COMPUTER PROGRAMS FOR COMPUTATIONAL ASSISTANCE IN THE STUDY OF LINEAR CONTROL THEORY

JAMES L. MELSA
STEPHEN K. JONES

SECOND EDITION

COMPUTER PROGRAMS FOR COMPUTATIONAL ASSISTANCE IN THE STUDY OF LINEAR CONTROL THEORY

SECOND EDITION

JAMES L. MELSA
Professor
Information & Control Sciences Center
Southern Methodist University

STEPHEN K. JONES
Visiting Assistant Professor
Information & Control Sciences Center
Southern Methodist University

McGRAW-HILL BOOK COMPANY
New York St. Louis San Francisco Düsseldorf Johannesburg Kuala Lumpur London Mexico Montreal New Delhi Panama Rio de Janeiro Singapore Sydney Toronto

COMPUTER PROGRAMS FOR COMPUTATIONAL ASSISTANCE IN THE STUDY OF LINEAR CONTROL THEORY

567890WHWH7987

Library of Congress Cataloging in Publication Data

Melsa, James L
Computer programs for computational assistance in the study of linear control theory.

1. Control theory – Computer programs.
I. Jones, Stephen K., joint author. II. Title.
QA402.3.M386 1973 629.8'32'0285425 72-13701
ISBN 0-07-041498-X

PREFACE

This book is intended to provide assistance in solving computational problems associated with the study and application of linear control theory. A series of thirteen digital computer programs is presented which permit one to carry out the analysis, design and simulation of a broad class of linear control problems. Although the book was written primarily to serve as a supplement to a basic course in linear control, the practicing engineer will also find it to be of considerable value for realistic design and analysis problems.

Extensive use has been made of subprograms in the development of the computer codes so that the material could be easily adapted for use on problems not treated here. All of the subprograms are described in some detail in two appendices in order to further facilitate flexibility. It is hoped that this book and its associated computer codes will not be considered as the final answer to computational problems of linear control but rather the basic essentials from which further developments may be extracted. In this regard, the authors would appreciate receiving any reports on uses or modification of the material contained in this book.

In order to use the codes described in this book, it is necessary to possess only a rudimentary knowledge of FORTRAN coding and an ability to use a card punch and to submit a FORTRAN program. For those readers who are more familiar with FORTRAN, listings of the programs as well as other information have been provided to assist in the understanding and possible modification of the codes. The reader is also assumed to be familiar with the basic concepts of linear control theory as might be obtained from any one of a number of available basic control texts.

The computer codes described in this book have been developed over a number of years and have been used by the authors and others to solve a wide variety of both academic and practical problems. Some typical applications are discussed in Chapter 4. Nevertheless, one can never completely test any program especially on all available FORTRAN compilers. The authors apologize for any problems which may arise and would appreciate information on any special difficulties encountered.

In order to facilitate the use of the computer codes described in this book, a prepunched deck of cards, which includes all programs, subprograms and example input, may be obtained by writing to James L. Melsa, Information and Control Sciences Center, Southern Methodist University, Dallas, Texas 75222. The programs are all written in basic FORTRAN IV language and use the IBM029 punch format.

The authors express their appreciation to many students and colleagues who assisted in this project. The inspiring leadership of Dr. Andrew P. Sage, Director of the Information and Control Sciences Center, SMU, and Dr. Thomas L. Martin, Dean of the Institute of Technology, SMU, has been a source of constant encouragement.

CONTENTS

1 INTRODUCTION

1.1 PURPOSE AND OUTLINE

In the study of linear control theory, one often encounters computations which, although theoretically simple, can be an impediment to the learning of the basic concepts. For example after completing a control system design, it may be very helpful to the educational process to evaluate this design. The labor involved in the determination of accurate frequency response or root locus plots or determining the step response of the system may be prohibitive. In addition, it is often desirable to study high-order realistic design problems in order to emphasize the practical usefulness of the theory. Without a method of computational assistance, such problems are usually avoided because the computational effort required far exceeds the benefits obtained.

The goal of this book is to provide a set of computer codes which solve almost all of the computational problems of linear control theory. These codes allow one to study realistic problems in detail without extensive computation effort. In addition, the use of these codes give experience with the concepts of computer aided design and analysis which are so popular in current engineering practice.

It is assumed that the reader is familiar with the basic concepts of linear control theory as may be obtained from any one of a number of available texts. Extensive attention is given to the state variable description of linear control systems and the use of linear state variable feedback. The interested reader will find a detailed treatment of this material in the book: Linear Control Systems.[1] The notation and system formulation used in this book follows closely that used in Linear Control Systems.

In order to use the material in this book, it is not required that the reader possess any knowledge of numerical methods. Only the very basic essentials of the FORTRAN language and some simple knowledge of computer card deck preparation are needed. For the reader who is familiar with FORTRAN, listings of the programs and subprograms as well as notes on coding are presented for reference.

The programs are divided into two broad classes: (1) State variable programs and (2) Transfer function programs. The five state variable programs described in Chapter 2 allow one to obtain certain matrix functions such as the inverse, determinant, resolvent matrix and state transition matrix as well as to analyze and design linear state variable feedback systems. Two programs are presented which compute the time response of linear state variable feedback systems. There is a program for studying the sensitivity of closed-loop systems to parameter variations and six programs for the design of linear systems.

---

[1] J. L. Melsa and D. G. Schultz: Linear Control Systems, McGraw-Hill Book Co., Inc., 1969.

Three transfer function programs are discussed in Chapter 3. These programs may be used to determine the frequency response and root locus of a system. The third transfer function program is used to obtain the partial fraction expansion of a rational function. Some typical applications of the computer programs are presented in Chapter 4.

All of the subprograms used are described in Appendix A. These subprogram descriptions are included to assist the interested reader in understanding the operation of the programs and also to permit the reader to generate new programs from these subprograms. Extensive use was made of subprograms in the development of these computer codes in order to facilitate such flexibility. For example one might wish to combine the sensitivity analysis and time response programs or the state variable feedback and root locus programs in order to carry out some particular study.

Appendix B presents a set of three graphical display subprograms which are used in the two preceding chapters. These display programs are quite general and allow one to obtain a quick graphical representation of the results of the programs on a line printer. The subprograms discussed are logarithmic plots, X - Y plot and a X - versus time plot.

Each of the thirteen program write-ups presented in Chapters 2 and 3 follow the same format. After a brief statement of the purpose of the program, the theoretical concepts involved in the development of the computer codes are discussed. The input and output format of the program is explained and then illustrated by means of an example. The input data deck is shown and the resulting program output is presented and discussed. In some cases it has been necessary to slightly modify or condense the computer output in order to fit it within page-size limitations. The output should however remain indicative of the type of output which will be obtained. Finally, the listing of each program is presented for reference.

In general, the programs have been limited to tenth-order problems. Although there is no basic limitation in the algorithms which make this constraint necessary, it was felt that almost all academic and most practical problems satisfy this limitation. If necessary, this limitation may be removed by extending the appropriate dimension statements. However it is not recommended that problems of excessively high order be treated since no major effort has been directed to a study of the efficiency of the numerical methods used. For an excellent discussion of numerical methods, the interested reader[1] is directed to the book: Numerical Methods for Scientists and Engineers.

1.2 INPUT FORMAT

The input format for each program is described in detail in Chapters 2 and 3; the purpose of this section is to comment on certain general features of the input formats which applied to all of the programs. A primary consideration in the selection of the input format was to make the use of the programs as simple as possible while still retaining sufficient flexibility. A second consideration was to make the input formats of the various programs as nearly uniform as possible. Because of these considerations, the input formats selected often contain more cards than necessary to provide the program with input data. On the other hand the input formats are easy to remember since a large majority of the cards have identical forms and the same general format is used in every program. In addition the input data deck is designed so as to be closely related to the basic problem statement so that it is easy to make modifications in the problem if desired.

---

[1] R. W. Hamming: Numerical Methods for Scientists and Engineers, McGraw-Hill Book Co., Inc., 1962.

All of the programs can be used to solve more than one problem in a single run by simply placing the various input decks one after another. In other words, an input deck is prepared for each problem and these decks are then added together to form a composite input deck for the program.

In the description of the input data format, the FORTRAN format type information is included for the reader who is familiar with the FORTRAN language. The first card of every input deck contains problem identification information which may be used for later reference. Any desired alphanumeric characters may be placed in the first twenty columns of this first card. If appropriate, the next two columns (21-22) of the first data card will contain information concerning the problem order in fixed-point format. The reader is reminded that fixed-point numbers are always right justified in their fields.

The remaining input cards will vary depending on the particular program under consideration. However there are some general format rules which are followed for entering information in the form of matrices and in terms of polynomials. Let us consider each of these types of input separately.

1.2-1 *MATRIX DATA* Only two types of matrices are considered in the programs described in this book: square matrices and column vectors. The elements of a square matrix are always read one row at a time much as they appear in the original array. The numbers are placed in ten-column fields in floating-point format. If the dimension N of the matrix is greater than eight, the elements continue on the next card. The elements of the next row, however, start on a new card. Therefore if $N \leq 8$, N cards are used to represent the matrix while if $8 < N \leq 10$, 2N cards are necessary. Suppose for example that we wish to enter the 3 by 3 matrix

$$\underset{\sim}{A} = \begin{bmatrix} 3 & 0 & 1 \\ -1 & 2 & 0 \\ 4 & 1.2 & 3 \end{bmatrix}$$

then the input cards would take the form shown in Table 1.2-1. Note that the decimal point should be punched since a floating point format is being used.

Table 1.2-1

Input Card for a 3 by 3 matrix

Card No.	Column No. 1–10	11–20	21–30
1	3.0	0.0	1.0
2	-1.0	2.0	0.0
3	4.0	1.2	3.0

In the case of column vectors, the elements are read in column form in order to minimize the number of input cards. In other words the elements are read as if the vector were a row vector. If the dimension N of the vector is less than or equal to eight, only one card will be used to input the vector; if $8 < N \leq 10$, then two cards will be needed. In the input format descriptions in Chapters 2 and 3, it has been assumed that $N \leq 8$ in order to simplify the discussion. Whenever $N > 8$, it is simply necessary to double the number of cards used to describe any vector or square matrix.

1.2-2 *POLYNOMIAL DATA* Polynomials may be entered in two different formats. All programs which accept polynomial inputs allow either format to be used. The polynomial may be entered as polynomial coefficients or as polynomial factors; these two formats will be referred to as P mode or F mode respectively.

If the P mode is selected the coefficients of the polynomials are entered in ten-column, floating-point fields just as if they were elements of a matrix. The constant term is given first and the coefficient of the highest term is assumed to be unity. This coefficient is read in although it is set to unity and not used. Note that if the degree of the polynomial is equal to eight it is necessary to add a second card which may be blank if desired. In other words, if an eighth order polynomial is to be read, the first eight coefficients are entered in the eight ten-column fields on one card, the coefficient of s^8 must then be placed on the next card even though it must be unity.

If it is desired to enter the polynomial in F mode as polynomial factors, then one factor is placed on each card. The real part of the factor is entered in the first ten columns and the imaginary part is in the next ten columns. The real part is positive is the factor is in the left half plane. For complex conjugate roots only one card is used; the program will supply the complex conjugate. The imaginary part will always be positive.

In order to indicate which of the two polynomial modes is to be used, a control card is placed in front of each polynomial input. A P or F is entered in the first column of this card to indicate which mode is being used and the degree of the polynomial is entered in fixed-point form in the next two columns. Let us consider the third-order polynomial P(s) given by

$$P(s) = 5 + 9s + 5s^2 + s^3 = (s + 1)[(s + 2)^2 + 1]$$

The input deck for this polynomial in P mode is shown in Table 1.2-2a and in F mode in Table 1.2-2b. Note that the coefficient of s^3 is assumed to be unity and need not be entered.

Table 1.2-2

Polynomial Input Deck: (a) P mode, (b) F mode.

(a)

Card No.	Column No. 1–10	11–20	21–30
1	P 3		
2	5.0	9.0	5.0

(b)

Card No.	Column No. 1–10	11–20	21–30
1	F 3		
2	1.0		
3	2.0	1.0	

1.3 OUTPUT FORMAT

The output of all of the programs is almost self explanatory. Only a few comments are indicated here to clarify any particular factors which might cause confusion. At the beginning of every program output, the problem identification supplied by the user and the name of the program is printed for reference. The input data is also listed for reference and finally the actual output of the program is presented. If the output has several parts, the problem identification information is repeated for each part. In the presentation of the actual program output for the example problems discussed in Chapters 2 and 3, certain modifications and condensations have been necessitated by the page size limitation. However the basic content and nature of the output should be unchanged. It is suggested that the reader prepare the input for and run the example problems for himself.

In the program output, square matrices are always written as they would normally appear with rows horizontally and columns vertically. Column vectors are printed in transposed form across the page just as they are read in.

Polynomials are given in both factored and unfactored form independent of which input form was used. The coefficients are given in ascending order with the constant term first. The factors are listed in the original form with negative real part indicating that the root is in the left half plane. The reader should note that there is a sign change between the input format for a factor and the way that it is listed.

2 STATE VARIABLE PROGRAMS

2.1 INTRODUCTION

In this chapter we discuss a group of ten programs which may be used for the analysis and design of linear control systems represented in state variable forms as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t)$$

$$u(t) = K[r(t) - \mathbf{k}^T\mathbf{x}(t)]$$

$$y(t) = \mathbf{c}^T\mathbf{x}(t)$$

The BASIC MATRIX (BASMAT) program, discussed in Sec. 2.2, allows one to compute the determinant, inverse, characteristic polynomial and eigenvalues of a square matrix $\mathbf{A}$. In addition BASMAT may be used to compute the resolvent matrix $(s\mathbf{I} - \mathbf{A})^{-1}$ and state transition matrix $\exp(\mathbf{A}t)$.

Two programs are provided for determining the time response of the linear feedback control system above. The RATIONAL TIME RESPONSE (RTRESP) program, Sec. 2.3, requires that the input function r(t) have a rational time response and that there be no repeated eigenvalues in the combination of the system and input. The RTRESP program gives a closed-form expression for the time response. The GRAPHICAL TIME RESPONSE (GTRESP) program of Sec. 2.4, is used to determine and graphically display the time response for an arbitrary input. Note that both of these programs may be used to study open-loop systems by letting $K = 0$ and unforced systems by setting $r(t) = 0$.

One of the basic problems of linear system analysis and design is the question of sensitivity to parameter variations. The SENSITIVITY ANALYSIS (SENSIT) program may be used to provide some information on sensitivity. The SENSIT program, which is discussed in Sec. 2.5, permits one to study the effect on the closed-loop poles of changes in elements of $\mathbf{A}$, $\mathbf{b}$, $\mathbf{k}$ or K. A graphical output is provided to assist in the evaluation.

The next section of this chapter discusses the STATE VARIABLE FEEDBACK (STVARFDBK) program. This program has been widely used in both academic and industrial environments for the analysis and particularly the design of linear state variable feedback control systems. The STVARFDBK program may be used to find both the open-loop plant and closed-loop system transfer functions and can in addition design closed-loop system from desired closed-loop transfer function specifications.

Sections 2.7 to 2.9 present three programs which may be used to design state variable feedback systems with inaccessible state variables. The program OBSERV is used to find the observability index of a system with determines the order of the compensator needed for either the Luenberger observer program (LUEN) or the series compensation program (SERCOM). The last two sections present two programs which may be used for designing state variable feedback structures for multiple-input, multi-output systems.

2.2 BASIC MATRIX (BASMAT)

Given a matrix $\underset{\sim}{A}$, the BASIC MATRIX program can compute the determinant of $\underset{\sim}{A}$, det $\underset{\sim}{A}$, the inverse of $\underset{\sim}{A}$, $\underset{\sim}{A}^{-1}$, the characteristic polynomial det($s\underset{\sim}{I} - \underset{\sim}{A}$) and eigenvalues of $\underset{\sim}{A}$, λ_i as well as the state transition matrix $\underset{\sim}{\Phi}(t) = \exp(\underset{\sim}{A}t)$ and the resolvent matrix $\underset{\sim}{\Phi}(s) = (s\underset{\sim}{I} - \underset{\sim}{A})^{-1}$.

2.2-1 *THEORY* The BASMAT program reads the elements of the matrix $\underset{\sim}{A}$ row by row and an option card which indicates what functions of $\underset{\sim}{A}$ are desired. The main program of BASMAT then calls subprograms to perform the appropriate calculations, these subprograms are discussed in Appendix A.

2.2-2 *INPUT FORMAT* The input data for the BASIC MATRIX programs consists of the usual identification card which also contains the dimension N of the matrix $\underset{\sim}{A}$ in columns 21-22. The elements of $\underset{\sim}{A}$ are then read row by row from the next N cards[1]. The last data card determines which computations are to be performed by BASMAT. If a column is zero or blank, the related operation is performed; if the column contains any non-zero integer (1 to 9), then the related operation is suppressed. If the last card is completely blank, all operations are performed. The input format is summarized in Table 2.2-1 for easy reference.

Table 2.2-1

Input Format for BASIC MATRIX

Card Number	Column Number	Description	Format
1	1-20 21-22	Problem identification N = dimension of $\underset{\sim}{A} \leq 10$	5A4, I2
2	1-10 11-20 etc.	a_{11} a_{12} ...	8E10.5
3	1-10 11-20 etc.	a_{21} a_{22} ...	8E10.5
N + 2	1 2 3 4 5 6	IDET $\neq$ 0 suppress determinant INV $\neq$ 0 suppress inverse NRM $\neq$ 0 suppress resolvent ICP $\neq$ 0 suppress characteristic polynomial IEIG $\neq$ 0 suppress eigenvalues ISTM $\neq$ 0 suppress state transition matrix	6I1

---

[1] N is assumed to be less than or equal to 8 for simplicity. See Sec. 1.2.

2.2-3 *OUTPUT FORMAT* After the problem identification, the matrix $\mathbf{A}$ is printed for reference. Then depending on the options selected, det($\mathbf{A}$), $\mathbf{A}^{-1}$, $\mathbf{\Phi}(s)$, the characteristic polynomial, eigenvalues, and $\mathbf{\Phi}(t)$ are printed. The resolvent matrix is written as $\mathbf{\Phi}(s) = \text{adj}(s\mathbf{I}-\mathbf{A})/\det(s\mathbf{I}-\mathbf{A})$ and the matrix numerator polynomial adj($s\mathbf{I}-\mathbf{A}$) is expressed as matrix coefficients of powers of s so that it takes the form

$$\text{adj}(s\mathbf{I}-\mathbf{A}) = \mathbf{F}_1 + \mathbf{F}_2 s + \ldots + \mathbf{F}_N s^{N-1}$$

The state transition matrix is expressed in a form similar to the adj($s\mathbf{I}-\mathbf{A}$) as matrix coefficients times the natural modes exp ($\lambda_i t$); complex eigenvalues are written as damped sine and cosine terms.

2.2-4 *EXAMPLE* In order to illustrate the use of the BASMAT program, let us consider the following matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -3 \end{bmatrix}$$

We will use all of the options available for this example so the last card is left blank. The form of the input data cards for this example is shown in Table 2.2-2.

Table 2.2-2
Input Data Deck for BASMAT Example

Card No.	Column No. 1–10	11–20	21–30
1	BASMAT EXAMPLE	3	3
2	0.0	1.0	0.0
3	0.0	0.0	1.0
4	-2.0	-3.0	-3.0
5	(BLANK CARD)		

The program output for this example problem is shown in Table 2.2-3. From this output, we see that $\phi_{11}(s)$ is

$$\phi_{11}(s) = \frac{3 + 3s + s^2}{2 + 3s + 3s^2 + s^3}$$

and the corresponding element of the state transition matrix is

$$\phi_{11}(t) = 0.667e^{-0.5t} \cos 0.866t + 1.15e^{-0.5t} \sin 0.866t + 0.333e^{-2t}$$

Table 2.2-3

Program Output for BASMAT Example

```
BASIC MATRIX PROGRAM
PROBLEM IDENTIFICATION:       BASMAT EXAMPLE 3

*****************************************

THE A MATRIX

 0.0                    1.0000000E 00        0.0
 0.0                    0.0                  1.0000000E 00
-2.0000000E 00         -3.0000000E 00       -3.0000000E 00

THE DETERMINANT OF THE MATRIX

-2.0000000E 00

THE INVERSE OF THE MATRIX

-1.5000000E 00         -1.5000000E 00       -5.0000000E-01
 1.0000000E 00          0.0                  0.0
 0.0                    1.0000000E 00        0.0

*****************************************

THE MATRIX COEFFICIENTS OF THE NUMERATOR OF THE RESOLVENT MATRIX

THE MATRIX COEFFICIENT OF S**2

 1.0000000E 00          0.0                  0.0
 0.0                    1.0000000E 00        0.0
 0.0                    0.0                  1.0000000E 00

THE MATRIX COEFFICIENT OF S**1

 3.0000000E 00          1.0000000E 00        0.0
 0.0                    3.0000000E 00        1.0000000E 00
-2.0000000E 00         -3.0000000E 00        0.0

THE MATRIX COEFFICIENT OF S**0

 3.0000000E 00          3.0000000E 00        1.0000000E 00
-2.0000000E 00          0.0                  0.0
 0.0                   -2.0000000E 00        0.0

*****************************************

THE CHARACTERISTIC POLYNOMIAL - IN ASCENDING POWERS OF S

 1.9999990E 00          3.0000000E 00        3.0000000E 00        1.0

*****************************************

THE EIGENVALUES OF THE A MATRIX
   REAL PART            IMAGINARY PART

-5.0000048E-01         -8.6602509E-01
-5.0000048E-01          8.6602509E-01
-1.9999990E 00          0.0
```

Table 2.2-3 (cont.)

```
*********************************************

 THE ELEMENTS OF THE STATE TRANSITION MATRIX

 THE MATRIX COEFFICIENT OF EXP(-5.000005E-01)T*COS( 8.660251E-01)

  6.6666591E-01       -3.3333445E-01       -3.3333409E-01
  6.6666842E-01        1.6666679E 00        6.6666794E-01
 -1.3333349E 00       -1.3333359E 00       -3.3333492E-01

 THE MATRIX COEFFICIENT OF EXP(-5.000005E-01)T*SIN( 8.660251E-01)

  1.1547022E 00        1.7320518E 00        5.7735133E-01
 -1.1547022E 00       -5.7735050E-01       -3.5762787E-07
  0.0                 -1.1547031E 00       -5.7735038E-01

 THE MATRIX COEFFICIENT OF EXP(-1.999999E 00)T

  3.3333433E-01        3.3333457E-01        3.3333439E-01
 -6.6666853E-01       -6.6666758E-01       -6.6666794E-01
  1.3333340E 00        1.3333340E 00        1.3333340E 00
```

2.2-5 *SUBPROGRAMS USED* The following subprograms are used by this program:

(1) DET
(2) CHREQ
(3) PROOT
(4) SIMEQ
(5) STMST

Please see Appendix A for a description of each of these subprograms.

2.2-6 *PROGRAM LISTING*

```
C   BASIC MATRIX PROGRAM
C   SUBPROGRAMS USED:  CHREQ, SIMEQ, STMST, PROOT, DET.
      DIMENSION A(10,10),EIGR(10),EIGI(10),C(11),AINV(10,10),
     *    NAME(5)
 2001 FORMAT (5A4,I2)
 2002 FORMAT (8E10.5)
 2003 FORMAT (1P6E20.7)
 2004 FORMAT (1H0,5X,16HTHE A MATRIX     ,/)
 2005 FORMAT (1H0,5X,32HTHE CHARACTERISTIC POLYNOMIAL -
     *   24HIN ASCENDING POWERS OF S    /)
 2006 FORMAT (1H0   ,5X,31HTHE EIGENVALUES OF THE A MATRIX)
 2007 FORMAT (9X,9HREAL PART,8X,14HIMAGINARY PART,/)
 2008 FORMAT (1H1,5X,20HBASIC MATRIX PROGRAM)
 2009 FORMAT (6X,23HPROBLEM IDENTIFICATION:,5X,5A4)
 2010 FORMAT (1H0,5X,29HTHE DETERMINANT OF THE MATRIX/)
 2011 FORMAT (1H0,5X,25HTHE INVERSE OF THE MATRIX/)
 2012 FORMAT (1H0,45(1H*))
 2013 FORMAT (6I1)
    4 READ (5,2001,END=10) (NAME(I),I=1,5),N
      DO 1 I=1,N
    1 READ 2002, (A(I,K),K=1,N)
      READ 2013, IDET,INV,NRM,ICP,IEIG,ISTM
      PRINT 2008
```

```
      PRINT 2009, (NAME(I),I=1,5)
      PRINT 2012
      PRINT 2004
      DO 2 I=1,N
    2 PRINT 2003, (A(I,K),K=1,N)
      IF (IDET.NE.0) GO TO 5
      D=DET(A,N)
      PRINT 2010
      PRINT 2003, D
    5 IF (INV.NE.0) GO TO 15
      PRINT 2011
      CALL SIMEQ(A,C,N,AINV,C,IERR)
      IF (IERR.EQ.0) GO TO 15
      DO 20 I=1,N
   20 PRINT 2003, (AINV(I,J),J=1,N)
   15 CALL CHREQ(A,N,C,NRM)
      CALL PROOT(N,C,EIGR,EIGI,+1)
      IF (ICP.NE.0) GO TO 30
      PRINT 2012
      PRINT 2005
      NN=N+1
      PRINT 2003, (C(I),I=1,NN)
   30 IF (IEIG.NE.0) GO TO 35
      PRINT 2012
      PRINT 2006
      PRINT 2007
      DO 3 I=1,N
    3 PRINT 2003, EIGR(I),EIGI(I)
   35 IF (ISTM.NE.0) GO TO 25
      CALL STMST(N,A,EIGR,EIGI,ISTM)
   25 GO TO 4
   10 STOP
      END
```

2.3 RATIONAL TIME RESPONSE (RTRESP)

The RTRESP program determines the time response in closed form of the closed-loop system

$$\dot{x}(t) = Ax(t) + bu(t)$$

$$u(t) = K[r(t) - k^T x(t)]$$

$$y(t) = c^T x(t)$$

due to any initial conditions $x(0)$ and input $r(t)$, $t \geq 0$ which has a rational Laplace transform $R(s)$ with a pole-zero excess of at least one.

2.3-1 *THEORY* Let the Laplace transform of $r(t)$ be given by

$$\mathcal{L}\{r(t)\} = R(s) = K_r \frac{N(s)}{D(s)}$$

where

$$N(s) = n_1 + n_2 s + \ldots + n_k s^{k-1} + s^k$$

$$D(s) = d_1 + d_2 s + \ldots + d_m s^{m-1} + s^m$$

and $m > k \geq 0$. The approach which we shall use will be to form an m^{th}-order dynamic system whose initial condition response (for a specific set of initial conditions) is equal to $r(t)$. Then we may combine this new system with the original system and find the complete response in closed form by the use of STMST (See Appendix A). Let us use phase variables to represent this new system in the form

$$\dot{x}_r(t) = A_r x_r(t)$$

$$y_r(t) = c_r^T x_r(t)$$

where

$$A_r = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \\ -d_1 & -d_2 & -d_3 & \ldots & -d_m \end{bmatrix}$$

and

$$c_r = \text{col}(K_r n_1, K_r n_2, \ldots, K_r, 0, \ldots, 0)$$

If we let the initial condition of the system be $x_r(0) = \text{col}(0, 0, \ldots, 0, 1)$, then the time response of the output $y_r(t)$ will be identical to $r(t)$.

Our original closed-loop system can be represented as

$$\dot{x}(t) = A_k x(t) + Kbr(t)$$

where $A_k = A - Kbk^T$. Since $y_r(t)$ is equal to $r(t)$ if the initial condition $x_r(0)$ is properly chosen, the closed-loop system can also be written as

$$\dot{x}(t) = A_k x(t) + Kbc_r^T x_r(t)$$

Now we form the $n + m^{th}$ order augmented system which is unforced