs. With the use of OPCON1, no restraints are put on the control required by the system which may result in the necessity of an unlimited supply of fuel to obtain the desired optimal results.

Again let us consider the system:

\[ \dot{x} = Fx + Du \]  

whose discrete solution is:

\[ x(k+1) = \phi x(k) + \Delta u(k) \]  

where u(k) and \( \Delta \) are defined in the same manner as in the previous chapter.

By starting with the last stage first, equation 45 becomes:

\[ J(N) = \min \left\{ x^T(N)Qx(N) + ru^2(N-1) \right\} + J(N-1) \]  

or by substitution of equation 4':

\[ J(N) = \min \left\{ (\phi x(N-1) + \Delta u(N-1))^T Q(\phi x(N-1) + \Delta u(N-1)) + ru^2(N-1) \right\} + J(N-1) \]
We now minimize $J(N)$ with respect to $u(N-1)$ noting that $J(N-1)$ is not a function of $u(N-1)$.

$$\frac{\partial J(N)}{\partial u(N-1)} = 0 = \Delta^T Q \phi x(N-1) + \Delta^T Q \Delta u(N-1) + ru(N-1)$$  \hspace{1cm} (48)

or,

$$u(N-1) = - \frac{\Delta^T Q \phi}{\Delta^T Q \Delta + r} x(N-1)$$  \hspace{1cm} (49)

As in Chapter II, the denominator of equation 49 will always be a scalar number; therefore, no matrix inversion is required.

By proceeding back one more stage, the control policy $u(N-2)$ may be optimized. Equation 45 becomes:

$$J(N) = \min \left[ x^T(N)Qx(N) + x^T(N-1)Qx(N-1) + ru^2(N-1) + ru^2(N-2) \right] + J(N-2)$$  \hspace{1cm} (50)

The following symbology will be used for clarity in this minimization:

$$A = \phi x(N-2) + \Delta u(N-2)$$  \hspace{1cm} (51)

and,

$$B = - \frac{\Delta^T Q \phi}{\Delta^T Q \Delta + r}$$  \hspace{1cm} (52)

By direct substitution using 4', 51, and 52, equation 50 becomes:

$$J(N) = \min \left\{ (\phi A + \Delta BA)^T Q (\phi A + \Delta BA) + A^T Q A + r (BA)^T (BA) + ru^2(N-2) \right\} + J(N-2)$$  \hspace{1cm} (53)

By rearranging terms,

$$J(N) = \min \left\{ A^T Q A + ((\phi + \Delta B) A)^T Q ((\phi + \Delta B) A) + r (BA)^2 + ru^2(N-2) \right\} + J(N-2)$$  \hspace{1cm} (54)

We now let,

$$a_{1}^T = B = - \frac{\Delta^T Q \phi}{\Delta^T Q \Delta + r}$$  \hspace{1cm} (55)

and

$$\psi_1 = \phi + \Delta a_{1}^T$$  \hspace{1cm} (56)

Equation 54 now becomes:

$$J(N) = \min \left\{ A^T Q A + A^T \psi_1 T Q \psi_1 A + r (a_{1}^T A)^2 + ru^2(N-2) \right\} + J(N-2)$$  \hspace{1cm} (57)
We now minimize \( J(N) \) with respect to \( u(N-2) \),

\[
\frac{\partial J(N)}{\partial u(N-2)} = 0 = \Delta^TQ\Delta + \Delta^T\psi_1^TQ\psi_1\Delta + r a_1^T a_1^T + r u(N-2)
\]  

(58)

Solving for \( u(N-2) \) by substitution of 51 into equation 58,

\[
u(N-2) = \frac{-\left\{\frac{\Delta^TQ\Delta + \Delta^T\psi_1^TQ\psi_1 + r a_1^T a_1^T}{\Delta^TQ\Delta + \Delta^T\psi_1^TQ\psi_1 + r a_1^T a_1^T}\right\} \phi}{\Delta^TQ\Delta + \Delta^T\psi_1^TQ\psi_1 + r a_1^T a_1^T + r} \chi(N-2)
\]  

(59)

We now let,

\[
P_1 = \psi_1^TQ\psi_1 + r a_1^T a_1^T
\]  

(60)

Equation 59 becomes:

\[
u(N-2) = -\frac{\Delta^T P_1 \phi}{\Delta^T P_1 \Delta + r} \chi(N-2)
\]  

(61)

As we did in the previous iteration, we let,

\[
a_2^T = -\frac{\Delta^T P_1 \phi}{\Delta^T P_1 \Delta + r}
\]  

(62)

and,

\[
\psi_2 = \phi + a_2^T
\]  

(63)

By proceeding back one more stage, a recursive relationship may be defined.

We shall define the following quantity for brevity in this development:

\[G = \phi \chi(N-3) + \Delta u(N-3)
\]  

(64)

The cost function, 45, may be written as,

\[J(N) = \text{minimum} \left[\chi^T(N)Q\chi(N) + \chi^T(N-1)Q\chi(N-1) + \chi^T(N-2)Q\chi(N-2) + ru^2(N-1) + ru^2(N-2) + ru^2(N-3) \right] + J(N-3)
\]  

(65)

By substitution of 4', 55, 56, 62, 63, and 64, equation 65 becomes:
\[ J(N) = \min \left\{ (\psi_2 \psi_1 G)^T Q (\psi_2 \psi_1 G) + r (\psi_2 a_1 G)^T (\psi_2 a_1 G) \\
+ (\psi_2 G)^T G (\psi_2 G) + r (a_2 G)^T (a_2 G) \\
+ G^T Q G + r u^2 (N-3) \right\} + J(N-3) \] (66)

Minimizing the cost over \( N \) stages with respect to \( u(N-3) \),
\[
\frac{\partial J(N)}{\partial u(N-3)} = 0 = \Delta T \psi_2^T \psi_1 Q \psi_2 + \Delta T Q + r \Delta T a_1 a_2^T + r \Delta T a_2 a_2^T \\
+ r \Delta T Q a_1 a_2^T + r \Delta T a_2 a_2^T + r \Delta T G \] (67)

Substituting 64 into equation 67 and solving for \( u(N-3) \), we find the optimum control policy:
\[
u(N-3) = \frac{\Delta T \psi_2^T \psi_1 Q \psi_2 + \Delta T Q + r \Delta T a_1 a_2^T}{\Delta T \psi_2^T \psi_1 Q + r \Delta T a_1 a_2^T} x(N-3) \] (68)

Combining terms,
\[
u(N-3) = \frac{\Delta T \psi_2^T (\psi_1 Q + r a_1 a_1^T) \psi_2 + \Delta T Q + r \Delta T a_2 a_2^T}{\Delta T \psi_2^T (\psi_1 Q + r a_1 a_1^T) \psi_2 + \Delta T Q + r \Delta T a_2 a_2^T} x(N-3) \] (69)

From equation 60, equation 69 becomes:
\[
u(N-3) = \frac{\Delta T \psi_2^T \psi_1 Q + r a_1 a_1^T}{\Delta T \psi_2^T (\psi_1 Q + r a_1 a_1^T)} x(N-3) \] (70)

We now let,
\[ P_2 = \psi_2^T \psi_1 Q + r a_2 a_2^T \] (71)

and equation 70 becomes:
\[
u(N-3) = \frac{\Delta T P_2 \Delta r}{\Delta T P_2 \Delta r} x(N-3) \] (72)

Let,
\[ a_3^T = - \frac{\Delta^T P_2 \phi}{\Delta^T P \Delta + r} \quad (73) \]

and,
\[ \psi_3 = \phi + \Delta a_3^T \quad (74) \]

As in Chapter II, a recursive relationship may now be shown without proceeding back another stage and may be shown to be true by an inductive proof. Let
\[ a_k^T = - \frac{\Delta^T P_{k-1} \phi}{\Delta^T P_{k-1} \Delta + r}, \quad a_0^T = 0 \quad (75) \]

and,
\[ \psi_k = \phi + \Delta a_k^T, \quad \psi_0 = 0 \quad (76) \]

where,
\[ P = \psi_k^T P_{k-1} \psi_k + Q + r a_k^T, \quad P_o = Q \quad (77) \]

and,
\[ u(N-k) = a_k^T x(N-k) \quad (78) \]

from,
\[ x(k+1) = \phi x(k) + \Delta u(k) \quad (79) \]

By letting \( r \) equal zero and solving the same illustrative example as in the previous chapter, the same results are obtained as illustrated below.

\[ a_1^T = - \frac{\Delta^T Q \phi}{\Delta Q \Delta + r} = \begin{bmatrix} -2.0 & -2.0 \end{bmatrix} \quad (80) \]

and,
\[ \psi_1 = \phi + \Delta a_1^T = \begin{bmatrix} 0 & 0 \\ -2.0 & -1.0 \end{bmatrix} \quad (81) \]

and,
\[ P_1 = \psi_1^T Q \psi_1 + r a_1^T = \begin{bmatrix} 1.0 & 0 \\ 0 & 0 \end{bmatrix} \quad (82) \]
and therefore,

\[ u(k) = a_1^T x(k) = -2x_1(k) - 2x_2(k) \]  \hspace{1cm} (83)

Equation 83 indicates the same results as in Chapter II and prescribes the same phase plane (Figure II-1).

Through the use of the recursion formulas 75, 76, 77, 78, and 79, PROGRAM OPCON2 was written. Figures III-1 through III-5 and Figures II-3 through II-6 represent abbreviated flow charts of the program and the subroutines used in the program.
Figure III-1: Flow Chart for PROGRAM OPCON2.
Figure III-1: (continued)
Figure III-1: (continued)
Figure III-1: (continued)
PRINT:
L, Z, X(1), and TCOST

```
DELTA(J,1) = Z * DEL(J,1)
X(J,1) = X(J,1) + DELTA(J,1)
AT(1,J) = ALFAT(I,J)
```

Figure III-1: (continued)
Figure III-1: (continued)
Figure III-1: (continued)
Figure III-2: Flow Chart for SUBROUTINE ATRAN(AT,P,PHI,DEL,R,N)
Figure III-3: Flow Chart for SUBROUTINE PPSI(PSI, PHI, DEL, AT, N)

Figure III-4: Flow Chart for SUBROUTINE COST(DOL, Y, Q, R, Z, N)
Figure III-5: Flow Chart for SUBROUTINE PP(P,PSI,P1,Q,AT,R,N)
Figure III-5: (concluded)
Figure III-1 shows that the inputs to the program must be read from data cards. Table III-1 shows the order in which these cards are read, the format required on each card, and the definitions of the variables concerned.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
<th>Format of Data Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Number of stages Plus one</td>
<td>I9</td>
</tr>
<tr>
<td>N</td>
<td>Order of system</td>
<td>I9</td>
</tr>
<tr>
<td>IT</td>
<td>Frequency of the desired printing of output</td>
<td>I9</td>
</tr>
<tr>
<td>r</td>
<td>Weighting constant of cost function</td>
<td>F10.6</td>
</tr>
<tr>
<td>A</td>
<td>Column vector of system impulse responses due to control inputs</td>
<td>8F10.6 (read from top to bottom of column)</td>
</tr>
<tr>
<td>φ</td>
<td>State transition matrix</td>
<td>nF10.6 (read in a row at a time)</td>
</tr>
<tr>
<td>Q</td>
<td>Weighting matrix of the cost function</td>
<td>nF10.6 (read in a row at a time)</td>
</tr>
<tr>
<td>x(0)</td>
<td>Initial conditions on the state vector</td>
<td>8F10.6 (read from top to bottom of column)</td>
</tr>
</tbody>
</table>

Table III-1: Definitions and Format of Data Cards for PROGRAM OPCODE.
The output of PROGRAM OPCON2 is the value of each state variable, the control signal, and the total cost of all previous stages including the present stage. The amount of printed output may be varied by the use of the IT variable's input data card. If all output is to be printed, then IT must be set to one by the data card. If the user wishes to have 1,000 stages and has a requirement of print out of only a sampling of these stages, IT should be set accordingly. By letting IT equal ten, only every tenth stage will be printed out. Another valuable output is the value of $a_T^T K$ which approaches the constant gain matrix necessary for optimal control of a continuous system. This is achieved by using a small sample period and many iterations thereby approaching the simulation of a continuous system. OPCON2 is limited for this use because the computer memory size limits the program to approximately 2,000 stages. If the user has no need for the outputs of the values of the control signals, state variables and total costs, then the program is unlimited in its number of stages because the value of $a_T^T K$ may be printed out as often as desired without storing it in memory. This alteration has been made in PROGRAM OPCON3, and a listing of this program may be found in the appendix.

The versatility of this program may be further shown in using it to simulate an optimum control in accordance with the cost function,

$$J(N) = \text{minimum} \left[ x_T^T(N) x(N) \right]$$

(84)

which we shall call the final value criterion. Using the same method of derivation as before, we find the following recursion formulas:
\[ \psi = \phi + \Delta a^T \quad \psi = 0 \quad (85) \]

\[ a^T_K = - \frac{\Delta^T P_{K-1} \phi}{\Delta^T P_{K-1} \Delta}, \quad a^T_0 = 0 \quad (86) \]

where,

\[ P_{K-1} = \psi^T_{K-1} P_{K-2} \psi_{K-1}, P_0 = I \quad (87) \]

and,

\[ u(N-K) = a^T_K(N-K) \quad (88) \]

To use OPCON2 for this cost function, we need only to set \( P_0 \) equal to the identity matrix, \( Q \) to the null matrix, and \( r \) equal to zero.

As an example of this cost function, we shall again use the same illustrative system. We find that:

\[ a^T_1 = - \frac{\Delta^T \phi}{\Delta^T \Delta} \left[ \begin{array}{c} -.4 \\ -1.2 \end{array} \right] \quad (89) \]

and,

\[ \psi_1 = \phi + \Delta a^T_1 = \left[ \begin{array}{cc} .8 & .4 \\ -.4 & -.2 \end{array} \right] \quad (90) \]

and,

\[ P_1 = \psi_1^T \psi_1 = \left[ \begin{array}{cc} .8 & .4 \\ .4 & .2 \end{array} \right] \quad (91) \]

and,

\[ a^T_2 = - \frac{\Delta^T P_1 \phi}{\Delta^T P_1 \Delta} \left[ \begin{array}{c} -1.0 \\ -1.5 \end{array} \right] \quad (92) \]

therefore,

\[ u(0) = -X_1(0) - 1.5X_2(0) \quad (93) \]
Given initial conditions of $X_1(0) = 1.0$ and $X_2(0) = 0.0$, equation 93 results in,

$$u(0) = -1.0$$

therefore,

$$X_1(1) = X_1(0) + 0.5u(0) = 0.5$$

and,

$$X_2(1) = X_2(0) + u(0) = -1.0$$

From equation 89,

$$u(1) = -0.4X_1(1) - 1.2X_2(1) = 1.0$$

and,

$$X_1(2) = X_1(1) + X_2(1) + 0.5u(1) = 0.0$$

and,

$$X_2(2) = X_2(1) + u(1) = 0.0$$

Figure III-6 is the phase plane resulting from the calculated controls and shows that the final states at $N$ equal two are definitely at a minimum and the cost function equals zero.

Figure III-6: Phase Plane for Example Problem.
With the use of OPCON2 and OPCON3, many control systems may be simulated. Using these two programs, a system may be designed for a number of optimization criteria while greatly reducing design time. In the next chapter, a typical system will be considered assuming an optimization criterion and the resulting design will be obtained.
CHAPTER IV

EXAMPLE PROBLEM

To show the use of the programs that were written and developed in the preceding chapters, the following example problem was chosen. The transfer function is typical of a servo motor, the roll attitude control of a missile, and many others. Figure IV-1 shows the open loop plant to be controlled.

\[ \dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \] (100)

With the use of PROGRAM PHIDEL and a sample period of one second, \( \phi \) and \( \Delta \) are:

\[ x(k+1) = \begin{bmatrix} 1.00 & .632 \\ 0.0 & .368 \end{bmatrix} x(k) + \begin{bmatrix} .368 \\ .632 \end{bmatrix} u(k) \] (101)

To show the flexibility of PROGRAM OPCON2, four different cost functions will be considered and compared. The first cost function will be the terminal value function for time optimal control. (Case I)

\[ J(N) = \text{minimum} \; x^T(N)x(N) \] (84')

This will then be compared to the terminal value control with an energy control shown by equation (102). (Case II)
\[ J(N) = \text{minimum} \sum_{k=1}^{k=N} x^T(N)x(N) + \sum_{k=1}^{k=N} u^2(K-1) \]  
(102)

The next case of interest minimizes the sum of the squares of the states with an energy limitation. This is represented by equation (45').

(Case III)

\[ J(N) = \text{minimum} \sum_{k=1}^{k=N} x^T(K)Qx(K)+ru^2(K-1) \]  
(45')

where,

\[ Q = I \quad \text{and} \quad r = 1.0 \]  
(103)

The final cost function will also be that of minimizing the states with an energy limitation; however, by use of PROGRAM OPCON3, a steady state feedback matrix was found. These values were then used in PROGRAM OPCON2 to obtain a transient solution. (Case IV) The elements of this control are:

\[ a^T(1,1) = -0.627797 \]  
(104)

and,

\[ a^T(1,2) = -0.615037 \]  
(105)

With the use of PROGRAM OPCON2, a transient solution was obtained in each of the four cases. Case II was computed for a twenty second trajectory in the phase plane, Figure IV-3. Cases III and IV were each run for ten second trajectories while Case I was optimized in two samples.

Figure IV-2 is a plot of control versus time for each of the cases. It is of interest to note that both cases III and IV utilize the same controls during the first seventy percent of their trajectories; only when the control is very small in the last couple of sampling points does the control required differ. This thereby leads to identical trajectories in the phase plane indicating that constant gain amplifiers in the
feedback loop are just as effective as variable gain amplifiers for compensation of this plant. Case II requires nearly a constant control effort after the initial control jump has been made. The required control for the twenty second trajectory is approximately equal to one half of the necessary control for a ten second trajectory.

Figure IV-3 is a phase plane plot of the four cases. One thing to note here is the trajectory of case II. As the total number of sample periods is increased, the necessary control is decreased as indicated above. This results in the trajectory approaching the $x_1$ axis as the number of sample points increases indefinitely. This means that the cost function used is optimized when the final value of the states is zero and the control is nearly zero. The point in the phase plane would drift in requiring only a very slight amount of control to stop it at its steady state value.
Figure IV-2: Plot of Control Versus Time for the Four Cases.
Figure IV-3: Phase Plane Plot of the Four Cases.


APPENDIX I

PROGRAM PHIDEL

PROGRAM PHIDEL
DIMENSION F(12,12),PHI(12,12),TERM(12,12),WORM(12,12)
IDEL(12),DELM(12,12),TELM(12,12),DELP(12,12),D(12)

C THIS PROGRAM CALCULATES THE STATE TRANSITION MATRIX, PHI, AND THE
C EXTERNAL FORCING FUNCTIONS IMPULSE RESPONSE MATRIX, DEL, FROM THE
C MATRICES F, E AND D. THIS IS ACCOMPLISHED BY USING THE TAYLOR SERIES
C EXPANSION FOR THE EXPONENTIAL FUNCTION. THE CALCULATION FOR THE
C DEL MATRIX IS GOOD FOR ONLY NON TIME VARYING DYNAMICS BECAUSE THE
C INTEGRATION IS ACCOMPLISHED WITHIN THE PROGRAM FOR THIS CASE ONLY.
C
C PHI=EXP**((F*T)=1+(F*T)+((F*T)**2)/(1.0*2.0)+...+((F*T)**N)/(1.0**(N))
C
C DEL=INTEGRAL FROM ZERO TO DT OF PHI(DT-TAU)*D*DTAU
C
C THE REQUIRED INPUT DATA CARDS ARE:
C
C (A) N (SIZE OF THE INPUT MATRIX; FORMAT 15)
C (B) DT (SAMPLING PERIOD; FORMAT 1F15.12)
C (C) TEST (TEST FOR MAXIMUM ALLOWABLE NUMBER IN A TERM TO TRUNCATE
C THE SERIES; FORMAT 1F15.12)
C (D) F (DISTRIBUTION MATRIX; FORMAT MUST BE INSERTED, STATEMENT
C NUMBER 31; MF10.6)
C (E) D (COLUMN VECTOR RELATING THE CONTROL VARIABLES TO THE SYSTEM
C DYNAMICS; FORMAT BF10.6)
C
C OTHER FORMAT CARDS WHICH MUST BE INSERTED ARE NRS 34 AND 35 WHICH

APPENDIX I 1-1 PROGRAM PHIDEL (CONTINUED)
C ALLOWS PHI AND THE EXPONENTIAL SERIES TERMS TO BE PRINTED OUT IN
C THEIR MATRIX FORM. THIS PROGRAM TERMINATES WHEN ALL ELEMENTS OF THE
C SERIES TERMS ARE LESS THAN THE TEST VARIABLE, THUS GUARANTEEING THAT
C THE SUM OF ALL TERMS TO FOLLOW IS LESS THAN THE LAST ELEMENTS
C CALCULATED.

C READ DATA AND INITIALIZE

READ 32 (N)
READ 33 (DT)
READ 33 (TEST)
READ 31 ((F(IR,I,C),IC=1,N),IR=1,N)
READ 31 (D(I),I=1,N)
TM=0.0
PRINT 56;DT
PRINT 54 ((F(IR,I,C),IC=1,N),IR=1,N)
PRINT 55 (D(I),I=1,N)
DO 40 IR=1,N
DO 40 IC=1,N
TERM(IR,IC)=0.0
WORM(IR,IC)=0.0
TERM(IR,IR)=1.0
TELX(IR,IC)=TERM(IR,IC)*DT
DELP(IR,IC)=TELX(IR,IC)
DELM(IR,IC)=0.0
DEL(IR)=0.0
40 PHI(IR,IC)=TERM(IR,IC)

C CALCULATE THE ELEMENTS OF EACH TERM IN THE SERIES

44 TM=1.0+TM

APPENDIX I I-2 PROGRAM PHIDEL (CONTINUED)
DO 50 IR=1,N
DO 50 IC=1,N
DO 50 JN=1,N
DL(IS, IC) = DL(IS, IC) + TELM(IS, JN) * (JN, IC) * DT / (TM+1.0)
50 WORM(IS, IC) = TELM(IS, JN) * EL(JN, IC)

DO 41 IR=1,N
DO 41 IC=1,N
TERM(IS, IC) = WORM(IS, IC)
TELM(IS, IC) = DL(IS, IC)
DEL P(IS, IC) = DELP(IS, IC) + TELM(IS, IC)
DEL M(IS, IC) = 0.0
41 WORM(IS, IC) = 0.0
M=TM
PRINT 24, M, ((TERM(IS, IC), IC=1,N), IR=1,N)

C SUMMATION OF THE TERMS IN THE SERIES

DO 51 IR=1,N
DO 51 IC=1,N
51 PHI(IS, IC) = PHI(IS, IC) + TERM(IS, IC)

C TEST FOR THE TERMINAL TERM IN THE SERIES AND PRINT THE PHI MATRIX

ABC=0.0
DO 42 IR=1,N
DO 42 IC=1,N
AA=TERM(IS, IC)
AB=ABSF(AA)
IF(ABC-AB)<43, 43, 42
43 ABC=ABC
42 CONTINUE
IF (TEST-ABC)45,45,46
45 GO TO 44
46 PRINT 35 ((PHI(IR,IC),IC=1,N),IR=1,N)

C CALCULATE AND PRINT THE DEL MATRIX

DO 52 I=1,N
DO 52 K=1,N
DO 52 J=1,N
52 DEL(I)=DEL(I)+PHI(I,J)*DELP(J,K)*D(K)
PRINT 53 (DEL(I),I=1,N)
31 FORMAT (8F10.8)
32 FORMAT (15)
33 FORMAT (1F15.12)
34 FORMAT (///7:HF(I,J),I2/(3F12.8))
35 FORMAT (///9X,8HPHI(I,J)///(3F12.8))
53 FORMAT (///9X,5HDEL(I)///(8F15.9)///)
54 FORMAT (///7HF(I,J)=///,(3F10.4))
55 FORMAT (///5HD(I)=///,(3F10.4))
56 FORMAT (///,3HDT=(1F10.8))
END
END

APPENDIX I  I-4  PROGRAM PHIDEL (CONCLUDED)
APPENDIX II

PROGRAM OPCON1

PPROGRAM OPCON1
ODIMENSION A(8,8),B(8,8),Q(8,8),D(8,8),X(8,40),G(40),H(8,8),
1P(40,8,8), R(40,8,8), Z(40,8,8), COST(40), Y(8,8), ZZ(8,8),
2P(R(8,8),P1(8,8),SM(40,8,8),F(40,9,8),AA(8,8),XX(8,8),E(8,8),
3MON(8,8),SMN(8,8),U(40),UU(5,8)

C THIS PROGRAM IS DERIVED USING THE COST FUNCTION, J(N)=MINIMUM(SUM
C X(N)T*Q*X(N)), THE PROGRAM WILL YIELD OPTIMUM CONTROL FOR UP TO AN
C EIGHTH ORDER SYSTEM OVER 40 STAGES. THE INPUT DATA CARDS ARE,
C
C (A) M, (NUMBER OF STAGES PLUS ONE, FORMAT 15)
C (B) N (ORDER OF SYSTEM, FORMAT 15)
C (C) DEL (COLUMN MATRIX READ IN ORDER BY ROW, FORMAT 8F10.6)
C (D) PHI (TRANSITION MATRIX READ IN A COLUMN AT A TIME, FORMAT 8F10.6)
C (E) Q (NXN WEIGHTING MATRIX READ IN SAME AS PHI, FORMAT 8F10.6)
C (F) X (COLUMN VECTOR OF INITIAL STATE CONDITIONS, FORMAT 8F10.6)

C THE FOLLOWING RECURSIVE EQUATIONS WERE DERIVED AND ARE USED AS THE
C BASIS FOR THE OPTIMIZATION OF CONTROL.

C PSI(K+1) = (I-(DEL*DELT*(Q+P(K)))/(DELT*(Q+P(K))*DEL)) * PHI
C (1)
C P(K-1) = PSIT(K-1)*P(K-2)*PSI(K-1)
C (2)
C U(N-K) = -((DELT*(Q+P(K-1))*PHI)/DELT*(Q+P(K-1))*DEL))*X(N-K)
C (3)
C X(K+1) = PHI*X(K) + DEL*U(K)
C (4)

APPENDIX II II-1 PROGRAM OPCON1 (CONTINUED)
C EQUATIONS 1 AND 2 CONSTITUTE SUBROUTINES IN THIS PROGRAM. ANOTHER
C SUBROUTINE GENERATES THE RESULTING COEFFICIENT MATRIX FOR X(N-K) IN
C EQUATION 3. (SUBROUTINE FF)

READ 202 (V)
READ 202 (N)
READ 201 (A(I,J), I=1,N)
READ 201 ((B(I,J) I=1,N) J=1,N)
READ 201 ((Q(I,J) I=1,N) J=1,N)
READ 201 (X(I,1), I=1,N)
PRINT 202 (M)
PRINT 202 (N)
PRINT 201 (A(I,1), I=1,N)
PRINT 201 ((B(I,J) I=1,N) J=1,N)
PRINT 201 ((Q(I,J) I=1,N) J=1,N)
PRINT 201 (X(I,1), I=1,N)
CALL DELTA (D,A,N)
DO 71 I=1,N
DO 71 J=1,N
Z(I,J)=Q(I,J)
71 ZZ(I,J)=Z(I,1,J)
DO 72 I=1,N
CALL PPS1 (D,ZZ,A:3,PP,N)
CALL PP (PP,C,P1,N)
CALL SU' (ZZ,O,P1,N)
DO 72 J=1,N
DO 72 K=1,N
P(I,J,K)=P1(J,K)
R(I,J,K)=R1(J,K)
72 Z(I+1,J,K)=ZZ(J,K)
PRINT 1004
PRINT 102 (((I,J,K,R(I,J,K),K=1,N), J=1,N), I=1,M)
PRINT 1005
PRINT 102 (((I,J,K,P(I,J,K),K=1,N), J=1,N), I=1,M)

APPENDIX II  II-2  PROGRAM ORCON1 (CONTINUED)
PRINT 1006
PRINT 1002 ( (I, J, K, Z(I, J, K), K=1, M), J=1, N), I=1, M)
DO 73, I=1, M
DO 73, J=1, N
DO 73, K=1, N
73  SM(I, J, K) = Z(M-1, J, K)
PRINT 1007
PRINT 1002 ( (I, J, K, SM(I, J, K), K=1, M), J=1, N), I=1, M)
DO 74, I=1, N-1
DO 75, J=1, N
DO 75, K=1, N
75  SMM(J, K) = SM(I, J, K)
CALL FA(H, A, B, SMM, N)
DO 20, J=1, N
DO 20, K=1, N
20  H(J, K) = -4(J, K)
DO 76, J=1, N
DO 76, K=1, N
76  Y(J, K) = 0.0
DO 79, J=1, N
79  Y(J, I) = X(J, I)
CALL PRD(UU, H, Y, 1, 1, N)
U(I)=UU(1, I)
CALL PRD(AA, A, UU, N; 1, I)
CALL PRD(XX, B, Y, N; 1, 1)
DO 77, J=1, N
77  X(J, I+1) = XX(J, I) + AA(J, I)
DO 78, J=1, N
F(I, 1, J) = H(I, J)
78  E(J, I) = X(J, I)
CALL MONEY(MON, F, 0, N)
PRINT 100 (I, MON(1, 1))
74  COST(I) = MON(I, 1)
U(N) = 0.0
DO 85 I=1,N
DO 86 J=1,N
85 F(M,I,J)=0.0
DO 86 I=1,N
86 E(I,1)=X(I)**M
CALL MONET("MON,F,O,N")
PRINT 100 ("*MON(1,1)")
COST(M)=MON(1,1)
PRINT 1000
PRINT 100 (I,U(I);I=1,M)
PRINT 1001
PRINT 101 ((J,I,X(J,I);J=1,N),I=1,M)
PRINT 1003
PRINT 101 ((J,I,F(I,1);J=1,N),I=1,M)
100 FORMAT (15,F20.15)
101 FORMAT (215,F20.15)
102 FORMAT (215,F20.15)
201 FORMAT(8F10.6)
202 FORMAT(15)
1000 FORMAT (///,25H M " " U(M) " " //)
1001 FORMAT (///,30H M " " X(M) " " //)
1002 FORMAT (///,25H M " " COST(M) " " //)
1003 FORMAT (///,30H N " " F(M,1,N) " " //)
1004 FORMAT (///,25H M " " ROW COL PSI(M,ROW,COL) " //)
1005 FORMAT (///,25H M " " ROW COL PSI(M,ROW,COL) " //)
1006 FORMAT (///,25H M " " ROW COL O+P(M,ROW,COL) " //)
1007 FORMAT (///,35H M " " ROW COL PSI(M,ROW,COL) " //)
END

C THE FOLLOWING SUBROUTINE CALCULATES PSI(N).

SUBROUTINE PSI (B,C,D,E,F,N)
ODIMENSION A(8,8),B(8,8),C(8,8),D(8,8),E(8,8),F(8,8),G(8,8),H(8,8),
1A(8,8),BB(8,8),DD(8,8),CC(8,8),GG(8,8)
CALL TRANSQ (B,G,N)

APPENDIX II II-4 PROGRAM OPCON1 (CONTINUED)
CALL PROD (AA,G,C,N,N,N)
CALL PROD (BB,B,AA,N,N,N,N)
CALL TRANSQ(D,DD,N)
CALL PROD (CC,C,D,N,1,N)
CALL PROD (EE,DD,CC,1,1,N)
DO 60 I=1,N
DO 60 J=1,N
  60 A(I,J)=0.0
  DO 61 I=1,N
  61 A(I,I)=1.0
  DO 62 I=1,N
  62 BB(I,J)=BB(I,J)/EE(I,I)
  DO 63 I=1,N
  63 BB(I,J)=A(I,J)-BB(I,J)
  DO 64 I=1,N
  CALL PROD (F,BB,E,N,N,N)
END

C THE FOLLOWING SUBROUTINE CALCULATES P.

SUBROUTINE PP(A,B,C,N)
DIMENSION A(B,B),B(B,B),C(B,B),D(B,B),AA(B,B)
CALL TRANSQ(A,AA,N)
CALL PROD(D,B,A,N,N,N)
CALL PROD(C,AA,D,N,N,N)
END

C THE FOLLOWING SUBROUTINE CALCULATES THE COEF. OF THE STATE VARIABLES
C FOR THE CONTROL POLICY.

SUBROUTINE FF(A,B,C,D,N)
DIMENSION A(B,B),B(B,B),C(B,B),D(B,B),E(B,B),BB(B,B),S(B,B),
  IH(B,B)
CALL PROD(E,D,C,N,N,N)

APPENDIX II    II-5    PROGRAM OPCONI (CONTINUED)
CALL TRANCOL(B, BB, N)
CALL PROD(A, BB, E, 1, N, N)
CALL PROD(G, D, B, N, 1, N)
CALL PROD(H, BB, S, 1, 1, N)
DO 64 I=1, N
64 A(1, I) = A(1, I) / H(1, 1)
END

C THIS SUBROUTINE CALCULATES THE COST AT EACH STAGE.

SUBROUTINE MONEY(MON, E, Q, N)
DIMENSION MON(8, 8), E(8, 8), ET(8, 8), Q(8, 8), MO(8, 8)
CALL TRANCOL(E, ET, N)
CALL PROD(MO, Q, E, N, 1, N)
CALL PROD(MON, ET, MO, 1, 1, N)
END

C THIS SUBROUTINE TRANSPOSES ANY COLUMN MATRIX (1, N)

SUBROUTINE TRANCOL(A, B, N)
DIMENSION A(8, 8), B(8, 8)
DO 12 I=1, N
12 B(1, I) = A(1, I)
END

C THIS SUBROUTINE TRANSPOSES ANY SQUARE MATRIX (N X N)

SUBROUTINE TRANSQ(A, B, N)
DIMENSION A(8, 8), B(8, 8)
DO 11 I=1, N
11 B(J, 1) = A(I, J)
DO 11 J=1, N
END

C THIS SUBROUTINE CALCULATES THE PRODUCT OF ANY TWO MATRICES AS LONG

APPENDIX II

II-6

PROGRAM OPCON1 (CONTINUED)
C AS THE PRODUCT IS ALGEBRAICALLY PROPER

SUBROUTINE PROD (A, B, C, L, M, N)
DIMENSION A(3,8), B(8,8), C(8,8)
DO 10 I=1,L
DO 10 J=1,N
A(I,J)=0.0
DO 10 K=1,N
10 A(I,J)=A(I,J)+B(I,K)*C(K,J)
END

SUBROUTINE DELTA(D, A, N)
DIMENSION D(8,8), A(8,8)
DO 42 I=1,8
DO 42 J=1,8
42 D(I,J)=0.0
DO 41 I=1,N
41 D(I,1)=A(I,1)
END

C THE FOLLOWING SUBROUTINE SUMS ANY TWO NXN MATRICES.

SUBROUTINE SUM(A, B, C, N)
DIMENSION A(8,8), B(8,8), C(8,8)
DO 51 I=1,N
DO 51 J=1,N
51 A(I,J)=B(I,J)+C(I,J)
END

APPENDIX II - II-7 PROGRAM OPCON1 (CONCLUDED)
APPENDIX III

PROGRAM OPCON2

PROGRAM OPCON2
ODIMENSION AT(8,8), ALFAT(2000,8), Q(6,8), P(6,8), P1(8,8), DEL(6,8), PSI(8,8), PHI(6,8), X(6,8), Y(8,8), DELTA(6,8), J(8,8)

C THIS PROGRAM HAS BEEN DERIVED USING A COST FUNCTION, J(N)=MINIMUM(SUM
C X(N)T*Q*X(N)+SUM R*U(N-1)**2). THE DIMENSION STATEMENTS LIMIT THE
C PROGRAM TO AN EIGHTH ORDER SYSTEM WITH 2000 ITERATIVE STAGES. THE
C FOLLOWING ITERATIVE EQUATIONS WERE DEVELOPED AND DERIVED USING R.E.
C DEHMANS PRINCIPLES OF DYNAMIC PROGRAMMING.
C A(K)T=-(DELT*P(K-1)*PHI)/(DELT*P(K-1)*DEL+R) (1)
C PSI(K)=PHI+DEL*A(K)T, PSI(0)=0 (2)
C P(K)=PSI(K)T*X(K-1)*PSI(K)+Q+R*A(K)*A(K)T, P(0)=Q (3)
C X(K)=PHI*X(K-1)+DEL*U(K-1) (4)
C U(N-K)=A(K)T*X(N-K) (5)

C EQUATIONS 1, 2, AND 3 CONSTITUTE SUBROUTINES IN THIS PROGRAM.
C THE DATA CARDS REQUIRED ARE LISTED IN ORDER,
C (A) M (NUMBER OF STAGES, FORMAT 15)
C (B) N (ORDER OF SYSTEM, FORMAT 15)
C (C) IT (MULTIPLE OF RESULTS OF STAGES TO BE PRINTED, FORMAT 15)

APPENDIX III I III-1 PROGRAM OPCON2 (CONTINUED)
C (D) R (CONSTANT IN THE COST FUNCTION WHICH WEIGHS THE SUM OF THE
C CONTROL, FORMAT 1F10.6)
C (E) DEL (THE SYSTEM IMPULSE RESPONSE MATRIX, FORMAT 8F10.6, READ
C IN BY COLUMN)
C (F) PHI (THE SYSTEM TRANSITION MATRIX, FORMAT NF10.6, READ IN BY
C ROWS)
C (G) Q (WEIGHTING MATRIX USED IN THE COST FUNCTION, FORMAT NF10.6,
C READ IN BY ROWS)
C (H) X (INITIAL CONDITIONS ON THE COLUMN VECTOR OF STATES, FORMAT
C 8F10.6, READ IN BY COLUMN)
C
C FORMAT STATEMENT NR 8 WILL HAVE TO BE ALTERED DEPENDING UPON THE
C ORDER OF THE SYSTEM, ITS FORMAT SHOULD BE NF10.6.

C READ DATA AND INITIALIZE VARIABLES.

READ 2 (H)
READ 2 (N)
READ 2 (IT)
READ 3 (R)
READ 1 (DEL(I,1),I=1,N)
READ 8 ((PHI(I,J),J=1,N),I=1,N)
READ 8 ((Q(I,J),J=1,N),I=1,N)
READ 1 (X(I,1),I=1,N)
PRINT 2 (M)
PRINT 2 (N)
PRINT 2 (IT)
PRINT 3 (R)
PRINT 1 (DEL(I,1),I=1,N)
PRINT 6 ((PHI(I,J),J=1,N),I=1,N)
PRINT 6 ((Q(I,J),J=1,N),I=1,N)
PRINT 1 (X(I,1),I=1,N)
TCOST=0.0
KN=0
DO 20 I=1,N
DO 20 J=1,N
20 PJ(I,J)=Q(I,J)

C CALCULATE AND PRINT A(K)T, PSI(K), AND P(K).

DO 21 I=1,M-1
CALL ATRAN(AT,P1 PHI DEL R,N)
DO 22 J=1,N
22 ALFAT(M-I,J)=AT(1,J)
CALL PPSI(PSI PHI DEL AT N)
CALL PP(P,PSI,P1 Q AT R,N)
KN=KN+1
IF (KN-IT)17,18,18
16 PRINT 4,I,(AT(I,J),J=1,N)
PRINT 5,I,((PSI(J,K)),K=1,N),J=1,N)
PRINT 6,I,((P(J,K)),K=1,N),J=1,N)
KN=0
17 CONTINUE
DO 21 J=1,N
DO 21 K=1,N
21 P1(J,K)=P(J,K)

C CALCULATE AND PRINT U(K), X(K), AND COST(K).

DO 23 I=1,N
23 AT(1,I)=ALFAT(1,I)
DO 28 J=2,N
DO 28 K=1,N
28 AT(J,K)=O.O
CALL PROD(U,AT,X,1,1,N)
Z=U(1,1)
CALL COST(DOL,X,Q,R,Z,N)
TCOST=TCOST+DOL
L=1
PRINT 7, L, Z, (X(I,1), I=1, N)
PRINT 9 (TCOST)

KN=1
DO 24 I=2, N
DO 25 J=1, N
25 Y(J,1)=X(J,1)
    CALL PROD(X, P HI, Y, N, 1, N)
DO 26 J=1, N
    DELTA(J,1)=Z*DEL(J,1)
    X(J,1)=X(J,1)+DELTA(J,1)
26 AT(1,J)=ALFAT(1,J)
DO 29 J=2, N
DO 29 K=1, N
29 AT(J,K)=0.0
    CALL PRCD(U, AT, X, 1, 1, N)
    Z=U(1,1)
    CALL COST (DOL, X, Q, R, Z, N)
    TCOST=TCOST+DOL
    KN=KN+1
    IF (KN-IT)24, 27, 27
27 PRINT 7, I, Z, (X(I,1), I=1, N)
PRINT 9 (TCOST)

KN=0
24 CONTINUE

1 FORMAT (8F10.6)
2 FORMAT (15)
3 FORMAT (1F10.6)
4 FORMAT (8X, 8F10.6, 8X, 15, 2HM=, 15, 2HAT, 8X, 8F10.6)
5 FORMAT (8X, 8F10.6, 8X, 15, 3HP=, 15, 2HX, 8F10.6)
6 FORMAT (8X, 8F10.6, 8X, 15, 9X, 6F10.6)
7 FORMAT (8X, 8F10.6, 8X, 15, 3X, 2HU=, 1F10.6, 8X, 10X, 2HX=, 5F10.6)
8 FORMAT (3F10.6)
9 FORMAT (30X, 6HTCOST=, 1F15.6)

END
C THIS SUBROUTINE CALCULATES A(K, T).

SUBROUTINE ATRAN (AT, P, PHI, DEL, R, N)
DIMENSION AT(8, 8), P(8, 8), PHI(8, 8), DEL(8, 8), DELT(8, 8), AB(8, 8),
1AC(8, 8), AD(8, 8)
CALL TRANCOL (DEL, DELT, N)
CALL PROD (AB, DELT, P, 1, N, N)
CALL PROD (AC, DEL, 1, 1, N)
CALL PROD (AD, AB, PHI, 1, N, N)
DO 14 I=1, N
14 AT(1, I) = -AD(1, I)/(AC(1, I) + R)
END

C THIS SUBROUTINE CALCULATES PSI(K).

SUBROUTINE PPsi (PSI, PHI, DEL, AT, N)
DIMENSION PSI(8, 8), PHI(8, 8), DEL(8, 8), AT(8, 8), AA(8, 8),
1PSIT(8, 8), AC(8, 8), AD(8, 8), AE(8, 8)
DO 15 I=1, N
15 AA(I, 1) = AT(I, I)
CALL TRANSQ (PSI, PSIT, N)
CALL PROD (AB, PSI, 1, N, N)
CALL PROD (AC, AB, PSI, N, N)
CALL PROD (AD, AA, AT, N, N, 1)
DO 16 I=1, N
DO 16 J=1, N
16 AD(I, J) = R * AD(I, J)

APPENDIX III  III-5  PROGRAM OPCONZ (CONTINUED)
CALL SUM (AE,AC,Q,N)
CALL SUM(P,AE,AD,N)
END

C THIS SUBROUTINE TRANSPOSES A COLUMN MATRIX HAVING A MAXIMUM OF
C EIGHT ELEMENTS.

SUBROUTINE TRANCOL (A,B,N)
DIMENSION A(8,8),B(8,8)
DO 12 I=1,N
  12 B(1,I)=A(I,1)
END

C THIS SUBROUTINE TRANSPOSES A SQUARE MATRIX OF MAXIMUM ORDER 8X8.

SUBROUTINE TRANSQ (A,B,N)
DIMENSION A(8,8),B(8,8)
DO 11 I=1,N
  DO 11 J=1,N
  11 B(J,I)=A(I,J)
END

C THIS SUBROUTINE CALCULATES THE COST AT EACH ITERATION AS DETERMINED
C BY THE COST FUNCTION.

SUBROUTINE COST (DOL,Y,Q,R,Z,N)
DIMENSION Y(8,8),Q(8,8),Z(8,8),YT(8,8),A(8,8),AA(8,8)
CALL TRANCOL(Y,YT,N)
CALL PROD(A,YT,Q,1,N,N)
CALL PROD(AA,A,Y,1,N,N)
DOL=AA(1,1)+R*Z**Z
END

C THIS SUBROUTINE MULTIPLIES ANY TWO MATRICES WHICH ARE LIMITED TO
C TWO 8X8S. THE ARGUMENTS ARE DEFINED AS FOLLOWS,
C (A) A THE PRODUCT
C (B) B THE MULTIPLICAND
C (C) C THE MULTIPLIER
C (D) L NUMBER OF ROWS OF THE MULTIPLICAND AND PRODUCT
C (E) M NUMBER OF COLUMNS OF THE MULTIPLIER AND PRODUCT
C (F) N NUMBER OF COLUMNS OF THE MULTIPLICAND AND THE NUMBER OF ROWS OF THE MULTIPLIER

SUBROUTINE PROD (A, B, C, L, M, N)
DIMENSION A(8,8), B(8,8), C(8,8)
DO 10 I=1, L
DO 10 J=1, M
A(I,J)=0.0
DO 10 K=1, N
10 A(I,J)=A(I,J)+B(I,K)*C(K,J)
END

C THIS SUBROUTINE FINDS THE SUM OF ANY TWO SQUARE MATRICES OF THE SAME
C DIMENSIONS UP TO AN 8x8.

SUBROUTINE SUM (A, B, C, N)
DIMENSION A(8,8), B(8,8), C(8,8)
DO 13 I=1, N
DO 13 J=1, N
13 A(I,J)=B(I,J)+C(I,J)
END

APPENDIX III
III-7
PROGRAM OPCONZ (CONCLUDED)
APPENDIX IV

PROGRAM OPCON3

*PROGRAM OPCON3*

CDIMENSION AT(8,8),Q(8,8),P(8,9),PL(8,9),DEL(8,8),PSI(8,8),
1PHI(8,8)

C THIS PROGRAM UTILIZES A COST FUNCTION: J(N)=MINIMUM(SUM X(N)*T*X(N)+
C SUM P*U(N-1)**2). AN UNLIMITED NUMBER OF ITERATIONS MAY BE MADE AT
C A COMPUTATION RATE OF 2000 PER MINUTE AFTER THE PROGRAM HAS BEEN
C COMPILED. THE OUTPUT OF THIS PROGRAM IS THE FEEDBACK GAIN MATRIX,
C A TRANSPOSE. THE FOLLOWING RECURSIVE EQUATIONS WERE DERIVED USING
C DYNAMIC PROGRAMMING,

C A(K)T=(DEL*P(K-1)+2)*PL/((DEL+2)*A(K)T)  \(1\)
C PSI(K)=PHI+DEL*A(K)T, PSI(0)=0  \(2\)
C P(K)=PSI(K)T*P(K-1)+PSI(K)+C+R*A(K)T *A(K)T, P(0)=C  \(3\)

C EQUATIONS 1, 2, AND 3 CONSTITUTE SUBROUTINES IN THIS PROGRAM.

C THE DATA CARDS REQUIRED ARE LISTED IN ORDER,

C (A) M (NUMBER OF STAGES, FORMAT 15)
C (B) N (ORDER OF SYSTEM, FORMAT 15)
C (C) IT (MULTIPLE OF RESULTS OF STAGES TO BE PRINTED, FORMAT 15)
C (D) R (CONSTANT IN THE COST FUNCTION WHICH WEIGHTS THE SUM OF THE
C CONTROL, FORMAT 1F10.6)

APPENDIX IV  IV-1  PROGRAM OPCON3 (CONTINUED)
C (E) DEL (THE SYSTEM IMPULSE RESPONSE MATRIX, FORMAT 8F10.6, READ IN BY COLUMN)
C (F) PHI (THE SYSTEM TRANSITION MATRIX, FORMAT NF10.6, READ IN BY ROWS)
C (G) Q (WEIGHTING MATRIX USED IN THE COST FUNCTION, FORMAT NF10.6, READ IN BY ROWS)
C FORMAT STATEMENT NR. 8 WILL HAVE TO BE ALTERED DEPENDING UPON THE ORDER OF THE SYSTEM. ITS FORMAT SHOULD BE NF10.6.

C READ DATA AND INITIALIZE VARIABLES.

READ 2(M)
READ 2(N)
READ 2(IT)
READ 3(R)
READ 1(DEL(I,1),I=1,N)
READ 8((PHI(I,J),J=1,N),I=1,N)
READ 8((Q(I,J),J=1,N),I=1,N)
PRINT 2(M)
PRINT 2(N)
PRINT 2(IT)
PRINT 3(R)
PRINT 1(DEL(I,1),I=1,N)
PRINT 8((PHI(I,J),J=1,N),I=1,N)
PRINT 8((Q(I,J),J=1,N),I=1,N)
KN=0
DO 20 I=1,N
DO 20 J=1,N
20 P(I,J)=Q(I,J)

C CALCULATE AND PRINT A(K)T, PSI(K), AND P(K).

DO 21 I=1,M-1

APPENDIX IV IV-2 PROGRAM OPCON3 (CONTINUED)
CALL ATRAN(AT,P1,PHI,DEL,R,N)
CALL PPSI(PSI,PHI,DEL,AT,N)
CALL R(PSI,PHI,P1,O,AT,R,N)
KN=KN+1
IF (KN-IT)17,18,18
18 PRINT 4,(AT(1,J),J=1,N)
PRINT 5,(PSI(J,K),K=1,N),J=1,N)
PRINT 6,(P(J,K),K=1,N),J=1,N)
KN=0
17 CONTINUE
DO 21 J=1,N
DO 21 K=1,N
21 P1(J,K)=P(J,K)
1 FORMAT (4F15.10)
2 FORMAT (15)
3 FORMAT (15F10.6)
4 FORMAT (12H2M=12./2HAT.9X.2F10.6,12)
5 FORMAT (12H2M=12./3HPSI.7X.8F10.6,12)
6 FORMAT (12H2M=12./1HP.9X.9F10.6,12)
8 FORMAT (2F10.6)
END

C THIS SUBROUTINE CALCULATES A(KT).
SUBROUTINE ATRAN (AT,P,PHI,DEL,R,N)
ODIMENSION AT(8,8),P(8,8),PHI(8,8),DEL(8,8),DELT(8,8),AB(8,8),AC(8,8),AD(8,8)
CALL TRANCOL (DEL,DELT,N)
CALL PROD(AB,DELT,P,1,N,N)
CALL PROD(AC,AB,DEL,1,1,N)
CALL PROD(AD,AB,PHI,1,1,N)
DO 14 I=1,N
14 AT(1,1)=-AD(1,1)/(AC(1,1)+R)
END
C THIS SUBROUTINE CALCULATES PSI(K).

SUBROUTINE PPSI(PSI,PHI,DEL,AT,N)
DIMENSION PSI(8,8),PHI(8,8),DEL(8,8),AT(8,8),AB(8,8)
CALL PROD (AB,DEL,AT,N,N,1)
CALL SUM (PSI,PHI,AB,N)
END

C THIS SUBROUTINE CALCULATES P(K).

SUBROUTINE PP(P,PSI,P1,Q,AT,N)
DIMENSION P(8,8),P1(8,8),PSI(8,8),Q(8,8),AT(8,8),AA(8,8),
1PSIT(8,8),AC(8,8),AD(8,8),AE(8,8)
DO 15 I=1,N
15 AA(I,1)=AT(I,1)
CALL TRANSQ (PSI,PSIT,N)
CALL PROD (AB,PSIT,P1,N,N,1)
CALL PROD (AC,AB,PSI,N,N,1)
CALL PROD (AD,CA,AT,N,N,1)
DO 16 I=1,N
DO 16 J=1,N
16 AD(I,J)=R*AD(I,J)
CALL SUM (AE,AC,Q,N)
CALL SUM (P,AE,AD,N)
END

C THIS SUBROUTINE TRANSPOSES A COLUMN MATRIX HAVING A MAXIMUM OF
C EIGHT ELEMENTS.

SUBROUTINE TRANCOL (A,3,N)
DIMENSION A(8,8),3(8,8)
DO 12 I=1,N
12 A(I,1)=A(I,1)
END

APPENDIX IV IV-6 PROGRAM OPCOMB (CONTINUED)
C THIS SUBROUTINE TRANSPOSES A SQUARE MATRIX OF MAXIMUM ORDER 8X8.

SUBROUTINE TRANSO (A,B,N)
DIMENSION A(8,8),B(8,8)
DO 11 I=1,N
DO 11 J=1,N
11 B(J,I)=A(I,J)
END

C THIS SUBROUTINE MULTIPLIES ANY TWO MATRICES WHICH ARE LIMITED TO
C TWO 8X8S. THE ARGUMENTS ARE DEFINED AS FOLLOWS:

C (A) A THE PRODUCT
C (B) B THE MULTIPLICAND
C (C) C THE MULTIPLIER
C (D) L NUMBER OF ROWS OF THE MULTIPLICAND AND PRODUCT
C (E) M NUMBER OF COLUMNS OF THE MULTIPLIER AND PRODUCT
C (F) N NUMBER OF COLUMNS OF THE MULTIPLICAND AND THE NUMBER OF ROWS
C OF THE MULTIPLIER

SUBROUTINE PROD (A,B,C,L,M,N)
DIMENSION A(8,8),B(8,8),C(8,8)
DO 10 I=1,L
DO 10 J=1,M
  A(I,J)=0.
DO 10 K=1,N
10  A(I,J)=A(I,J)+B(I,K)*C(K,J)
END

C THIS SUBROUTINE FINDS THE SUM OF ANY TWO SQUARE MATRICES OF THE SAME
C DIMENSIONS UP TO AN 8X8.

SUBROUTINE SUM (A,B,C,N)
DIMENSION A(8,8),B(8,8),C(8,8)
DO 13 I=1,N

APPENDIX IV  IV-5  PROGRAM OPCODE (CONTINUED)
Optimum digital control synthesis.