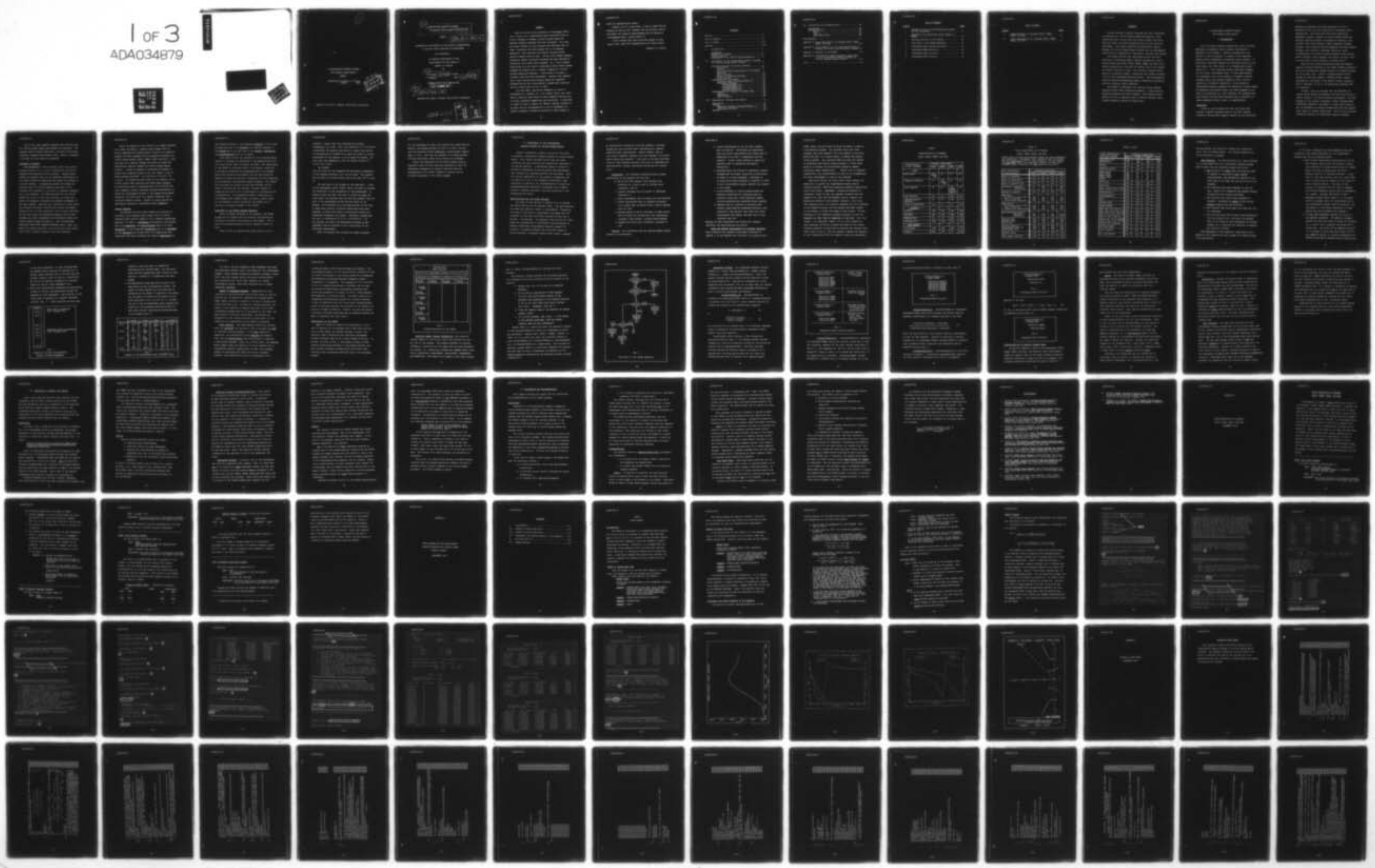
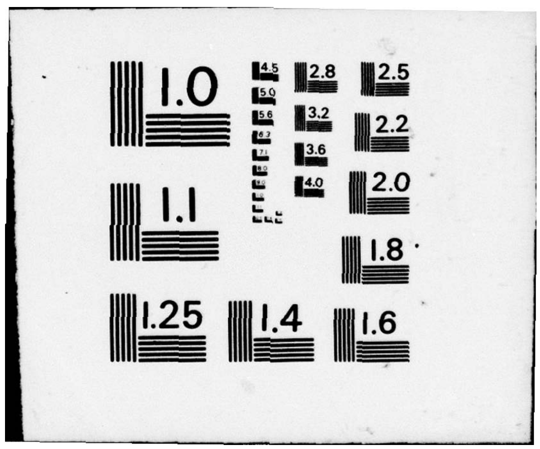
AD-A034 879 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 9/2
A CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN (CCPC--ETC(U)
DEC 76 F L O'BRIEN
UNCLASSIFIED GE/EE/76D-38 NL

1 OF 3
ADA034879




1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

A CONSOLIDATED COMPUTER PROGRAM

FOR CONTROL SYSTEM DESIGN

THESIS

GE/EE/76D-38 Frederic L. O'Brien
Captain USAF

A CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN (CCPCSD).

THESIS

Master's thesis,

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Frederic L. O'Brien B.S.E.E.

Captain USAF

Graduate Electrical Engineering

219p.

December 1976

GE/EE/76D-38

Approved for public release; distribution unlimited.

012225

LB

Preface

While at the Air Force Institute of Technology (AFIT), other students and I experienced sheer frustration when we tried to analyze or design a control system using the separate computer programs that were available. The input and output formats for each program were different and, at times confusing, and each program had to be attached separately, then executed, and finally returned before another program could be used. The Air Force Flight Dynamics Laboratory (AFFDL) personnel expressed the same feelings of frustration when using these programs. So, I have plunged head-strong into the task of combining as many control design programs as possible into a "usable" program for control systems design and analysis. Given another two years, I probably could have been successful. However, the comments that I have received from persons using the "separate" programs and the new "consolidated" program have indicated that my efforts were not all in vain.

I am sure that I had become somewhat of a pest to Professors C.W. Richard, Jr., T.E. Reeves, and G. Orr, and I wish to publicly thank them for their patience with me, and for their competent suggestions and directions. I wish also to thank Dr. Gary B. Lamont for taking a special interest in my task, and for encouraging me to try to accomplish it. Similar thoughts of thanks go directly to Frank George of

AFFDL for sponsoring my thesis.

Finally, but in no way least, I wish to thank God for helping me, and my wife, Barbara, and my children, Noelle and Kathy, for being so understanding and for giving me encouragement when I really needed it.

I will confess that some errors may appear in this report, and I take full responsibility for these errors.

Frederic L. O'Brien

Contents

List of Figures

List of Tables

ABSTRACT

Several different computer programs have been independently developed for control system design and analysis. Each program may have its own input and output formats (different from any other), and each program may have its own operating philosophy. This report describes a Consolidated Computer Program for Control System Design (CCPCSD) and its development: specifications, organization, realization, testing, and results.

The CCPCSD contains four previously written Air Force Institute of Technology (AFIT) computer programs for control system design algorithms (ROOTL, PARTL, FREQR, and POLY) and a lead-in program which controls the overall flow of the CCPCSD. An overlay structure is used. The structure consists principally of an executive (main) overlay and four primary overlays (one for each subprogram mentioned above). Each subprogram is modified towards standardizing inputs, termination procedures, and data structure (where feasible).

The CCPCSD is developed to be used by control systems engineers and/or AFIT engineering students. A user-oriented approach permeates the entire program. This computer-aided design tool can be operated from an intercom terminal, where operator-computer interaction takes place.

A CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN

I. Introduction

One of the most important present-day tools for either designing or analyzing control systems (continuous or digital) is the computer. Computer programs have been written to aid the engineer or student in the sometimes awesome task of designing or analyzing control systems. It seems that individual programs are willingly written as the need arises, but with little, or no thought about future needs (i.e., ...for combining several individual programs for one large, consolidated task...or more specifically, for combining a root locus program, a frequency response program, and a time response program so that a control system can be analyzed completely.). Well structured consolidated computer programs are definitely in great demand by engineers and students alike, but these programs do not seem to be available (or, quite possibly, the programs have been written, but are made available only to personnel in small unexposed offices, firms, or organizations).

Background

Prior to, and including the time this thesis was written, computer programs stored in the AFIT computer library of the CDC 6600 computer system, and the Electrical

Engineering Department were made available for use of personnel at the Air Force Institute of Technology (AFIT), building 640. The library and Electrical Engineering Department possess a variety of control system design and analysis computer programs (continuous and digital), such as ROOTL (Root Locus Program), FREQR (Frequency Response Program), PARTL (Partial Fraction Expansion and Time Response Program), POLY (Roots of Polynomial Function), DIGIS (Z-Transforms), FIR (Finite Impulse Response Filter Design), OPTCON (Optimal Control Design), and others, no doubt, that are still part of some professor's "private stock."

There are also other specialized design and analysis computer programs available at the Flight Dynamics Laboratory and at the Aeronautical Systems Division of Wright-Patterson Air Force Base, Ohio. DIGICON ("Digital Flight Control Systems for Tactical Fighter," AFFDL-TR-74-69; computer algorithm for the design of the digital control systems of the F-4 Aircraft) is one of the specialized programs that are available.

Further, there are programs that are available for control system design and analysis that have been written to fulfill master's, or doctor's degree requirements. One such program is "The Design of Automatic Control Systems Using Interactive Graphics," by Kenneth R. Young, which allows a user an advantage of working interactively with the computer (real time) for control system work. The Air Force has also contracted industry for specialized computer programs.

All in all, many computer programs that could be used for control system design and analysis are available. The engineer and/or student would like to be able to use these programs---computer-aided design tools. Hence, a statement of purpose for this thesis is presented.

Statement of Purpose

The professional control engineer and/or the control engineer student are, of course, saddled with the task of designing and analyzing control systems. Computer-aided design and analysis tools are an important contribution to the timely success of completing the design and analysis of a control system. To adequately perform the design and analysis task, the control engineer must have at his disposal (as a minimum) a root locus, a frequency response, and a time response. The characteristics of the control system are shown by these three methods, and these then enable the designer to analyze the complete performance of his basic system. There are also extensions to these design and analysys techniques (Nichols Chart design and Nyquist design are two of these extensions) that are also available, but the root locus, the frequency response, and the time response are still considered as the principal methods for control system design and analysis. And, as previously mentioned, three AFIT computer programs (ROOTL, FREQR, and PARTL) are already available, and they provide these basics (root locus, frequency response, and time response).

Hence, the purpose of this effort is to clearly develop, by at least one means, a computer-aided design tool for control engineers and students which contains as an absolute minimum four AFIT programs that are presently available for control system design (ROOTL, FREQR, PARTL, and POLY). This tool must be a computer program that shall be called the Consolidated Computer Program for Control System Design (CCPCSD). It must be capable of providing an engineer or student with repeated access to one, two, three, or all of these programs mentioned - all during one sitting at the terminal. The capability must exist for the operator to be in full control when selecting and using these programs. An architecture must be developed which allows adding some other programs, without requiring extensive modification to either the consolidated program or the program being added. Therefore, the CCPCSD must be a useful, versatile, self-contained computer program - capable of being operated by experienced engineers or beginning student engineers.

General Approach

The approach that was taken towards the objective of providing a good computer-aided design tool for control engineers might best be thought of as a "general engineering approach," in that: 1) information pertaining to the problem has to be researched, and specifications have to be fully determined; 2) the researched information has to be evaluated so that decisions can be made about the capabilities and trade-offs that would influence the overall organization of the problem solution; 3) the organized solution, in this case an algorithm, has to be realized in a computer program; 4) the program has to be test; and 5) the results, conclusions, and recommendations have to be stated.

This report covers this approach in the following manner. Chapter II includes the definition of the specifications and the actual method taken to achieve the task. The organization of the CCPCSD is discussed fully in the chapter, along with a discussion concerning the standardization of the program inputs and termination procedures, and the data structure. Chapter III shows the process of realizing the total algorithm as an efficient, error-proofed computer program. The chapter also discusses the testing criteria and methods, and the results obtained from the testing. Finally, Chapter IV summarizes the effort with a presentation of conclusions and recommendations. The appendixes contain a description of the programs ROOTL, FREQR, PARTL, and POLY; a user's guide for the CCPCSD; a source listing for the CCPCSD; and a procedure for adding programs to the CCPCSD.

Highlights of the CCPCSD Program Operation

From a terminal (teletype or CRT display), the CCPCSD program can be attached from the library, and executed by following the printed instructions as they appear. Now to look at the program operation from an operator's point of view.

First of all, an operator must state that he is at a terminal. Typing "TTY" and returning the carriage accomplishes this task. Next, and explanation of the program is provided; the operator may select or reject this option. The explanation is intended to be used for either an initial declaration of the program, or as a recall for review. For each case, the explanation tells the general sequence of events that occurs.

The program

that the roots for the numerator and denominator polynomials of the transfer function, G(s), can be found. The operator may not need to use this option, and he can type "NO" to move on.

The next part of the program is very important. A list of the subprograms (ROOTL, FREQR, PARTL, and POLY) is printed out along with a brief explanation of each subprogram. To conserve paper and time, the list is printed out only one time; hence, the operator must jot down the symbolic name for each of the subprograms for use then and later in the program. (This decision for a one-time printout of the list is based upon previous irritating experiences with other programs, in which long lists of options were printed out repeatedly throughout the program. Hopefully, copying down the list of subprograms is not too inconvenient.) Now the operator can select the subprogram he wishes to use. (Appendix A may be referred to for a description of the individual subprograms.)

After the operator has provided the inputs required for the subprogram he chose, and obtained the output that he desired, the program allows the option to either continue with the presently chosen subprogram, choose another subprogram, or stop. Each time the operator is finished working with any subprogram, these above options are presented.

Hence, the highlights of the CCPCSD program (from an operator's point of view) have been presented to show the sequential arrangement of specific functions that are accomplished by the CCPCSD. Chapter II follows with an in-depth discussion of the CCPCSD program.

II. Development of the Consolidated Computer Program for Control System Design

Chapter I presented a general overview of the effort to develop a consolidated computer program for control system design. This chapter expounds further upon the areas concerning the specifications and organization discussed in the general approach. A substantial amount of effort is devoted to discussing the specifications and requirements that afforded the finalized approach that was taken. Next, the overall organization of the program is explained, and the make-up of each of the overlays is described. Some discussion is devoted to the standardization of inputs and program/subprogram termination procedures, and, finally, pertinent comments relating to the data structures of the subprograms are stated.

Specifications and the Actual Approach

The theme of this section of Chapter II is to discuss the specifications and the approach taken. The specifications are based upon specific needs, or requirements. Discussing (in terms of the needs or requirements) the means that are available to meet each need allow choosing the best way to proceed - the approach. The approach, when carried out fully, meets the intended goal of "providing a unified package of computer algorithms (consolidated computer program) for in-house, or acamemic analysis and conceptual design of control systems...contained totally within itself...capable of enabling user interaction with the computer...one-time program use with one-time input (satisfying all computer algorithm input requirements)...useful outputs, such as root locus, frequency response, and time response, for control system design and analysis." This goal is the sole directing force for this important section, and prior to discussing the specifications, certain constraints and factors are exposed below.

Constraints. The following constraints place certain restrictions on the approach and the goal:

a. use of CDC 6600 computer only; maximum core available for intercom jobs is $50{,}000_8$ words. (Reference 2 and 3)
b. computer language must be Fortran IV (Extended) (Reference 1)
c. input requirements must be simple and user-oriented.
d. output requirements must be complete and useful (in a format easy to analyze from a control systems point of view)
e. maximum amount of plots, printouts, or data stored must be within the $50{,}000_8$ word core requirement. (program + data storage = $50{,}000_8$ words maximum)
f. program must allow user interaction (terminal or crt)

Factors. The constraints that are imposed demand certain factors to be addressed.

a. A good understanding of the CDC 6600 computer must be acquired. Terminology and appropriate input and output requirements and formats must be understood to be able to communicate with the computer. Correct codes relating to syntax and semantics must be understood and applied efficiently.

b. Knowledge about the Fortran IV (Extended) computer language must be sought, along with a proficiency in using it to write programs. A knowledge of overlays and overlay structure is required in order to put a consolidated program together and conserve on core usage.

c. A good knowledge of control system design and analysis terminology and techniques must be acquired, along with a valid approach towards properly analyzing and designing control systems.

d. Research must be conducted to determine what computer algorithms are presently available, what function(s) they perform, what input and output requirements they demand, and what type of data structure they use.

Knowing now what constraints and factors were imposed initially, the specifications can unfold.

Input and Outputs Requirements for Programs Available. After reviewing the programs (algorithms) mentioned in Chapter I, it was decided that initially the programs ROOTL, FREQR, PARTL, and POLY would be chosen as basics to make up a consolidated program. The rationale being that these programs, when used together, are very capable of providing sufficient data, in a useful format, to design and analyze control systems. And, although FREQR and ROOTL are capable of performing some digital system analysis, a "first step" would be to develop the consolidated program to handle continuous domain systems only. Tables I and II indicate, respectively, the characteristics of each of the programs, and the input requirements of each of the programs.

Also worth noting, the "first step" would consider the capability to operate the consolidated program from an intercom terminal. BATCH and interactive graphics capabilities could follow later. **Rationale** - interactive graphics is considered "ideal" in that the engineer interacts with the computer in a real-time environment, and he "creates" his design and analysis at the graphics terminal. Root locus, time response plots, and frequency response plots are displayed on the graphics console, and the engineer may make changes to his system, or accept the design he has created. The key advantage to the interactive graphics approach is that the engineer can "see" what is happening in real-time. But, even though it is considered "ideal," it is a very complex and time consuming job which is beyond the scope of this thesis. Intercom operation is preferred by engineers and students over BATCH jobs, in that it, like the graphics, enables the operator to act interactively with the computer (input the parameters,

Table I

Characteristics of Programs

ROOTL, FREQR, PARTL, and POLY

Characteristics of each Program	Programs (Algorithms)			
	ROOTL	FREQR	PARTL	POLY
...Job...				
Root Locus	yes (order 50)	no	no	no
Frequency Response	no	yes (order 20)	no	no
Time Response	no	no	yes (order Num:12 Den:24)	no
Time Function	no	no	yes	no
Partial Fraction Expansion	no	no	yes	no
Roots of polynomial	yes	no	no	yes
...Outputs...				
Listing	yes	yes	yes	yes
TTY Plot	no	yes	yes	N/A
Printer Plot	no	no	yes	N/A
Calcomp Plot	yes	yes	yes	N/A
...Type System...				
Continuous	yes	yes	yes	N/A
Digital	part.	yes	no	N/A

Table II

*Input Requirements for Programs

ROOTL, FREQR, PARTL, and POLY

*NOTE: Table II shows the different "ways" the input parameters are "asked for." The symbol "N/A" indicates that the program does not explicitly "ask" for that parameter, but the program may implicitly require the parameter. Hence, at times three different ways of entering the same parameter may occur.

Input Requirements For Each Program	Programs (Algorithms)			
	ROOTL	FREQR	PARTL	POLY
Order of Numerator	N/A	yes	yes	N/A
Order of Denominator	N/A	yes	yes	N/A
Power of Repeated Root	N/A	no	yes	N/A
Polynomial: Numerator	no	yes	no	yes
Denominator	no	yes	no	yes
Numerator Constant	yes	yes	yes	N/A
Numerator:				
No. Factors Given Order	N/A	yes	N/A	N/A
Order of Factors	N/A	yes	N/A	N/A
Coefficients Numerator	N/A	yes	N/A	N/A
Denominator:				
No. Factors Given Order	N/A	yes	N/A	N/A
Order of Factors	N/A	yes	N/A	N/A
Coefficients Denominator	N/A	yes	N/A	N/A
Real/Imaginary Factors:				
Numerator	N/A	N/A	yes	roots
Denominator	N/A	N/A	yes	roots
Initial Time	N/A	N/A	yes	N/A
Duration Time	N/A	N/A	yes	N/A
Delta Time	N/A	N/A	yes	N/A
Initial Omega (w)	N/A	yes	N/A	N/A
Incremental Omega (w)	N/A	yes	N/A	N/A
Final Omega (w)	N/A	yes	N/A	N/A
Restart Array	N/A	yes	N/A	N/A
Time Delay, exp(-Ts)	yes	yes	no	N/A

Table II cont.

Input Requirements For Each Program	Programs (Algorithms)			
	ROOTL	FREQR	PARTL	POLY
Magnitude Plot Log-dB Plot	N/A N/A	yes yes	N/A N/A	N/A N/A
Titles	yes	yes	yes	N/A
Sampling Period	no	yes	no	N/A
Right Bound Left Bound Upper Bound Lower Bound Bound	yes yes yes yes yes	no no no no no	no no no no no	N/A N/A N/A N/A N/A
Plot Size	yes	yes	yes	N/A
Step Size	yes	N/A	N/A	N/A
Dist. Between Plot. Pts.	yes	no	no	N/A
Sensitivity, K	yes	N/A	N/A	N/A
Gain Bandwidth	yes	N/A	N/A	N/A
Zeta	yes	N/A	N/A	N/A
Initial Step	yes	N/A	N/A	N/A
Figure Number	yes	no	no	N/A
Real Part Init. Test Pt.	yes	N/A	N/A	N/A
Imag. Part Init. Test Pt	yes	N/A	N/A	N/A
Type of Init. Test Pt.	yes	N/A	N/A	N/A
Number Pts. to be Called	yes	N/A	N/A	N/A
Delete Tnsfr Function	yes	no	no	N/A
Multiple Graphs	yes	no	no	N/A
Plots Disposed	yes	yes	yes	N/A
Plot Destination	yes	yes	yes	N/A

see and analyze the printed out results, and continue or stop), but to a lesser degree. The intercom operation is not as difficult to program.

Data Structure. The specifications for a data structure to be incorporated into the consolidated computer program must be such that the following criteria can be met:

a. One structure for input data; the structure must be capable of being used by all programs simultaneously, thus allowing an operator to put input data in one time, but use any or all of the programs.

b. The structure must allow changing of only the data wished to be changed by the operator without altering the data entered previously.

c. The structure must allow future addition of programs - extending the amount of data stored without changing the data structure.

d. The structure must allow data to be read out, but not destroyed.

e. The structure must allow certain output parameters from one program (i.e., the roots from program POLY) to be stored as input parameters for another program (i.e., stored as the numerator or denominator factors of ROOTL).

Data management is very important, especially as the amount of core required for each program to be added becomes a critical factor.

Of the basic techniques for data management that were researched, four seemed applicable for the consolidated program, and they are discussed below.

a. Mass storage input/output subroutines for the CDC 6600 computer system allow the creation, access, and modification of multi-record files on a random basis, and does not depend upon the file's physical position or internal structure. A good point is that each record in the file, when either read or written into at random, does not have its other contents affected. This type of file storage is consistent with the idea of changing only certain parameters rather than all parameters of a system. Although this method of data management seemed almost ideal, it is not necessarily an optimum choice for input/output use in the consolidated program (could be tailored), and furthermore, it was nearly impossible to locate persons who would recommend this approach.

b. A second data management approach is the use of the NAMELIST statement. The NAMELIST statement permits the input and output of groups of variables (i.e., poles, zeros, constant) and arrays with a unique name. Key points of the NAMELIST data are: (1) a variable or array name may belong to one or more NAMELIST groups, (2) specific parameters in a NAMELIST group may be changed without altering any of the other parameters, (3) when entering data, the NAMELIST group name must be preceded by a $ (dollar sign), and after all of the data has been entered, another $ must be used as a termination. The use of the NAMELIST statement is a familiar method for use as a data management tool.

c. Where only single-valued parameters are manipulated, strings of vectors (one-dimensional arrays) are adequate for data management. Each position in the vector must be assigned as a specific parameter location (Fig. 1). These positions can have data

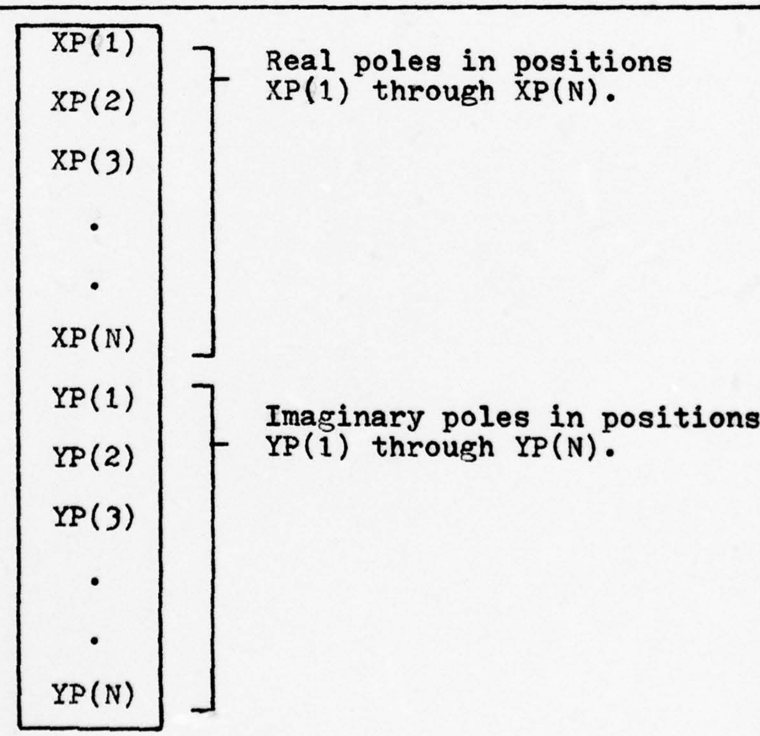


Fig. 1

Example of Vectors (One-Dimension Arrays) for Data Management (specific) read into them, or removed for calculations and returned again. The key point being that the program must keep "record" of each position - putting in, or taking out only when needed.

d. Two-dimensional arrays may also be used in the same manner as the one-dimensional arrays. The number of words reserved for the array will be determined by the number of elements in the array, and the number is figured by the product of the array subscripts (i.e., A(3,3)= 3 x 3 = 9 words). The use of the two-dimensional array requires pre-planning of the parameter positions as well as exact data control (entering and retrieving data) by the program (Fig. 2).

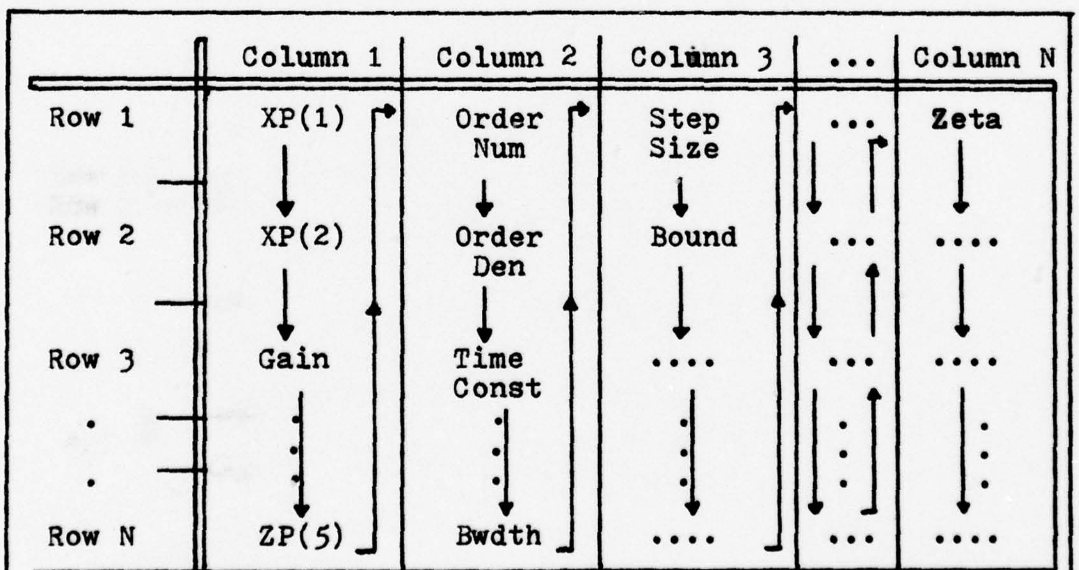


Fig. 2
Example of Two-Dimensional Array (Order of Entry)

In view of the four methods of data management that have been described, methods b and c are chosen for the consolidated program - ROOTL already utilizes the NAMELIST statement; FREQR, PARTL, and POLY already utilize the vector storage method. And, considering the "first Step" approach to a usable consolidated program, these data structures are to remain relatively unchanged.

<u>Method of Combining Programs</u>. Remembering the chief constraint of having available only $50{,}000_8$ words of core for intercom use, the method for combining the programs ROOTL, FREQR, PARTL, and POLY is limited to the use of overlays and overlay structure only. By using the overlay structure, the amount of core required is reduced (only part of the total consolidated program would be in the memory at one time), and field length is used more efficiently. The details and explanation of the overlay structure is given later in this chapter as the organization of the CCPCSD is discussed.

<u>Final Approach</u>. The final approach takes into consideration the <u>knowledge</u> (of the writer) that is at hand, the <u>goal</u> to "develop a unified package...," the <u>time</u> element for completion, the availability of the <u>computer</u> and computer time, and the <u>specifications</u> that are pertinent to describe a consolidated program. Hence, as a "first step," the programs ROOTL, FREQR, PARTL, and POLY are to be put together into a consolidated computer program which is then, as a minimum, capable of performing each task (root locus, frequency response, time response and roots of a polynimial) for a

continuous domain control system design and analysis. The consolidated program is to be user-oriented, allowing the user to control the destiny of the control system he is designing and analyzing by choosing whichever program he wishes (repeatedly, if necessary). The programs are to be modified to fit in a usable architecture of overlays, and the data structures and input/output requirements presently used within each program are to remain in effect, unless glaring discrepancies, or conflicts arise. The major intent is to standardize operations and instructions wherever possible. It must be understood that the "first step" finished product shall be definitely useful, but certainly not as optimal of a solution as might be desired. However, the time and efforts of another interested individual may be channeled towards the goal of optimizing the "first step."

Organization of the CCPCSD

Again, to strive to reduce the actual amount of core that is required for the program, and to make more efficient use of field length, an overlay structure is used. The executive (main) overlay and the individual subprograms (primary overlays) make up the overall program structure. In general, as shown by Fig. 3, the primary overlay of ROOTL contains subroutines pertinent to the primary overlay, and also contains secondary overlays which have within their structure other subroutines pertinent only to the secondary overlay.

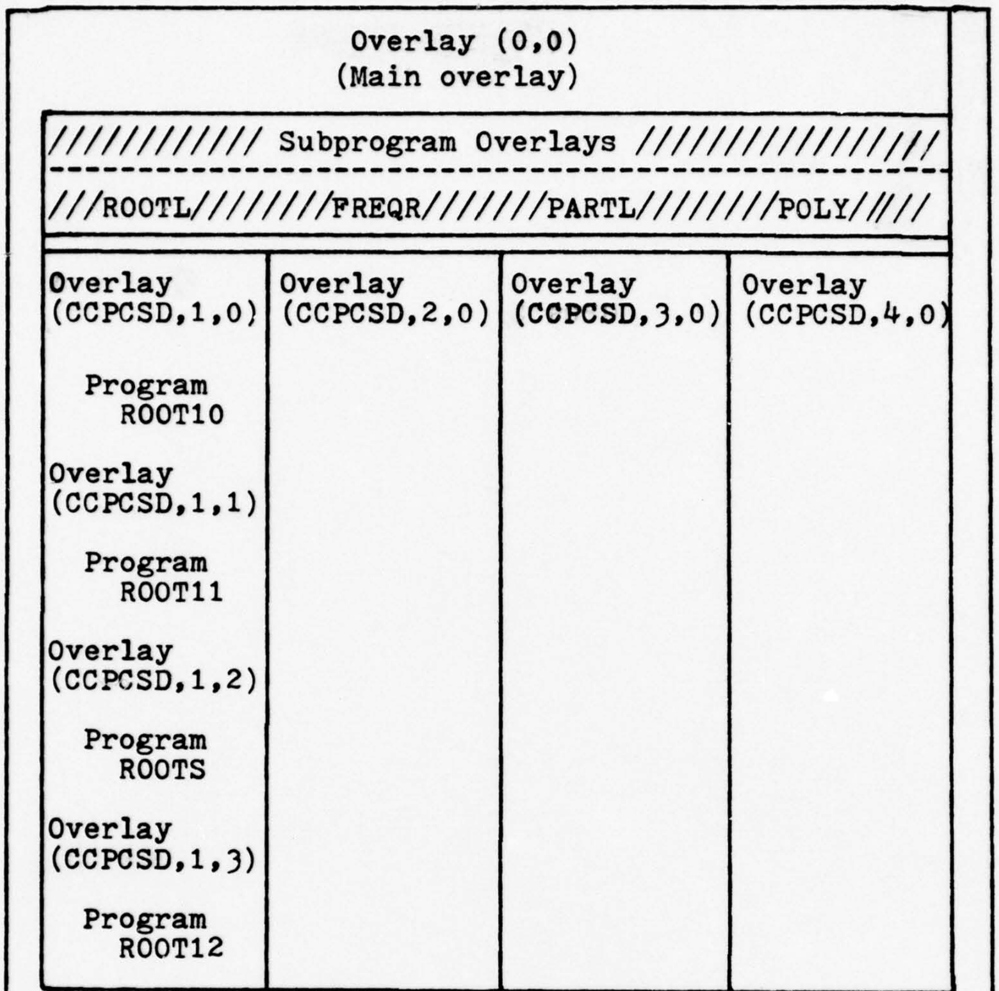


Fig. 3

Overlay Structure of the CCPCSD

Executive (Main) Overlay (CCPCSD,0,0). The executive overlay remains in core at all times, and controls the overall flow of the program. The program statement for Overlay (CCPCSD,0,0) specifies the file names that are used throughout the CCPCSD; i.e., INPUT=100B, OUTPUT=100B, ANSWER, PPLOT, PLOT, CFILE, TAPE5=INPUT, TAPE6=ANSWER, TAPE7=OUTPUT, TAPE8=PPLOT, and TAPE9=100B. File names do not appear in any other overlay.

And, of course, Overlay(CCPCSD,0,0) precedes all other overlays.

The executive overlay provides the following questions, with subsequent direction consistent with the response of the operator:

a. "PLEASE TYPE 'TTY' IF YOU ARE AT AN INTERCOM TERMINAL."

b. "DO YOU WANT AN EXPLANATION OF THE PROGRAM?"

c. "YOU WILL NEED ROOTS AND/OR FACTORS OF THE NUMERATOR AND DENOMINATOR OF THE TRANSFER FUNCTION. IF YOU DO NOT HAVE THESE ROOTS AND/OR FACTORS, TYPE 'POLY'. OTHERWISE, TYPE 'NO'."

d. "TYPE THE SYMBOLIC NAME OF THE RESPONSE YOU DESIRE. EXAMPLE: PARTL"

e. "IF YOU ARE FINISHED, TYPE 'STOP'. IF YOU DESIRE TO USE ANOTHER SUBPROGRAM, TYPE THE CORRECT SYMBOLIC NAME FOR THAT SUBPROGRAM."

Typing 'POLY' in c above will alert the executive overlay to call Overlay(CCPCSD,4,0), which is a subprogram to factor any polynomial. Typing either 'ROOTL', 'FREQR', 'PARTL', or 'POLY' in d or e above will alert the executive overlay to call Overlay(CCPCSD,1,0), Overlay(CCPCSD,2,0), Overlay(CCPCSD, 3,0), or Overlay(CCPCSD,4,0), respectively. And, as might be expected, typing 'STOP' in e above will terminate the program. The flow chart in Fig. 4 illustrates the operation of the CCPCSD program as controlled by the executive overlay (Overlay (CCPCSD,0,0)).

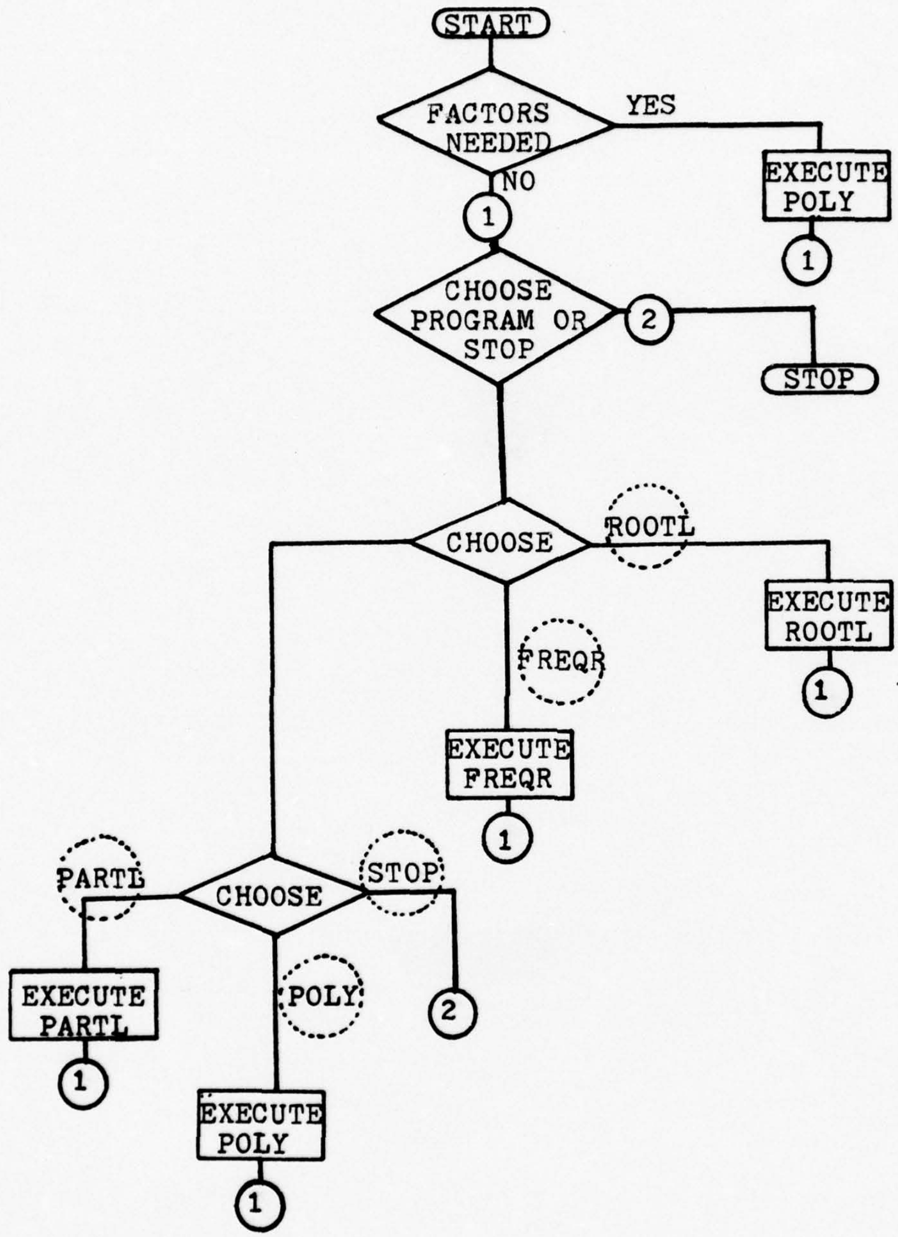


Fig. 4

Flow Chart of the CCPCSD Operation

Subprogram Overlays. The subprogram overlays (Overlay CCPCSD,1,0) - ROOTL; Overlay(CCPCSD,2,0) - FREQR; Overlay (CCPCSD,3,0) - PARTL; and Overlay(CCPCSD,4,0) - POLY) are primary overlays and are called up by the executive overlay (Overlay(CCPCSD,0,0)). They may be considered as totally separable from one another, but, at the same time, not totally independent when considering their individual functions as used in control system design and analysis.

Overlay(CCPCSD,1,0). Overlay(CCPCSD,1,0) represents the subprogram ROOTL, which is a program developed to determine the root locus of a given transfer function by finding the solution to its characteristic equation

$$1 + KG(s)H(s) = 0 \qquad (1)$$

or

$$\frac{K(s-z_1)\ldots(s-z_{m-1})}{(s-p_1)\ldots(s-p_n)} = -1 \qquad (2)$$

as a function of K (K varying from 0 to $\pm$ infinity). Appendix A may be referred to for more specific information about ROOTL's theory and operation.

Structurally, ROOTL consists of the overlays and subroutines shown in Fig. 5. The primary overlay (Overlay (CCPCSD,1,0)) acts as an executive overlay by controlling the overall flow of subprogram ROOTL. The three secondary overlays provide the specific tasks of setting up for a calcomp plot, printing and plotting the root locus, and determining the roots of interest for a specified gain.

Overlay(CCPCSD,1,0) Program ROOT10 Subroutine BANG Subroutine BOX Subroutine GANG1 Subroutine BLEND Subroutine SMULR Subroutine ANG1	Primary Overlay (Directs flow of ROOTL)
Overlay(CCPCSD,1,1) Program ROOT11 Subroutine CNTER Subroutine DNTER Subroutine GROUP	Secondary Overlay (Sets up Calcomp Plot)
Overlay(CCPCSD,1,2) Program ROOTS	Secondary Overlay (Determines Roots of Interest for Specified Gain)
Overlay(CCPCSD,1,3) Program ROOT12 Subroutine BREAK Subroutine SEEK	Secondary Overlay (Trace Root Locus, Print, Plot per Option LL Code)

Fig. 5

Subprogram ROOTL Overlay Structure

<u>Overlay(CCPCSD,2,0)</u>. Overlay(CCPCSD,2,0) represents the subprogram FREQR, which will provide a polar plot or Bode plot frequency response for the given transfer functions G(.) and H(.). (The arguments G(.) and H(.) can be either s=jw or z=exp(jwT), where w, omega, is in radians per second and T is the sample period in seconds.) Subprogram FREQR, Overlay (CCPCSD,2,0), consists only of a primary overlay, but it has

six specialized subroutines to complete its task (Fig. 6).

```
Overlay(CCPCSD,2,0)
  Program FREQR

    Subroutine BODPLOT
    Subroutine POLPLOT
    Subroutine POLYVAL
    Subroutine POLARD
    Subroutine POLYRD
    Subroutine POLYMUL
```

Fig. 6

Subprogram FREQR Structure

Overlay(CCPCSD,3,0). Overlay(CCPCSD,3,0) represents subprogram PARTL, which will calculate the partial fraction expansion coefficients of a transfer function of the form

$$\frac{K(s+a_1+jb_1)(s+a_2+jb_2)\ldots(s+a_m+jb_m)}{(s+g)^L(s+c_1+jd_1)(s+c_2+jd_2)\ldots(s+c_n+jd_n)}, \qquad (3)$$

print the time function of the transfer function, and tabulate or plot the time response. Similar in structure to Overlay (CCPCSD,2,0) for subprogram FREQR, Overlay(CCPCSD,3,0) contains only a primary overlay, along with one subroutine and one FUNCTION routine (Fig. 7).

Overlay(CCPCSD,4,0). Overlay(CCPCSD,4,0) is the last overlay to consider. It represents subprogram POLY, which as its only function, calculates the roots of a polynomial

Overlay(CCPCSD,3,0)
Program PARTL

Subroutine POLAR

FUNCTION FT

Fig. 7

Subprogram PARTL Structure

equation of the form

$$P_n(x) = A_1x^n + A_2x^{n-1} + \ldots + A_nx + A_{n+1} = 0 \ . \quad (4)$$

It, too, is structured with only a primary overlay, subroutine, and FUNCTION routine (Fig. 8).

Overlay(CCPCSD,4,0)
Program POLY

Subroutine DMULR

FUNCTION RESID

Fig. 8

Subprogram POLY Structure

Standardization of Specific Program Areas

As noted in Chapter I, each of the subprograms (ROOTL, FREQR, PARTL, and POLY) had different input requirements (Table II, pages 13 and 14) and output formats, along with different procedures to begin and stop the programs. The following subsections describes the specific areas of each program that have been standardized.

Inputs. The inputs for FREQR, PARTL, and POLY are basically the same as the original programmers had written them. Each subprogram consists of printed statements that "ask" the operator to submit the specified input in a required format.

Only the order and coefficients of a desired polynomial are needed as inputs for POLY. PARTL and FREQR basically require the factors of the numerator and denominator of a transfer function, plus specific parameters which control the type, quantity, duration, and limits of the tabulations, printouts, and/or plots. These specific parameters are again "asked for" by the subprograms.

ROOTL, on the other hand, required substantial alteration to allow standardization with the other subprograms. First of all, the front end of Overlay(CCPCSD,1,0) was changed to include a list of options (if the operator desired). The choice of inputting all of the parameters in either one NAMELIST statement, or in a step-by-step manner (the subprogram "asking" for the parameters by associative groupings) was added to assist the operator. Finally, the signs were changed for the parameters XP(.) (real poles), YP(.) (imaginary poles), XZ(.) (real zeros), and YZ(.) (imaginary zeros) to allow ROOTL to accept factors instead of roots.

Therefore, each subprogram has the capabilities of providing a list of options, of "asking" the operator to input specific parameters, and of accepting factors of the numerator and denominator of the transfer function instead of roots.

Termination. Although the actual method of terminating each of the subprograms seemed small and near insignificant, standardizing the method has been accomplished. For each of the subprograms (ROOTL, FREQR, PARTL, and POLY), the method for termination is to type a '0' (zero) and return the carriage. A note of caution: typing a zero at any point in the program will not necessarily cause termination. The program must present the list of options which include termination before the operator types the zero. However, to terminate the total CCPCSD program (not just the subprogram), the operator may type a '%A'. All prior information and calculated results are lost once the %a is entered and the program stops.

Data Structure. The ideal data structure for this consolidated computer program must be one in which data, such as the numerator coefficients (zeros), denominator coefficients (poles), gain constant, and other similar parameters, are entered one time - all of the subprograms using this one-time input - and that data changes could be made to only the parameters desired to be changed without effecting any of the remaining parameters. However, for reasons stated in the first section of this chapter, the data structure for each of the subprograms was not altered towards achieving a standardized structure. Subprogram ROOTL makes use of the NAMELIST statement for parameters that are entered as inputs, called out for processing, put back as either the same parameter or as an altered parameter, and used for tabulations and/or plots as required. COMMON statements are used as vehicles for data transfer from overlay to subroutine and overlay to overlay. Subprogram FREQR, PARTL, and POLY made use of COMMON statements only. If a change in a parameter is desired, the subprogram must be re-started and all of the data must be entered; this is unlike ROOTL's NAMELIST data capability of changing only selected parameters (knowing that all others will remain unchanged and at their initialized value).

Standardizing the data structure for the CCPCSD program remains to be a complex task yet to be fully accomplished. Chapter IV discusses some recommendations which affect the data structure selection.

III. Realization, Testing, and Results

Once a total algorithm and flow chart analysis had been formulated, and the total overlay structure for the CCPCSD had been determined, the job of realization of the algorithm as an efficient, usable program became a reality. The following subsections contain a discussion concerning the realization of the total algorithm, the testing criteria and methods, and the results obtained from the testing.

Realization

In order, first, to make sure that continuity throughout all of the subprograms would be accomplished, the program statement for the executive overlay (Overlay(CCPCSD,0,0)) was modified to include all inputs, outputs, and tape files. It reads:

```
PROGRAM CCPCSD(INPUT=100B,OUTPUT=100B,ANSWER,PPLOT,
CFILE,PLOT,TAPE5=INPUT,TAPE6=ANSWER,TAPE7=OUTPUT,
TAPE8=PPLOT,TAPE9=100B)
```

Next, to incorporate the modified file system, the input statements for the subprograms ROOTL and POLY were updated where necessary. Subprograms PARTL and FREQR did not require any modification to their file system. Only the executive overlay (Overlay(CCPCSD,0,0)) contained the program statement with the specified file names required for all of the overlay levels in the CCPCSD program. Hence, each of the separate subprograms were properly corrected for consolidation.

Overlay statements were written; "return" statements were substituted for "stop" and "end" statements (as needed); and all COMMON and data statements for each of the subprograms were checked for duplication which could cause erroneous results (i.e., storing and manipulating different parameters from the same locations - unintentionally).

The executive overlay (Overlay(CCPCSD,0,0)), subprogram ROOTL, and subprogram POLY were "put together" first. The errors were corrected, and finally these three parts functioned properly together. (Testing is discussed in the following subsection.) Subprograms FREQR and PARTL were added to the consolidated program sequentially, and, after each was added, the same error-correcting test procedure was applied until the total CCPCSD program functioned without errors. At this stage, the CCPCSD program was ready for testing.

Testing

Testing centered basically around four areas:

a. inducing errors intentionally
b. using each option for every subprogram
c. checking for correctiveness of results by comparing with a set of controlled test problems
d. making sure that the termination option functioned properly.

If plots or printed outputs could be "sent" to a central location, (i.e., AFIT computer room, building 640), they were indeed "sent," and were verified as reaching that location by physically picking them up and checking them for correctness.

Executive Overlay (Overlay(CCPCSD,0,0)). Each option in the executive overlay was tested for proper functioning, and each performed properly without fault. Intentional errors, such as typing a symbolic name incorrectly, returning the carriage without any information being previously entered, and omitting part of an instruction, were induced and the response was noted. Error statements were printed out each time an error was entered intentionally, along with a comment for the operator to "try again by typing...". Because the executive overlay must maintain the overall flow of the CCPCSD program, it was extremely important that no error introduced, by mistake or intentionally, would cause the program to jump to another location, or halt without proper cause. The error checks tested operated satisfactory. The executive overlay was not tested for "correctness of results," as it does not perform any calculations, but the terminating operation was performed to insure that the total CCPCSD stopped and that a CP time was given. Hence, the executive overlay (Overlay (CCPCSD,0,0)) was subjected to each of the applicable test areas.

Subprogram Overlays. Initially, each of the subprograms were test for termination upon command. Next, to insure that all of the options for every subprogram worked, and that each worked correctly, the prior-tested examples contained in the original user's guide (ROOTL, FREQR, PARTL,and POLY) were used as the controlled test problems. Every option was tested, and the results of the CCPCSD program were compared with the results of the sample problems. Finally, intentional errors were introduced randomly throughout each subprogram. Some of the errors used were: exceeding the order of a system beyond its capability; typing in too much data at one time (i.e., too many coefficients, constants, or other parameters); typing an option number that did not exist; and creating a situation in which the computer had "trouble calculating the results within the limits/bounds" (i.e., very small numerator and extremely large denominator of a transfer function with a division process taking place).

Results

The results of testing the CCPCSD program were indeed favorable and encouraging. Each of the subprograms, plus the executive overlay, did terminate upon command. At any point in the program, typing the "%A" did cause complete program termination - as predicted.

All of the options for every subprogram operated as planned; each of the problem results obtained from the CCPCSD program did compare exactly with the results of each of the sample problems; and, finally, the error checking capability of each subprogram proved adequate, but not optimal. Common errors (wrong options, too many constants) were "caught" by a subprogram, and the operator was given instructions to "try again;" however, errors forcing equations to become mathematically unsolvable caused the total CCPCSD program to be terminated.

Although the greater portion of the CCPCSD program worked well, the subprogram ROOTL does contain one annoying characteristic. The operator is given the choice to enter the data and parameters either as <u>one</u> NAMELIST statement, or as a <u>step-by-step</u> grouping of parameters, with each grouping being "printed out" for the operator to do with as he please. Accomplishing the one-time NAMELIST data is relatively simple if one knows what he wishes to input prior to typing the statement. Accomplishing the step-by-step procedure can be annoying, but its procedure is still simple. As an example, the following statement may be printed out for the operator:

> "ENTER NUMBER OF LOCUS POINTS COMPUTED,'JK=';
> DAMPING RATIO,'ZETA='; AND STEP-SIZE ON RADIAL
> SEARCH FOR GIVEN ZETA,'RAD='."

If the operator does <u>not</u> wish to change any of the parameters mentioned, he must start in column two, and type '$INPUD $'. Hence, regardless of whether or not data is entered, the operator must start in column two and type '$INPUD', then skip a space and type either another '$' for no data change, or type the data and a '$' at the end of the data. The process is at times annoying, but certainly not unbearable.

All in all, the results from testing the CCPCSD program indicated that the program operated as planned, and that it provided correct results (compared to the testing sample problems). The CCPCSD program is usable.

IV. Conclusions and Recommendations

This chapter provides the reader with the conclusions and recommendations for the CCPCSD program.

Conclusions

Completion of the Consolidated Computer Program for Control System Design (CCPCSD) is definitely a positive step towards providing the control engineer and/or engineering student with the optimum computer-aided design tool. Although the CCPCSD program is not the "optimal program," it is definitely a working program, and it can be used in its present state for field use in control system design and analysis work.

The overlay structure insures that the core used at any time is at an absolute minimum. The overlay structure also demonstrates current state of the art practices for developing large program/subprogram combinations, and it possesses the capability to add other programs/subprograms as desired with little modification. In short, the overlay structure is effective.

In the view of being a useful product, the CCPCSD does meet the following criteria:

a. It works effectively, and it has understandable instructions.

b. It provides correct results (calculations, plots, tabulations).

c. It contains error diagnosis/statements.

d. It is usable on computers with Fortran IV (Extended) language (CDC 6600, in particular).

The CCPCSD, however, does not meet the criteria for a single data structure that (1) enables the operator to enter the input data one time - this input satisfying all of the subprograms input requirements; and (2) enables information to be passed from one subprogram to another.

The CCPCSD offers considerable advantage (time and convenience) to the engineer or student by providing each design tool (root locus, frequency response, and time response) at his fingertips (one call up of the computer yields all of the programs). And, based upon the results of this thesis (relating to the criteria and tests), the CCPCSD is a useful program for control system design and analysis. It must now be "tried and proven" by serious engineers and/or students - improving its performance wherever possible.

Recommendations

Two distinct areas for immediate/near-future development are recommended:

1. To continue with the present CCPCSD, improving it and expanding its capabilities.
2. To convert the present CCPCSD into an interactive graphics package.

Case 1. If case one is selected, the data structure should be attacked with a goal to make one data structure which is used by all of the programs in the CCPCSD. Provisions should be made to accept added programs without any affect on the data structure - a challenging job. Also, the CCPCSD should be expanded to accomplish both continuous and digital control system design and analysis. The main overlay should contain the necessary logic, with user-instructions for the added capability. The CCPCSD should also be capable of being run as BATCH jobs.

More programs, as listed in Chapter I, should be added to the CCPCSD, as long as duplications are removed and non-productive programs are eliminated. Prospective avenues for improving and expanding the present CCPCSD are almost limitless.

Case 2. If case 2 is selected, the task of converting the CCPCSD to operate from a graphics terminal would involve restructuring both the data structure and some of the CCPCSD architecture. However, there is a graphics package already developed for program ROOTL (called GROOTL; available in the AFIT library of the CDC 6600 computer system). This graphics package could be a good starting point for a CCPCSD graphics package. Regardless, a graphics package with even the present CCPCSD capabilities is definitely an option urgently needed by engineers and/or engineering students.

Long Range Goals. Long range goals could call for the CCPCSD to be capable, first of all, of operating BATCH, at an intercom terminal, or from a graphics terminal/console. As mentioned before, the amount of inputs required should be at an absolute minimum, and they should be entered only one time to accomplish each task for all of the programs.

Further, the CCPCSD could be expanded to do several tasks both within and without the realm of control system design and analysis. The CCPCSD could be expanded to do:

a. synthesis of control systems

b. block diagram manipulations (build-up and reduction)

c. optimal estimation and control design (Kalman filter design)

d. statistical analysis

e. continuous-to-digital and digital-to-continuous transformations

f. specific filter designs (Butterworth, Chebyshev, Finite Impulse Response)

g. compensation networks - design and analysis

There are already algorithms that have been developed to accomplish some of the tasks mentioned above. But, actually, before one accepts the challenge of developing the CCPCSD to accomplish all of the tasks listed, the "ground floor" must be done first---attack the data structure; insure that extra programs can be added without altering the data structure, the main CCPCSD, and, if possible, the program being added; and make the CCPCSD concept totally user-oriented and simple to understand. Now the executive overlay (Overlay(CCPCSD,0,0)) can be shaped with the necessary logic to accomplish each newly added task: step-by-step. (Note: All efforts to expand the capabilities of the CCPCSD should, of course, be coordinated with knowledgeable control systems personnel in both the field and the academic environment.)

An evaluation of the algorithms (programs) already available (by job and effectiveness---excellent, adequate, poor) might seem worthy, but past experience has shown that it is nearly impossible to keep track of these algorithms due to changes in location, task, updating, and growth. The confidence factor becomes short termed. Therefore, to help eliminate any frustrating experiences for some other soul who might "trust" that which is in print as being totally correct and up-to-date, the evaluation of the algorithms is not included.

> ". . . up-to-date information must be
> updated frequently to insure its
> validity. . ." (unsigned)

Bibliography

1. Control Data Corporation. Fortran Extended Version 4 Reference Manual (Revision H, Publication Number 60305601). Sunnyvale, California: Control Data Corporation, July 1975.

2. Control Data Corporation. SCOPE Reference Manual Version 3.4.4 (Revision J, Publication Number 60307200). Sunnyvale, California: Control Data Corporation, August 1975.

3. Control Data Corporation. Intercom Reference Manual Models 72, 73, 74 Version 4, 6000 Version 4 (Revision E, Publication Number 60307100). Sunnyvale, California: Control Data Corporation, August 1974.

4. Jensen, R.W. and L.P. McNamee, "On Benchmarking CAD Programs." The UCLA Computer Science Department Quarterly, Research and Education, Vol. 4, No. 2: 61-68 (April 1976).

5. Thompson, G.W. and K.R. Young, Utilization of a CRT Display Light Pen in the Design of Feedback Control Systems. NASA-CR-11574. Washington: National Aeronautical and Space Administration, May 1972.

6. Young, K.R. The Design of Automatic Control Systems Using Interactive Computer Graphics. unpublished PhD Dissertation. The University of Texas at Austin, May 1974.

7. AFFDL-TR-74-69. Digital Flight Control Systems for Tactical Fighters. Honeywell, Inc. Wright-Patterson Air Force Base, Ohio: Air Force Flight Dynamics Laboratory, July 1974.

8. AFIT/EN. ROOTL User's Manual. Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, March 1975.

9. AFIT/EN. PARTL. Heaviside Partial Fraction Expansion and Time Response Program. (Revised) Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, December 1974.

10. AFIT/EN. GROOTL User's Manual. AFIT. Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, March 1975.

11. AFIT/ENC. POLY. (Revised User's Manual). AFIT. Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, October 1974.

12. AFIT/EN. FREQR. Frequency Response Program. AFIT. Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, November 1975.

13. D'Azzo, J. J., and C. H. Houpis. Linear Control System Analysis and Design: Conventional and Modern. McGraw-Hill Book Co., New York, 1975.

APPENDIX A

BRIEF DESCRIPTION OF PROGRAMS
ROOTL, FREQR, PARTL, AND POLY
(DECEMBER 1976)

Brief Description of Programs
ROOTL, FREQR, PARTL, and POLY

This appendix is indeed a brief description of each of the programs - ROOTL, FREQR, PARTL, and POLY. Initially, the approach to this appendix was to obtain every bit of information about each of the programs, and then to write a brief, but thorough, description of each program using the information that had been gathered. However, it is determined that the original user's guides for each of the programs is an "authority" for not only its description, but also for its operation. Hence, with an intent not to draw from that which is already in the original user's guides (duplication of effort is both unprofitable and unnecessary), but yet to provide a better understanding of the programs in their present state (in the CCPCSD), it is decided instead to make the reader aware of each user's guide, and to provide all of the changes to each program since incorporating each into the CCPCSD.

ROOTL (Root Locus Program)

The user's guide for program ROOTL is:

NAME: ROOTL USER'S MANUAL
A DIGITAL COMPUTER PROGRAM TO CALCULATE
AND PLOT THE ROOT LOCUS

DATE: March 1975

DEPARTMENT: Air Force Institute of Technology (AFIT/EN)
Wright-Patterson Air Force Base, OHio 45433

The following changes have been made to ROOTL:

1. Factors instead of roots are now entered for poles and zeros (i.e., for poles (s+3)(s+6): BEFORE - XP(1)=-3,-6 (as roots); NOW - XP(1)=3,6 (as factors).
2. The operator is provided with a list of options (if he desires).
3. The operator may elect to enter the input parameters either in one NAMELIST statement, or in groups of parameters in the form of a step-by-step procedure - several NAMELIST entries (each "asked" by the computer). For each entry, a '$INPUD' must start in column two, and a '$' must be added at the end of the entry.

EXAMPLE: a. Regular one-NAMELIST entry:

```
$INPUD AA=12,BB=13,CC=-10,DD=-14,
N=2,M=1,XP(1)=2,4,YP(1)=0,XZ(1)=3,
YZ(1)=0$
```

b. Reply after being "asked" about option by the step-by-step procedure:

```
$INPUD LL=1$
```

c. Wanting no change to parameters being "asked" by the step-by-step procedure:

```
$INPUD $
```

FREQR (Frequency Response Program)

The user's guide for program FREQR is:

NAME: FREQR
FREQUENCY RESPONSE PROGRAM

DATE: November 1975

DEPARTMENT: Air Force Institute of Technology (AFIT/EN)
Wright-Patterson Air Force Base, Ohio 45433

Program FREQR remains relatively unchanged with the only addition being a list of options (operator requested).

PARTL (Time Response Program)

The user's guide for program PARTL IS:

NAME: PARTL
HEAVISIDE PARTIAL FRACTION EXPANSION AND
TIME RESPONSE PROGRAM

DATE: December 1974 (Revised)

DEPARTMENT: Air Force Institute of Technology (AFIT/EN)
Wright-Patterson Air Force Base, Ohio 45433

The version of program PARTL that is presently in the CCPCSD is, unfortunately, not the latest AFIT version. However, the only change that the updated version has incorporated in it is in the format of the coefficients for the partial fraction expansion. (The version in the CCPCSD was received from the AFIT CDC 6600 computer library as the "latest" version of PARTL.)

Present (CCPCSD) PARTL: (Coefficient printout)

ROOT		POWER	COEFFICIENT	
REAL	IMAG		REAL	IMAG
...	...		...	...
REAL	IMAG		REAL	IMAG
...	...		...	...

Updated version of PARTL: (Coefficient Printout)

FACTOR		POWER			COEFFICIENT	
REAL	IMAG		REAL	IMAG	MAGNITUDE	ANGLE
...	...		...	...	...	...

It is quite possible that the truly updated version of PARTL is available now.

The only change to program PARTL is the termination procedure. A '0' (zero) is used to terminate normally instead of a '6' (six). Also, the operator must remember to dispose his plots to the calcomp plotter.

POLY (Polynomial Factoring Program)

The user's guide for program POLY is:

NAME: POLY
CDC 6600 PROGRAM TO FIND THE ZEROS OF
A POLYNOMIAL

DATE: October 1974 (Revised)

DEPARTMENT: Air Force Institute of Technology (AFIT/ENC)
Wright-Patterson Air Force Base, Ohio 45433

Program POLY does not have any changes or additions since its incorporation into the CCPCSD program.

It should be noted that, even without the program descriptions or the original user's guide for each of the programs, programs POLY, PARTL, and FREQR (in the CCPCSD) could be used satisfactorily by the operator. ROOTL is more complicated and, because of its input requirements, requires the user's guide as a reference for the symbols (LL, GTOL, GANE, XP(1), and others). These input symbols may be foreign to the operator. Therefore, the user's guides for programs ROOTL, FREQR, PARTL, and POLY should be obtained for references before using the CCPCSD.

APPENDIX B

USER'S MANUAL FOR THE CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN (CCPCSD)

(DECEMBER 1976)

Contents

CCPCSD

USER'S MANUAL

Introduction

The CCPCSD is developed to be completely user-oriented. With this in mind, an engineer or student need only know how to call it up (if it is already in the CDC 6600 computer library), and what commands to give the computer to begin executing the program. Therefore, it is assumed that the CCPCSD is in the permanent files in binary deck form and source deck form. A discussion follows now for using both forms. (Underlined commands and instructions indicate the computer commands; the replies from the operator are not underlined.)

CCPCSD In Binary Deck Form

When the CCPCSD is in the CDC 6600 computer in binary form, the commands to use the program are as follows:

(NOTE: PROJX is the file name for the CCPCSD.)

PLEASE LOGIN

LOGIN,(your problem number),(your password),(terminal ID number)

COMMAND - ATTACH,PROJX,(file name PROJX cataloged under),CY=(cycle PROJX cataloged),ID=(ID number PROJX cataloged under if different from yours),PW=(password if used when PROJX cataloged)

COMMAND - ATTACH,CCAUX,ID=X654321,SN=AFIT

COMMAND - LIBRARY,CCAUX

COMMAND - PROJX

The CCPCSD program now begins to execute. From this point, the operator need only follow the directions as they are presented, as they are completely self explanatory.

CCPCSD In Source Deck Form

If the source deck for the CCPCSD is in the permanent files, the following procedure is to be used to make the CCPCSD operational: (PROJX is the file name for the CCPCSD.)

PLEASE LOGIN

LOGIN,(your problem number),(your password), (terminal ID number)

COMMAND - ATTACH,PROJX,(file name PROJX cataloged under), CY=(cycle PROJX cataloged), ID= (ID number PROJX cataloged under if different from yours),PW=(password if used when PROJX cataloged)

COMMAND - ATTACH,CCAUX,ID=X654321,SN=AFIT

COMMAND - LIBRARY,CCAUX

COMMAND - RU,F=PROJX,FTN

The CCPCSD is now ready for execution. It is compiled (approximately 19 seconds of compilation time), and after being compiled, it begins execution by presenting a program title banner and the statement, "PLEASE TYPE 'TTY' IF YOU ARE AT AN INTERCOM TERMINAL." Again, the operator need only follow the directions as they are presented, as they are completely self explanatory.

Statements The CCPCSD Presents To The Operator

The executive overlay (Overlay(CCPCSD,0,0)) of the

CCPCSD presents the following directions, questions, statements, and explanations in the order given below:

a. "DO YOU WANT AN EXPLANATION OF THE PROGRAM? TYPE 'YES' OR 'NO'."

b. If your reply is 'YES', the following explanation is presented:

" YOU ARE ABOUT TO BEGIN A CONSOLIDATED COMPUTER PROGRAM WHICH CAN BE USED AS AN AID IN DESIGNING AND ANALYZING CONTROL SYSTEMS. YOU MUST HAVE YOUR SYSTEM REDUCED TO A SINGLE TRANSFER FUNCTION OF THE FORM:

$$\frac{K\ (S + Z_1)\ \ldots\ (S + Z_M)}{(S + P_1)\ \ldots\ (S + P_N)} ,$$

WHERE K=GAIN CONSTANT, Z=ZEROS, P=POLES OF THE SYSTEM; OR OF THE FORM:

$$\frac{K\ (A_1X^M + A_2X^{M-1} + \ldots + A_MX + A_{M+1}}{(B_1X^N + B_2X^{N-1} + \ldots + B_NX + B_{N+1}} .$$

IF YOU HAVE THE LAST FORM SHOWN, YOU WILL BE GIVEN AN OPPORTUNITY TO USE PROGRAM 'POLY' TO OBTAIN FACTORS (ROOTS) OF THE NUMERATOR AND DENOMINATOR FOR YOUR TRANSFER FUNCTION. THEN YOU WILL BE PROVIDED WITH A LIST OF THE INDIVIDUAL SUBPROGRAMS THAT ARE PRESENTLY AVAILABLE FOR YOUR USE. THE INPUT REQUIREMENTS WILL BE SPECIFIED AT THE BEGINNING OF YOUR RUN. AT THE END OF EACH RUN, YOU WILL BE GIVEN THE CHOICE TO STOP, CONTINUE WITH THE PRESENT RUN USING NEW DATA, OR GO TO ANOTHER SUBPROGRAM. IN ANY CASE, YOU WILL BE PROVIDED THE INITIAL LISTING OF SUBPROGRAMS ONLY AT THE BEGINNING; SO, COPY THE LIST AS SOON AS IT IS SHOWN IF YOU INTEND TO DO MORE THAN ONE DESIGN."

c. " YOU WILL NEED THE ROOTS AND/OR FACTORS OF THE NUMERATOR AND DENOMINATOR OF THE TRANSFER FUNCTION. IF YOU DO NOT HAVE THESE ROOTS AND/OR FACTORS, TYPE 'POLY'. OTHERWISE, TYPE 'NO'. (POLY WILL BE AVAILABLE AGAIN AS A SUBPROGRAM.)"

d. " THE FOLLOWING SUBPROGRAMS WITH EXPLAINED OUTPUTS ARE AVAILABLE:

PARTL...PARTIAL FRACTION EXPANSION AND TIME RESPONSE OUTPUT
FREQR...FREQUENCY RESPONSE WITH POLAR PLOT OR BODE PLOT OR BOTH
ROOTL...ROOT LOCUS WITH PRINTS OR PLOTS OR BOTH
POLY....DETERMINES ROOTS OF POLYNOMIAL

TYPE THE SYMBOLIC NAME OF THE RESPONSE YOU DESIRE. EXAMPLE: PARTL."

e. Once you have finished using any one of the programs shown in d above, the following statement is presented:

" IF YOU ARE FINISHED, TYPE 'STOP'. IF YOU DESIRE TO USE ANOTHER SUBPROGRAM, TYPE THE CORRECT SYMBOLIC NAME FOR THAT SUBPROGRAM."

Each of the statements shown above is easy to understand, and the operator should not have any problem directing the CCPCSD with these statements.

Do's And Don't's

Do: a. Make sure the CCPCSD is in the permanent files before attempting to attach the program. Know if the CCPCSD is in source or binary form.

b. Remember to attach the auxiliary calcomp routines (CCAUX) and to LIBRARY,CCAUX.

c. After commanding the CCPCSD to stop, always check FILES to make sure the file PLOT is disposed. If it is still present, type DISPOSE,PLOT,PT=IBB.

Don't:

a. Do not make any mistake while entering the input data into subprogram ROOTL. This will cause the total CCPCSD program to terminate.

b. Do not forget to double check each entry you make before returning the carriage.

Sample Problem

The following sample problem demonstrates the usefulness and versatility of the CCPCSD.

The system to be analyzed is assumed to be described by the equation

$$C(s) = \frac{4}{s(s^2 + s + 1)(s^2 + 0.4s + 4)}$$

$$= \frac{4}{s(s + 0.5 \pm j0.866)(s + 0.2 \pm j1.99)} \; .$$

File ANSWER is connected to obtain the printed output of the partial fraction expansion from subprogram PARTL. Options 1 (tabular listing) and 4 (calcomp plot) are chosen from subprogram PARTL. Option 1 (begin new problem), -1 (open-loop response, tabular listing), and -3 (calcomp plot of Bode diagram of the frequency responce) are chosen from subprogram FREQR. Option 3 (root locus, calcomp plot) is chosen for subprogram ROOTL. For the executive overlay, the explanation of the CCPCSD is printed out. For each of the subprograms, the list of options is printed out. The plot file (PLOT) is not disposed at the end of the total run, because subprogram ROOTL automatically disposes the file. (If subprogram ROOTL is not used, then the operator must dispose the plot file by giving the command DISPOSE,PLOT,PT= IBB before LOGOUT.) The terminal printout and calcomp plots are now shown.

```
COMMAND- ATTACH,FRED,OBRIEN,CY=5,PW=FLOB1

COMMAND- ATTACH,CCAUX,ID=X654321,SN=AFIT

 PFN IS
 CCAUX
 PF CYCLE NO. = 001
COMMAND- LIBRARY,CCAUX

COMMAND- CONNECT,ANSWER

COMMAND- FRED


****************************************************
*******START OF CONSOLIDATED COMPUTER PROGRAM*******
*******FOR CONTROL SYSTEM DESIGN.*******************
****************************************************


PLEASE TYPE 'TTY' IF YOU ARE AT AN INTERCOM TERMINAL.TTY

  DO YOU WANT AN EXPLANATION OF THE PROGRAM?
 TYPE EITHER 'YES' OR 'NO'.
YES

****************************************************
     YOU ARE ABOUT TO BEGIN A CONSOLIDATED COMPUTER
PROGRAM WHICH CAN BE USED AS AN AID IN DESIGNING AND ANALYZING
CONTROL SYSTEMS.  YOU MUST HAVE YOUR SYSTEM REDUCED TO A SINGLE
TRANSFER FUNCTION OF THE FORM*

                  K (S + Z ) ... (S + Z )
                          1              M
             ---------------------------

                    (S + P ) ... (S + P )
                          1              N



WHERE K = GAIN CONSTANT, Z = ZEROS, P = POLES OF THE SYSTEM,
OR OF THE FORM:

                 M         M-1        M-2
             K (A X  + A X      + A X      +...+ A X + A
                 1       2          3              M      M+1
        -------------------------------------------------

                N         N-1        N-2
             B X   + B X      + B X      +...+ B X + B
              1        2          3              N      N+1
```

PROJX

```
     IF YOU HAVE THE LAST FORM SHOWN, YOU WILL BE GIVEN AN
OPPORTUNITY TO USE PROGRAM 'POLY' TO OBTAIN FACTORS (ROOTS)
OF THE NUMERATOR AND DENOMINATOR FOR YOUR TRANSFER FUNCTION.
THEN YOU WILL BE PROVIDED WITH A LIST OF THE INDIVIDUAL SUB-
PROGRAMS THAT ARE PRESENTLY AVAILABLE FOR YOUR USE. THE
INPUT REQUIREMENTS WILL BE SPECIFIED AT THE BEGINNING OF
YOUR RUN. AT THE END OF EACH RUN, YOU WILL BE GIVEN THE
CHOICE TO STOP, CONTINUE WITH THE PRESENT RUN USING NEW
DATA, OR GO TO ANOTHER SUBPROGRAM.  IN ANY CASE, YOU WILL
BE PROVIDED THE INITIAL LISTING OF SUBPROGRAMS ONLY AT THE
BEGINNING; SO, COPY THE LIST AS SOON AS IT IS SHOWN IF YOU
INTEND TO DO MORE THAN ONE DESIGN.
***************************************************


---------------------------------------------------
        YOU WILL NEED THE ROOTS AND/OR FACTORS OF THE
NUMERATOR AND DENOMINATOR OF THE TRANSFER FUNCTION. IF
YOU DO NOT HAVE THESE ROOTS AND/OR FACTORS, TYPE 'POLY'.
OTHERWISE, TYPE 'NO'. (POLY WILL BE AVAILABLE AGAIN AS A
SUBPROGRAM.) NO


---------------------------------------------------

***************************************************
  THE FOLLOWING SUBPROGRAMS WITH EXPLAINED OUTPUTS ARE
AVAILABLE:

      PARTL...PARTIAL FRACTION EXPANSION WITH TIME RESPONSE OUTPUT.
      FREQR...FREQUENCY RESPONSE WITH POLAR PLOT OR BODE PLOT OUTPUT.
      ROOTL...ROOT LOCUS WITH PRINTS OR PLOTS OR BOTH.
      POLY....DETERMINES ROOTS OF POLYNOMIAL.


   TYPE THE SYMBOLIC NAME OF THE RESPONSE YOU DESIRE.
EXAMPLE:  PARTLPARTL

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


===================================================================

**************START OF PROGRAM 'PARTL'**************
TYPE NO. OF ZEROS, POLES, POWER OF REPEATED ROOT, AND NUMERATOR CONSTANT
0,5,0,4

TYPE REAL AND IMAG PARTS OF DENOMINATOR FACTOR.0,0

TYPE REAL AND IMAG PARTS OF DENOMINATOR FACTOR.0.5,0.866

CONJUGATE ASSUMED
TYPE REAL AND IMAG PARTS OF DENOMINATOR FACTOR.0.2,1.99

CONJUGATE ASSUMED
```

```
          ROOT                     POWER              COEFFICIENT

  REAL            IMAG                          REAL          IMAG
 0.              0.                  1         1.0000        0.
 .50000          .86600              1        -.64516       -.24834
 .50000        -.86600               1        -.64516        .24834
 .20000         1.9900               1         .14515       -.39441E-01
 .20000        -1.9900               1         .14515        .39441E-01

  REAL            IMAG                        MAGNITUDE       ANGLE
 0.              0.                  1         1.0000         0.000
 .50000          .86600              1         .69131      -158.947
 .50000        -.86600               1         .69131       158.947
 .20000         1.9900               1         .15042       -15.201
 .20000        -1.9900               1         .15042        15.201


THE TIME FUNCTION IS
F(T)=
          1.0000    EXP( 0.       T)
          1.3826    EXP(-.50000   T) SIN( .86600    *T+ 248.947)
          .30083    EXP(-.20000   T) SIN( 1.9900    *T+ 105.201)


CALCULATIONS COMPLETE
FOR TABULAR LISTING TYPE 1
FOR TELETYPE PLOT TYPE 2
FOR PRINTER PLOT TYPE 3
FOR CALCOMP PLOT TYPE 4
FOR ANOTHER PROBLEM TYPE 5
TO TERMINATE TYPE 0
TYPE OPTION1

TYPE INITIAL TIME, LISTING TIME DURATION, DELTA TIME.0,5,.5

        T                  F(T)
   0.               -.10730929E-07
   .50000000         .87120957E-02
   1.0000000         .10776403
   1.5000000         .38884255
   2.0000000         .80505999
   2.5000000         1.1809903
   3.0000000         1.3540184
   3.5000000         1.3005925
   4.0000000         1.1378976
   4.5000000         1.0151548
   5.0000000         .99951412
TYPE OPTION4

TYPE INITIAL TIME, PLOT DURATION.0,5

TYPE TITLE - 50 CHARACTERS MAX

SAMPLE CCPCSD---PARIL PORTION
```

```
TYPE NO. OF CHARACTERS, INCL SPACES.29

TYPE OPTION0


        --------END OF PROGRAM 'PARTL'--------


*****************************************************
 IF YOU ARE FINISHED, TYPE 'STOP'. IF YOU DESIRE TO USE
ANOTHER SUBPROGRAM, TYPE THE CORRECT SYMBOLIC NAME FOR THAT
SUBPROGRAM.
FREQR
 *****************************************************

==========================================================================

***************START OF PROGRAM 'FREQR'**************


 DO YOU WANT A LIST OF OPTIONS AVAILABLE FOR 'FREQR'?
TYPE 'YES' OR 'NO'.
YES


*****************************************************
THE FOLLOWING OPTIONS ARE AVAILABLE FOR PROGRAM 'FREQR':

    1...TO BEGIN A NEW PROBLEM. INPUT OF G(JW) OR G(Z).
   -1...FOLLOWED BY W (OMEGA); CALCULATES G(JW) USING FUNCTION
        ALREADY READ INTO MEMORY AFTER OPTION 1.
   -2...CLOSED-LOOP RESPONSE, C(JW)/R(JW).
    3...PRODUCES TTY PLOT OF OPTIONS -1 OR -2, OR BOTH.
   -3...PRODUCES CALCOMP PLOT OF BODE DIAGRAM (FREQ RESPONSE).
    4...TO READ IN A H(JW), FEEDBACK TRANSFER FUNCTION.
    5...ADDS A TRANSPORT LAG TERM (E(EXP JWT)) TO G(JW).
    6...INPUT NEW FORWARD GAIN CONSTANT ONLY.
   -6...INPUT NEW FEEDBACK GAIN CONSTANT ONLY.
    7...ALLOWS AUTOMATIC DISPOSE BODE PLOTS TO BLDG 640, 'BB'.
    9...TO SEE PRESENT G(S), H(S), OF G(Z) FUNCTIONS.
  -11...SAME AS -1, EXCEPT G(Z) IS CALCULATED.
  -12...SAME AS -2, EXCEPT CLOSED-LOOP Z-DOMAIN IS CALCULATED.
   15...TO SET A SAMPLE PERIOD FOR THE SAMPLED DATA SYSTEM.
    0...CAUSES THE PROGRAM TO STOP.
*****************************************************
TYPE OPTION NUMBER -- 1


  FORWARD TRANSFER FUNCTION -- G(S)

TYPE GAIN CONSTANT K --4
```

```
GAIN CONSTANT =    4.000000

TYPE ORDER OF NUMERATOR --0

TYPE NUMBER OF FACTORS OF A GIVEN ORDER --1

TYPE ORDER OF THOSE FACTORS --0

TYPE  1 COEFFICIENT(S)
1

COEFFICIENT(S) OF NUMERATOR
 1.000000E+00 S( 0)

TYPE ORDER OF DENOMINATOR --5

TYPE NUMBER OF FACTORS OF A GIVEN ORDER --1

TYPE ORDER OF THOSE FACTORS --1

TYPE  2 COEFFICIENT(S)
1,0

COEFFICIENT(S) OF DENOMINATOR
 1.000000E+00 S( 1)   0.             S( 0)

TYPE NUMBER OF FACTORS OF A GIVEN ORDER --2

TYPE ORDER OF THOSE FACTORS --2

TYPE  6 COEFFICIENT(S)
1,1,1,1,.4,4

COEFFICIENT(S) OF DENOMINATOR
 1.000000E+00 S( 2)   1.000000E+00 S( 1)   1.000000E+00 S( 0)

 1.000000E+00 S( 2)   4.000000E-01 S( 1)   4.000000E+00 S( 0)

TYPE OPTION NUMBER -- -1

TO RESTART PLOT ARRAY STORAGE, ENTER A '1'.

1

TYPE INITIAL, INCREMENTAL, AND FINAL OMEGA.
0.1,.8,8
```

```
    G -- OPEN-LOOP RESPONSE

          W            MAGNITUDE               MAG. IN DB        ANGLE IN DEGREES
        .10           10.074552                20.064515          -96.3423
        .90           1.5051088                3.5513577         -174.5180
       1.70           .71105541               -2.9619311         -259.5217
       2.50           .11175204              -19.034891           -40.5742
       3.30           .16572214E-01          -35.612389           -60.7022
       4.10           .46252095E-02          -46.697372           -68.1662
       4.90           .17258231E-02          -55.260074           -72.3841
       5.70           .76724325E-03          -62.301338           -75.1645
       6.50           .38438311E-03          -68.304714           -77.1566
       7.30           .21018778E-03          -73.547850           -78.6622
  TYPE OPTION NUMBER -- -3

   IF YOU WANT A MAG VS W PLOT , ENTER A 0

   IF YOU WANT A DB VS LOG W PLOT,ENTER A 1

   IF YOU WANT BOTH TYPES OF PLOTS, ENTER A 2 2

  TYPE IN TITLE OF M-VS-W PLOT, 30 CHARACTERS MAX

  TITLE IS SAMPLE CCPCSD---FREQR PORTION

   TYPE IN TITLE OF DB VS LOG W PLOT, 30 CHARACTERS MAX

  TITLE IS SAMPLE CCPCSD---FREQR PORTION

  TYPE OPTION NUMBER -- 0


          -------END OF PROGRAM 'FREQR'-------


  ******************************************************
   IF YOU ARE FINISHED, TYPE 'STOP'. IF YOU DESIRE TO USE
  ANOTHER SUBPROGRAM, TYPE THE CORRECT SYMBOLIC NAME FOR THAT
  SUBPROGRAM.
  ROOTL

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

```
***************START OF PROGRAM 'ROOTL'**************

/########################################################
  DO YOU WANT A LIST OF OPTIONS AVAILABLE FOR 'ROOTL'?
TYPE 'YES' OR 'NO'.YES


*******************************************************
THE FOLLOWING OPTIONS ARE AVAILABLE FOR 'ROOTL':

     0...TO STOP THE PROGRAM
     1...COMPUTES EACH BRANCH OF THE LOCUS OF THE TRANSFER
         FUNCTION OVER SPECIFIED BOUNDED REGION. TABULAR OUTPUT
         FOR BRANCHES, PLUS PLOT, IF DESIRED.
     2...SIMILAR TO '1', EXCEPT STEP SIZE CAN BE SPECIFIED.
     3...SIMILAR TO '1', EXCEPT ZETA, DAMPING RATIO, SPECIFIED.
     4...TO INVESTIGATE PARTS OF LOCUS OF SPECIAL INTEREST.
     5...SIMILAR TO '4', WITH BOUNDARIES OF '1' - '3'; STARTING
         POINT MUST BE SPECIFIED.
     6...TRUNCATED VERSION OF '1'. ONLY ROOTS OF INTEREST FOR A
         SPECIFIED GAIN OF INTEREST (GANE) AND TOLERANCE (GTOL)
         ARE LISTED. NO PLOT GENERATED. GTOL NOT EQUAL TO ZERO.
     7...SIMILAR TO '6'. FOR SPECIFIED DAMPING RATIO (ZETA), ROOTS
         OF INTEREST ARE SPECIFIED. (GANE,GTOL, AND TIME ARE
         NOT SPECIFIED.
*******************************************************

*******************************************************
  NOW YOU WILL HAVE THE CHOICE OF EITHER ENTERING THE
INPUT DATA IN ONE NAMELIST STATEMENT (ASSUMING YOU KNOW EACH
PARAMETER AND HAVE THE LIST PREPARED), OR FOLLOWING A STEP BY
STEP PROCEDURE TO INPUT EACH PARAMETER. IF YOU INTEND TO USE
ONE NAMELIST STATEMENT, TYPE 'ONE'; FOR A STEP BY STEP METHOD,
TYPE 'STEP'.
ONE

-------------------------------------------------------


ENTER NAMELIST INPUD. START IN COLUMN 2 WITH A DOLLAR
SIGN AND INPUD ($INPUD). PUT COMMA BETWEEN EACH ENTRY. END
WITH A DOLLAR SIGN. (EXAMPLE: $INPUD LL=3,N=2,XP(1)=2,4$)
 $INPUD LL=3,N=5,M=0,XP(1)=0,0.5,0.5,0.2,0.2,YP(1)=0,0.866,
 -0.866,1.99,-1.99,GA=4,ZETA=.7,FPN=1,ITITL=29,IPLOT=7,
 MAXDSP=1$

#######################################################



ENTER ITITL TITLESAMPLE CCPCSD---ROOTL PORTION


SAMPLE CCPCSD---ROOTL PORTION
```

ROOT LOCUS USING OPTION NUMBER 3

CODE
0-LOCUS PT. 2-ZERO 4-IMAGINARY AXIS
1-POLE 3-BREAK PT. 5-SENSITIVITY PT.

5 POLES AT

X = -0.	Y = 0.
X = -.50000	Y = -.86600
X = -.50000	Y = .86600
X = -.20000	Y = -1.9900
X = -.20000	Y = 1.9900

TIME DELAY (TIME) = 0. , GAIN CONSTANT (GA) = 4.0000000

DAMPING FACTOR OF INTEREST (ZETA) = .7000

REGION OF CALCULATION-REAL -4.50 TO .500
IMAG -3.50 TO 3.50

BRANCH NUMBER 1

CALCULATION STEP SIZE = .1000
PRINTING STEP SIZE = .2000

LOCUS REAL	LOCUS IMAG	DIST TO ORIGIN	GAIN	ZETA	CD
-0.	0.	0.	0.	1.00000	1
-.20000000	0.	.20000000	.166315	1.00000	0
-.40000000	0.	.40000000	.303990	1.00000	0
-.60000000	0.	.60000000	.469664	1.00000	0
-.80000000	0.	.80000000	.725739	1.00000	0
-1.0000000	0.	1.0000000	1.14997	1.00000	0
-1.2000000	0.	1.2000000	1.84509	1.00000	0
-1.4000000	0.	1.4000000	2.94837	1.00000	0
-1.6000000	0.	1.6000000	4.64125	1.00000	0
-1.8000000	0.	1.8000000	7.15894	1.00000	0
-2.0000000	0.	2.0000000	10.8000	1.00000	0
-2.2000000	0.	2.2000000	15.9359	1.00000	0
-2.4000000	0.	2.4000000	23.0208	1.00000	0
-2.6000000	0.	2.6000000	32.6009	1.00000	0
-2.8000000	0.	2.8000000	45.3243	1.00000	0
-3.0000000	0.	3.0000000	61.9501	1.00000	0
-3.2000000	0.	3.2000000	83.3589	1.00000	0
-3.4000000	0.	3.4000000	110.561	1.00000	0
-3.6000000	0.	3.6000000	144.709	1.00000	0
-3.8000000	0.	3.8000000	187.102	1.00000	0
-4.0000000	0.	4.0000000	239.200	1.00000	0
-4.2000000	0.	4.2000000	302.634	1.00000	0
-4.4000000	0.	4.4000000	379.210	1.00000	0
-4.6000000	0.	4.6000000	470.925	1.00000	0

BOUNDARY

BRANCH NUMBER 2

CALCULATION STEP SIZE = .1000
PRINTING STEP SIZE = .2000

LOCUS REAL	LOCUS IMAG	DIST TO ORIGIN	GAIN	ZETA	CD
-.50000000	-.86600000	.99997800	0.	.50001	1
-.30448854	-.82957040	.88368565	.246524	.34457	0
-.11262071	-.88257788	.88973431	.493125	.12658	0
0.	-.94134735	.94134735	.699934	.00000	4
.17382086	-1.0402725	1.0546945	1.15627	.16481	0
.35058902	-1.1338059	1.1867723	1.85652	.29541	0
.53042467	-1.2213151	1.3315258	2.94947	.39836	0

BOUNDARY

BRANCH NUMBER 3

CALCULATION STEP SIZE = .1000
PRINTING STEP SIZE = .2000

LOCUS REAL	LOCUS IMAG	DIST TO ORIGIN	GAIN	ZETA	CD
-.50000000	.86600000	.99997800	0.	.50001	1
-.30448854	.82957040	.88368565	.246524	.34457	0
-.11262071	.88257788	.88973431	.493125	.12658	0
0.	.94134735	.94134735	.699934	.00000	4
.17382086	1.0402725	1.0546945	1.15627	.16481	0
.35058902	1.1338059	1.1867723	1.85652	.29541	0
.53042467	1.2213151	1.3315258	2.94947	.39836	0

BOUNDARY

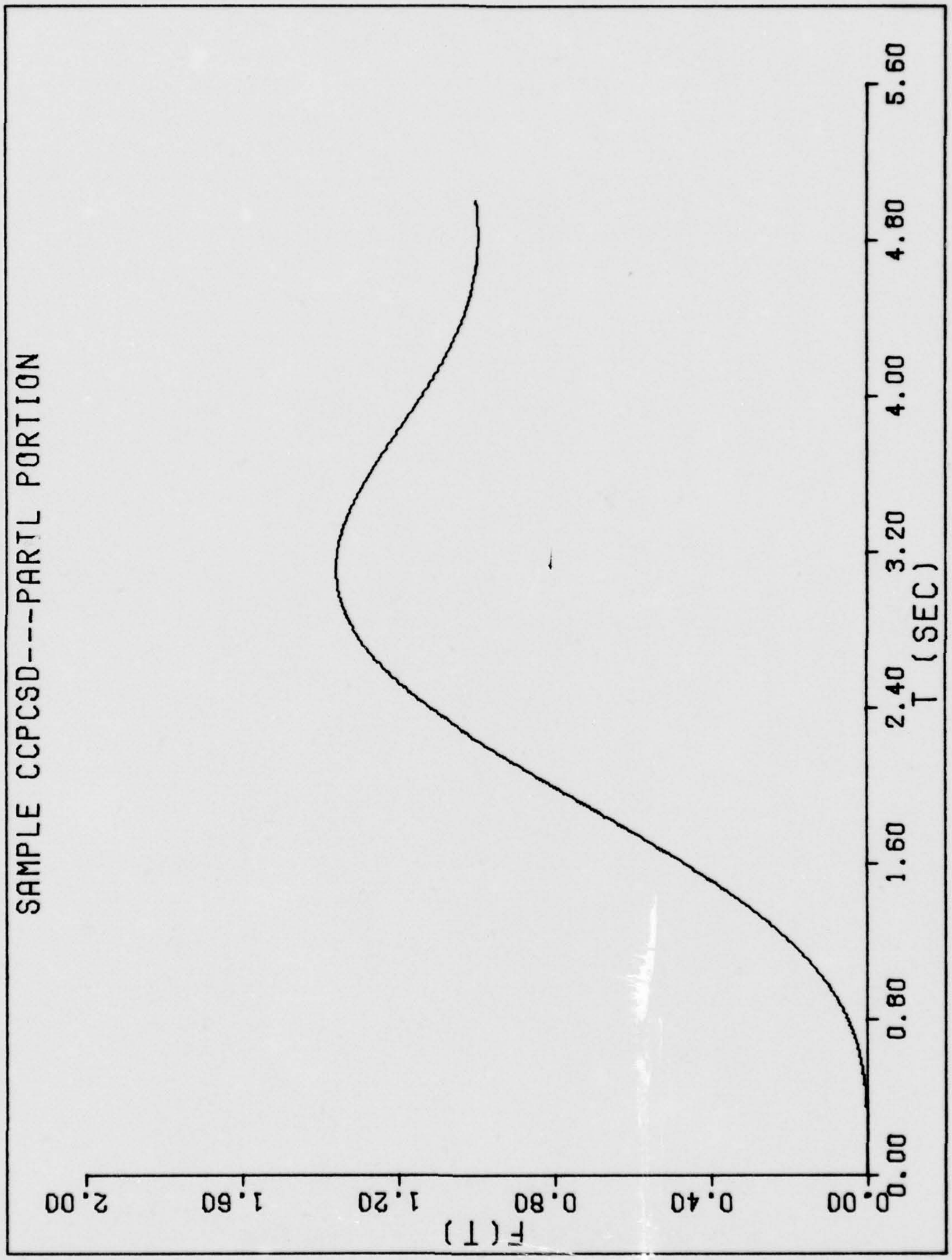
BRANCH NUMBER 4

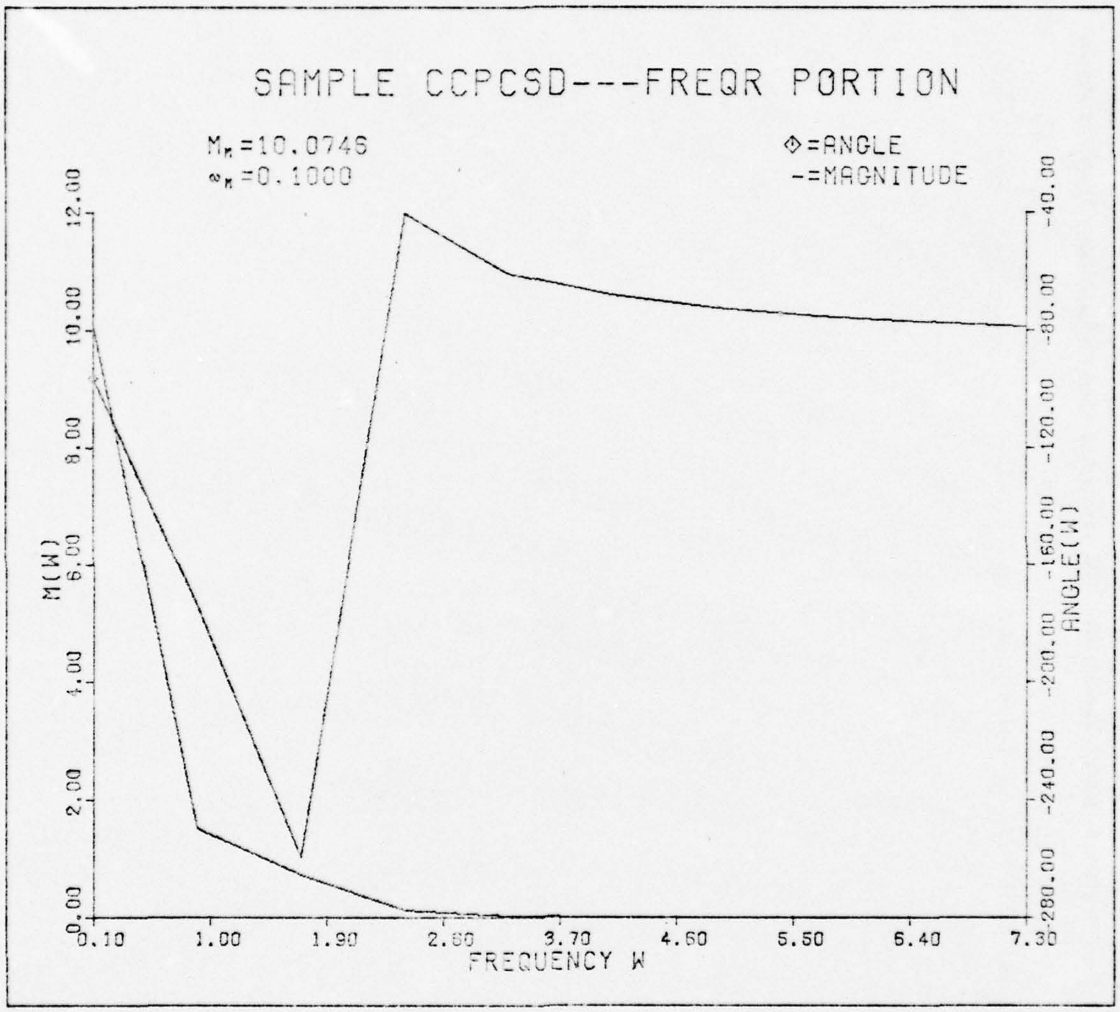
CALCULATION STEP SIZE = .1000
PRINTING STEP SIZE = .2000

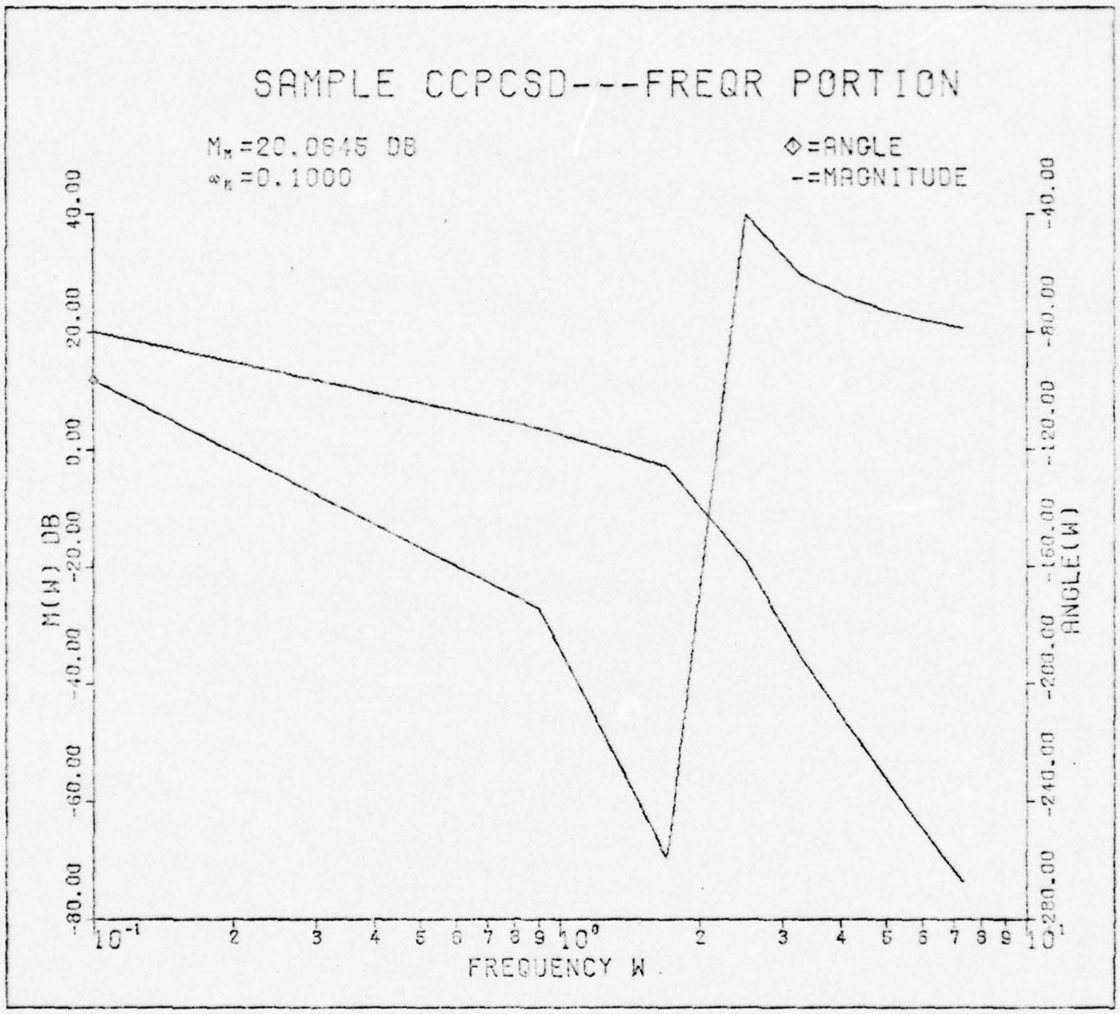
LOCUS REAL	LOCUS IMAG	DIST TO ORIGIN	GAIN	ZETA	CD
-.20000000	-1.9900000	2.0000250	0.	.10000	1
-.39519276	-2.0125619	2.0509955	1.33897	.19268	0
-.55125307	-2.1363737	2.2063482	3.31859	.24985	0
-.67354420	-2.2944421	2.3912604	6.60731	.28167	0
-.77726025	-2.4654006	2.5850210	11.7819	.30068	0
-.87005745	-2.6425529	2.7821009	19.4947	.31273	0
-.95587349	-2.8231994	2.9806289	30.5102	.32070	0
-1.0369288	-3.0060350	3.1798534	45.7200	.32609	0
-1.1145739	-3.1903461	3.3794354	66.1536	.32981	0
-1.1896814	-3.3757066	3.5792090	92.9887	.33239	0
-1.2628424	-3.5618444	3.7790880	127.560	.33417	0

BOUNDARY

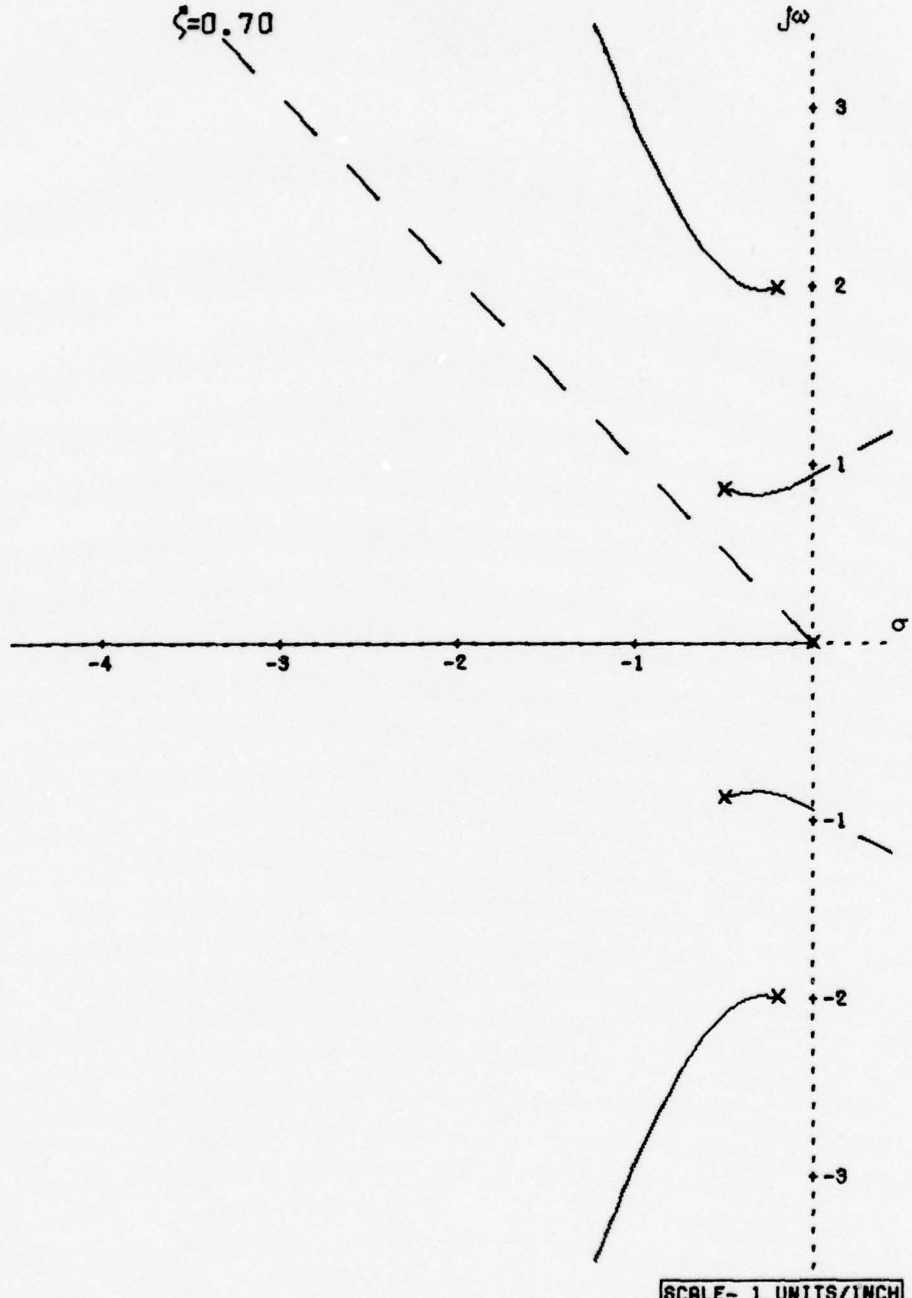
```
                    BRANCH NUMBER    5

     CALCULATION STEP SIZE =   .1000
     PRINTING STEP SIZE    =   .2000

   LOCUS REAL       LOCUS IMAG     DIST TO ORIGIN      GAIN           ZETA     CD
 -.20000000        1.9900000       2.0000250       0.               .10000    1
 -.39519276        2.0125619       2.0509955       1.33897          .19268    0
 -.55125307        2.1363737       2.2063482       3.31859          .24985    0
 -.67354420        2.2944421       2.3912604       6.60731          .28167    0
 -.77726025        2.4654006       2.5850210       11.7819          .30068    0
 -.87005745        2.6425529       2.7821009       19.4947          .31273    0
 -.95587349        2.8231994       2.9806289       30.5102          .32070    0
 -1.0369288        3.0060350       3.1798534       45.7200          .32609    0
 -1.1145739        3.1903461       3.3794354       66.1536          .32981    0
 -1.1896814        3.3757066       3.5792090       92.9887          .33239    0
 -1.2628424        3.5618444       3.7790880       127.560          .33417    0
        BOUNDARY

/##################################################
   DO YOU WANT A LIST OF OPTIONS AVAILABLE FOR 'ROOTL'?
 TYPE 'YES' OR 'NO'.NO

**************************************************
   NOW YOU WILL HAVE THE CHOICE OF EITHER ENTERING THE
 INPUT DATA IN ONE NAMELIST STATEMENT (ASSUMING YOU KNOW EACH
 PARAMETER AND HAVE THE LIST PREPARED), OR FOLLOWING A STEP BY
 STEP PROCEDURE TO INPUT EACH PARAMETER. IF YOU INTEND TO USE
 ONE NAMELIST STATEMENT, TYPE 'ONE'; FOR A STEP BY STEP METHOD,
 TYPE 'STEP'.ONE

 --------------------------------------------------


 ENTER NAMELIST INPUD. START IN COLUMN 2 WITH A DOLLAR
 SIGN AND INPUD ($INPUD). PUT COMMA BETWEEN EACH ENTRY. END
 WITH A DOLLAR SIGN. (EXAMPLE: $INPUD LL=3,N=2,XP(1)=2,4$)
  $INPUD LL=0$

##################################################

  1 PLOTS DISPOSED TO TERMINAL 7

        -------END OF PROGRAM 'ROOTL'------


**************************************************
  IF YOU ARE FINISHED, TYPE 'STOP'. IF YOU DESIRE TO USE
 ANOTHER SUBPROGRAM, TYPE THE CORRECT SYMBOLIC NAME FOR THAT
 SUBPROGRAM.
 STOP

  **************************************************
      STOP
```


SAMPLE CCPCSD---PARTL PORTION
F(T)
0.00
0.40
0.80
1.20
1.60
2.00
T (SEC)
0.00
0.80
1.60
2.40
3.20
4.00
4.80
5.60


SAMPLE CCPCSD---FREQR PORTION
M_M=10.0746
ω_M=0.1000
◇=ANGLE
-=MAGNITUDE
M(W)
0.00
2.00
4.00
6.00
8.00
10.00
12.00
ANGLE(W)
-40.00
-80.00
-120.00
-160.00
-200.00
-240.00
-280.00
0.10
1.00
1.90
2.80
3.70
4.60
5.50
6.40
7.30
FREQUENCY W


SAMPLE CCPCSD---FREQR PORTION
M_M=20.0645 DB
ω_M=0.1000
◇=ANGLE
-=MAGNITUDE
M(W) DB
40.00
20.00
0.00
-20.00
-40.00
-60.00
-80.00
ANGLE(W)
-40.00
-80.00
-120.00
-160.00
-200.00
-240.00
-280.00
FREQUENCY W

SAMPLE CCPCSD---ROOTL PORTION



$$G(S)H(S)=\frac{K}{S(S^2+1S+1.000)(S^2+0.4S+4)}$$

FIGURE 1 - ROOT LOCUS

APPENDIX C

LISTING OF THE CCPCSD

(DECEMBER 1976)

LISTING OF THE CCPCSD

This appendix contains the source listing for the Consolidated Computer Program for Control System Design (CCPCSD). The sequence numbers are on the listing in an effort to maintain the order of the listing, as it is physically long, and, hopefully, to assist anyone who wishes to work with the program.

```
      OVERLAY(CCPCSD,0,0)
      PROGRAM CCPCSD(INPUT=100B,OUTPUT=100B,ANSWER,PPLOT,CFILE,PLOT,
     +TAPE5=INPUT,TAPE7=OUTPUT,TAPE6=ANSWER,TAPE8=PPLOT,TAPE9=100B)
      LOGICAL TTY
      TTY=.TRUE.
      PRINT 888
888   FORMAT(" ************************************************************
     +"/" *******START OF CONSOLIDATED COMPUTER PROGRAM*******"/
     +" *******FOR CONTROL SYSTEM DESIGN.*******************"/
     +" *******************************************************
     +"//)
      PRINT 999
999   FORMAT (" PLEASE TYPE 'TTY' IF YOU ARE AT AN INTERCOM TERMINAL.")
      READ 1000,I
1000  FORMAT (A10)
      IF (I.EQ.3HTTY) GO TO 1
1     PRINT 1013
1013  FORMAT("    DO YOU WANT AN EXPLANATION OF THE PROGRAM?"/
     +"   TYPE EITHER 'YES' OR 'NO'."/)
15    READ 1014,K
1014  FORMAT(A10)
      IF(K.EQ.3HYES) GO TO 15
      IF(K.EQ.2HNO) GO TO 13
      PRINT 1015
1015  FORMAT("    SORRY, TRY AGAIN...TYPE EITHER 'YES' OR 'NO'.")
      GO TO 16
15    PRINT 1001
1001  FORMAT(" ************************************************************
     +"/"        YOU ARE ABOUT TO BEGIN A CONSOLIDATED COMPUTER"/
     +" PROGRAM WHICH CAN BE USED AS AN AID IN DESIGNING AND ANALYZING"
```

```
     +/" CONTROL SYSTEMS.  YOU MUST HAVE YOUR SYSTEM REDUCED TO A SINGLE
     +"/" TRANSFER FUNCTION OF THE FORM*
     +"/"                                                                 "
     +/"                      K (S + Z ) ... (S + Z )                     "
     +/"                              1             M                     "
     +/"                    --------------------------                    "
     +/"                                                                  "
     +/"                        (S + P ) ... (S + P )                     "
     +/"                              1             N         ")
      PRINT 5005
 5005 FORMAT(/"  "/
     +/"                                                                  "
     +/" WHERE K = GAIN CONSTANT, Z = ZEROS, P = POLES OF THE SYSTEM,"/
     +" OR OF THE FORM:                                                  "/
     +/"                   M       M-1       M-2                         "/
     +"              K (A X + A X       + A X     +...+ A X + A          "/
     +"                  1     2          3             M     M+1         "/
     +"            -------------------------------------------------------"//
     +"                  N       N-1       N-2                           "/
     +"               B X  + B X     + B X     +...+ B X + B             "/
     +"                1     2         3             N     N+1    "//)
      PRINT 5010
 5010 FORMAT(/"    "/
     +"      IF YOU HAVE THE LAST FORM SHOWN, YOU WILL BE GIVEN AN  "/
     +" OPPORTUNITY TO USE PROGRAM 'POLY' TO OBTAIN FACTORS (ROOTS)"/
     +" OF THE NUMERATOR AND DENOMINATOR FOR YOUR TRANSFER FUNCTION."/
     +" THEN YOU WILL BE PROVIDED WITH A LIST OF THE INDIVIDUAL SUB-"/
     +" PROGRAMS THAT ARE PRESENTLY AVAILABLE FOR YOUR USE. THE "/
     +" INPUT REQUIREMENTS WILL BE SPECIFIED AT THE BEGINNING OF"/
     +" YOUR RUN. AT THE END OF EACH RUN, YOU WILL BE GIVEN THE"/
```

```
     +" CHOICE TO STOP, CONTINUE WITH THE PRESENT RUN USING NEW"/
     +" DATA, OR GO TO ANOTHER SUBPROGRAM.  IN ANY CASE, YOU WILL"/
     +" BE PROVIDED THE INITIAL LISTING OF SUBPROGRAMS ONLY AT THE"/
     +" BEGINNING: SO, COPY THE LIST AS SOON AS IT IS SHOWN IF YOU"/
     +" INTEND TO DO MORE THAN ONE DESIGN."/
     +" *****************************************************")
13    PRINT 1002
1002 FORMAT(//" -----------------------------------------------------------
     +"/"          YOU WILL NEED THE ROOTS AND/OR FACTORS OF THE"/
     +" NUMERATOR AND DENOMINATOR OF THE TRANSFER FUNCTION. IF"/
     +" YOU DO NOT HAVE THESE ROOTS AND/OR FACTORS, TYPE 'POLY'."/
     +" OTHERWISE, TYPE 'NO'. (POLY WILL BE AVAILABLE AGAIN AS A "/
     +" SUBPROGRAM.)")
12    READ 1003,TF
1003 FORMAT(A7)
      IF(TF.EQ.4HPOLY) GO TO 3
      IF(TF.EQ.2HNO) GO TO 2
      PRINT 1010
1010 FORMAT("   SORRY, TRY AGAIN...TYPE 'POLY' OR 'NO'."/)
      GO TO 12
3     PRINT *
      PRINT *," ----------------------------------------------------"
      CALL OVERLAY(5HCCPCSD,4,0,0)
      GO TO 3055
2     PRINT *
      PRINT *," ----------------------------------------------------"
3055 IF(TTY) PRINT 1005
1005 FORMAT(/" *****************************************************"/
     +"   THE FOLLOWING SUBPROGRAMS WITH EXPLAINED OUTPUTS ARE"/
     +" AVAILABLE:"//"        PARTL...PARTIAL FRACTION EXPANSION WITH TIM
```

```
     +E RESPONSE OUTPUT."/"        FREQR...FREQUENCY RESPONSE WITH POLAR
     +PLOT OR BODE PLOT OUTPUT."/"        ROOTL...ROOT LOCUS WITH PRINTS
     +OR PLOTS OR BOTH."/"       POLY....DETERMINES ROOTS OF POLYNOMIAL.
     +"///"    TYPE THE SYMBOLIC NAME OF THE RESPONSE YOU DESIRE.
     +"/" EXAMPLE:   PARTL")
7     READ 1011,J
1011 FORMAT(A10)
     PRINT *," ************************************************************"
18   IF(J.EQ.5HPARTL) GO TO 9
     IF(J.EQ.5HFREQR) GO TO 11
     IF(J.EQ.5HROOTL) GO TO 17
     IF(J.EQ.4HPOLY) GO TO 20
     IF(J.EQ.4HSTOP) STOP
     PRINT 1008
1008 FORMAT("     ERROR...THE SUBPROGRAM YOU HAVE LISTED IS INCORRECT"/
     +" OR NOT AVAILABLE.  RETYPE THE FIVE-LETTER SYMBOLIC NAME"/
     +" FOR THE SUBPROGRAM."/)
     GO TO 7
9    CALL OVERLAY(6HCCPCSD,3,0,0)
10   PRINT 1009
1009 FORMAT(/" ***********************************************************
     +"/"  IF YOU ARE FINISHED, TYPE 'STOP'. IF YOU DESIRE TO USE "/
     +" ANOTHER SUBPROGRAM, TYPE THE CORRECT SYMBOLIC NAME FOR THAT"/
     +" SUBPROGRAM."/)
     READ 1012,J
1012 FORMAT(A10)
     PRINT *," ************************************************************"
     IF(J.NE.4HSTOP) GO TO 18
     STOP
20   CALL OVERLAY(6HCCPCSD,4,0,0)
```

```
      GO TO 10
11    CALL OVERLAY(6HCCPCSD,2,0,0)
      GO TO 10
17    CALL OVERLAY(6HCCPCSD,1,0,0)
      GO TO 10
      END


      OVERLAY(CCPCSD,1,0)
      PROGRAM ROOT10
C
C    THIS PROGRAM READS INPUT DATA, ESTABLISHES DEFAULT VALUES, AND
C    DIRECTS THE FLOW OF THE OVERALL PROGRAM
C
      COMMON /MAIN01/BOUND,DZW(8),FPN,IFOLD,IFPN,ISTAM,ITITL,ITHES,RAD
     +   ,THESI(8),ZETA,MULT
      COMMON /MAIN02/CGIN,CNIG,GAG,II,LN,DELPR(50)
      COMMON /MAIN012/GANE,NOPLOT,GTOL,IIV,LL,V,XAX,YAX,DEL(50),
     +  XDISL(50),XDISR(50),YDISD(50),YDISU(50),XOT,YOT,CB
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGSS/A,B,ANG,PI
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      COMMON /ANGUU/ AA,BB,CC,DD,D,DA, JK,SANG,CA
      LOGICAL NOPLOT
      NAMELIST/INPUD/LL,N,M,GA,TIME,XP,YP,XZ,YZ,DEL,AA,BB,CC,DD,GANE
     1,GTOL,X,Y,LN,JK,SNG,FPN,ITITL,IFOLD,ISTAM,ZETA,ITHES,BOUND,RAD
     2,MULT,IPLOT,MAXDSP,DELPR
      DATA GANE/1000./           ,FPN/0/,
```

```
     1AGIN/6H    GA/,ANIG/6HIN     /,BGIN/6HSENSIT/,BNIG/6HIVITY /
      DATA MULT/0/,AA,BB,CC,DD/.5,3.5,-4.5,-3.5/,PI/3.1415926535/
      DATA E/.0001/,N/1/,M,ITHES/2*0/
      DATA NUMPLOT/0/,MAXDSP/4/,FRSTPLT/0./,NNTERM/0/
      CALL CONNEC(9)
      PRINT 891
  891 FORMAT(//" ===================================================
     +=================="//" ***************START OF PROGRAM 'ROOTL'*****
     +**********")
C
C     INITIALIZE VALUES
C
1     GA=1.
      DO 1000 I=1,50
 1000 DELPR(I)=DEL(I)=0
      NTERM=IPLOT=ISTAM=IFOLD=ITITL=LN=GTOL=ZETA=TIME=0
      LL=J=II=1
      BOUND=SNG=1.
      NOPLOT=.TRUE.
      CGIN=BGIN
      CNIG=BNIG
      JK=50
      RAD=.01
C
C     READ INPUT DATA
C
      WRITE(9,3000)
 3004 READ(5,3001) JI
      IF(JI.EQ.3HYES) GO TO 3005
      IF(JI.EQ.2HNO) GO TO 3007
```

```
      WRITE(9,3003)
C
C     LIST OF OPTIONS
C
      GO TO 3004
 3005 WRITE(9,3006)
C
C     CHOICE OF INPUT FORMATS
C
 3007 WRITE(9,3008)
 3011 READ(5,3009) KL
      WRITE(9,3013)
      IF(KL.EQ.3HONE) GO TO 3012
      IF(KL.EQ.4HSTEP) GO TO 3014
      WRITE(9,3010)
      GO TO 3011
C
C     INSTRUCTIONS FOR 'SEPARATE' INPUT. STATEMENTS FOLLOWING.
C
 3014 WRITE(9,3015)
      WRITE(9,3016)
      READ(5,INPUD)
      WRITE(9,3017)
      READ(5,INPUD)
      WRITE(9,3018)
      READ(5,INPUD)
      WRITE(9,3019)
      READ(5,INPUD)
      WRITE(9,3020)
      READ(5,INPUD)
```

```
      WRITE(9,3021)
      READ(5,INPUD)
      WRITE(9,3022)
      READ(5,INPUD)
      WRITE(9,3023)
      READ(5,INPUD)
      WRITE(9,3024)
      READ(5,INPUD)
      WRITE(9,3025)
      READ(5,INPUD)
      WRITE(9,3026)
      READ(5,INPUD)
      WRITE(9,3027)
      READ(5,INPUD)
      WRITE(9,3028)
      READ(5,INPUD)
      GO TO 3023
C
C     ONE-TIME INPUT OF NAMELIST DATA
C
 3012 WRITE(9,200)
      READ(5,INPUD)
      WRITE(9,3002)
C
C     CHANGE SIGNS SO 'FACTORS' ENTERED --- NOT 'ROOTS'
C
 3029 DO 3040 I=1,M
      XZ(I)=-XZ(I)
      YZ(I)=-YZ(I)
 3040 CONTINUE
```

```
      DO 3041 J=1,N
      XP(J)=-XP(J)
      YP(J)=-YP(J)
 3041 CONTINUE
      IF(EOF(5).NE.0.) LL=0
      IF(LL.GT.4) IPLOT=0
      IF(IPLOT .EQ. 0) GO TO 300
      NOPLOT = .FALSE.
      NNTERM=IPLOT
C    IF THIS IS FIRST PLOT TO BE GENERATED,
C     INITIALIZE PLOTTER
      IF(FRSTPLT .NE. 0) GO TO 300
      FRSTPLT=1.
        CALL PLOT (0.,-3.,-3)
        CALL PLOT (0.,1.,-3)
C     DISPOSE PLOTS IF NECESSARY
  300 IF(((LL .NE. 0) .AND. ((NUMPLOT .LT. MAXDSP) .OR. (MULT .NE. 0)))
     +  .OR. (NUMPLOT .EQ. 0)) GO TO 301
      LLL = 0
      IF(LL.EQ.0) LLL=1
      CALL DSP(NNTERM,LLL)
      WRITE(9,201)NUMPLOT,NNTERM
      NUMPLOT=0
  301 IF(LL .EQ. 0) GO TO 1249
      TIME=ABS(TIME)
      GA = ABS(GA)
      IF (GA.EQ.1.) GO TO 3
      GA=1./GA
      CGIN=AGIN
      CNIG=ANIG
```

```
C     PAGE EJECT FOR NEW PROBLEM.
 3     WRITE(7,174)
C
C      TEST FOR FIRST TITLE
C       0 MEANS NONE
C      >0 MEANS GET ONE
C      <0 MEANS USE THE OLD ONE
C
       IF (ITHES.LE.0) GO TO 4
       ITHES=-1
C
C     READ AND WRITE: TITLES, FIGURE #, OPTION # AND CODE FOR POINTS.
C
       WRITE(9,202)
       READ (5,181) THESI
 4     IF(ITHES.LT.0)WRITE(7,179)THESI
       IF(ITITL .LE. 0) GO TO 5
       WRITE(9,203)
       READ(5,181) DZW
       WRITE(7,179)DZW
 5     IFPN=FPN
       IF(IFPN.NE.0)WRITE(7,180)IFPN
       WRITE(7,168)LL
       IF (IABS(LL).GE.6) GO TO 400
       WRITE(7,1680)
 400   IF (LL.GT.0) GO TO 12
C
C      IF LL IS NEGATIVE XP AND XZ ARE DENOMINATOR AND NUMERATOR
C       COEFFICIENTS OF THE UNFACTORED TRANSFER FUNCTION
C
```

```
      LL=IABS(LL)
      GA=GA*ABS(XP(1)/XZ(1))
      IF(GA.NE.1.)DGIN=AGIN
      IF(GA.NE.1.)DNIG=ANIG
      IF (XP(1).EQ.0.) GO TO 1
      DO 8 I=1,N
8     XDISR(I+1)=XP(I+1)/XP(1)
      XDISR(1)=1.
      CALL SMULR(XDISR,N,XDISL,YDISD)
      DO 9 I=1,N
      XP(I)=XDISL(I)
9     YP(I)=YDISD(I)
      IF (M.LT.1) GO TO 12
      IF (XZ(1).EQ.0.) GO TO 1
      DO 10 I=1,M
10    XDISR(I+1)=XZ(I+1)/XZ(1)
      XDISR(1)=1.
      CALL SMULR(XDISR,M,XDISL,YDISD)
      DO 11 I=1,M
      XZ(I)=XDISL(I)
11    YZ(I)=YDISD(I)
C
C     XP,YP AND XZ,YZ NOW CONTAIN POLES AND ZEROS
C      (N POLES AND M ZEROS)
C
C    ORDER FACTORS.
C
12    CALL BLEND
      SIG=1.
      GAG = GA
```

```
      IF(SNG .GE. 0) GO TO 7
      GAG = -GAG
      SIG = 0
C
C     WRITE OUT POLES AND ZEROS TO BE USED
C
 7    WRITE(7,189)N
      DO 13 I=1,N
 13   WRITE(7,191)XP(I),YP(I)
      IF (M) 16,15,14
 14   WRITE(7,190)M
      DO 15 I=1,M
 15   WRITE(7,191)XZ(I),YZ(I)
 16   GAGI=1./GAG
C    WRITE OUT TIME DELAY AND GAIN CONSTANT.
      WRITE(7,169)TIME,GAGI
C
C     BACK UP ORIGIN IF THIS IS A MULTIPLE PLOT
C
      IF (NOPLOT) GO TO 17
      IF(MULT .EQ. 0) GO TO 20
      CALL PLOT(-4.-XOT,0,-3)
      GO TO 17
C
C     CALL OVERLAY TO DO INITIAL PLOTTING
C
 20   NUMPLOT = NUMPLOT + 1
 17   CALL OVERLAY(6HCCPCSD,1,1,0)
C
C     THE DEFAULT SIZES ARE
```

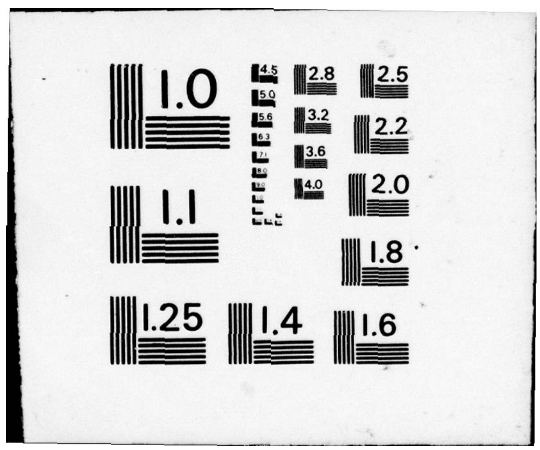
```
C          CALCULATION:  25 POINTS PER INCH OVER THE AREA CONSIDERED
C                        OR .1 WHICH EVER IS LARGER.
C          PRINTING STEP SIZE:  5 POINTS PER INCH
C      IN ANY CASE,  THE PRINTING STEP SIZE IS NEVER LESS THAN THE
C      CALCULATION STEP SIZE.
C
       IF(LL .GE. 6) GO TO 22
       NTERM = 1
       IF(LL .EQ. 2) NTERM = N+1
       DO 1700 I=1,NTERM
       IF(DEL(I) .EQ. 0) DEL(I)=AMAX1( .1,.04/V)
       IF(DELPR(I) .EQ. 0) DELPR(I) = .2/V
  1700 IF(DEL(I) .GT. DELPR(I))DELPR(I) = DEL(I)
       D = DEL(1)
C
C     IF OPTION 4 OR 5, GO TO OVERLAY TO CALCULATE AND PRINT LOCUS BRANCH
C      BUT FOR OPTION 5, FIRST PRINT BOUNDARIES
C
       IF(LL-4)22,19,18
C
C     WRITE OUT THE DESIRED ZETA,GAIN OF INTEREST AND ITS TOLERANCE.
C
 22    IF(ZETA.NE.0)WRITE(7,177)ZETA
       IF(GTOL.NE.(0.).AND.GA.NE.1.)WRITE(7,170)GANE,GTOL
       Z=GANE/GA
       A=GTOL/GA
       IF(GTOL.NE.0.)WRITE(7,178)Z,A
       IF(NOPLOT) GO TO 21
C      SET VALUES FOR PLOTTING IN ROOTS
       XX=XAX
```

```
      YY=YAX
      G=V
      FX=XOT
      FY=YOT
C
C     CALL OVERLAY TO CALCULATE THE CLOSED ROOTS AT DESIRED GAIN.
C
 21   IF((GTOL.NE.0).AND.(TIME.EQ.0))CALL OVERLAY(5HCCPCSD
     + ,1,2,0)
C
C     IF OPTION #6 OR #7, GO TO A NEW PROBLEM.
C     BUT FOR OPTION 7, FIRST PRINT BOUNDARIES.
C
      IF (LL.EQ.6) GO TO 1
C     WRITE OUT REGION OF CALCULATION
 18   WRITE(7,172)CC,AA,DD,BB
      IF(LL .EQ. 7) GO TO 1
C
C     CALL OVERLAY TO TRACE ROOT LOCUS AND PRINT AND PLOT AS DESIRED
C
 19   CALL OVERLAY(5HCCPCSD,1,3,0)
C
C     MOVE OFF PLOT AND DUMP FINAL PLOT BUFFER
C       (ORIGIN WILL HAVE TO BE RESET IF MULTIPLE PLOT WANTED)
C
      IF (NOPLOT) GO TO 1
      CALL PLOT(XOT+4.,0,-3)
      GO TO 1
C
C     FORMATS
```

```
C
 158   FORMAT (//,5X,31HROOT LOCUS USING OPTION NUMBER ,I2)
 1580  FORMAT (/,6X,4HCODE,/,8X,11H0-LOCUS PT.,10X,6H2-ZERO,15X,15H4-IMAG
      1INARY AXIS,/,8X,5H1-POLE,15X,11H3-BREAK PT.,10X,17H5-SENSITIVITY P
      2T.)
 159   FORMAT (//,6X,13HTIME DELAY (TIME) =,G10.4,22H, GAIN CONSTANT (GA)
      1 =,G16.8)
 170   FORMAT (/,6X,25HGAIN OF INTEREST (GANE) =,G13.4,5X,11H+OR- GTOL =,
      1G12.4)
 172   FORMAT (/,6X,27HREGION OF CALCULATION-REAL ,G10.3,2X,3HTO ,G10.3,/
      1,28X,5HIMAG ,G10.3,2X,3HTO ,G10.3)
174    FORMAT (1H1)
 177   FORMAT (/,6X,35HDAMPING FACTOR OF INTEREST (ZETA) =,G10.4)
 178   FORMAT (/,6X,29HSENSITIVITY OF INTEREST (K) =,G13.4,1X,11H+OR- KTO
      1L =,G12.4)
 179   FORMAT("0",8A10)
 180   FORMAT (/,5X,6HFIGURE,I4)
 181   FORMAT(8A10)
 189   FORMAT (//,4X,I3,1X,8HPOLES AT)
 190   FORMAT (//,4X,I3,1X,8HZEROS AT)
 191   FORMAT (8X,3HX =,G13.5,9X,3HY =,G13.5)
 200   FORMAT(" ENTER NAMELIST INPUD. START IN COLUMN 2 WITH A DOLLAR"/
      +" SIGN AND INPUD ($INPUD). PUT COMMA BETWEEN EACH ENTRY. END"/
      +" WITH A DOLLAR SIGN. (EXAMPLE: $INPUD LL=3,N=2,XP(1)=2,4$)"/)
 201   FORMAT(I3," PLOTS DISPOSED TO TERMINAL",I2)
 202   FORMAT(".ENTER THESI TITLE")
 203   FORMAT(".ENTER ITITL TITLE")
 3000  FORMAT(/"//##################################################
      +"/"    DO YOU WANT A LIST OF OPTIONS AVAILABLE FOR 'ROOTL'?"/
      +" TYPE 'YES' OR 'NO'.")
```

AD-A034 879 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 9/2
A CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN (CCPC--ETC(U)
DEC 76 F L O'BRIEN
UNCLASSIFIED GE/EE/76D-38 NL

2 OF 3
ADA034879


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3

```
3001 FORMAT(A10)
3002 FORMAT(" ##########################################################"/)
3003 FORMAT("    SORRY, TRY AGAIN...TYPE 'YES' OR 'NO'.")
3006 FORMAT(/" ********************************************************
    +"/" THE FOLLOWING OPTIONS ARE AVAILABLE FOR 'ROOTL':"//
    +"      0...TO STOP THE PROGRAM"/
    +"      1...COMPUTES EACH BRANCH OF THE LOCUS OF THE TRANSFER"/
    +"          FUNCTION OVER SPECIFIED BOUNDED REGION. TABULAR OUTPUT"/
    +"          FOR BRANCHES, PLUS PLOT, IF DESIRED."/
    +"      2...SIMILAR TO '1', EXCEPT STEP SIZE CAN BE SPECIFIED."/
    +"      3...SIMILAR TO '1', EXCEPT ZETA, DAMPING RATIO, SPECIFIED."/
    +"      4...TO INVESTIGATE PARTS OF LOCUS OF SPECIAL INTEREST."/
    +"      5...SIMILAR TO '4', WITH BOUNDARIES OF '1' - '3'; STARTING"/
    +"          POINT MUST BE SPECIFIED."/
    +"      6...TRUNCATED VERSION OF '1'. ONLY ROOTS OF INTEREST FOR A"/
    +"          SPECIFIED GAIN OF INTEREST (GANE) AND TOLERANCE (GTOL)"/
    +"          ARE LISTED. NO PLOT GENERATED. GTOL NOT EQUAL TO ZERO."/
    +"      7...SIMILAR TO '6'. FOR SPECIFIED DAMPING RATIO (ZETA), ROOT
    +S"/"           OF INTEREST ARE SPECIFIED. (GANE,GTOL, AND TIME ARE"/
    +"          NOT SPECIFIED."/
    +" ********************************************************"/)
3008 FORMAT(" ********************************************************
    +"/"    NOW YOU WILL HAVE THE CHOICE OF EITHER ENTERING THE"/
    +" INPUT DATA IN ONE NAMELIST STATEMENT (ASSUMING YOU KNOW EACH"/
    +" PARAMETER AND HAVE THE LIST PREPARED), OR FOLLOWING A STEP BY"/
    +" STEP PROCEDURE TO INPUT EACH PARAMETER. IF YOU INTEND TO USE"/
    +" ONE NAMELIST STATEMENT, TYPE 'ONE'; FOR A STEP BY STEP METHOD,"/
    +" TYPE 'STEP'.")
3009 FORMAT(A10)
3013 FORMAT(" ------------------------------------------------------
```

```
     +"//)
3010 FORMAT("    SORRY, TRY AGAIN...TYPE 'ONE' OR 'STEP'.")
3015 FORMAT(/" *********************IMPORTANT*********************"/
     +" BE SURE TO START EACH ENTRY IN COLUMN 2. BEGIN WITH '$INPUD';"/
     +" SKIP A SPACE; ENTER THE DATA; END WITH A DOLLAR SIGN, $. "/
     +" EXAMPLE:  $INPUD LL=3,N=2,M=3,XP(1)=2,4$  ALSO, IF YOU DO NOT"/
     +" WANT TO ENTER/CHANGE THE PARTICULAR PARAMETER SHOWN, ENTER"/
     +"  $INPUD $. TO STOP THE PROGRAM, ENTER ' $INPUD LL=0$."/
     +" ******************************************************"/)
3016 FORMAT(/" CHOOSE OPTION (0 THRU 7): PRECEDE ENTRY BY 'LL='."/)
3017 FORMAT(/" ENTER NUMBER OF POLES (DENOMINATOR ORDER) AS 'N=', AND"/
     +" NUMBER OF ZEROS (NUMERATOR ORDER) AS 'M='."/)
3018 FORMAT(/" ENTER DENOMINATOR FACTORS: REAL PARTS AS 'XP( )=',AND"/
     +" IMAGINARY PARTS AS 'YP( )='."/)
3019 FORMAT(/" ENTER NUMERATOR FACTORS: REAL PARTS AS 'XZ( )=', AND"/
     +" IMAGINARY PARTS AS 'YZ( )='."/)
3020 FORMAT(/" ENTER THE OPEN-LOOP GAIN, 'GA=': SIGN OF OPEN-LOOP GAI,
     +"/" 'SNG='; AND TIME DELAY, 'TIME='."/)
3021 FORMAT(/" ENTER THE RH BOUNDARY, 'AA='; UPPER BOUNDARY, 'BB=';"/
     +" LH BOUNDARY,'CC=-': AND LOWER BOUNDARY, 'DD=-'; EACH SPECIFYING
     +"/" THE BOUNDARY FOR THE REGION OF CALCULATION."/)
3022 FORMAT(/" ENTER BOUNDARY SCALE FACTOR, 'BOUND='."/)
3023 FORMAT(/" ENTER REAL AXIS LENGTH, 'IFOLD=': CALCULATION STEP SIZE,
     +"/" 'DEL='; AND PRINT STEP SIZE, 'DELPR='."/)
3024 FORMAT(/" ENTER CLOSED-LOOP GAIN OF INTEREST, 'GANE=': AND THE"/
     +" REGION OF INTEREST ABOUT GANE, 'GTOL='."/)
3025 FORMAT(/" ENTER REAL PART STARTING POINT, 'X='; IMAGINARY PART"/
     +" STARTING POINT, 'Y=': AND STARTING POINT TYPE NUMBER, 'LN='."/)
3026 FORMAT(/" ENTER NUMBER OF LOCUS POINTS COMPUTED, 'JK='; DAMPING"/
     +" RATIO, 'ZETA=': AND STEP-SIZE ON RADIAL SEARCH FOR GIVEN ZETA,"/
```

```
     +" 'RAD='."/)
3027 FORMAT(/" FOR PLOTS: ENTER FIGURE NUMBER, 'FPN=': FIRST TITLE FLAG
     +,"/" 'ITHES=': SECOND TITLE FLAG, 'ITITL=': AND THIRD TITLE FLAG,"
     +/" 'ISTAM='."/)
3028 FORMAT(/" FOR PLOTS: ENTER MULTIPLE PLOT FLAG, 'MULT=': PLOTTING"/
     +" TERMINAL INDICATOR (7 FOR 3B AT BLDG 640), 'IPLOT=': AND DISPOSE
     +"/" LIMIT FOR PLOTS, 'MAXDSP='."/)
1249 WRITE(9,3050)
3050 FORMAT(//"          -------END OF PROGRAM 'ROOTL'------          "//)
     RETURN
     END


     SUBROUTINE BANG
C
C    THIS SUBROUTINE COMPUTES THE ANGLE OF A VECTOR FROM ITS RECTANGULA
C    COMPONENTS, A, B, BY FINDING ARCTAN(B/A)
C
     COMMON/ANGSS/A,B,ANG,PI
     ANG=0.
     IF(A) 2,1,3
1    IF(B) 5,6,4
2    BAA=B/A
     ANG=ATAN(BAA)+PI
     GO TO 6
3    BAA=B/A
     ANG=ATAN(BAA)
     GO TO 6
```

```
4       ANG=PI/2.
        GO TO 6
5       ANG=-PI/2.
5       RETURN
        END


        SUBROUTINE BOX
C
C     THIS SUBROUTINE REDUCES AN ANGLE TO ITS PRIMARY VALUE BETWEEN
C     PLUS AND MINUS PI (I.E. REMOVES MULTIPLES OF PI)
C
        COMMON/ANGSS/A,B,ANG,PI
        LM=ABS(ANG)/6.283185
        T=LM
        IF (ANG) 1,5,2
1       ANG=ANG+T*2.*PI
        GO TO 3
2       ANG=ANG-T*2.*PI
3       T=1.
        IF (ANG+PI) 1,5,4
4       IF (ANG-PI) 5,2,2
5       RETURN
        END
```

```
      SUBROUTINE GANG1
C
C    THIS SUBROUTINE COMPUTES PHASE ANGLE OF TEST POINT
C     GANG=ANG((T-Z(1))+(T-Z(2))+...+(T-Z(M))) -
C          ANG((T-P(1))+(T-P(2))+...+(T-P(N))) - PI*SIG - TAU*YT
C    WHERE: T=(XT,YT)                            TEST POINT
C          Z(J)=(XZ(J),YZ(J))                      ZERO
C          P(I)=(XP(I),YP(I))                      POLE
C    NOTE: SIG=1.    FOR GA>0.
C    NOTE: SIG=0.    FOR GA<0.
C    AS DEFINED ABOVE, GANG=0 DEG. - - - - LOCUS POINT FOR GA>0.
C    AS DEFINED ABOVE, GANG=180 DEG. - - - LOCUS POINT FOR GA<0.
C
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGSS/A,B,ANG,PI
      GANG=0.
      DO 2 I=1,N
      A=X-XP(I)
      B=Y-YP(I)
      CALL BANG
      GANG=GANG-ANG
      IF (I-M) 1,1,2
1     A=X-XZ(I)
      B=Y-YZ(I)
      CALL BANG
      GANG=GANG+ANG
2     CONTINUE
      ANG=GANG-PI*SIG-TIME*Y
      CALL BOX
      GANG=ANG
```

```
      RETURN
      END


      SUBROUTINE BLEND
C
C     THIS SUBROUTINE ORDERS ROOTS PRIOR TO PLOTTING LOCUS
C
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      DATA E/.0001/
C
C     IF POLE OR ZERO IS ALMOST REAL, MAKE IT REAL.
C     SETS IMAGINARY PARTS <10**-4 EQUAL TO 0.
C
      DO 1 I=1,N
      IF (ABS(YP(I)).LE.E) YP(I)=0.
      IF (M.LT.I) GO TO 1
      IF (ABS(YZ(I)).LE.E) YZ(I)=0.
1     CONTINUE
C
C     ARRANGE COMPLEX POLE PAIRS IN SEQUENCE
C
      IF (N.LE.2) GO TO 8
      NNN=N-1
C
C      GET CONJUGATE PAIRS OF POLES ADJACENT
C
      DO 4 I=1,NNN
```

```
      IF (YP(I).EQ.0.) GO TO 4
      MMM=I+1
      DO 3 JM=MMM,N
      IF ((ABS(XP(I)-XP(JM)).GT.E).OR.(ABS(YP(I)+YP(JM)).GT.E)) GO TO 3
      IIM=JM-I
      DO 2 JJM=1,IIM
      IPM=JM-JJM
      XP(IPM+1)=XP(IPM)
2     YP(IPM+1)=YP(IPM)
      YP(I+1)=-YP(I)
3     CONTINUE
4     CONTINUE
C
C     MOVE ALMOST IDENTICAL POLES ADJACENT
C
      DO 7 I=1,NNN
      MMM=I+1
      DO 6 JM=MMM,N
      IF ((ABS(XP(I)-XP(JM)).GT.E).OR.(ABS(YP(I)-YP(JM)).GT.E)) GO TO 6
      IIM=JM-I
      DO 5 JJM=1,IIM
      IPM=JM-JJM
      XP(IPM+1)=XP(IPM)
5     YP(IPM+1)=YP(IPM)
6     CONTINUE
7     CONTINUE
8     IF (M.LE.2) RETURN
      NNN=M-1
C
C     GET CONJUGATE PAIRS OF ZEROS ADJACENT
```

```
C
      DO 11 I=1,NNN
      IF (YZ(I).EQ.0.) GO TO 11
      MMM=I+1
      DO 10 JM=MMM,M
      IF ((ABS(XZ(I)-XZ(JM)).GT.E).OR.(ABS(YZ(I)+YZ(JM)).GT.E)) GO TO 10
      IIM=JM-I
      DO 9 JJM=1,IIM
      IPM=JM-JJM
      XZ(IPM+1)=XZ(IPM)
9     YZ(IPM+1)=YZ(IPM)
      YZ(I+1)=-YZ(I)
10    CONTINUE
11    CONTINUE
C
C     MOVE ALMOST IDENTICAL ZEROS ADJACENT
C
C
      DO 14 I=1,NNN
      MMM=I+1
      DO 13 JM=MMM,M
      IF ((ABS(XZ(I)-XZ(JM)).GT.E).OR.(ABS(YZ(I)-YZ(JM)).GT.E)) GO TO 13
      IIM=JM-I
      DO 12 JJM=1,IIM
      IPM=JM-JJM
      XZ(IPM+1)=XZ(IPM)
12    YZ(IPM+1)=YZ(IPM)
13    CONTINUE
14    CONTINUE
      RETURN
```

```
      END

      SUBROUTINE SMULR (COE,N1,ROOTR,ROOTI)
C
C     THIS SUBROUTINE IS USED IN FACTORING THE INPUT NUMERATOR AND
C     DENOMINATOR POLYNOMIALS WHEN THE INPUT TRANSFER FUNCTION IS IN
C     POLYNOMIAL FORM.
C
      DIMENSION COE(1),ROOTR(1),ROOTI(1)
      N4=0
      I=N1+1
1     IF (COE(I)) 3,2,3
2     N4=N4+1
      ROOTR(N4)=0.0
      ROOTI(N4)=0.0
      I=I-1
      IF (N4-N1) 1,20,1
3     CONTINUE
4     AXR=0.8
      AXI=0.0
      L=1
      N3=1
      ALP1R=AXR
      ALP1I=AXI
      M=1
      GO TO 25
5     BET1R=TEMR
```

```
      BET1I=TEMI
      AXR=0.85
      ALP2R=AXR
      ALP2I=AXI
      M=2
      GO TO 26
6     BET2R=TEMR
      BET2I=TEMI
      AXR=0.9
      ALP3R=AXR
      ALP3I=AXI
      M=3
      GO TO 26
7     BET3R=TEMR
      BET3I=TEMI
8     TE1=ALP1R-ALP3R
      TE2=ALP1I-A_P3I
      TE5=ALP3R-ALP2R
      TE6=ALP3I-ALP2I
      TEM=TE5*TE5+TE6*TE6
      TE3=(TE1*TE5+TE2*TE6)/TEM
      TE4=(TE2*TE5-TE1*TE6)/TEM
      TE7=TE3+1.0
      TE9=TE3*TE3-TE4*TE4
      TE10=2.0*TE3*TE4
      DE15=TE7*BET3R-TE4*BET3I
      DE16=TE7*BET3I+TE4*BET3R
      TE11=TE3*BET2R-TE4*BET2I+BET1R-DE15
      TE12=TE3*BET2I+TE4*BET2R+BET1I-DE15
      TE7=TE9-1.0
```

```
      TE1=TE9*BET2R-TE10*BET2I
      TE2=TE9*BET2I+TE10*BET2R
      TE13=TE1-BET1R-TE7*BET3R+TE10*BET3I
      TE14=TE2-BET1I-TE7*BET3I-TE10*BET3R
      TE15=DE15*TE3-DE16*TE4
      TE16=DE15*TE4+DE16*TE3
      TE1=TE13*TE13-TE14*TE14-4.0*(TE11*TE15-TE12*TE16)
      TE2=2.0*TE13*TE14-4.0*(TE12*TE15+TE11*TE16)
      TEM=SQRT(TE1*TE1+TE2*TE2)
      IF (TE1) 9,9,10
9     TE4=SQRT(0.5*(TEM-TE1))
      TE3=0.5*TE2/TE4
      GO TO 13
10    TE3=SQRT(0.5*(TEM+TE1))
      IF (TE2) 11,12,12
11    TE3=-TE3
12    TE4=0.5*TE2/TE3
13    TE7=TE13+TE3
      TE8=TE14+TE4
      TE9=TE13-TE3
      TE10=TE14-TE4
      TE1=2.0*TE15
      TE2=2.0*TE16
      IF (TE7*TE7+TE8*TE8-TE9*TE9-TE10*TE10) 14,14,15
14    TE7=TE9
      TE8=TE10
15    TEM=TE7*TE7+TE8*TE8
      TE3=(TE1*TE7+TE2*TE8)/TEM
      TE4=(TE2*TE7-TE1*TE8)/TEM
      AXR=ALP3R+TE3*TE5-TE4*TE6
```

```
      AXI=ALP3I+TE3*TE6+TE4*TE5
      ALP4R=AXR
      ALP4I=AXI
      M=4
      GO TO 26
 16   IF (ABS(HELL)+ABS(BELL)-1.0E-20) 18,19,17
 17   TE7=ABS(ALP3R-AXR)+ABS(ALP3I-AXI)
      IF (TE7/(ABS(AXR)+ABS(AXI))-1.0E-7) 19,19,18
 18   N3=N3+1
      ALP1R=ALP2R
      ALP1I=ALP2I
      ALP2R=ALP3R
      ALP2I=ALP3I
      ALP3R=ALP4R
      ALP3I=ALP4I
      BET1R=BET2R
      BET1I=BET2I
      BET2R=BET3R
      BET2I=BET3I
      BET3R=TEMR
      BET3I=TEMI
      IF (N3-100) 8,19,19
 19   N4=N4+1
      ROOTR(N4)=ALP4R
      ROOTI(N4)=ALP4I
      N3=0
      IF (N4-N1) 21,20,20
 20   RETURN
 21   IF (ABS(ROOTI(N4))-1.0E-5) 4,4,22
 22   GO TO (23,4), L
```

```
23    AXR=ALP1R
      AXI=-ALP1I
      ALP1I=-ALP1I
      M=5
      GO TO 26
24    BET1R=TEMR
      BET1I=TEMI
      AXR=ALP2R
      AXI=-ALP2I
      ALP2I=-ALP2I
      M=6
      GO TO 26
25    BET2R=TEMR
      BET2I=TEMI
      AXR=ALP3R
      AXI=-ALP3I
      ALP3I=-ALP3I
      L=2
      M=3
26    TEMR=COE(1)
      TEMI=0.0
      DO 27 I=1,N1
      TE1=TEMR*AXR-TEMI*AXI
      TEMI=TEMI*AXR+TEMR*AXI
27    TEMR=TE1+COE(I+1)
      HELL=TEMR
      BELL=TEMI
      IF (N4) 28,30,28
28    DO 29 I=1,N4
      TEM1=AXR-ROOTR(I)
```

```
      TEM2=AXI-ROOTI(I)
      TE1=TEM1*TEM1+TEM2*TEM2
      TE2=(TEMR*TEM1+TEMI*TEM2)/TE1
      TEMI=(TEMI*TEM1-TEMR*TEM2)/TE1
29    TEMR=TE2
30    GO TO (5,6,7,16,24,25), M
      END


      SUBROUTINE ANG1
C
C    THIS SUBROUTINE COMPUTES MAGNITIUDE AND PHASE ANGLE
C    (SEE SUBROUTINE GANG1)
C    WHERE "MAGNITUDE" IS
C    GAIN = GA*(!T-P(1)!*!T-P(2)!*...*!T-P(N)!)
C             /(!T-Z(1)!*!T-Z(2)!*...*!T-Z(M)!)
C    NOTE: GA USED HERE IS INVERSE OF INPUT VALUE
C    !T-P(I)!=SQRT(A**2+B**2)
C
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGSS/A,B,ANG,PI
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      GANG=0.
      GAIN=GA
      IF(TIME.NE.0.) GAIN=GAIN*(2.71828183**(TIME*X))
      FX=0.
      FY=-TIME
      DO 1 I=1,N
```

```
      A=X-XP(I)
      B=Y-YP(I)
      G=A*A+B*B
      IF (G .GE. E*E) GO TO 5
      J=5
      GO TO 1
5     FX=FX+B/G
      FY=FY-A/G
      CALL BANG
      GANG=GANG-ANG
1     GAIN=GAIN*SQRT(G)
      IF (M) 4,4,2
2     DO 3 I=1,M
      A=X-XZ(I)
      B=Y-YZ(I)
      G=A*A+B*B
      IF(G .GE. E*E) GO TO 6
      GAIN = 0
      GO TO 3
6     FX=FX-B/G
      FY=FY+A/G
      CALL BANG
      GANG=GANG+ANG
      GAIN=GAIN/SQRT(G)
3     CONTINUE
4     ANG=GANG-PI*SIG-TIME*Y
      CALL BOX
      GANG=ANG
      RETURN
      END
```

```
*     SUBROUTINE DSP            DISPOSE PLOT FILE
*
*          AFIT VERSION FOR SCOPE 3.4   APRIL 74
*
*          CALL DSP(ID)
*          CALL DSP(ID,LAST)
*
*          WHERE   ID = EITHER: A TWO CHARACTER (HOLLERITH CONSTANT)
*                               GIVING THE SCOPE 3.4 TERMINAL ID
*                           OR: AN INTEGER CONSTANT GIVING THE
*                               OLD TERMINAL ID. SEE THE IDTABLE BELOW
*                               FOR CORRESPONDENCE.
*
*                LAST = AN OPTIONAL 2ND ARGUMENT. IF LAST IS NOT USED OR
*                       IF IT HAS ZERO VALUE, THE PLOT FILE IS RE-OPENED
*                       AFTER THE DISPOSE AND A BANNER IS DRAWN WITH
*                       THE JOBNAME, DATE AND TIME.
*                       IF LAST .NE.0 THE PLOT FILE IS NOT RE-OPENED.
*
*          IN THIS VERSION OF DSP AN ENDING BANNER IS NOT DRAWN
*          BEFORE THE DISPOSE.
*
         IDENT   DSP
         LIST M,S
         SPACE   1
CALL     MACRO   SUB,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,TRAC
         LOCAL   SKIP,PARLST
         SPACE   1
         IF      -DEF,SUB,1
         EXT     SUB
```

```
         EQ     SKIP
PARLST   BSS    0
         IFC    NE,**A1*
         VFD    60/A1
         IFC    NE,**A2*
         VFD    60/A2
         IFC    NE,**A3*
         VFD    60/A3
         IFC    NE,**A4*
         VFD    60/A4
         IFC    NE,**A5*
         VFD    60/A5
         IFC    NE,**A6*
         VFD    60/A6
         IFC    NE,**A7*
         VFD    60/A7
         IFC    NE,**A8*
         VFD    60/A8
         IFC    NE,**A9*
         VFD    60/A9
         IFC    NE,**A10*
         VFD    60/A10
         ENDIF
         DATA   0
SKIP     SA1    PARLST
+        RJ     SUB
         IFC    EQ,**TRAC*,1
-        VFD    12/*-TRACE.,18/TRACE.
         IFC    NE,**TRAC*,1
-        VFD    12/*-TRACE.,18/TRAC
```

```
CALL      ENDM
          SPACE  1
          MACRO  TRACEM,TRAC,NAME,INTNAME
          SPACE  1
          ENTRY  NAME
X         MICRO  1,,=NAME=               REMOVE POSSIBLE =
SKP1      IFC    NE,*TRAC*TRACE.*
          IF     -DEF,TRACE.,1
TRACE.    BSS    0
SKP1      ENDIF
TRAC      VFD    42/7L"X"
          VFD    18/NAME
          IF     -DEF,TEMPA0.,1
TEMPA0.   BSS    1
          IFC    NE,*NAME*INTNAME*,1
NAME      BSS    0
          IFC    EQ,**INTNAME*,1
          DATA   0
TRACEM    ENDM
          SPACE  1
FTNLNK    MACRO  NOARG                   NOARG LIMIT IS 6.
          SPACE  1
          IFLE NOARG,6
          SX6  A0                        REVISED FOR FORTRAN
          SA0  A1                        EXTENDED LINKEAGES.
          SA6  TEMPA0.                   SAVE A0.
          IFC  NE,**NOARG*
          IFNE NOARG,0
          IFGE NOARG,1
          SB1  X1                        ARG1.
```

```
         IFGE  NOARG,2
         SA1   A0+1
         SB2   X1                  ARG2.
         IFGE  NOARG,3
         SA1   A0+2
         SB3   X1                  ARG3.
         IFGE  NOARG,4
         SA1   A0+3
         SB4   X1                  ARG4.
         IFGE  NOARG,5
         SA1   A0+4
         SB5   X1                  ARG5.
         IFGE  NOARG,6
         SA1   A0+5
         SB6   X1                  ARG6.
         ENDIF
         IFGT  NOARG,6
*********************************************************************
*                                                                   *
 ***THIS MACRO WILL PROCESS A MAX. OF SIX ARGUMENTS*****************
*                                                                   *
*********************************************************************
         ENDIF
FTNLNK   ENDM
         SPACE  1
         TRACEM DSP
         SPACE  1
         FTNLNK  2
         SPACE  1
         MX0  12                   ..
```

```
         SA4  31                            ..
         BX3  X0*X4                         EXTRACT TERMINAL ID.
         NZ     X3,ARGC       JUMP IF NOT INTEGER ID
         MX0    56            MASK INTEGER ID
         BX3    -X0*X4
         SA3    X3+IDTABLE    GET NEW CHARACTER ID FROM TABLE
ARGC     SA2  DISP                          ..
         MX0  36                            ..
         LX3  24                            POSITION TERMINAL ID.
         BX6  X0*X2                         CLEAR DISPOSE COMPLETION BIT.
         IX6  X6+X3                         INSERT TERMINAL NUMBER.
         SA6  DISP                          ..
         MX7    0
         EQ     B0,B2,GO      JUMP IF ONLY ONE ARG
         SA3    B2
         BX7    X3
GO       SA7    LAST          STORE 2ND ARG IN LAST
         SPACE  1
         SA4  JOBNAME                       ..
         NZ   X4,ID5                        BYPASS IF WE ALREADY HAVE JOBNAME.
         SPACE  1
         SA5  JOBID                         SET UP PP CALL
         BX7  X5                            TO GET JOBNAME.
ID3      SA4  1                             ..
         NZ   X4,ID3                        LOOP IF RA+1 IS BUSY.
         SA7  1                             ISSUE REQUEST.
ID3A     SA4  1                             ..
         NZ   X4,ID3A                       LOOP UNTIL ACCEPTED.
ID4      SA5  JOBNAME                       ..
         NZ   X5,ID5                        CONTINUE WHEN JOBNAME RETURNED.
```

```
         RJ   RELEAS                 GO INTO RECALL.                          010280
         EQ   ID4                    ..                                       010290
ID5      EQ   *+1                    ..                                       010300
         SPACE  1                                                             010310
*  DELETE CARDS DSP00033 THRU DSP00039 TO OMIT ENDING BANNER                  010320
DISP     BSS  0                                                               010330
         SPACE  1                                                             010340
         DISPOSE   PLOT,PT=IXX,RECALL                                         010350
         SPACE  1                                                             010360
         SA3    LAST            OMIT OPEN BANNER                              010370
         NZ     X3,EXIT            IF LAST.NE.0                               010380
         CALL PLOT,=0.25,=10.0,=3                                             010390
         CALL PLOT,=0.25,=8.555,=2                                            010400
         CALL PLOT,=0,=8.555,=2                                               010410
         CALL SYMBOL,=0.1875,=8.805,=0.105,OPNPLT,=90.0,=9                    010420
         CALL PLOT,=0,=0,=3                                                   010430
         CALL GILSID,JOBNAME                                                  010440
         SPACE  1                                                             010450
EXIT     SA5  TEMPA0.                ..                                  AFIT-010460
         SA0  X5                     RESET A0                                 010470
         EQ   DSP                    AND EXIT.                                010480
         SPACE  1                                                             010490
RELEAS   DATA 0                      ..                                       010500
         SX6  2203148                .SET RCL (RELEASE CP)                    010510
         LX6  42                     .LEFT JUSTIFY                            010520
         SA6  1                      .SET IN RA+1 (RECALL)                    010530
         EQ   RELEAS                 ..                                       010540
         SPACE  1                                                             010550
*ENDDSP  DIS  1, END  DSP    **DELETE THIS CARD FOR AFIT-DSP             AFIT-010560
OPNPLT   DIS  1,OPEN PLOT                                                     010570
```

```
JOBID     VFD  18/3HPID,24/0,18/JOBNAME
JOBNAME   DATA 0
LAST      DATA    0
IDTABLE   DATA    2LAB          T0   ASD/AD   B675 (NO ONLINE PLOTTER)
          DATA    2LBA          T1   AFFDL    B45
          DATA    2LAD          T2   ASD/ENF  B17
          DATA    2LAC          T3   ASD/XR   B52
          DATA    2LAE          T4   AFAPL    B18
          DATA    2LAF          T5   AMRL     B441
          DATA    2LBC          T6   AFWAL    B450
          DATA    2LBB          T7   AFIT     B640
          DATA    2LBD          T8   AFAL     B22
          END
      OVERLAY(CCPCSD,1,1)
      PROGRAM ROOT11
C
C     THIS OVERLAY SETS UP THE CALCOMP PLOT
C
      COMMON /MAIN01/BOUND,DZW(8),FPN,IFOLD,IFPN,ISTAM,ITITL,ITHES,RAD
     +  ,THESI(8),ZETA,MULT
      COMMON /MAIN012/GANE,NOPLOT,STOL,IIV,LL,V,XAX,YAX,DEL(50),
     +  XDISL(50),XDISR(50),YDISD(50),YDISU(50),XOT,YOT,CB
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGSS/A,B,ANG,PI
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      COMMON /ANGJU/ AA,BB,CC,DD,D,DA, JK,SANG,CA
      LOGICAL NOPLOT
      DIMENSIONRZQ(50),IID(50),JJD(50),XOTVAL(5)
      DATA XOTVAL /6.,13.,19.,23.,30./
C
```

```
C     SKIP ALL PLOTTING AND PLOT SETUP IF THIS IS TO BE
C     A LOCUS ON AN EXISTING PLOT
C
      IF(MULT .NE. 0) GO TO 69
C
C     SCALE S-PLANE BOUNDARIES
C
      AA=AA*BOUND
      BB=BB*BOUND
      CC=CC*BOUND
      DD=DD*BOUND
      AAB=AA-CC
      BBD=BB-DD
C
C     DETERMINE PLOT SIZE AND ORIENTATION AND PRINT OUTSIDE TITLE
C
      YOT=9.
      XOT = XOTVAL(MAXO(MINO(4,IFOLD),0)+1)
      IF (IFOLD.NE.0) GO TO 20
      IF (BBD.GT.AAB) GO TO 20
      XOT=9.
      YOT=6.
      IF (NOPLOT) GO TO 21
      IF(ITHES .LT. 0) CALL SYMBOL(9.375,6.0,.105,THESI,-90.,80)
      GO TO 21
 20   IF (NOPLOT) GO TO 21
      IF(ITHES .LT. 0) CALL SYMBOL(0,9.375,.105,THESI,0,80)
C
C     READJUST BB, CC, AND DD TO FIT THE LENGTH AND WIDTH OF THE PLOT
C
```

```
21    WV=(YOT-2.)/(XOT-1.)
      CBZ=WV*AAB
      CC=AA-BBD/WV
      IF (BBD.GE.CBZ) GO TO 22
      BB=BB+.5*(CBZ-BBD)
      DD=DD-.5*(CBZ-BBD)
      CC=AA-AAB
C
C     COMPUTE SCALE FACTOR AND RECIPROCAL
C
C    SCALE FACTOR IN INCHES PER UNIT
22    V=(XOT-1.)/(AA-CC)
C    RECIPROCAL SCALE FACTOR IN UNITS PER INCH
      VW=(AA-CC)/(XOT-1.)
      XAA=XOT-.5
      YBB=YOT-.75
C
C     NUMBER OF DASHES IN X AND Y AXES
C
      IAY=(YOT-2.)*10.+E
      IAX=(XOT-1.)*10.+E
C
C     COMPUTE COORDINATES OF (0,0)
C
      YAX=.5-V*CO
      XAX=1.25-V*DD
      IF ((LL.EQ.4).OR.(LL.EQ.5)) GO TO 750
      IF(NOPLOT) GO TO 25
C
C     DRAW OUTLINE
```

```
C
      CALL PLOT (0.0,0.0,3)
      CALL PLOT (0.0,YOT,2)
      CALL PLOT (XOT,YOT,2)
      CALL PLOT (XOT,0.0,2)
      CALL PLOT (0.0,0.0,2)
      XYAX = AINT(10.*(YAX-.5))
      YXAX = AINT(10.*(XAX-1.25))
C
C     CENTER AND PRINT FIGURE NUMBER
C
      CBZ=XOT
      IF(IFPN.EQ.0) GO TO 223
      AAB=CBZ/2.-1.54
      CALL SYMBOL (AAB,.20,.14,22HFIGURE     - ROOT LOCUS,0.0,22)
      CALL NUMBER (AAB+.98,.20,.14,FPN,0.0,-1)
C
C     CENTER AND PRINT INSIDE TITLE
C
 223  IF(ITITL.LE.0)GO TO 24
      TITSIZ = .175
      IF(ITITL*.175 .GT. XOT-.25) TITSIZ=(XOT-.25)/ITITL
      AAB = CBZ/2.-(TITSIZ*.5*ITITL)
      CALL SYMBOL(AAB,YOT-.5,TITSIZ,DZW,0,ITITL)
C
C     PRINT SCALE FACTOR AND BOX AROUND IT
C
24    CBZ=VW
      JK=4
      CALL DNTER (CBZ,JK)
```

```
      ASP=CBZ*.070
      CBZ=VW
      CALL CNTER (CBZ)
      ASP=ASP+CBZ*.070
      CEA=CBZ
      ASP=XOT-1.73-ASP
      CALL PLOT (ASP,1.03,3)
      CALL PLOT (XOT-.5,1.03,2)
      CALL PLOT (XOT-.5,1.165,2)
      CALL PLOT (ASP,1.165,2)
      CALL PLOT (ASP,1.03,2)
      ASP=ASP+.03
      CALL SYMBOL (ASP,1.055,.07,7HSCALE- ,0.0,7)
      ASP=ASP+.49
      CALL NUMBER (ASP,1.055,.07,VW,0.0,JK)
      ASP=XOT-1.27
      CALL SYMBOL (ASP,1.055,.07,11H UNITS/INCH,0.0,11)
C
C     SEARCH ALONG RADIAL LINE FOR SOLUTION AT DESIRED ZETA
C     AND COMPUTE GAIN FACTOR.
C
 25   IF (.NOT.((LL.EQ.3).OR.(LL.EQ.7)).OR.(ZETA.GT..91)) GO TO 31
      XWORK = -ZETA
      YWORK=SQRT(1.-ZETA*ZETA)
      XX = X = XWORK*RAD
      YY = Y = YWORK*RAD
      CALL GANG1
      D = DI = DEL(1)
      IF(D .EQ. 0) D = DI = AMAX1(.1,.04*VW)
25    X=XX+D*XWORK
```

```
      Y=YY+D*YWORK
      CB=GANG
      CALL GANG1
      IF (CB*GANG) 28,29,27
27    IF ((X .LT.CC).OR.( Y.GT.BB)) GO TO 31
      XX=X
      YY=Y
      GO TO 26
28    IF(ABS(GANG) .GT. 2.)GO TO 25
      GANG = CB
      D=D/4.
      IF (( DI/256.-D).LT.0.) GO TO 26
29    CALL ANG1
      J=1
      GANE=GAIN
      GTOL=.1*GANE
31    IF (NOPLOT) GO TO 32
      IF (GTOL.EQ.0.) GO TO 32
C
C     COMPUTE SIZE OF GAIN BOX
C
       CBZ=GANE
      CALL CNTER (CBZ)
      JK=2
      XWORK = CBZ
      CBZ=GANE
      CALL DNTER (CBZ,JK)
      CBZ = CBZ+XWORK
      XWORK=.825
      IF(GA .NE. 1.) XWORK=.85
```

```
      ASP=XWORK+CBZ*.070
C
C     PLOT GAIN BOX AND GAIN OF INTEREST
C
      CALL PLOT (.5,1.03,3)
      CALL PLOT (ASP,1.03,2)
      CALL PLOT (ASP,1.165,2)
      CALL PLOT (.5,1.165,2)
      CALL PLOT (.5,1.03,2)
      CALL SYMBOL (.565,1.09,.07,2,0.0,-1)
      CALL SYMBOL (.6,1.055,.07,2H K,0.0,2)
      IF(GA.NE.1.)CALL SYMBOL (.74,1.055,.035,1HN,0.0,1)
      CALL SYMBOL(XWORK-.08,1.055,.07,1H=,0,1)
      CALL NUMBER(XWORK,1.055,.07,GANE,0,JK)
32    IF (ISTAM.LT.0) GO TO 50
      IF (NOPLOT) GO TO 320
C
C     DRAW BOX FOR TRANSFER FUNCTION
C
      CALL PLOT (.5,.5,3)
      CALL PLOT (XOT-.5,.5,2)
      CALL PLOT (XOT-.5,1.,2)
      CALL PLOT (.5,1.,2)
      CALL PLOT (.5,.5,2)
320   IF (ISTAM.GT.0) GO TO 48
      IF (NOPLOT) GO TO 50
C
C     SET UP TO PRINT TRANSFER FUNCTION
C     DETERMINE SIZE OF TRANSFER FUNCTION FOR PLOT
C
```

```
      JROOT=0
      IIM=0
      KKV=0
      KROOT=1
      JJN=N
      DO 33 I=1,N
      YDISD(I)=YP(I)
33    XDISL(I)=XP(I)
34    CALL GROUP (XDISL,XDISR,YDISD,YDISJ,IID,KKV,JJN)
C
C     IID= 0 - DAMPING TERM IS ZERO
C     IID= 1 - ROOT AT ORIGIN
C     IID= 2 - REPEATED ROOT AT ORIGIN
C     IID= 4 - REAL ROOT
C     IID= 5 - REPEATED REAL ROOT
C     IID= 6 - PURE IMAGINARY COEFFICIENT
C     IID= 7 - DAMPING TERM COEFFICIENT
C     IID= 8 - ZEROETH ORDER COEFFICIENT OF COMPLEX ROOT
C
      XWORK=0
      DO 35 I=KROOT,KKV
      AAB=IID(I)
C
C     AT ORIGIN
      IF ( AAB  .LT.3) GO TO 35
C
C     COMPLEX PAIR
      IF ( AAB .EQ. 8.) AAB=0
C
C     IMAGINARY PAIR
```

```
      IF ( AAB .EQ. 6.) AAB=5.
      CBZ=XDISR(I)
      CALL CNTER (CBZ)
      ASP=CBZ
      CBZ=XDISR(I)
      JK=3
      CALL DNTER (CBZ,JK)
      JJD(I)=JK
      AAB=AAB+CBZ+ASP
C
C    # OF SPACES FOR COEFFICIENT "XDISR(I)"
      RZO(I)=CBZ+ASP
 35   XWORK=XWORK+AAB
      EEA=XWORK
      IF (IIM.EQ.1) GO TO 37
      IIM=1
C    LENGTH OF DENOMINATOR
      EEP=XWORK
      EEA=0.
      JROOT=KKV
      IF (M.EQ.0) GO TO 37
      JJN=M
      KROOT=KKV+1
      DO 36 I=1,M
      XDISL(I)=XZ(I)
36    YDISD(I)=YZ(I)
      GO TO 34
C    LENGTH OF NUMERATOR
37    EEA=EEA+1.
C
```

```
C     DETERMINE SIZE FOR PLOTTING TIME DELAY IF APPROPRIATE
C
      IF (TIME.LT.0.) GO TO 38
      CBZ=-TIME
      CALL CNTER (CBZ)
      TIMA=CBZ
      CBZ=-TIME
      JK=3
      CALL DNTER (CBZ,JK)
      TIMA=TIMA+CBZ
      NID=JK
      EEA=EEA+TIMA+2.
C
C     SELECT LETTER SIZE
C
38    ASP=EEP
      IF (EEA.GT.EEP) ASP=EEA
      CBZ=ASP*.105
      IF((XOT-1.945).GT.CBZ)GO TO 40
      CBZ=ASP*.07
      IF((XOT-1.945).GT.CBZ)GO TO 40
      IF((XOT-1.).GT.CBZ)GO TO 39
      CBZ=.035*ASP
C     DETERMINE IF TRANSFER FUNCTION FITS BOX
      IF((XOT-1.).LT.CBZ)GO TO 50
C
C     USE FULL LABEL
C     THERE IS ROOM OVER THE TRANSFER FUNCTION
C
 39   VIX=(XOT-3.57)/2.
```

```
      CALL SYMBOL (VIX,.843,.105,34HTHE OPEN LOOP TRANSFER FUNCTION IS,0
     1.0,34)
      VIY=.67
      VIX=XOT/2.
      AAB=VIX-CBZ/2.
      IF (AAB.LT..5) GO TO 50
      GO TO 41
C
C     USE CONDENSED LABEL
C     PUT IT IN FRONT OF THE TRANSFER FUNCTION
C
 40   VIX=(XOT-CBZ)/2.-.472
      VIY=.75
      CALL SYMBOL (VIX,.7,.105,3HG(S)H(S)=,0.0,9)
      VIX=VIX+CBZ/2.+.945
C
C     NOW PRINT TRANSFER FUNCTION
C
41    BBD=CBZ/ASP
      AAB=VIX-CBZ/2.
      CALL PLOT (AAB,VIY,3)
      AAB=VIX+CBZ/2.
      CALL PLOT (AAB,VIY,2)
      CBZ=BBD
      AAB=VIX-EEA*BBD/2.
      VIX=VIX-EEP*BBD/2.
      BBD=VIY
      VIY=BBD-1.6*CBZ
      DO 47 I=1,KKV
      JJN=IID(I)
```

```
      IF (JJN.EQ.1.OR.JJN.EQ.2) GO TO 44
      IF (JJN.EQ.6.OR.JJN.EQ.7) GO TO 42
      CALL SYMBOL (VIX,VIY,CBZ,2H(S,0.0,2)
      VIX=VIX+2.*CBZ
      IF (JJN.EQ.8.OR.JJN.EQ.0) GO TO 45
42    IF (XDISR(I).LT.0.) GO TO 43
      CALL SYMBOL (VIX,VIY,CBZ,1H+,0.0,1)
      VIX=VIX+CBZ
43    JK=JJD(I)
      EEP=XDISR(I)
      CALL NUMBER (VIX,VIY,CBZ,EEP,0.0,JK)
      VIX=VIX+CBZ*RZQ(I)
      IF (JJN.EQ.8) GO TO 44
      CALL SYMBOL (VIX,VIY,CBZ,1H),0.0,1)
      VIX=VIX+CBZ
      IF (JJN.EQ.5) GO TO 45
      GO TO 46
44    CALL SYMBOL (VIX,VIY,CBZ,1HS,0.0,1)
      VIX=VIX+CBZ
      IF (JJN.EQ.8.OR.JJN.EQ.1) GO TO 46
45    EEP=2.
      IF (JJN.EQ.5.OR.JJN.EQ.2) EEP=YDISJ(I)+E
      ASP=.7*CBZ
      EEA=VIY+.6*CBZ
      CALL NUMBER (VIX,EEA,ASP,EEP,0.0,-1)
      VIX=VIX+.75*CBZ
      IF (JJN.EQ.8) GO TO 42
46    IF (I.NE.IROOT) GO TO 47
      VIY=.30*CBZ+BBD
      VIX=AAB
```

```
      CALL SYMBOL (VIX,VIY,CBZ,1HK,0.0,1)
      IF(SIG.EQ.0.) CALL SYMBOL (VIX-CBZ,VIY,CBZ,1H-,0.0,1)
      VIX=VIX+CBZ
47    CONTINUE
C
C     COMPLETE TRANSFER FUNCTION IF THERE IS A TIME DELAY
C
      IF (TIME.LT.E) GO TO 50
      EEP=.8*CBZ
      CALL SYMBOL (VIX,VIY,EEP,85,0.0,-1)
      EEA=VIY+.4*CBZ
      BBD=VIX+.2*CBZ
      EEP=.4*CBZ
      CALL SYMBOL (BBD,EEA,EEP,81,0.0,-1)
      VIX=VIX+CBZ
      VIY=VIY+.6*CBZ
      EEP=CBZ*.7
      EEA=-TIME
      CALL NUMBER (VIX,VIY,EEP,EEA,0.0,NID)
      VIX=VIX+.70*TIMA*CBZ
      CALL SYMBOL (VIX,VIY,EEP,46,0.0,-1)
      GO TO 50
C
C     PRINT 3 LINES IN PLACE OF TRANSFER FUNCTION
C
48    VIY=.849
      TITSIZ  = AMIN1(.105,(XOT-1.)/ISTAM)
      VIX=(XOT-ISTAM*TITSIZ)*.5
      WRITE(7,200)
      DO 49 I=1,3
```

```
      READ(5,181)(RZQ(IIM),IIM=1,8)
      WRITE(6,179)(RZQ(IIM),IIM=1,8)
      IF (NOPLOT) GO TO 480
      CALL SYMBOL(VIX,VIY,TITSIZ,RZQ,0,ISTAM)
480   VIY=VIY-.148
43    CONTINUE
C
C     CHECK FOR REPEATED OPEN-LOOP ZEROES
C
50    IIV=0
      IF (M.LT.2) GO TO 55
      LQ=1
      DO 54 I=2,M
      IF (XZ(I).EQ.XZ(I-1).AND.YZ(I).EQ.YZ(I-1)) GO TO 53
51    IF (LQ.EQ.1) GO TO 54
      ASP=XZ(I-1)*V+YAX
      EEP=YZ(I-1)*V+XAX
      IF ((ASP.GT.(XOT-.5)).OR.(ASP.LT..5).OR.(EEP.LT.1.25).OR.(EEP.GT.(
     1YOT-.75))) GO TO 52
      IF (NOPLOT) GO TO 510
C
C     PLOT ANY REPEATED OPEN-LOOP ZEROES
C
      CALL SYMBOL (ASP,EEP,.035,1,0.0,-1)
      CALL SYMBOL (ASP+.055,EEP+.03,.07,3H( ),0.0,3)
510   BBD=LQ
      IF (NOPLOT) GO TO 511
      CALL NUMBER (ASP+.125,EEP+.03,.07,BBD,0.0,-1)
511   IIV=IIV+1
      YDISD(IIV)=EEP+.02
```

```
      YDISU(IIV)=EEP+.11
      XDISL(IIV)=ASP+.05
      XDISR(IIV)=ASP+.23
52    LQ=0
53    LQ=LQ+1
      IF (I.EQ.M) GO TO 51
54    CONTINUE
55    IF (NOPLOT) GO TO 75
C
C     PLOT AND LABEL REAL AXIS
C     VW=UNITS/IN.              V=1/VW
C     YYB=YOT-.75
C     XAA=XOT-.5
C     XAX=1.25-V*DD
C     YAX=.5-V*CC
C     CEA=# CIPHERS LEFT OF DECIMAL IN VW
C
      CBZ=VW
      JK=2
      CALL DNTER (CBZ,JK)
      NID=JK
      ASP=.07*CBZ
      DXAX=1.2
      EXAX=1.34
      DYAX=XAA-CEA*.07-ASP-.07
      EYAX=DYAX-.07
      IF (XAX.LT.(1.25).OR.XAX.GT.YBB) GO TO 58
      IF (YAX.GT.(XAA-.4)) GO TO 56
C     DRAW SIGMA
      CALL SYMBOL (XOT-.55,XAX+.05,.14,108,0,-1)
```

```
      IIV=IIV+1
      XDISR(IIV)=XOT-.4
      XDISL(IIV)=XOT-.57
      YDISU(IIV)=XAX+.13
      YDISD(IIV)=XAX+.04
56    DXAX=XAX-.14
      EXAX=XAX
      XWORK=YAX-XYAX/10.-.01
      DO 57 I  =1,IAX
C     DRAW X AXIS
      CALL PLOT (XWORK,XAX,3)
      CALL PLOT (XWORK+.02,XAX,2)
 57   XWORK = XWORK + .1
C
C    PLOT AND LABEL IMAGINARY AXIS
C
58    IF (YAX.LT.(.5).OR.YAX.GT.XAA) GO TO 60
      IIV=IIV+1
C    DRAW J-OMEGA
      CALL SYMBOL (YAX-.19,YOT-.81,.09,37,0.0,-1)
      CALL SYMBOL (YAX-.113,YOT-.73,.14,114,0,-1)
      CALL SYMBOL (YAX-.15,YOT-.7,.07,74,0.0,-1)
      XDISR(IIV)=YAX-.045
      XDISL(IIV)=YAX-.2
      YDISU(IIV)=YOT-.7
      YDISD(IIV)=YOT-.82
      DYAX=YAX+.07
      EYAX=YAX
      YWORK = XAX - YXAX/10. - .01 +(IAY-1)*.1
      DO 59 I  =1,IAY
```

```
C     DRAW Y AXIS
      CALL PLOT(YAX,YWORK+.02,3)
      CALL PLOT(YAX,YWORK,2)
 59   YWORK = YWORK - .1
C
C     SET UP TO DRAW SCALES
C
60    IF(IFPN.GT.0) FPN=FPN+1.
      IVW=BB/VW
      WV=IVW
C    TOTAL # OF UNITS ON + IMAGINARY AXIS
      CBZ=WV*VW
      CALL CNTER (CBZ)
      EEA=CBZ
      IVW=DD/VW
      WV=IVW
C    TOTAL # OF UNITS ON - IMAGINARY AXIS
      CBZ=WV*VW
      CALL CNTER (CBZ)
      IF (CBZ.GT.EEA) EEA=EEA+1.
      CEA=EEA*.07+ASP+DYAX-.01
C
C     DRAW Y AXIS SCALE
C
61    WV=IVW
      WVW=WV*VW
      IF (WVW.GT.BB) GO TO 63
      WV=XAX+WV
      CALL SYMBOL (EYAX,WV,.035,3,0.0,-1)
      IF (WVW.EQ.0.) GO TO 62
```

```
      WV=WV-.035
      CBZ=WVW
      CALL CNTER (CBZ)
      DTAX=DYAX+(EEA-CBZ)*.07
      CALL NUMBER (DTAX,WV,.07,WVW,0.0,NID)
      IIV=IIV+1
      XDISR(IIV)=CEA
      XDISL(IIV)=DTAX-.01
      YDISD(IIV)=WV-.01
      YDISU(IIV)=WV+.08
62    IVW=IVW+1
      GO TO 61
63    IVW=CC/VW
C
C     DRAW X AXIS SCALE
C
64    WV=IVW
      WVW=WV*VW
      IF (WVW.GT.AA) GO TO 66
      WV=YAX+WV
      CALL SYMBOL (WV,EXAX,.035,3,0.0,-1)
      IF (WVW.EQ.0.) GO TO 65
      CBZ=WVW
      CALL CNTER (CBZ)
      CEA=CBZ*.035+ASP/2.
      WV=WV-CEA
      CALL NUMBER (WV,DXAX,.070,WVW,0.0,NID)
      IIV=IIV+1
      XDISL(IIV)=WV-.01
      XDISR(IIV)=WV+2.*CEA-.01
```

```
      YDISD(IIV)=DXAX-.01
      YDISU(IIV)=DXAX+.08
65    IVW=IVW+1
      GO TO 64
C
C     DRAW ZETA LINE
C
66    IF ((ZETA.LT.(.09)).OR.(ZETA.GT.(.31))) GO TO 69
      CALL PLOT (YAX,XAX,3)
      EEA=SQRT(1.-ZETA*ZETA)
      VIY=XAX
      VIX=YAX
      NID=2
67    VIX=VIX-.25*ZETA
      VIY=VIY+.25*EEA
      IF ((((VIY+.90).GT.YDT).OR.(VIX.LT..8)).AND.(NID.EQ.3)) GO TO 68
      CALL PLOT (VIX,VIY,NID)
      NID = (NID+4)/NID
      GO TO 67
C
C     PRINT ZETA
C
68    VIY=VIY-.2*EEA
      VIX=VIX-.1*ZETA
      CALL SYMBOL(VIX-.05,VIY-.03,.20,96,0,-1)
      CALL SYMBOL (VIX+.05,VIY,.095,1H=,0.0,1)
      CALL NUMBER (VIX+.144,VIY,.105,ZETA,0.0,2)
      IIV=IIV+1
      XDISL(IIV)=VIX-.050
      YDISD(IIV)=VIY-.035
```

```
      YDISU(IIV)=VIY+.11
      XDISR(IIV)=VIX+.52
C
C     PLOT ANY ZERO THAT COINCIDES WITH A POLE
C     AND REMOVE BOTH FROM FURTHER CONSIDERATION
C
69    IF (M.EQ.0) GO TO 75
      DO 74 I=1,M
      DO 73 IIM=1,N
      IF (ABS(XZ(I)-XP(IIM)).GT.E.OR.ABS(YZ(I)-YP(IIM)).GT.E) GO TO 73
      VIX=XZ(I)*V+YAX
      VIY=YZ(I)*V+XAX
      IF ((VIX.LT..5).OR.(VIX.GT.(XOT-.5)).OR.(VIY.LT.1.25).OR.
     1(VIY.GT.(YOT-.75))) GO TO 695
      CALL SYMBOL (VIX,VIY,.07,1,0.0,-1)
      CALL SYMBOL (VIX,VIY,.07,4,0.0,-1)
 695  N=N-1
      IF (IIM.GT.N) GO TO 71
      DO 70 JUNK=IIM,N
      XP(JUNK)=XP(JUNK+1)
70    YP(JUNK)=YP(JUNK+1)
 71   M=M-1
      IF ( I.GT.M) GO TO 75
      DO 72 JUNK=I,M
      XZ(JUNK)=XZ(JUNK+1)
72    YZ(JUNK)=YZ(JUNK+1)
      GO TO 69
73    CONTINUE
74    CONTINUE
75    JK=0
```

```
750   CONTINUE
C
C      FORMATS
C
 173  FORMAT("0"8A10)
 181  FORMAT(8A10)
 200  FORMAT(".ENTER THREE LINE TITLE")
      END


      SUBROUTINE CNTER (CBZ)
C
C     COMPUTES THE NUMBER OF CHARACTERS IN INTEGER PART OF CBZ
C     RETURNS ANSWER IN CBZ
C
      RILA=CBZ
      CBZ=1.
      IF (RILA.LT.0.) CBZ=2.
      RILA=ABS(RILA)
      CBZ = CBZ + MAX0(IFIX(ALOG10(RILA)+.00000001),0)
      RETURN
      END


      SUBROUTINE ONTER (CBZ,JK)
C
```

```
C      INPUT
C            CBZ  VALUE
C            JK   MAXIMUM NUMBER OF DECIMAL PLACES
C      OUTPUT:
C            CBZ  NUMBER OF DECIMAL PLACES TO PLOT + 1 FOR DECIMAL POINT
C             JK  NECESSARY NUMBER OF DECIMAL PLACES OR -1 FOR NONE
C
      Q=.000001
      NJK=-1
      RILA=ABS(CBZ)
      IAAW=RILA+Q
      AAW=IAAW
      RILA=RILA-AAW
      IAAW=RILA*(10.**JK)+.49
      IF (IAAW.EQ.0) GO TO 2
      DO 1 I=1,JK
      NJK=I
      IAAW=10.*(RILA+Q)
      AAW=IAAW
      RILA=10.*RILA-AAW
      IF (RILA.LE.Q) GO TO 2
1     CONTINUE
2     JK=NJK
      APA=JK
      CBZ=APA+1.
      RETURN
      END
```

```
      SUBROUTINE GROUP (XDISM,XDISS,YDISE,YDISV,IIE,KKV,JJN)
C
C     THIS SUBROUTINE IS USED IN DETERMINNING THE SIZE OF THE TRANSFER
C     FUNCTION  FOR PLOTTING IT
C
      DIMENSION XDISM(1),XDISS(1),YDISE(1),YDISV(1),IIE(1)
      E=.0001
      LQ=0
      DO 2 I=1,JJN
      IF (LQ.GE.I) GO TO 2
      LQ=I
      IF ((ABS(YDISE(I)).GT.E).OR.(ABS(XDISM(I)).GT.E)) GO TO 2
      KKV=KKV+1
      IIE(KKV)=1
      YDISV(KKV)=1.
    1 IF (LQ.GE.JJN) GO TO 2
      IF (YDISE(I).NE.YDISE(LQ+1).OR.XDISM(I).NE.XDISM(LQ+1)) GO TO 2
      IIE(KKV)=2
      LQ=LQ+1
      YDISV(KKV)=YDISV(KKV)+1.
      GO TO 1
    2 CONTINUE
      LQ=0
      DO 4 I=1,JJN
      IF (LQ.GE.I) GO TO 4
      LQ=I
      IF ((ABS(YDISE(I)).GT.E).OR.(ABS(XDISM(I)).LT.E)) GO TO 4
      KKV=KKV+1
      IIE(KKV)=4
      YDISV(KKV)=1.
```

```
      XDISS(KKV)=-XDISM(I)
 3    IF (LQ.GE.JJN) GO TO 4
      IF (YDISE(I).NE.YDISE(LQ+1).OR.XDISM(I).NE.XDISM(LQ+1)) GO TO 4
      IIE(KKV)=5
      LQ=LQ+1
      YDISV(KKV)=YDISV(KKV)+1.
      GO TO 3
 4    CONTINUE
      LQ=0
      DO 5 I=1,JJN
      IF (LQ.GE.I) GO TO 5
      LQ=I
      IF (ABS(YDISE(I)).LT.E) GO TO 5
      KKV=KKV+1
      IIE(KKV)=0
      XDISS(KKV)=0.
      KKV=KKV+1
      LQ=LQ+1
      IIE(KKV)=5
      XDISS(KKV)=XDISM(I)*XDISM(I)+YDISE(I)*YDISE(I)
      IF (ABS(XDISM(I)).LT.E) GO TO 5
      IIE(KKV-1)=8
      IIE(KKV)=7
      XDISS(KKV-1)=-2.*XDISM(I)
 5    CONTINUE
      RETURN
      END
```

```
      OVERLAY(COPCSD,1,2)
      PROGRAM ROOTS
C
C     THIS PROGRAM IS USED TO FIND THE ROOTS OF INTEREST FOR A
C     SPECIFIED GAIN, FOR OPTIONS 1, 2, 3, 6, AND 7
C
      COMMON /MAIN012/ GANE,NOP_OT
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      DIMENSION XXP(50),YYP(50),ZZP(50)
      LOGICAL NOP_OT
C     COMPUTE NUMERATOR POLYNOMIAL COEFFICIENTS
      NN=N
      A=1.
      IF(SIG.EQ.0.) A=-1.
      LLI=0
      ZZP(1)=1.
      DO 4 I=1,N
      IF(LLI.EQ.1) GO TO 3
      C=0.
      B=-XP(I)
      IF(YP(I).EQ.0.) GO TO 1
      C=B*B+YP(I)*YP(I)
      B=2.*B
      LLI=1
    1 KK=I
      XXP(1)=0.
      XXP(2)=0.
      DO 2 L=1,KK
      XXP(L+2)=0.
```

```
      XXP(L)=XXP(L)+ZZP(L)
      XXP(L+1)=XXP(L+1)+B*ZZP(L)
      XXP(L+2)=XXP(L+2)+C*ZZP(L)
    2 ZZP(L)=XXP(L)
      ZZP(I+1)=XXP(I+1)
      ZZP(I+2)=XXP(I+2)
      GO TO 4
    3 LLI=0
    4 CONTINUE
      YYP(1)=1.
      IF(M.EQ.0) GO TO 9
C     NOW COMPUTE DENOMINATOR
      LLI=0
      ZZP(1)=1.
      DO 8 I=1,M
      IF(LLI.EQ.1) GO TO 7
      C=0.
      B=-XZ(I)
      IF(YZ(I).EQ.0.) GO TO 5
      C=B*B+YZ(I)*YZ(I)
      B=2.*B
      LLI=1
    5 KK=I
      YYP(1)=0.
      YYP(2)=0.
      DO 6 L=1,KK
      YYP(L+2)=0.
      YYP(L)=YYP(L)+ZZP(L)
      YYP(L+1)=YYP(L+1)+B*ZZP(L)
      YYP(L+2)=YYP(L+2)+C*ZZP(L)
```

```
5     ZZP(L)=YYP(L)
      ZZP(I+1)=YYP(I+1)
      ZZP(I+2)=YYP(I+2)
      GO TO 8
7     LLI=0
8     CONTINUE
9     LLI=N+1
      KK=N-M
      DO 10 I=1,LLI
      ZZP(I)=XXP(I)
      IF(I.LE.KK) GO TO 10
      LI=I-KK
      ZZP(I)=ZZP(I)+YYP(LI)*A*GANE/GA
10    CONTINUE
      CALL SMULR (ZZP,NN,XXP,YYP)
      WRITE(6,12)
      DO 11 I=1,N
      IF(ABS(XXP(I)).LT.E) XXP(I)=0.
      IF(ABS(YYP(I)).LT.E) YYP(I)=0.
      IF (NOPLOT) GO TO 11
      VIX=YY+G*XXP(I)
      VIY=XX+G*YYP(I)
      IF ((VIX.LT..5).OR.(VIX.GT.(FX-.5)).OR.(VIY.LT.1.25).OR.(VIY.GT.(F
     1Y-.75))) GO TO 11
      CALL SYMBOL (VIX,VIY,.07,2,0.0,-1)
11    WRITE(6,13) XXP(I),YYP(I)
      WRITE(6,14)
C
C    FORMATS
C
```

```
12    FORMAT (/,6X,17HROOTS OF INTEREST)
13    FORMAT (8X,3HX =,G13.5,9X,3HY =,G13.5)
14    FORMAT(2X)
      END


      OVERLAY(CCPCSD,1,3)
      PROGRAM ROOT12
C
C     TRACE ROOT LOCUS AND PRINT AND PLOT AS INDICATED BY LL CODE
C
       COMMON /MAIN02/CGIN,CNIG,GAG,II,LN,DPR(50)
C     DPR IS CALLED DELPR IN ROOT10
      COMMON /MAIN012/GANE,NOPLOT,STOL,IIV,LL,V,XAX,YAX,DEL(50),
     +  XDISL(50),XDISR(50),YDISD(50),YDISU(50),XOT,YOT,CB
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,SANG,TIME
      COMMON/ANGSS/A,B,ANG,PI
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      COMMON /ANGUU/ AA,BB,CC,DD,D,DA, JK,SANG,CA
      LOGICAL NOPLOT
      DATA DAN/.2/,LIE/0/,EEB/.00001/
      JJ = JJV = 0
      DI = D
      TDPR=DPR(1)
      GO TO (79,79,79,76,78) LL
C
C     CODE FOR OPTIONS 4 AND 5
C
```

```
C     BEGIN ITERATION, OPTION 4
75     WRITE (6,171) JK
C     BEGIN ITERATION, OPTION 5
 78    WRITE (6,173) X,Y,D,CGIN,CNIG
C
C      IF LN NE 0 START TRACE AT A POLE
C
       IF (LN.NE.0) GO TO 80
       CALL ANG1
       A=-FY
       B=FX
       CALL BANG
       CALL BOX
       SANG=ANG
       GO TO 89
C
C      START A BRANCH TRACE
C
C     BEGIN ITERATION, OPTIONS 1, 2 & 3
 79    WRITE(6,175)II,D,TDPR
       WRITE(6,1751)CGIN,CNIG
       JJV = 2
       J=1
       X=XP(II)
       Y=YP(II)
80     CALL GANG1
C
C     TEST FOR MULTIPLE POLES AND PLOT
C     JJ=0, INITIALLY
C
```

```
      IF (JJ-1) 82,88,88
81    II=II+1
      GO TO 85
82    IJ=II+JJ
      IF (XP(IJ)-X) 86,83,86
83    IF (YP(IJ)-Y) 86,84,86
84    JJ=JJ+1
85    IF (JJ+II-N) 82,82,87
86    IF (LL-4) 87,81,81
87    ZP=JJ
      IF(NOPLOT) GO TO 88
      IF (JJ.EQ.1) GO TO 88
      VIX=V*X+YAX
      VIY=V*Y+XAX
      IF ((VIX.LT..5).OR.(VIX.GT.(XOT-.5)).OR.(VIY.LT.1.25).OR.(VIY.GT.(
     1YOT-.75))) GO TO 88
      CALL SYMBOL (VIX,VIY,.07,11,0.0,-1)
      CALL SYMBOL (VIX+.055,VIY+.03,.07,3H( ),0.0,3)
      CALL NUMBER (VIX+.125,VIY+.03,.07,FLOAT(JJ),0.0,-1)
      IIV=IIV+1
      YDISD(IIV)=VIY+.02
      YDISU(IIV)=VIY+.11
      XDISL(IIV)=VIX+.05
      XDISR(IIV)=VIX+.23
88    Z=JJ
      JJ=JJ-1
C     ADJUST PHASE ANGLE AT REPEATED POLE
      ANG=(GANG+2.*PI*Z)/ZP
      LN=1
      GAIN=0.
```

```
      CALL BOX
      SANG=ANG
      CA=SANG
 89   DA=0
      DS=0
      L=1
      JL=1
      K=1
 90   XX=X
      YY=Y
      GEN=GAIN
      A=X
      B=Y
      WNN=SQRT(A*A+B*B)
      CALL BANG
      ZE=ABS(COS(ANG))
      DE=1.02*DI
      IF (LL.EQ.4) GO TO 91
C
C     CHECK FOR AT AN OUTSIDE BOUNDARY
C
      IF (XX.GT.AA.OR.YY.GT.BB.OR.XX.LT.CC.OR.YY.LT.DD) GO TO 162
C
C    TEST TO SEE IF POINT SHOULD BE PRINTED OR SKIPPED,
C    PRINT EVENLY SPACED POINTS ACCORDING TO PRINTING STEP SIZE
C
 91   IF ((LN.NE.0).OR.(LL.GE.4)) GO TO 92
      SDPR=SDPR+D
      IF(SDPR .LT. TDPR) GO TO 93
C     WRITE LOCUS POINT AND RESET SPACING COUNTER
```

```
 92   WRITE (6,185) XX,YY,WIN,GAIN,ZE,LN
      SDPR=TDPR*.000001
C
C     SKIP PLOT IF NOPLOT
C
 93   IF(NOPLOT) GO TO 99
C
C     SET UP TO PLOT POINT
C
      VIX=V*X+YAX
      VIY=V*Y+XAX
C     ONLY PLOT AN X FOR A POLE
      IF (LN.EQ.1) CALL SYMBOL(VIX,VIY,.07,4,0,-1)
C
C    CHECK BLOCKED OUT AREAS OF PLOT
C
      IF(JJV .EQ. 3) CALL PLOT(VIX,VIY,3)
      JJV=3
      DO 94 I=1,IIV
      IF ((VIX.LT.XDISR(I)).AND.(VIX.GT.XDISL(I)).AND.(VIY.LT.YDISU(I)).
     1AND.(VIY.GT.YDISD(I))) GO TO 96
 94   CONTINUE
      JJV=2
C    PLOT LOCUS POINT - PEN UP IF IN BLOCKED OUT AREA
 96   CALL PLOT(VIX,VIY,JJV)
 99   IF(M .EQ. 0) GO TO 110
C
C     TEST POINT TO SEE IF IT IS NEAR A ZERO
C
      DO 109 I=1,M
```

```
      IF (ABS(XX-XZ(I))-DE) 101,101,109
101   IF (ABS(YY-YZ(I))-DE) 102,102,109
C
C     IT IS CLOSE - SEE IF THERE IS A POLE CLOSER
C
102   DO 107 IJ=1,N
      IF (ABS(XZ(I)-XP(IJ))-2.*DE) 103,103,107
103   IF (ABS(YZ(I)-YP(IJ))-2.*DE) 104,104,107
104   A=XZ(I)-XP(IJ)
      B=YZ(I)-YP(IJ)
      DA=.2*SQRT(A*A+B*B)
      IF (ABS(XX-XZ(I))-DA) 105,105,106
105   IF (ABS(YY-YZ(I))-DA) 108,105,106
106   IF (DA.LT.(.0001)) GO TO 108
      D=DA
      GO TO 110
107   CONTINUE
C
C     A ZERO IS REACHED, WRITE AND PLOT IT
C
108   GAIN=0.
      LN=2
      A=XZ(I)
      B=YZ(I)
      WNN=SQRT(A*A+B*B)
      CALL BANG
      ZE=ABS(COS(ANG))
      WRITE (6,185) XZ(I),YZ(I),WNN,GAIN,ZE,LN
      IF (NOPLOT) GO TO 153
      VIX=V*XZ(I)+YAX
```

```
      VIY=V*YZ(I)+XAX
      CALL SYMBOL (VIX,VIY,.07,1,0.0,-2)
C
C     NEXT BRANCH OR LOCUS
C
      GO TO 163
103   CONTINUE
C
C     NOT A ZERO - GO ON
C
      DA=DI
      D=DS
C     J = 4 FOR A BREAK POINT
110   GO TO (111,111,116,153,111), J
C
C     SEARCH ALONG REAL AXIS
C
111   IF (ABS(YY)-E) 112,112,121
112   CB=1.
      CAA=0.
      CA=0.
      IF (ABS(SANG)-.3) 115,116,113
113   IF (ABS(SANG+PI)-.3) 115,115,114
114   IF (ABS(SANG-PI)-.3) 115,115,121
115   CB=-1.
      CAA=-PI
      CA=-PI
116   X=XX+CB*D
      YY=0.
      Y=0.
```

```
      J=3
      CALL ANG1
      IF (J.EQ.3) GO TO 117
      J=3
      GO TO 120
117   IF (ABS(GANG).GT.(.5)) GO TO 120
      SANG=0.
      IF (FY.GT.0.) SANG=-PI
      IF (ABS(FY).EQ.0.) GO TO 119
      IF ((ABS(ABS(SANG)-ABS(CA))).GT.2.5) GO TO 118
      IF (LIE.NE.1) GO TO 1170
      X=XX
 1170 LIE=0
      IF (K.GE.5.OR.K.EQ.1) GO TO 132
      K=K+1
      XY=XX
      XX=X
      GO TO 115
118   IF (LIE.EQ.1) GO TO 152
      K=2
      D=D/4.
      IF (DA/255.-D) 116,115,152
119   XX=X
      LIE=1
      GO TO 115
C
C     REDUCE STEPPING DISTANCE AND TRY AGAIN
C
120   D=D/4.
      IF (D.LT.(DA/250.)) GO TO 150
```

```
      IF (J.EQ.3) GO TO 115
C
C     SEARCH COMPLEX POINTS
C
121   J=1
      CA=SANG
C
C     MOVE TO A NEW POINT
C
122   X=XX+D*COS(SANG)
      Y=YY+D*SIN(SANG)
      CALL ANG1
      IF (J.EQ.5) GO TO 120
      IF (ABS(GANG)-.00001) 124,124,123
C
C     IF OFF LOCUS, TRY TO IMPROVE
C
 123  IF ((L.EQ.2).AND.(JL.GE.5)) GO TO 130
      ANG=SANG-GANG/(FY*D*COS(SANG)-FX*D*SIN(SANG))
      CALL BOX
      SANG=ANG
      JL=JL+1
      IF (JL-20) 122,122,125
124   GO TO (126,130,126), L
125   JL=1
      X=XX+D*COS(SANG)
      Y=YY+D*SIN(SANG)
      IF (ABS(GANG)-.0005) 129,129,161
C
C     PERTURB POINT BY EEB IN EACH DIRECTION
```

```
C       IF ERROR ANGLE CHANGES IN EITHER DIRECTION WE ARE WITHIN
C       .00001 OF THE LOCUS
C
125     R=SQRT(D*D+EEB*EEB)
        XXX=X
        YYY=Y
        ED=EEB/D
        GAG=SANG+ATAN(ED)
        X=XX+R*COS(GAG)
        Y=YY+R*SIN(GAG)
        CB=GANG
        CALL  GANG1
C
C       MY MODIFICATION ALLOWS FOR 1E-10 ERROR IN THE ANGULAR COMPUTATION
C
        IF(GANG*CB-1.E-20) 128,128,127
127     GAG=SANG-ATAN(ED)
        X=XX+R*COS(GAG)
        Y=YY+R*SIN(GAG)
        CALL  GANG1
        IF(GANG*CB-1.E-20 .LE. 0) GO TO 128
        GANG=CB
        GO TO 123
C
C       WE ARE ON LOCUS - PROCEED
C
128     JL=1
        X=XXX
        Y=YYY
123     CAA=CA
```

```
      CA=SANG
C
C     COMPUTE S_OPE AT THE NEW POINT
C     IF THE CHANGE IN SLOPE FROM THE OLD POINT (DANG IS THE CHANGE)
C     IS GREATER THAN PI/5, ASSJME THAT WE HAVE PASSED A BREAK POINT
C
130   A=-FY
      B=FX
      CAPOT=SANG
      CALL BANG
      CALL BOX
      SANG=ANG
      IF (L.EQ.2) GO TO 155
      DANG=ABS(SANG-CA)
      EANG=ABS(CAA-CA)
      IF (EANG.GT.1.4) GO TO 149
      IF ((EANG.GT..3).AND.(K.GE.2)) GO TO 149
 131  IF(DANG .GT. .6) GO TO 150
C     ALLOW A SMALLER CHANGE IF WE THINK WE ARE NEAR A BREAK POINT
      IF ((DANG.GT.DAN).AND.(K.GE.2)) GO TO 150
132   XY=XX
      YX=YY
      XX=X
      YY=Y
      GO TO (134,147,147,147,133), K
133   LIE=0
      K=1
      D=DA
134   IF (XY*XX) 136,135,140
135   IF (XY) 136,140,136
```

```
136   LN=4
      XX=XY
      YY=YX
      D=D/4.
      IF (DA/255.-D) 137,137,133
137   L=3
      IF (J-3) 122,139,122
138   Y=YX-(YX-Y)*XY/(XY-X)
139   X=0.
      D=DA
      CALL ANG1
      GO TO 89
140   IF (L-3) 141,122,141
141   D=DA
      LN=0
      GO TO (143,143,143,142,143), LL
C
C     DECREMENT AND TEST POINT COUNTER FOR OPTION 4
C
142   JK=JK-1
      IF (JK) 153,153,143
C
C    DETERMINE WHETHER SOLUTION IS AT GAIN OF INTEREST
C     THEN GO TO PLOT IT
C
143   IF (GAIN.LT.(GANE-GTOL)) GO TO 90
      IF (GTOL.EQ.0.) GO TO 145
      IF (((GEN-GANE).LT.0.).AND.((GAIN-GANE).GE.0.)) GO TO 144
      IF (GAIN.GT.(GANE+GTOL)) GO TO 145
      GO TO 145
```

```
144   IF (NOPLOT) GO TO 145
      VIY=Y*V+XAX
      VIX=V*X+YAX
      IF(TIME.NE.0.)CALL SYMBOL (VIX,VIY,.07,2,0.0,-2)
C     REDUCE STEP SIZE AROUND GAIN OF INTEREST
145   DS=DI/5.
      D=DS
      LN=5
      GO TO 90
146   DS=DI
      D=DS
      GO TO 90
C
C     SNEAK UP ON A BREAK POINT - INITIAL ENTRY AT 150
C
147   K=K+1
148   IF (J-3) 122,116,122
149   IF ((ABS(CAA).GT.2.5).AND.(ABS(CA).GT.2.5)) GO TO 131
      CA=CAA
      GO TO 151
150   IF ((ABS(CA).GT.2.8).AND.(ABS(SANG).GT.2.8)) GO TO 132
151   SANG=CA
      K=2
      D=D/4.
      IF (D.LT.E) GO TO 152
      IF (DA/256.-D) 148,148,152
C
C     COMPUTE EXACT BREAK POINT AND CHECK TO SEE IF ON REAL AXIS
C
152   CALL BREAK
```

```
      IF (J.EQ.3) YY=0.
      J=4
      IF (ABS(YY).LT..001) YY=0.
      CA=CAA
      LIE=0
      X=XX
      Y=YY
      LN=3
      CALL ANG1
      J=4
      GO TO 90
C
C     APPEARS TO BE ROUTINE TO LEAVE BREAK POINT
C
153   K=2
      D=DA/100.
154   DO 158 JM=1,6
      J=1
      ANG=CA-(11-JM )*PI/12.
      CALL BOX
      PHI=ANG
      SANG=ANG
      L=2
      GO TO 122
C
C    CHECK PHASE CRITERION
C
155   JL=1
      IF (ABS(CAPOT-PHI).LT.(.27)) GO TO 156
      IF ((ABS(CAPOT).LT.(2.9)).OR.(ABS(PHI).LT.(2.9))) GO TO 158
```

```
      IF ((ABS(ABS(CAPOT)-ABS(PHI))).GT.(.27)) GO TO 158
156   IF (ABS(CAPOT-SANG).LT.(.5)) GO TO 157
      IF ((ABS(SANG).LT.(2.9)).OR.(ABS(CAPOT).LT.(2.9))) GO TO 158
      IF ((ABS(ABS(CAPOT)-ABS(SANG))).GT.(.5)) GO TO 158
157   D=DA
      SANG=CAPOT
      CA=SANG
      CAA=CA
      L=1
      K=1
      GO TO 111
158   CONTINUE
      IF (D.GE.(DA/11.)) GO TO 159
      D=DA/10.
      GO TO 154
C
C     TROUBLE - PRINT MESSAGE AND TERMINATE BRANCH
C
159   WRITE (6,188)
      GO TO 163
160   WRITE (6,182)
      GO TO 163
161   WRITE (6,187)
      GO TO 163
C
C     WRITE BOUNDARY MESSAGES
C
C    POLE OUTSIDE RANGE
162   IF(LN .NE. 1) GO TO 167
      WRITE(6,183)X,Y
```

```
      GO TO 163
C     LOCUS CROSSED BOUNDARY
 157  WRITE(6,185)XX,YY,WNN,GAIN,ZE,LN
      WRITE(6,186)
C
C     FINISHED A BRANCH - ACTION DEPENDS ON OPTION
C
163   D=DI
      SDPR=0.
      GO TO (156,154,155,200,200) LL
 154  IF (II.GT.N) GO TO 155
      D=DEL(II+1)
      TDPR=DPR(II+1)
      DI=0
155   IF (JK.EQ.5) GO TO 156
      II=II+1
      IF (II.LE.N) GO TO 79
C
C     SEARCH BOUNDARIES
C
156   CALL SEEK
C
C     LOCATE BRANCH STARTING OUTSIDE RANGE
C
      IF(JK .EQ. 1) GO TO 200
      WRITE(6,1730)X,Y,D,TDPR
      WRITE(6,1751)CGIN,CNIG
      IF(NOPLOT) GO TO 111
      VIX=V*X+YAX
      VIY=V*Y+XAX
```

```
      CALL PLOT(VIX,VIY,3)
      JJV = 2
      GO TO 111
C
C     EXIT
C
  200 CONTINUE
C
C     FORMATS
C
  171 FORMAT (/,6X,23HNUMBER OF POINTS (JK) =,I5)
  173 FORMAT (///,9X,25HSUB-BRANCH STARTING AT X =,G13.5,4X,3HY =,G13.5,
     1//,6X,45HCALCULATION STEP SIZE = PRINTING STEP SIZE = ,G10.4,//,3X
     2,10HLOCUS REAL,5X,10HLOCUS IMAG,3X,14HDIST TO ORIGIN,1X,2A6,4X,4HZ
     3ETA,4X,2HCD)
 1730 FORMAT (///,9X,25HSUB-BRANCH STARTING AT X =,G13.5,4X,3HY =,G13.5,
     1//,6X,24HCALCULATION STEP SIZE = ,G10.4,/,6X,24HPRINTING STEP SIZE
     2     = ,G10.4)
  175 FORMAT (///,23X,14HBRANCH NUMBER ,I3,//,6X,24HCALCULATION STEP SIZ
     1E = ,G10.4,/,6X,24HPRINTING STEP SIZE     = ,G10.4)
 1751 FORMAT (/,3X,10HLOCUS REAL,5X,10HLOCUS IMAG,3X,14HDIST TO ORIGIN,1
     1X,2A6,4X,4HZETA,4X,2HCD)
 182  FORMAT (10X21HTROUBLES NEAR A POLE./10X32HREDUCING THE STEP SIZE M
     1AY HELP.)
  183 FORMAT (10X,15HTHE POLE AT X =,G13.5,8H AND Y =,G13.5,/,10X,24HIS 
     1OUTSIDE THE BOUNDARY.)
  185 FORMAT (1X,3(G14.8,1X),G12.6,1X,F9.5,2X,I2)
 186  FORMAT (10X,8HBOUNDARY)
 187  FORMAT (10X,59HTROUBLE IN FINDING THE NEXT POINT TO WITHIN 10E-5 A
     1CCURACY./10X,32HREDUCING THE STEP SIZE MAY HELP.)
```

```
188   FORMAT (10X,30HTROUBLE LEAVING A BREAK POINT./10X49HUSE OPTION 5 T
     10 GET A CONTINUATION OF THIS BRANCH)
      END


      SUBROUTINE BREAK
C
C     ZERO IN ON A BREAK POINT
C
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      IIM=0
1     IF (IIM.GT.10) GO TO 4
      FX=0.
      FY=-TIME
      FXX=0.
      FXY=0.
      DO 3 I=1,N
      G=(XX-XP(I))*(XX-XP(I))+(YY-YP(I))*(YY-YP(I))
      IF( G .EQ. 0) GO TO 2
      FX=FX+(YY-YP(I))/G
      FY=FY-(XX-XP(I))/G
      FXX=FXX-2.*(XX-XP(I))*(YY-YP(I))/(G*G)
      FXY=FXY+((XX-XP(I))*(XX-XP(I))-(YY-YP(I))*(YY-YP(I)))/(G*G)
 2    IF(M .LT. I) GO TO 3
      G=(XX-XZ(I))*(XX-XZ(I))+(YY-YZ(I))*(YY-YZ(I))
      IF(G .EQ. 0) GO TO 3
      FX=FX-(YY-YZ(I))/G
```

```
      FY=FY+(XX-XZ(I))/G
      FXX=FXX+2.*(XX-XZ(I))*(YY-YZ(I))/(G*G)
      FXY=FXY-((XX-XZ(I))*(XX-XZ(I))-(YY-YZ(I))*(YY-YZ(I)))/(G*G)
3     CONTINUE
      IF( ABS(FXX) + ABS(FXY) .LT. 1.E-10) GO TO 4
      XX=XX-(FX*FXX+FY*FXY)/(FXX*FXX+FXY*FXY)
      YY=YY+(FY*FXX-FX*FXY)/(FXX*FXX+FXY*FXY)
      IIM=IIM+1
      IF ((ABS(FY).GT.0.).OR.(ABS(FX).GT.0.)) GO TO 1
4     RETURN
      END


      SUBROUTINE SEEK
C
C    THIS SUBROUTINE SEARCHES THE BOUNDARIES FOR ANY LOCUS BRANCHES
C    WHICH RE-ENTER THE REGION OF CALCULATION AFTER LEAVING IT OR
C    FOR ANY LOCUS BRANCHES WHICH START OUTSIDE OF THE REGION OF
C    CALCULATION AND THEN ENTER IT
C
      COMMON/ANGRR/XP(50),YP(50),XZ(50),YZ(50),N,M,X,Y,SIG,GANG,TIME
      COMMON/ANGSS/A,B,ANG,PI
      COMMON/ANGTT/FX,FY,GAIN,GA,G,J,E,XX,YY
      COMMON /ANGUJ/ AA,BB,CC,DD,D,DA, JK,SANG,CA
      LOGICAL LOOP
C
C     ENTRY POINT (JK=5 IS A REENTRY)
C
```

```
      DA = D
      LOOP = .FALSE.
      IF(JK .EQ. 5) GO TO 18
      ISIDE = 1
      AX = 1.
      AY = 0
    1 Y=DD
      X=CC
C
C     LOOP CONTROL AND HUNT LOOP
C
    2 CALL GANG1
    3 XA=X
      YA=Y
      CG = GANG
    4 X=XA+D*AX
      Y=YA+D*AY
      CALL GANG1
      IF(GANG*CG .LE. 0) GO TO 11
      IF(LOOP) GO TO 3
      GO TO (5,5,6,6)ISIDE
    5 IF(X .GT. AA) GO TO (7,8)ISIDE
      GO TO 3
    6 IF(Y .GT. BB) GO TO (9,9,9,10)ISIDE
      GO TO 3
    7 Y=BB
      X=CC
      ISIDE=2
      GO TO 2
    8 AX=0
```

```
      AY=1.
      ISIDE=3
      GO TO 1
9     X=AA
      Y=DD
      ISIDE=4
      GO TO 2
C
C     ALL DONE SEARCH - JK=1 IS FLAG TO CALLER
C
10    JK=1
      RETURN
C
C     A BRANCH IS CROSSED
C
11    IF(ABS(GANG) .GT. PI*.5) GO TO 3
      CALL ANG1
      J = 1
      A=-FY
      B=FX
      CALL BANG
      CALL BOX
      SANG=ANG
      GO TO (12,13,14,15)ISIDE
12    IF(SANG)3,3,16
13    IF(SANG)16,3,3
14    IF(ABS(SANG)-PI*.5)16,3,3
15    IF(ABS(SANG)-PI*.5)3,3,16
C
C     HAVE AN ENTRY BRANCH - REFINE IF NECESSARY
```

```
C
 15     IF(GANG .EQ. 0) GO TO 17
        LOOP = .TRUE.
        D=D*.25
        IF(D .GT. DA/1024.) GO TO 4
C
C       HAVE FOUND A GOOD ENTRY POINT - USE IT
C
 17     YA=Y
        XA=X
        D=DA
        JK=5
        CG=GANG
        CA=SANG
        XX=X
        YY=Y
        RETURN
C
C       REENTRY - MOVE A LITTLE FROM THE OLD POINT AND RESTART
C
 18     X=XA+D*AX*.0001
        Y=YA+D*AY*.0001
        GO TO 2
        END


        OVERLAY(CCPCSD,2,0)
        PROGRAM FREQR
```

```
      COMMON/COM/X(502),Y(502),Z(502),XD3(502),ICOUNT,TTY,A(21),B(21),
     1 E(21),F(21),C(40),CTEMP(40),KORD(40),D(21)
      COMMON /CONVERT/ RAD2DEG,DEG2RAD
      LOGICAL TTY,KFLAG
      DATA RAD2DEG/57.2957795/,DEG2RAD/.01745329252/
      PRINT 400
400   FORMAT(//" ==========================================================
     +=================="//" ***************START OF PROGRAM 'FREQR'*****
     +**********")
      IFRSEQ=2
      TTY = .TRUE.
      LPLOT=0
      PRINT 2050
2050  FORMAT(//"  DO YOU WANT A LIST OF OPTIONS AVAILABLE FOR 'FREQR'?"/
     +" TYPE 'YES' OR 'NO'."/)
2051  READ 2052,KI
2052  FORMAT(A10)
      IF(KI.EQ.3HYES) GO TO 2053
      IF(KI.EQ.2HNO) GO TO 1
      PRINT 2055
2055  FORMAT("    SORRY,TRY AGAIN...TYPE 'YES' OR 'NO'.")
      GO TO 2051
2053  PRINT 2054
2054  FORMAT(/" *****************************************************
     +"/" THE FOLLOWING OPTIONS ARE AVAILABLE FOR PROGRAM 'FREQR':"//
     +"     1...TO BEGIN A NEW PROBLEM. INPUT OF G(JW) OR G(Z)."/
     +"    -1...FOLLOWED BY W (OMEGA); CALCULATES G(JW) USING FUNCTION"/
     +"         ALREADY READ INTO MEMORY AFTER OPTION 1."/
     +"    -2...CLOSED-LOOP RESPONSE, C(JW)/R(JW)."/
     +"     3...PRODUCES TTY PLOT OF OPTIONS -1 OR -2, OR BOTH."/
```

```
     +"     -3...PRODUCES CALCOMP PLOT OF BODE DIAGRAM (FREQ RESPONSE)."/
     +"      4...TO READ IN A H(JW), FEEDBACK TRANSFER FUNCTION."/
     +"      5...ADDS A TRANSPORT LAG TERM (E(EXP JWT)) TO G(JW)."/
     +"      6...INPUT NEW FORWARD GAIN CONSTANT ONLY."/
     +"     -6...INPUT NEW FEEDBACK GAIN CONSTANT ONLY."/
     +"      7...ALLOWS AUTOMATIC DISPOSE BODE PLOTS TO BLDG 640, 'BB',"/
     +"      9...TO SEE PRESENT G(S), H(S), OF G(Z) FUNCTIONS."/
     +"    -11...SAME AS -1, EXCEPT G(Z) IS CALCULATED."/
     +"    -12...SAME AS -2, EXCEPT CLOSED-LOOP Z-DOMAIN IS CALCULATED."/
     +"     15...TO SET A SAMPLE PERIOD FOR THE SAMPLED DATA SYSTEM."/
     +"      0...CAUSES THE PROGRAM TO STOP."/
     +" *****************************************************")
1     PRINT 405
405   FORMAT(" TYPE OPTION NUMBER -- ")
      READ *,FIND
4     IND = FIND
      IF(IND .EQ. -1 .OR. IND .EQ. -2) GO TO 50
      MF = 0
      IF (IND.EQ.0) GO TO 700
      IF(IND .EQ. 1) GO TO 8
      IF(IND .EQ. 4) GO TO 10
      IF(IND .EQ. 3) GO TO 200
      IF (IND.EQ.-3) GO TO 300
      IF(IND .EQ. 5) GO TO 20
      IF(IND.EQ.6)GO TO 500
      IF(IND.EQ.-6) GO TO 550
      IF(TTY) GO TO 1
      PRINT 405
406   FORMAT("0ERROR IN OPTION.  RUN TERMINATED.")
      STOP
```

```
C
C
C     READ NUMERATOR AND DENOMINATOR OF G(S)
C
C
 8    KFLAG = .TRUE.
      IFEED = 1H
      PRINT 402
 402  FORMAT("1  FORWARD TRANSFER FUNCTION -- G(S)")
      CALL POLYRD(A,N,B,MD,OGN)
      TAUD = 0.0
      ICOUNT = 0
      GO TO 1
  500  CONTINUE
      ICOUNT =0
      IF (TTY) PRINT 501
  501 FORMAT("0TYPE NEW GAIN CONSTANT K-- ")
      READ *,GNW
      PRINT 503,GNW
 503  FORMAT("1    NEW GAIN CONSTANT = ",G15.7)
      GFACT=GNW/OGN
      OGN=GNW
      NN=N+1
      DO 510 II=1,NN
  510 A(II)=A(II)*GFACT
      GO TO 1
  550 KFLAG=.FALSE.
      ICOUNT =0
      IFEED=1HH
      IF(TTY) PRINT 551
```

```
551 FORMAT("0TYPE NEW FEEDBACK GAIN CONSTANT K-- ")
    READ *,FBNG
    PRINT 553,FBNG
553 FORMAT("1     NEW FEEDBACK GAIN CONSTANT =",G15.7)
    IF(FBNG.EQ.0.) GO TO 558
    FBFACT=FBNG/FBOG
    FBOG=FBNG
    NE1=NE+1
    DO 555 II=1,NE1
555 E(II)=E(II)*FBFACT
    GO TO 1
558 KFLAG=.TRUE.
    IFEED=1H
    PRINT 559
559 FORMAT(*1       SYSTEM NOW UNITY FEEDBACK, H=1  *)
    NE=0
    NF=0
    E(1)=1.0
    F(1)=1.0
    GO TO 1
C
C
C     READ NUMERATOR AND DENOMINATOR OF FEEDBACK TRANSFER FUNCTION
C
C
10  KFLAG = .FALSE.
    ICOUNT =0
    IFEED = 1HH
    PRINT 409
409 FORMAT("-  FEEDBACK TRANSFER FUNCTION -- H(S)")
```

```
      CALL POLYRD(E,NE,F,NF,FROG)
      GO TO 1
C
C
C     READ TRANSPORT DELAY
C
C
 20   IF(TTY) PRINT 415
 415  FORMAT("0TYPE TRANSPORT DELAY - TAU -- ")
      ICOUNT =0
      READ *,TAUD
      PRINT 416,TAUD
 416  FORMAT("0TRANSPORT DELAY =",F15.5)
      TAUD = TAUD * RAD2DEG
      GO TO 1
C
C
C     REQUEST PARAMETERS AND THEN COMPUTE AND PRINT RESPONSE VALUES
C
C
 50   CONTINUE
      IF(TTY) PRINT 450
 450  FORMAT(" TO RESTART PLOT ARRAY STORAGE, ENTER A '1'."/)
      IF(TTY) READ *,IFRSEQ
      IF(IFRSEQ.EQ.1) ICOUNT =0
      IF(TTY) PRINT 417
 417  FORMAT(" TYPE INITIAL, INCREMENTAL, AND FINAL OMEGA."/)
      READ *,WO,DELTW,WMAX
      W = WO
      IND = -IND
```

```
      IF(IND .EQ. MF) GO TO 100
      MF = IND
      IF(IND .EQ. 1) PRINT 420
      IF(IND .EQ. 2) PRINT 421, IFEED
      PRINT 422
  420 FORMAT("0     G -- OPEN-LOOP RESPONSE"/)
  421 FORMAT("0     G/(1+",A1,"G) -- CLOSED-LOOP RESPONSE"/)
  422 FORMAT(8X"W"8X,"MAGNITUDE",10X,"MAG. IN DB",6X,"ANGLE IN DEGREES")
C
C     COMPUTE NUMERATOR
C
  100 CALL POLYVAL(W,A,N,PR,PI)
      CALL POLARD(PR,PI,PMAG,PANG)
      IF(TAUD.EQ.0.) GO TO 101
      PANG = PANG + W*TAUD
      PR=PMAG*COS(PANG*DEG2RAD)
      PI=PMAG*SIN(PANG*DEG2RAD)
  101 CONTINUE
      IF(PMAG .NE. 0) GO TO 140
  102 PRINT 423,W
  423 FORMAT(F12.2,4X,"0.")
      GO TO 192
C
C     COMPUTE DENOMINATOR
C
  140 CALL POLYVAL(W,B,MD,PRD,PID)
      IF(IND .EQ. 1) GO TO 160
C
C     CLOSED LOOP MODIFICATIONS
C
```

```
      IF(KFLAG) GO TO 158
      CALL POLYVAL(W,E,NE,PRE,PIE)
      CALL POLARD(PRE,PIE,EMAG,EANG)
      CALL POLYVAL(W,F,NF,PRF,PIF)
      CALL POLARD(PRF,PIF,FMAG,FANG)
      IF(FMAG .NE. 0) GO TO 159
      PMAG = 0.
      GO TO 102
159   EMAG = (EMAG/FMAG)*PMAG
      EANG = ((EANG-FANG)+PANG) * DEG2RAD
      PRD = PRD + EMAG*COS(EANG)
      PID = PID + EMAG*SIN(EANG)
      GO TO 160
158   PID = PID + PI
      PRD = PRD + PR

      PRD = PRD + PR
160   CALL POLARD(PRD,PID,DMAG,DANG)
      IF(DMAG .NE. 0) GO TO 180
      PRINT 424,W
424   FORMAT(F12.2,10X,"*** POLE ***")
      GO TO 194
180   PMAG = PMAG  / DMAG
      PANG = PANG-DANG
      PDB = 8.6858896 * ALOG(ABS(PMAG))
      PRINT 425,W,PMAG,PDB,PANG
425   FORMAT(F12.2,G18.8,G20.8,F11.4)
192   ICOUNT = ICOUNT + 1
      IF(ICOUNT .GT. 500) GO TO 198
      X(ICOUNT) = PMAG
```

```
      Y(ICOUNT) = PANG
      Z(ICOUNT)=W
      XDB(ICOUNT)=PDB
194   W = W + DELTW
      IF(W-WMAX) 100,100,1
198   PRINT 426
426   FORMAT("0TOO MANY POINTS.  EITHER PLOT WHAT YOU HAVE OR RESTART")
      GO TO 1
200   CALL POLPLOT
      GO TO 1
 300  LPLOT=1
      CALL BODPLOT
      GO TO 1
700   PRINT 1500
1500  FORMAT(//"        -------END OF PROGRAM 'FREQR'-------         "//)
      IF(LPLOT.EQ.1) GO TO 701
      RETURN
 701  CALL PLOTE
      RETURN
      END


      SUBROUTINE BODPLOT
      COMMON/COM/X(502),Y(502),Z(502),XDB(502),ICOUNT,TTY,A(21),B(21),
     1 E(21),F(21),C(40),CTEMP(40),KORD(40),D(21)
      DIMENSION TITLE(3)
      LOGICAL TTY
      DATA BLANK/10H          /
```

```
10    FORMAT(3A10)
15    FORMAT("0TYPE IN TITLE OF M-VS-W PLOT, 30 CHARACTERS MAX")
16    FORMAT ("0TITLE IS ")
17    FORMAT(*0 IF YOU WANT A DB VS LOG W PLOT,ENTER A 1 *)
18    FORMAT(*0 IF YOU WANT A MAG VS W PLOT , ENTER A 0 *)
20    FORMAT(*0 IF YOU WANT BOTH TYPES OF PLOTS, ENTER A 2 *)
21    FORMAT(*0 INVALID PLOT CODE, TRY AGAIN*)
22    FORMAT(*0 TYPE IN TITLE OF DB VS LOG W PLOT, 30 CHARACTERS MAX *)
      IDB=0
99    IF(TTY) PRINT 18
      IF(TTY) PRINT 17
      IF(TTY) PRINT 20
      READ *,IDB
      IDC=IDB
      IF(IDB.LT.0.OR.IDB.GT.2) GO TO 892
      IF(ICOUNT.EQ.0) GO TO 900
      XMAX=-1.0E10
      DO 30 I=1,ICOUNT
      IF(X(I).GT.XMAX) GO TO 25
      GO TO 30
25    XMAX=X(I)
      XDBMAX=XDB(I)
      ZMX=Z(I)
30    CONTINUE
100   CONTINUE
      TITLE(1)=BLANK
      TITLE(2)=BLANK
      TITLE(3)=BLANK
      CALL PLOT(5.0,-12.0,-3)
      CALL PLOT(0.0,2.0,-3)
```

```
      CALL PLOT(-.75,-.75,3)
      CALL PLOT(-.75,7.75,2)
      CALL PLOT(8.75,7.75,2)
      CALL PLOT(8.75,-.75,2)
      CALL PLOT(-.75,-.75,2)
      CALL PLOT(0.0,0.0,3)
      IF(IDB.EQ.1) GO TO 50
      IF(TTY) PRINT 15
      IF (TTY) PRINT 16
      READ 10,TITLE
      ZMIN=Z(1)
      ZDA=(Z(ICOUNT)-ZMIN)/8.0
      Z(ICOUNT+1)=ZMIN
      Z(ICOUNT+2)=ZDA
      CALL SCALE(X,6.0,ICOUNT,1)
      CALL SCALE(Y,5.0,ICOUNT,1)
      XMIN=X(ICOUNT+1)
      XDA=X(ICOUNT+2)
      YMIN=Y(ICOUNT+1)
      YDA=Y(ICOUNT+2)
      CALL AXIS(0.0,0.0,11HFREQUENCY W,-11,8.0,0.0,ZMIN,ZDA)
      CALL AXIS(0.0,0.0,4HM(W),4,6.,90.,XMIN,XDA)
      CALL AXIS(8.0,0.0,8HANGLE(W),-8,5.,90.,YMIN,YDA)
      CALL LINE(Z,X,ICOUNT,1,0,0)
      CALL LINE(Z,Y,ICOUNT,1,10,5)
      GO TO 70
   50 CONTINUE
      IDB=3
      IF(TTY) PRINT 22
      IF(TTY) PRINT 16
```

```
      READ 10,TITLE
      XMAX=XDBMAX
      CALL SCALE(XDB,6.,ICOUNT,1)
      IF(IDC.EQ.1) CALL SCALE(Y,6.0,ICOUNT,1)
      CALL LGSCAL(Z,8.,ICOUNT)
      ZMIN=Z(ICOUNT+1)
      ZDA=Z(ICOUNT+2)
      XMIN=XDB(ICOUNT+1)
      XDA=XDB(ICOUNT+2)
      YMIN=Y(ICOUNT+1)
      YDA=Y(ICOUNT+2)
      CALL LGAXIS(0.0,0.0,11HFREQUENCY W,-11,8.,0.,ZMIN,ZDA)
      CALL AXIS(0.,0.,7HM(W) DB,7,6.,90.,XMIN,XDA)
      CALL AXIS(8.,0.,8HANGLE(W),-8,6.,90.,YMIN,YDA)
      CALL LGLINE(Z,XDB,ICOUNT,0,0,-5)
      CALL LGLINE(Z,Y,ICOUNT,10,5,-5)
   70 CONTINUE
      CALL SYMBOL(1.4,7.0,0.21,TITLE,0.0,30)
      CALL SYMBOL(6.0,6.24,0.14,11H-=MAGNITUDE,0.0,11)
      CALL SYMBOL(6.0,6.57,0.14,5,0.0,-1)
      CALL SYMBOL(6.14,6.5,0.14,6H=ANGLE,0.0,6)
      CALL SYMBOL(1.0,6.5,0.14,3HM =,0.0,3)
      CALL NUMBER(999.,999.,0.14,XMAX,0.0,4)
      IF(IDB.EQ.3) CALL SYMBOL(999.,999.,0.14,3H DB,0.0,3)
      CALL SYMBOL(1.14,6.465,0.07,1HM,0.0,1)
      CALL SYMBOL(1.0,6.24,0.14,114,0.0,-1)
      CALL SYMBOL(1.14,6.205,0.07,1HM,0.0,1)
      CALL SYMBOL(1.28,6.24,0.14,1H=,0.0,1)
      CALL NUMBER(999.,999.,0.14,ZMX,0.0,4)
      CALL PLOT(9.0,0.0,-3)
```

```
      IF(IDB.EQ.2) GO TO 800
      IF(IDB.EQ.3) GO TO 890
      GO TO 890
  800 IDB=1
      GO TO 100
  892 IF(TTY) GO TO 893
      PRINT 21
      GO TO 890
  893 PRINT 21
      GO TO 99
  890 ICOUNT =0
  900 CONTINUE
      RETURN
      END


      SUBROUTINE POLPLOT
      COMMON/COM/X(502),Y(502),Z(502),XDB(502),ICOUNT,TTY,A(21),B(21),
     1 E(21),F(21),C(40),CTEMP(40),KORD(40),D(21)
      EQUIVALENCE (X(1),IX(1)),(Y(1),IY(1))
      COMMON /CONVERT/ RAD2DEG,DEG2RAD
      DIMENSION IX(500),IY(500),LINE(121)
      INTEGER TOPFORM(3),BOTFORM(3)
      LOGICAL TTY
      INTEGER DOT,ZERO,BLANK
      DATA DOT/1H./, BLANK/1H /, ZERO/1H0/, STAR/1H*/
      IF(ICOUNT .EQ. 0) GO TO 20
C
```

```
C
C     CONVERT TO RECTANGULAR COORDINATES
C     AT THE SAME TIME FIND MAX AND MIN VALUES
C
      XMAX = XMIN = YMAX = YMIN = 0
      DO 1 I=1,ICOUNT
      YRAD = Y(I)*DEG2RAD
      XTEMP = X(I)*COS(YRAD)
      YTEMP = X(I)*SIN(YRAD)
      XMAX = AMAX1(XTEMP,XMAX)
      XMIN = AMIN1(XTEMP,XMIN)
      YMAX = AMAX1(YTEMP,YMAX)
      YMIN = AMIN1(YTEMP,YMIN)
      X(I) = XTEMP
    1 Y(I) = YTEMP
C
C     COMPUTE SCALE FACTORS
C
      XRAN = XMAX - XMIN
      YRAN = YMAX - YMIN
      RANGE = AMAX1(XRAN,YRAN)
      XBIAS = 31.0
      YBIAS = 13.0
      IXMAX = 61
      IYMAX = 37
      IF(TTY) GO TO 3
      XBIAS = 61.0
      YBIAS = 37.0
      IXMAX = 121
      IYMAX = 73
    3 XFACT = IXMAX/RANGE
```

```
      YFACT = IYMAX/RANGE
      XBIAS = XBIAS -(XMIN + .5*XRAN)*XFACT
      YBIAS = YBIAS - (YMIN + .5*YRAN)*YFACT
      IY1 = IYMAX + 1
      DO 4 I=1,ICOUNT
      IXX = X(I)*XFACT + XBIAS
      IF(IXX .LT. 1) IXX = 1
      IF(IXX .GT. IXMAX) IXX = IXMAX
      IX(I) = IXX
      IYY = Y(I)*YFACT + YBIAS
      IF(IYY .LT. 1) IYY = 1
      IF(IYY .GT. IYMAX) IYY = IYMAX
C        Y MUST BE INVERTED SINCE WE PLOT FROM THE TOP DOWN
    4 IY(I) = IY1 - IYY
C
C     SORT X,Y PAIRS ON Y
C     USES A SHELL SORT
C
      IF(ICOUNT .LE. 1) GO TO 107
      ID = 1
  110 ID = SHIFT(ID,1)
      IF(ID .LE. ICOUNT) GO TO 110
      ID = SHIFT(ID,-1)-1
  101 DO 106 I=1,ID
      J1 = I
      J2 = I+ID
      GO TO 105
  102 IF(IY(J1) .LE. IY(J2)) GO TO 104
      J3 = J1
      J4 = J2
```

```
103   ITEMP = IY(J3)
      IY(J3) = IY(J4)
      IY(J4) = ITEMP
      ITEMP = IX(J3)
      IX(J3) = IX(J4)
      IX(J4) = ITEMP
      J4 = J3
      J3 = J3-ID
      IF(J3 .LT. 1) GO TO 104
      IF(IY(J3) .GT. IY(J4)) GO TO 103
104   J1 = J2
      J2 = J2 + ID
105   IF(J2 .LE. ICOUNT) GO TO 102
106   CONTINUE
      ID = ID/2
      IF(ID .GE. 1) GO TO 101
C
C     DO ACTUAL PLOT
C
107   IXZ = XBIAS
      IF(IXZ .LT. 1) IXZ = 1
      IF(IXZ .GT. IXMAX) IXZ = IXMAX
      IYZ = YBIAS
      IF(IYZ .LT. 1) IYZ = 1
      IF(IYZ .GT. IYMAX) IYZ = IYMAX
      IYZ = IY1 - IYZ
      I1MAX = IYZ-1
      I2MIN = IYZ+1
      IXZZ = IXZ - 4
      IF(IXZZ .LT. 0) IXZZ = 0
```

```
      ENCODE(23,5,TOPFORM)IXZZ
      PRINT TOPFORM
5     FORMAT(5H("1",,I5,134X,"-270 DEG"))
      PRINT 6,(STAR,I=1,IXMAX),STAR
6     FORMAT(" *",122A1)
      IC = 1
      IF(I1MAX .EQ. 0) GO TO 10
      DO 9 I=1,I1MAX
      DO 7 J=1,IXMAX
7     LINE(J) = BLANK
      LINE(IXZ) = DOT
8     IF(IC .GT. ICOUNT) GO TO 9
      IF(IY(IC) .GT. I) GO TO 9
      LINE(IX(IC)) = ZERO
      IC = IC + 1
      GO TO 8
9     PRINT 6,(LINE(J),J=1,IXMAX),STAR
10    DO 11 J=1,IXMAX
11    LINE(J) = DOT
12    IF(IC .GT. ICOUNT) GO TO 13
      IF(IY(IC) .GT. IYZ) GO TO 13
      LINE(IX(IC)) = ZERO
      IC = IC + 1
      GO TO 12
13    PRINT 6,(LINE(J),J=1,IXMAX),STAR
      IF(I2MIN .GT. IYMAX) GO TO 17
      DO 16 I=I2MIN,IYMAX
      DO 14 J=1,IXMAX
14    LINE(J) = BLANK
      LINE(IXZ) = DOT
```

```
15    IF(IC .GT. ICOUNT) GO TO 16
      IF(IY(IC) .GT. I) GO TO 15
      LINE(IX(IC)) = ZERO
      IC = IC + 1
      GO TO 15
16    PRINT 6,(LINE(J),J=1,IXMAX),STAR
17    PRINT 6,(STAR,J=1,IXMAX),STAR
      ENCODE(26,18,BOTFORM)IXZZ
      PRINT BOTFORM
18    FORMAT(5H(" ",,I5,15HX,"-30 DEG"/"1"))
      ICOUNT = 0
      RETURN
20    PRINT 21
21    FORMAT("-NO POINTS COMPUTED - NO PLOT AVAILABLE")
      RETURN
      END


      SUBROUTINE POLYVAL(W,COEF,ORDER,PR,PI)
      INTEGER ORDER
      DIMENSION COEF(1)
      Z = -W*W
      IF((ORDER .AND. 1) .EQ. 0) GO TO 6
C
C                 ORDER IS ODD
C
      PR = COEF(2)
      PI = COEF(1)
```

```
      IF(ORDER .EQ. 1) GO TO 4
      DO 2  I=3,ORDER,2
      PR = PR*Z + COEF(I+1)
 2    PI = PI*Z + COEF(I)
 4    PI = PI*W
      RETURN
C
C                  ORDER IS EVEN
C
 5    IF (ORDER .EQ. 0) GO TO 12
      PR = COEF(1)*Z + COEF(3)
      PI = COEF(2)
      IF (ORDER .EQ. 2) GO TO 10
      DO 8  I=4,ORDER,2
      PR = PR*Z + COEF(I+1)
 8    PI = PI*Z + COEF(I)
 10   PI = PI*W
      RETURN
C
C                  ORDER IS 0
C
 12   PI = 0.
      PR = COEF(1)
      RETURN
      END


      SUBROUTINE POLARD(REAL,IMAG,MAG,ANG)
```

```
C
C       COMPUTES ANGLE  IN RANGE 0-360
C
        COMMON /CONVERT/ RAD2DEG,DEG2RAD
        REAL MAG,IMAG
        MAG = SQRT(REAL**2 + IMAG**2)
        IF(REAL) 6,1,5
  1     IF(IMAG)2,3,4
  2     ANG = 270.
        RETURN
  3     ANG = 0.
        RETURN
  4     ANG = 90.
        RETURN
  5     ANG = ATAN(IMAG/REAL)*RAD2DEG
        IF(ANG .LT. 0) ANG = ANG + 360.
        RETURN
  6     ANG = ATAN(IMAG/REAL)*RAD2DEG + 180.
        RETURN
        END


        SUBROUTINE POLYRD(A,N,B,MD,OGN)
        DIMENSION A(21),B(21)
        COMMON/COM/X(502),Y(502),Z(502),XD3(502),ICOUNT,TTY,DJM(42),
       1 E(21),F(21),C(40),CTEMP(40), KORD(40),D(21)
        LOGICAL TTY,MM
        IF(TTY) PRINT 35
```

```
35    FORMAT("0TYPE GAIN CONSTANT K -- ")
      READ *,D(1)
      PRINT 36,D(1)
      OGN=D(1)
      FBOG=D(1)
36    FORMAT("0GAIN CONSTANT =",G15.7)
      MULT = 1
      IF(TTY) PRINT 1
1     FORMAT("0TYPE ORDER OF NUMERATOR -- ")
 4    READ *,FMD
      MD = FMD
      IF(MD .GE. 0 .AND. MD .LE. 20) GO TO 5
2     PRINT 3
3     FORMAT("0ERROR IN INPUT DATA.   RUN TERMINATED.")
      STOP
5     MM = .TRUE.
      ND = 1
6     IF(TTY) PRINT 7
7     FORMAT("0TYPE NUMBER OF FACTORS OF A GIVEN ORDER -- ")
      READ *,FNUM
      NUM = FNUM
      IF(TTY) PRINT 9
9     FORMAT("0TYPE ORDER OF THOSE FACTORS -- ")
      READ *,FMORD
      MORD = FMORD
      MORD1 = MORD + 1
      K = NUM*MORD1
      K1 = K + 1
      IF(TTY) PRINT 10 ,K
10    FORMAT(" TYPE",I3," COEFFICIENT(S)"/)
```

```
      DO 12 I=1,K
      READ *,CTEMP(I)
      C(K1-I) = CTEMP(I)
12    CONTINUE
      IF(.NOT.(TTY.OR.MM))GO TO 16
      IF(MULT .EQ. 1) PRINT 13
13    FORMAT("0COEFFICIENT(S) OF NUMERATOR")
      IF (MULT .NE. 1) PRINT 14
14    FORMAT("0COEFFICIENT(S) OF DENOMINATOR")
      MM = .FALSE.
16    DO 18 I=1,MORD1
18    KORD(I) = I-1
      DO 22 J=1,K,MORD1
      JEND = J + MORD
      PRINT 19 ,(CTEMP(I),KORD(JEND+1-I),I=J,JEND)
19    FORMAT(" ",3(1PE13.6," S(",I2,") ")/(21X,E13.6," S(",I2,") ",
     +   E13.6," S(",I2,")"))
      CALL POLYMUL(C(J),MORD1,D,ND,B,NB)
      DO 20 L=1,NB
20    D(L) = B(L)
22    ND = NB
      IF(MD-ND+1) 2,24,6
24    IF(MULT .NE. 1) GO TO 30
      DO 28 I=1,ND
28    A(I) = D(ND+1-I)
      N = MD
      MULT = 2
      IF(TTY) PRINT 29
29    FORMAT("0TYPE ORDER OF DENOMINATOR -- ")
      D(1) = 1.
```

```
      GO TO  4
30    DO 32 I=1,ND
32    B(I) = D(ND+1-I)
      RETURN
      END


      SUBROUTINE POLYMUL(A,NA,B,NB,C,NC)
      DIMENSION A(1),B(1),C(1)
      NC = NA+NB-1
      DO 1 I=1,NA
1     C(NB-1+I)  =  A(I)*B(NB)
      IF(NB .EQ. 1) RETURN
      NB1 = NB-1
      DO 2 I=1,NB1
2     C(I)  = A(1)*B(I)
      IF(NA .EQ. 1) RETURN
      DO 3 I=2,NA
      DO 3 J=1,NB1
3     C(I+J-1)  =  C(I+J-1)  +  A(I)*B(J)
      RETURN
      END


      OVERLAY(COPCSD,3,0)
      PROGRAM PARTL
```

```
      COMMON /POLARC/ AC,BO,FACT,FACTR
      COMMON Z(12),XR,NXX,Y(24),W(24),CR(24),EC(24),OM(24),FE(24),L
      INTEGER DOT,BLANK,CROSS,COLON,TLINE
      LOGICAL ONPAGE,TELL
      DIMENSION A(24),B(24),C(24),D(24),E(24),F(24),P(12),Q(12)
      DIMENSION TLINE(102),FTOR(12),X(503),Y1(503),TITLE(5)
      DATA FACTR/1./,FACT/0./,PI/3.1415927/,DEGRE/57.29578/
      DATA PI2/6.283185307/
      DATA DOT/1H./,BLANK/1H /,CROSS/1HX/,COLON/1H:/,IZERO/1H0/
      DATA FTOR/1.,1.,2.,6.,24.,120.,720.,5040.,40320.,
     +   362880.,3628800.,39916800./
      DATA TELL/.TRUE./
C    INITIALIZE ARRAYS
C    OVERRUN ON Z(I) ALSO SETS XR AND NXX
 1    DO 2   I=1,24
 2    Y(I)=W(I)=CR(I)=OM(I)=FE(I)=EC(I)=Z(I)=0
      H=0
      FIN=0.
C
C     INPUT SECTION
C
C     K = NUMBER OF ZEROS
C     L = NUMBER OF NON-REPEATED POLES IN DENOMINATOR
C     M = POWER OF REPEATED POLE IN DENOMINATOR
C     CONST = CONSTANT IN NUMERATOR
C
      WRITE(7,300)
 5    WRITE(7,223)
      READ *,FK,FL,FM,CONST
      K = FK   $ L = FL   $ M = FM
```

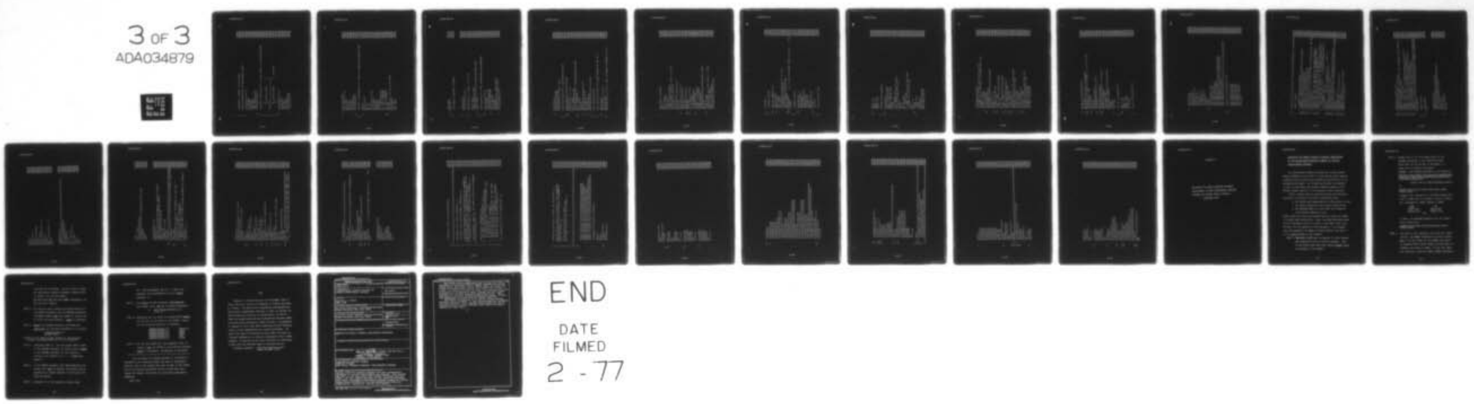
```
C
C     TEST CONSTRAINTS ON K, L, AND M
C
      IF(K .LT. 0  .OR.  K .GT. 12) GO TO 8
      IF(L .LT. 0  .OR.  L .GT. 24) GO TO 8
      IF(M .LT. 0  .OR. M .EQ. 1  .OR.  M .GT. 12) GO TO 8
      IF(K .GE. L + M) GO TO 8
      IF(K .GT. L + 1) GO TO 8
      GO TO 10
C
C     ILLEGAL INPUT
C
 8    WRITE(7,280)
      GO TO 6
C
C     ACCEPT NUMERATOR VALUES
C
 10   IF(K .EQ. 0) GO TO 18
 12   JY=0
 14   JY=JY+1
      WRITE(7,224)
      READ *,A(JY),B(JY)
      IF(B(JY).EQ.0) GO TO 16
      JY=JY+1
      A(JY)=A(JY-1)
      B(JY)=-B(JY-1)
      WRITE(7,255)
 16   IF(JY.LT.K)GO TO 14
C
C     ACCEPT NON REPEATED VALUES IN DENOMINATOR
```

```
C
 18     IF(L .EQ. 0) GO TO 28
        JX=0
 20     JX=JX+1
        WRITE(7,225)
        READ *,C(JX),D(JX)
        IF(D(JX).EQ.0) GO TO 22
        JX=JX+1
        C(JX)=C(JX-1)
        D(JX)=-D(JX-1)
        WRITE(7,255)
 22     IF(JX.LT.L)GO TO 20
C
C       THIS ALGORITHM BASED ON A PAPER BY DOV HAZONY AND JACK RILEY
C       PRESENTED ON 21 AUG, 1959 AT THE AUTOMATIC CONTROL SESSION OF THE
C       WESTERN ELECTRONIC SHOW AND CONVENTION
C
C       THIS SECTION SETS (E(N),F(N)) AS POLAR COORDINATE FORM OF
C
C                               K     DO 30                 !
C             (S+C(N)+JD(N))    TT  (S+A(I)+JB(I))          !
C                               I=1                         !
C       R(N1)=______________________________________________!
C                               L                           !
C             (S+G+JH)**M       TT  (S+C(J)+JD(J))          ! S=-(C(N)+JD(N))
C               DO 38           J=1      DO 34              !
C
C           =E(N)+F(N)
C
 28     IF(L.EQ.0)GO TO 36
```

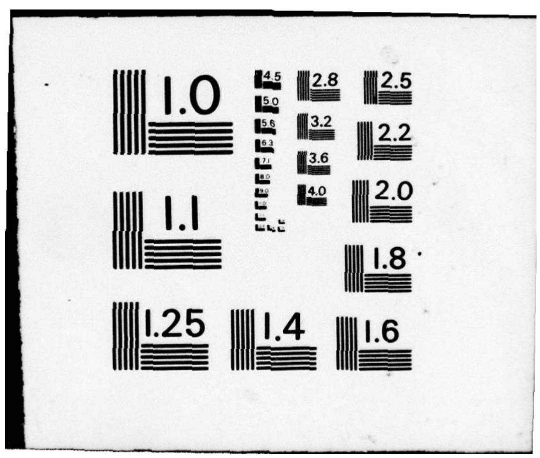
```
      DO 34 N=1,L
      E(N)=1.
      F(N)=0.
      IF(K .EQ. 0) GO TO 32
      DO 30 I=1,K
      AC=A(I)-C(N)
      BD=B(I)-D(N)
      CALL POLAR
      F(N)=F(N)+FACT
   30 E(N)=E(N)*FACTR
   32 E(N)=E(N)*CONST
      DO 34 J=1,L
      IF( J-N  .EQ. 0) GO TO 34
      AC=C(J)-C(N)
      BD=D(J)-D(N)
      CALL POLAR
      E(N)=E(N)/FACTR
      F(N)=F(N)-FACT
   34 CONTINUE
C
C     PARTIAL FRACTION EXPANSION IS COMPLETE IF M = 0
C
   36 IF(M.EQ.0)GO TO 56
C
C     ACCEPT REPEATED VALUES
C
      WRITE(7,226)
      READ *,G
      P(M) = 1.
      Q(M)=0.
```

AD-A034 879 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 9/2
A CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN (CCPC--ETC(U)
DEC 76 F L O'BRIEN
UNCLASSIFIED GE/EE/76D-38 NL

3 OF 3
ADA034879



END
DATE
FILMED
2 - 77


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

```
C
C     PROCEDURE TO HANDLE REPEATED ROOT
C
C     MODIFY COEFFICIENTS OF NON-REPEATED ROOTS
C
      IF(L.EQ.0)GO TO 40
      DO 38 N=1,L
      AC=G-C(N)
      BD=H-D(N)
      CALL POLAR
      E(N)=E(N)/FACTR**M
   38 F(N)=F(N)-FACT*M
C
C     GENERATE COEFFICIENT OF THE M'TH POWER OF THE REPEATED ROOT
C
C           K                        !
C           TT (S+A(I)+JB(I))        !
C           I=1                      !
C     C(M)=______________________!
C           L                        !
C           TT (S+C(J)+JD(J))        !  S=-(G+JH)
C           J=1                      !
C
   40 IF(K.EQ.0)GO TO 44
      DO 42 I=1,K
      AC=A(I)-G
      BD=B(I)-H
      CALL POLAR
      P(M)=P(M)*FACTR
   42 Q(M)=Q(M)+FACT
```

```
44    P(M)=P(M)*CONST
      IF(L.EQ.0)GO TO 48
      DO 46 J=1,L
      AC=C(J)-G
      BD=D(J)-H
      CALL POLAR
      P(M)=P(M)/FACTR
46    Q(M)=Q(M)-FACT
C
C     GENERATE COEFFICIENTS FOR 1 THRU M-1 POWERS OF REPEATED ROOT
C
48    MI=M-1
      DO 54 N=1,MI
      NA=N-1
      AA = 0
      AB = 0
      IF(L.EQ.0)GO TO 52
      DO 50 J=1,L
      AC=G-C(J)
      BD=H-D(J)
      CALL POLAR
      R = E(J)*FACTR**NA
      S = F(J) + FACT*NA
      AA = AA + R*COS(S)
50    AB = AB + R*SIN(S)
52    DELTA=0
      IF( M+L-<-N  .EQ. 0) DELTA=CONST
      AC=DELTA-AA
      BD = -AB
      CALL POLAR
```

```
      P(N)=FACTR
 54   Q(N)=FACT
C
C     END PARTIAL FRACTION EXPANSION

C
 56   WRITE(6,200)
      IF(L.EQ.0)GO TO 60
C
C     REDUCE ANGLES TO RANGE -PI TO +PI
C
      DO 58 J=1,L
      TEMP = AMOD(F(J),PI2)
      IF(ABS(TEMP).GT. PI) TEMP=TEMP-SIGN(PI2,TEMP)
 58   F(J) = TEMP
C
C     PRINT ROOTS WITH THEIR COEFFICIENTS (RECTANGULAR)
C
 60   IF(M.EQ.0)GO TO 66
      DO 62 N=1,M
      A(N)=P(N)*COS(Q(N))
 62   B(N)=P(N)*SIN(Q(N))
      DO 64 NAA=1,M
      N=M+1-NAA
      IF(ABS(B(N)).LE..0001) B(N)=0
 64   WRITE(6,203)G,H,N,A(N),B(N)
      IF(L.EQ.0)GO TO 70
```

```
66    DO 68 N=1,L
      A(N)=E(N)*COS(F(N))
      B(N)=E(N)*SIN(F(N))
      IF(ABS(B(N)).LE..0001) B(N)=0
C
C     PRINT ROOTS WITH THEIR COEFFICIENTS (POLAR)
C
68    WRITE(6,204)C(N),D(N),A(N),B(N)
70    WRITE(6,201)
      IF (M .EQ. 0) GO TO 76
      DO 72 J=1,M
      TEMP = AMOD(Q(J),PI2)
      IF(ABS(TEMP) .GT. PI) TEMP=TEMP-SIGN(PI2,TEMP)
72    A(J) = TEMP*DEGRE
      DO 74 JA=1,M
      J=M+1-JA
74    WRITE(6,202)G,H,J,P(J),A(J)
      IF(L.EQ.0)GO TO 80
76    DO 78 N=1,L
      B(N)=F(N)*DEGRE
78    WRITE(6,205)C(N),D(N),E(N),B(N)
C
C     TIME FUNCTION OUTPUT
C
C     THIS SECTION ALSO SETS UP BLANK COMMON FOR USE BY FT
C
80    WRITE(7,228)
      IF (M .EQ. 0) GO TO 88
C     REAL REP ROOT
C     MULTIPLICATION BY ONE IS USED TO CHANGE A POSSIBLE -0 TO 0
```

```
C       FOR PRINTING PURPOSES
        NXX=1
        XR = 1.*(-G)
        DO 86  JX=1,M
        JY=M+1-JX
        Z(JY) = P(JY)/FTOR(JY)
        IF(ABS(A(JY)) .GE. 170.) Z(JY) = -Z(JY)
        IF(JY .LE. 2) GO TO 82
        JY = JY - 1
        WRITE(7,229) Z(JY+1),JY,XR
        GO TO 86
  82    IF(JY .EQ. 1) GO TO 84
        WRITE(7,237)Z(2),XR
        GO TO 86
  84    WRITE(7,230)Z(1),XR
  86    CONTINUE
  88    IF (L.EQ.0) GO TO 96
        DO 94  NR=1,L
        IF (D(NR))90,92,94
  90    CR(NR)=2.*E(NR)
        EC(NR) = 1.*(-C(NR))
        OM(NR) = 1.*(-D(NR))
        FE(NR)=B(NR)+90.
        WRITE(7,231)CR(NR),EC(NR),OM(NR),FE(NR)
        GO TO 94
  92    Y(NR) = E(NR)
        IF(ABS(B(NR)) .GE. 170.) Y(NR) = -E(NR)
        W(NR) = 1.*(-C(NR))
        WRITE(7,230)Y(NR),W(NR)
        IF(C(NR).EQ.0.)FIN=FIN+Y(NR)
```

```
   94  CONTINUE
C
C      OPTIONS PORTIONS
C
   95  IF(TELL) WRITE(7,227)
       TELL = .FALSE.
   98  WRITE(7,250)
       READ *,FNIX
       NIX = FNIX
       IF((NIX.LT.0.).OR.NIX.GT.5) GO TO 98
       IF(NIX.EQ.0.) GO TO 100
       GO TO (102,106,146,148,1) NIX
  100  WRITE(7,3000)
 3000  FORMAT(//"         --------END OF PROGRAM 'PARTL'--------         "//)
       RETURN
C
C      TABULAR LISTING
C
  102  WRITE(7,250)
       READ *,T,DUR,DT
       N=DUR/DT+1.
       IF(N.GT.200)N=200
       WRITE(6,248)
       DO 104 I=1,N
       FU=FT(T)
       WRITE(6,235)T,FU
  104  T=T+DT
       GO TO 95
C
C      SET UP FOR TTY PLOT
```

```
C
 106  NTOP = 50
      NFILE = 7
      NLINE = 61
      BIASL = 50.
C
C     PRINTED PLOT GENERATOR
C
 108  WRITE(7,234)
      READ *,T,DUR,DT,BIAS
      BIASP = BIAS + 1.5
C     RECORD ORIGINAL TIME
      TORG=T
      N = (DUR/DT) + 1.
C     LIMIT NUMBER OF POINTS PLOTTED
      IF (N .GT. NTOP) N = NTOP
      FMAX=0.
      TP=0.
      DO 110 I=1,N
      FU = ABS(FT(T))
C      TP WILL BE TIME TO THE PEAK VALUE
       IF(FMAX .GE. FU) GO TO 110
      TP = T
      FMAX = FU
 110  T=T+DT
C        RESET TIME
      T=TORG
      SGNE=FT(TP)
      WRITE(NFILE,245)FIN
      WRITE(NFILE,246)TP
```

```
      WRITE(NFILE,249)SGNE
C     IF ORIGIN IS ON PAGE MARK IT WITH A ZERO
      IF(BIASP .LT. 1 .OR. BIAS .GT. BIASL) GO TO 114
      ONPAGE = .TRUE.
      DO 112 J=1,NLINE
  112 TLINE(J)=BLANK
      TLINE(BIASP)=IZERO
      WRITE(NFILE,240)(TLINE(J),J=1,NLINE)
      GO TO 122
  114 ONPAGE = .FALSE.
      IF(SGNE) 116,116,118
  116 FLOW = -FMAX
      FHIGH = (BIASL-BIAS)*FMAX/BIAS
      GO TO 120
  118 FLOW = -BIAS*FMAX/(BIASL-BIAS)
      FHIGH = FMAX
  120 WRITE(NFILE,295)FLOW,FHIGH
C     MAKE Y AXIS WITH DOTS
  122 DO 124 J=1,NLINE
  124 TLINE(J)=DOT
      WRITE(NFILE,240)(TLINE(J),J=1,NLINE)
C     COMPUTE THE MAX AT THE PEAK
      DO 126 J=1,NLINE
  126 TLINE(J)=BLANK
      DO 128 J=1,NLINE,10
  128 TLINE(J) = COLON
C     PUT DOT ALONG THE X AXIS IF WE CAN
      IF(ONPAGE) TLINE(BIASP) = DOT
      IF(SGNE) 130,130,132
      IF(SGNE) 130,130,132
```

```
130   NFINAL=FIN/FMAX*BIAS+BIASP
      GO TO 134
132   NFINAL=FIN/FMAX*(BIASL-BIAS)+BIASP
134   IF(NFINAL .GT. NLINE .OR. NFINAL .LT. 1) GO TO 136
      TLINE(NFINAL) = DOT
136   DO 144 I=1,N
      FU=FT(T)
      IF(SGNE)140,140,138
138   NFU=FU/FMAX*(BIASL-BIAS)+BIASP
      GO TO 142
140   NFU=FU/FMAX*BIAS+BIASP
142   IF(NFU.GT.NLINE.OR.NFU.LT.1) NFU=102
      NTEMP = TLINE(NFU)
      TLINE(NFU)=CROSS
      WRITE(NFILE,240)(TLINE(J),J=1,NLINE)
      TLINE(NFU) = NTEMP
      T=T+DT
144   CONTINUE
      GO TO 96
C
C     SET UP FOR PRINTER PLOT
C
146   NTOP = 200
      NFILE = 8
      NLINE = 101
      BIASL = 100.
      GO TO 108
C
C     SET UP FOR CALCOMP PLOT
C
```

```
148   WRITE(7,285)
      READ *,T,DUR
      WRITE(7,255)
      READ(5,310) TITLE
      WRITE(7,275)
      READ *,FLET
      LETTERS = FLET
      DT=DUR/500.
      DO 150 J=1,501
      Y1(J)=FT(T)
      X(J)=T
150   T=T+DT
      CALL PLOT(0,-3.,-3)
      CALL PLOT(0,3.,-3)
      CALL SCALE(X,7.,501,1)
      CALL SCALE(Y1,5.,501,1)
      CALL AXIS(0,0,4HF(T),4,5.,90.,Y1(502),Y1(503))
      CALL AXIS(0,0,7HT (SEC),-7,7.,0,X(502),X(503))
      CALL LINE(X,Y1,501,1,0,0)
      CALL SYMBOL((7.-LETTERS*.125)*.5,5.30,.125,TITLE,0,LETTERS)
C
C     DRAW 6" X 9" BOX AROUND PLOT
C
      CALL PLOT(-.5,-.5,3)
      CALL PLOT(7.5,-.5,2)
      CALL PLOT(7.5,5.5,2)
      CALL PLOT(-.5,5.5,2)
      CALL PLOT(-.5,-.5,2)
      CALL PLOT(10.,0,-3)
      GO TO 96
```

```
200  FORMAT("-",12X,"ROOT",15X,"POWER",15X,"COEFFICIENT"/"0  REAL",12X,
C
C      OUTPUT FORMATS
C
    +  "IMAG",23X,"REAL",14X,"IMAG")
 201  FORMAT("0  REAL",12X,"IMAG",21X,"MAGNITUDE",10X,"ANGLE")
 202  FORMAT(G12.5,G15.5,I9,G19.5,F14.3)
 203  FORMAT(G12.5,G15.5,I9,G19.5,G17.5)
 204  FORMAT(G12.5,G15.5,8X,"1",G19.5,G17.5)
 205  FORMAT(G12.5,G15.5,8X,"1",G19.5,F14.3)
 223  FORMAT(".TYPE NO. OF ZEROS, POLES, POWER OF REPEATED ROOT, AND NUM
    +ERATOR CONSTANT.")
 224  FORMAT(".TYPE REAL AND IMAG PARTS OF NUMERATOR FACTOR.")
 225  FORMAT(".TYPE REAL AND IMAG PARTS OF DENOMINATOR FACTOR.")
 226  FORMAT(".TYPE ONLY REAL PART OF REPEATED ROOT.")
 227  FORMAT("-CALCULATIONS COMPLETE"/" FOR TABULAR LISTING TYPE 1"/
    +  " FOR TELETYPE PLOT TYPE 2" / " FOR PRINTER PLOT TYPE 3" /
    +" FOR CALCOMP PLOT TYPE 4" /   " FOR ANOTHER PROBLEM TYPE 5" /
    +" TO TERMINATE TYPE 0")
 228  FORMAT(        "-THE TIME FUNCTION IS" /   " F(T)= ")
 229  FORMAT(G19.5,"T**",I2,"  EXP(",G11.5,"T)")
 230  FORMAT(G19.5,"EXP(",G11.5,"T)")
 231  FORMAT(G19.5,"EXP(",G11.5,"T) SIN(",G11.5,"*T+",F8.3,")")
 234  FORMAT(".TYPE INITIAL TIME, PLOT DURATION, DELTA TIME, AND ORDINAT
    +E BIAS.")
 235  FORMAT(1X,2G16.8)
 237  FORMAT(G19.5,"T  EXP(",G11.5,"T)")
 240  FORMAT(1X,101A1)
 245  FORMAT("1",31X,"FINAL VALUE =",G15.8)
 246  FORMAT(3X,"TP =",G13.6)
```

```
243   FORMAT(8X,"T",15X,"F(T)")
249   FORMAT(32X,"PEAK VALUE =",G15.8)
250   FORMAT(".TYPE INITIAL TIME, LISTING TIME DURATION, DELTA TIME.")
255   FORMAT( " CONJUGATE ASSUMED")
260   FORMAT( ".TYPE OPTION")
265   FORMAT( " TYPE TITLE - 50 CHARACTERS MAX"/".",5(9X,","))
275   FORMAT( ".TYPE NO. OF CHARACTERS, INCL SPACES.")
280   FORMAT(" ILLEGAL INPUT FOR NUMBER OF ROOTS OR POLES.")
285   FORMAT(".TYPE INITIAL TIME, PLOT DURATION.")
295   FORMAT("  GRAPH LIMITS ARE",G14.7," TO",G14.7)
300   FORMAT(//" ==========================================================
     +=================="//" ***************START OF PROGRAM 'PARTL'*****
     +**********")
C
C     INPUT FORMATS
C
310   FORMAT(5A10)
      END


      SUBROUTINE POLAR
      COMMON /POLARC/ AC,BD,FACT,FACTR
      DATA PI/3.1415926535/,HALFPI/1.5707963285/
      FACTR = SQRT(AC**2 + BD**2)
      IF(AC) 6,1,5
1     IF(BD) 2,3,4
2     FACT = -HALFPI
      RETURN
```

```
3     FACT = 0
      RETURN
4     FACT = HALFPI
      RETURN
5     FACT = ATAN(BD/AC)
      RETURN
6     IF(BD) 7,8,9
7     FACT = ATAN(BD/AC) - PI
      RETURN
8     FACT = -PI
      RETURN
9     FACT = ATAN(BD/AC) + PI
      RETURN
      END


      FUNCTION FT(T)
      COMMON Z(12),XR,NXX,Y(24),W(24),CR(24),EC(24),OM(24),FE(24),L
      RAD = 3.1415926535/180.
      IF(NXX.EQ.0)GO TO 2
      EX=EXP(XR*T)
      F1 = 0
      DO 1  I=1,12
      F1 = F1 + Z(I)*EX
1     EX = EX*T
      GO TO 3
2     F1=0.
3     F2=0.
```

```
      F3=0.
      DO 4   I=1,L
      F2=F2+Y(I)*EXP(W(I)*T)
    4 F3=F3+CR(I)*EXP(EC(I)*T)*SIN(OM(I)*T+FE(I)* RAD)
      FT=F1+F2+F3
      RETURN
      END


      OVERLAY(CCPCSD,4,0)
      PROGRAM POLY
C     ROOTS OF A NTH DEGREE POLYNOMIAL  (MULLER'S METHOD)
      COMMON RR,RI,A(99),DA(99),DRR(99),DRI(99)
      DOUBLE PRECISION DA,DRR,DRI
      COMPLEX CR
      EQUIVALENCE (CR,RR)
      PRINT 890
  890 FORMAT(//" ===========================================================
     +================"//" ***************START OF PROGRAM 'POLY'******
     +**********")
  211 WRITE(7,4)
    4 FORMAT("0TYPE N, DEGREE OF POLYNOMIAL, ON 1ST LINE, "
     +"OR ENTER '0' TO STOP."/
     -" THEN TYPE N+1 REAL COEFFICIENTS "
     -"(HIGHEST DEGREE COEFF. FIRST),"/
     -" WITH UP TO 80 CHAR. PER LINE, NUMBERS SEPARATED BY COMMAS.")
   40 WRITE(7,5)
    5 FORMAT("0? N= ")
```

```
      READ(5,*) N
      IF(N.LE.0.OR.EOF(5).NE.0) GO TO 500
      IF(N.LT.99) GO TO 50
      WRITE(7,*) "N > 98. TRY AGAIN"
      GO TO 40
 50   M=N+1
 40   WRITE(7,6)
 6    FORMAT(" ? A(I)= ")
      READ(5,*) (A(L),L=1,M)
      AMAX=0.
      WRITE(7,105)
 105  FORMAT(*0 I *5X*A(I)*)
      DO 101 L=1,M
      WRITE(7,110) L,A(L)
 110  FORMAT(1H I2,2X,G15.8)
      DA(L)=A(L)
      IF(ABS(A(L)).GT.AMAX) AMAX=ABS(A(L))
 101  CONTINUE
      DO 220 N1=1,M
      IF(A(N1).NE.0.) GO TO 225
 220  CONTINUE
      WRITE(7,230)
 230  FORMAT(" ERROR...ALL COEFF'S ARE ZERO")
      GO TO 40
 225  NX=M-N1
      CALL DMULR(DA(N1),NX,DRR,DRI)
      PRINT 1004
 1004 FORMAT(//" *********************************************************
     +"/"    PLEASE COPY DOWN THE REAL AND IMAGINARY PARTS (THAT"/
     +" WILL BE SHOWN) IN ROOT AND FACTOR FORM: I.E., ROOT...S=-2;"/
```

```
     +" FACTOR...S+2."/
     +" ********************************************************
     +"//)
      WRITE(7,300)
300   FORMAT(*0 I *2X*REAL:ROOT(I)* 4X*IMAG:ROOT(I)* 4X*RESIDUAL*)
      DO 250 I=1,NX
      RR=DRR(I)
      RI=DRI(I)
      ERR=RESID(A(N1),NX,CR)/AMAX
250   WRITE(7,310) I,CR,ERR
310   FORMAT(1H I2,2X,G15.8,1X,G15.8,1X,1PE8.1)
      GO TO 40
500   CONTINUE
      PRINT 501
501   FORMAT(//"        ---------END OF PROGRAM 'POLY'--------       "//)
      END


      FUNCTION RESID(A,N,X)
      DIMENSION A(1)
      COMPLEX X,C
      C=A(1)
      DO 1 I=1,N
1     C=C*X+A(I+1)
      RESID=CABS(C)
      RETURN
      END
```

```
      SUBROUTINE DMULR (COE,N1,ROOTR,ROOTI)
C
C
C*****************************************************************************
C
C     POLYNOMIAL ROOT FINDER SUBROUTINE ....
C
C     ITERATIVE METHOD FOR POLYNOMIAL EQUATIONS ....
C
C     THIS METHOD APPROXIMATES THE FUNCTION F(Z) BY A QUADRATIC
C   WHICH MAY ,IN GENERAL, HAVE COMPLEX COEFFICIENTS AND DOES NOT
C   REQUIRE A KNOWLEDGE OF THE DERIVATIVE OF F(Z) THOUGH
C   THE FUNCTION F(Z) MUST BE EVALUATED ONCE PER ITERATION ....
C
C     THIS SUBROUTINE FINDS REAL AND COMPLEX ROOTS OF A POLYNOMIAL
C   WITH REAL COEFFICIENTS ....
C
C
C     USE OF MULLER SUBROUTINE ....
C   1.   CALL DMULR  (COE,N1,ROOTR,ROOTI) ....
C   2.   COE IS THE TAG OF THE ARRAY OF COEFFICIENTS.
C        THE COEFFICIENTS MUST BE ORDERED FROM HIGHEST DEGREE
C        TO LOWEST DEGREE .
C   3.   N1 IS DEGREE OF THE POLYNOMIAL .
C   4.   ROOTR IS THE TAG OF THE ARRAY WHERE THE REAL PARTS
C        OF THE COMPLEX ROOTS ARE STORED .
C   5.   ROOTI IS THE TAG OF THE ARRAY WHERE THE IMAGINARY
C        PARTS OF THE COMPLEX ROOTS ARE STORED ....
C
C     ALL ARITHMETIC IS IN THE COMPLEX MODE ....
```

```
C     THEREFORE UNDER-FLOW IS NORMAL FOR REAL ROOTS ....
C
C     MULTIPLE ROOTS DECREASES ACCURACY OF THIS SUBROUTINE .
C     WHEN MULTIPLICITY IS FOUR THE ACCURACY DECREASES TO
C     ABOUT TWO PLACES ....
C
C     DEGREE SQUARED DIVIDED BY TWENTY ....
C     FOR DEGREE ELEVEN IT TAKES SIX SECONDS ....
C
C
C
C*********************************************************************
C
C
C
      DOUBLE PRECISION ROOTR,ROOTI,AXR,AXI,ALP1R,ALP1I,TEM
      DOUBLE PRECISION BET1R,BET1I,ALP2R,ALP2I,BET2R,BET2I
      DOUBLE PRECISION TEMR,TEMI,ALP3R,ALP3I,BET3R,BET3I
      DOUBLE PRECISION ALP4R,ALP4I,TEM1,TEM2,HELL,BELL
      DOUBLE PRECISION TE1,TE2,TE3,TE4,TE5,TE6,TE7,TE8,TE9,TE10
      DOUBLE PRECISION TE11,TE12,TE13,TE14,TE15,TE16,DE15,DE16,COE
C
      DIMENSION COE(1),ROOTR(1),ROOTI(1)
C
      N2=N1+1
      N4=0
      I=N1+1
10    IF(COE(I))9,7,9
7     N4=N4+1
      ROOTR(N4)=0.0D0
```

```
      ROOTI(N4)=0.0D0
      I=I-1
      IF(N4-N1)19,37,19
9     CONTINUE
C
10    AXR=0.8D0
      AXI=0.0D0
      L=1
      N3=1
      ALP1R=AXR
      ALP1I=AXI
      M=1
      GO TO 99
C
11    BET1R=TEMR
      BET1I=TEMI
      AXR=0.85D0
      ALP2R=AXR
      ALP2I=AXI
      M=2
      GO TO 99
C
12    BET2R=TEMR
      BET2I=TEMI
      AXR=0.9D0
      ALP3R=AXR
      ALP3I=AXI
      M=3
      GO TO 99
C
```

```
13    BET3R=TEMR
      BET3I=TEMI
14    TE1=ALP1R-ALP3R
      TE2=ALP1I-ALP3I
      TE5=ALP3R-ALP2R
      TE6=ALP3I-ALP2I
      TEM=TE5*TE5+TE6*TE6
      TE3=(TE1*TE5+TE2*TE6)/TEM
      TE4=(TE2*TE5-TE1*TE6)/TEM
      TE7=TE3+1.0D0
      TE9=TE3*TE3-TE4*TE4
      TE10=2.0D0*TE3*TE4
      DE15=TE7*BET3R-TE4*BET3I
      DE16=TE7*BET3I+TE4*BET3R
      TE11=TE3*BET2R-TE4*BET2I+BET1R-DE15
      TE12=TE3*BET2I+TE4*BET2R+BET1I-DE16
      TE7=TE9-1.0D0
      TE1=TE9*BET2R-TE10*BET2I
      TE2=TE9*BET2I+TE10*BET2R
      TE13=TE1-BET1R-TE7*BET3R+TE10*BET3I
      TE14=TE2-BET1I-TE7*BET3I-TE10*BET3R
      TE15=DE15*TE3-DE16*TE4
      TE16=DE15*TE4+DE16*TE3
      TE1=TE13*TE13-TE14*TE14-4.0D0*(TE11*TE15-TE12*TE16)
      TE2=2.0D0*TE13*TE14-4.0D0*(TE12*TE15+TE11*TE16)
      TEM=DSQRT(TE1*TE1+TE2*TE2)
      IF(TE1)113,113,112
113   TE4=DSQRT(0.5D0*(TEM-TE1))
      TE3=0.5D0*TE2/TE4
      GO TO 111
```

```
C
112   TE3=DSQRT(0.5D0*(TEM+TE1))
      IF(TE2)110,200,200
110   TE3=-TE3
200   TE4=0.5D0*TE2/TE3
111   TE7=TE13+TE3
      TE8=TE14+TE4
      TE9=TE13-TE3
      TE10=TE14-TE4
      TE1=2.0D0*TE15
      TE2=2.0D0*TE16
      IF(TE7*TE7+TE8*TE8-TE9*TE9-TE10*TE10)204,204,205
204   TE7=TE9
      TE8=TE10
205   TEM=TE7*TE7+TE8*TE8
      TE3=(TE1*TE7+TE2*TE8)/TEM
      TE4=(TE2*TE7-TE1*TE8)/TEM
      AXR=ALP3R+TE3*TE5-TE4*TE6
      AXI=ALP3I+TE3*TE6+TE4*TE5
      ALP4R=AXR
      ALP4I=AXI
      M=4
      GO TO 99
C
15    N6=1
C*********************************************************************
33    IF(DABS(HELL)+DABS(BELL)-1.0D-20)18,18,16
16    TE7=DABS(ALP3R-AXR)+DABS(ALP3I-AXI)
      IF(TE7/(DABS(AXR)+DABS(AXI))-1.0D-7)18,18,17
C*********************************************************************
```

```
17     N3=N3+1                                                          038590
       ALP1R=ALP2R                                                      038600
       ALP1I=ALP2I                                                      038610
       ALP2R=ALP3R                                                      038620
       ALP2I=ALP3I                                                      038630
       ALP3R=ALP4R                                                      038640
       ALP3I=ALP4I                                                      038650
       BET1R=BET2R                                                      038660
       BET1I=BET2I                                                      038670
       BET2R=BET3R                                                      038680
       BET2I=BET3I                                                      038690
       BET3R=TEMR                                                       038700
       BET3I=TEMI                                                       038710
       IF(N3-100)14,18,18                                               038720
18     N4=N4+1                                                          038730
       ROOTR(N4)=ALP4R                                                  038740
       ROOTI(N4)=ALP4I                                                  038750
       N3=0                                                             038760
41     IF(N4-N1)30,37,37                                                038770
37     RETURN                                                           038780
C***********************************************************************038790
30     IF(DABS(ROOTI(N4))-1.0D-5)10,10,31                               038800
31     GO TO (32,10),L                                                  038810
32     AXR=ALP1R                                                        038820
       AXI=-ALP1I                                                       038830
       ALP1I=-ALP1I                                                     038840
       M=5                                                              038850
       GO TO 99                                                         038860
33     BET1R=TEMR                                                       038870
       BET1I=TEMI                                                       038880
```

```
      AXR=ALP2R
      AXI=-ALP2I
      ALP2I=-ALP2I
      M=6
      GO TO 99
C
34    BET2R=TEMR
      BET2I=TEMI
      AXR=ALP3R
      AXI=-ALP3I
      ALP3I=-ALP3I
      L=2
      M=3
99    TEMR=COE(1)
      TEMI=0.0D0
      DO 100 I=1,N1
      TE1=TEMR*AXR-TEMI*AXI
      TEMI=TEMI*AXR+TEMR*AXI
100   TEMR=TE1+COE(I+1)
      HELL=TEMR
      BELL=TEMI
42    IF(N4)102,103,102
102   DO 101 I=1,N4
      TEM1=AXR-ROOTR(I)
      TEM2=AXI-ROOTI(I)
      TE1=TEM1*TEM1+TEM2*TEM2
      TE2=(TEMR*TEM1+TEMI*TEM2)/TE1
      TEMI=(TEMI*TEM1-TEMR*TEM2)/TE1
101   TEMR=TE2
103   GO TO (11,12,13,15,33,34),M
      END
```

APPENDIX D

PROCEDURE FOR ADDING COMPUTER PROGRAMS (ALGORITHMS) TO THE CONSOLIDATED COMPUTER PROGRAM FOR CONTROL DESIGN (CCPCSD)

(DECEMBER 1976)

Procedure for Adding Computer Programs (Algorithms) to the Consolidated Computer Program for Control System Design (CCPCSD)

The Consolidated Computer Program for Control System Design (CCPCSD) is structured to allow adding other computer programs without causing any substantial modification to the programs being added. The following procedure is prepared to help an individual add another computer program to the CCPCSD. Please read all of the procedure before starting.

(NOTE: Caution must be exercised before the following procedure is started as two major limitations exist.

a. The CCPCSD uses approximately $25{,}000_8$ words of core; the added program must not exceed $25{,}000_8$ words.

b. The maximum amount of files that can be used at the intercom terminal is ten.

Hence, make sure that your program does not cause the CCPCSD to exceed these limitations. Also, the steps mentioned below do not necessarily have to be done in the order given. And, further, for the purpose of this procedure, it is assumed that the program to be added is called "PROJX," and that it is the <u>fifth</u> program in the CCPCSD.)

STEP 1. IMPORTANT: PROJX must be capable of fully independent operation from an intercom terminal. (All of the errors must have been removed <u>before</u> adding the program to the CCPCSD.)

STEP 2. Compare all of the file names listed in the PROGRAM statements of the CCPCSD and PROJX. PROJX must now be altered, if necessary, to agree with the CCPCSD file names.

EXAMPLE: The PROGRAM statement for the CCPCSD is:

```
PROGRAM CCPCSD(INPUT=100B,OUTPUT=100B,ANSWER,PPLOT,
CFILE,PLOT,TAPE5=INPUT,TAPE6=ANSWER,TAPE7=OUTPUT,
TAPE8=PPLOT,TAPE9=100B).
```

Assume that the PROJX PROGRAM statement is:

```
PROGRAM PROJX(INPUT,OUTPUT,TAPE1=INPUT,TAPE2=
OUTPUT,COUNT).
```

a. All of the statements in the PROJX program that refer to TAPE1 must be changed to refer to TAPE5; and, consequently, TAPE2 changed to TAPE7.

From	To
READ(1,200)	READ(5,200)
WRITE(2,210)	WRITE(7,210)

ETC.

b. Make a new PROGRAM statement with the CCPCSD files corrections:

```
PROGRAM PROJX(INPUT,OUTPUT,TAPE5=INPUT,TAPE7=
OUTPUT,COUNT).
```

STEP 3. This step is very important, and could get rather tedious. If COMMON statements are used in PROJX, all of the file names for the COMMON statements in programs ROOTL, PARTL, FREQR, and POLY must be compared with those in PROJX. If any file names are identical, modify the PROJX COMMON statements by using new file names. (Do not forget to make the appropriate changes throughout program PROJX to reflect the new file names.)

If PROJX does not have any COMMON statements, you may proceed to STEP 4.

STEP 4. Now that you have finished the modifications for the COMMON statements and the PROGRAM statements throughout PROJX, test the program to insure that it still functions properly. Debug, if necessary.

STEP 5. Remove the PROGRAM statement from PROJX and substitute the following statements in its place:

```
OVERLAY(CCPCSD,5,0)
PROGRAM PROJX
```

Now you are ready to make changes to the executive overlay (Overlay(CCPCSD,0,0)) of the CCPCSD.

STEP 6. (Reference STEP 2) - The file named COUNT, shown in the PROGRAM statement for PROJX, must be added to the PROGRAM statement for the executive overlay of the CCPCSD (i.e., ". . .TAPE9=100B, COUNT)").

STEP 7. To the FORMAT statement 1005 (approximately line number 730), add the symbolic name PROJX, and an explanation of PROJX (similar to those given for ROOTL and PARTL).

STEP 8. Statement 18 of the executive overlay reads

"18 IF(J.EQ.5HPARTL) GO TO 9." Insert the statement "IF(J.EQ.5HPROJX) GO TO 21" after statement 18.

STEP 9. Just before the END statement (approximately line number 1110), add the following statements:

```
21    CALL OVERLAY(CCPCSD,5,0,0)
      GO TO 10
```

STEP 10. Physically put the cards for program PROJX behind (at the end) of the cards in the CCPCSD. Hence, the new structure would be (in general):

```
OVERLAY(CCPCSD,0,0)        Executive
OVERLAY(CCPCSD,1,0)        ROOTL
OVERLAY(CCPCSD,2,0)        FREQR
OVERLAY(CCPCSD,3,0)        PARTL
OVERLAY(CCPCSD,4,0)        POLY
OVERLAY(CCPCSD,5,0)        PROJX
```

STEP 11. Put the new CCPCSD into the permanent files, or library. Test the CCPCSD at the intercom terminal. Debug, if necessary. Re-catalog, if necessary.

* *

This procedure for adding programs to the CCPCSD is developed to be relatively simple and easy to understand. However, some of the changes that must be made to the CCPCSD and to the program being added should be made with care. Typing the changes incorrectly can cause many unnecessary headaches.

Good luck.

VITA

Frederic L. O'Brien was born on 20 November 1943 in Ilion, New York, the son of Frederick B. O'Brien and Helen M. O'Brien. Six Months after graduating from Zephyrhills High School, Zephyrhills, Florida, in 1961, he entered the United States Air Force as an Airman Basic. In 1967 he began the Airman Education and Commissioning Program (AECP) at Arizona State University, Tempe, Arizona. He graduated in January of 1971, and, after completing Officer Training School, he was commissioned as a Second Lieutenant. He spent four years at Griffiss Air Force Base, New York, as a Project Engineer for a Rome Air Development Center (RADC) program. He entered the Air Force Institute of Technology in May 1975 with fourteen years of military service.

Permanent Address: 4299 West Humphrey Street
Tampa, Florida 33614

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER GE/EE/76D-38	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A CONSOLIDATED COMPUTER PROGRAM FOR CONTROL SYSTEM DESIGN (CCPCSD)		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Frederic L. O'Brien Capt, USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology(AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Flight Dynamics Laboratory(AFFDL) Wright-Patterson AFB, Ohio 45433		12. REPORT DATE December 1976 13. NUMBER OF PAGES 220
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17
Jerrel F. Guess, Captain, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Consolidated Computer Program
Combined Program
Root Locus, Frequency Response, Time Response Programs

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Several different computer programs have been independently developed for control system design and analysis. Each program may have its own input and output formats (different from any other), and each program may have its own operating philosophy. This report describes a Consolidated Computer Program for Control System Design (CCPCSD) and its development, specifications, organization, realization, testing, and results. (cont on p 1473B)

DD FORM 1 JAN 73 1473 A EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

The CCPCSD contains four previously written Air Force Institute of Technology (AFIT) computer programs for control system design algorithms (ROOTL, FREQR, PARTL, and POLY) and a lead-in program which controls the overall flow of the CCPCSD. An overlay structure is used. The structure consists principally of an executive (main) overlay and four primary overlays (one for each subprogram mentioned above). Each subprogram is modified towards standardizing inputs, termination procedures, and data structure (where feasible).

The CCPCSD is developed to be used by control systems engineers and/or students. A user-oriented approach permeates the entire program. This computer-aided design tool can be operated from an intercom terminal, where operator-computer interaction takes place.