1976-06

Computer aided control system design using frequency domain specifications.

Mancini, Anthony J.

Monterey, California. Naval Postgraduate School

COMPUTER AIDED CONTROL SYSTEM DESIGN USING FREQUENCY DOMAIN SPECIFICATIONS

Anthony J. Mancini

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

COMPUTER AIDED CONTROL SYSTEM DESIGN USING
FREQUENCY DOMAIN SPECIFICATIONS

by

Anthony J. Mancini

June 1976

Thesis Advisor: Dr. G. J. Thaler

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Computer Aided Control System Design Using Frequency Domain Specifications		5. TYPE OF REPORT & PERIOD COVERED Engineer's Thesis (June 1976) 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Anthony J. Mancini		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE June 1976 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		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

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

Control System Design
Computer Aided Control System Design
Series Compensation
Linear Control Systems
Computer Aided Design
Transfer Function Synthesis
Frequency Response Techniques

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

The primary intent of this work is to investigate computer aided compensator design using classical frequency response techniques in conjunction with the techniques of modern mathematical programming. A computer program to perform the automated design task is presented. This particular algorithm is based on the constrained optimization technique introduced by M. J. Box.

In particular, the desired open loop frequency response is

20\. specified for a number of discrete frequency points over the frequency range of interest. Then the minimization routine is used to vary the compensator parameters in such a manner as to minimize a cost functional based on the difference between the actual and desired open loop frequency response of the compensated system. To illustrate the algorithm several detailed examples using the program are presented.

COMPUTER AIDED CONTROL SYSTEM DESIGN USING FREQUENCY DOMAIN SPECIFICATIONS

by

Anthony Joseph Mancini
Lieutenant United States Navy
B.S.E.E., University of Oklahoma, 1970
M.S.E.E., Naval Postgraduate School, 1975

Submitted in partial fulfillment of the requirements for the degree of

ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL

June 1976

ABSTRACT

The primary intent of this work is to investigate computer aided compensator design using classical frequency response techniques in conjunction with the techniques of modern mathematical programming. A computer program to perform the automated design task is presented. This particular algorithm is based on the constrained optimization technique introduced by M. J. Box.

In particular, the desired open loop frequency response is specified for a number of discrete frequency points over the frequency range of interest. Then the minimization routine is used to vary the compensator parameters in such a manner as to minimize a cost functional based on the difference between the actual and desired open loop frequency response of the compensated system. To illustrate the algorithm several detailed examples using the program are presented.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

The author would like to express his sincere thanks and appreciation to Dr. G. J. Thaler for his continued encouragement, advice, and unselfish sharing of professional expertise during the research and development of this thesis.

A word of thanks is also owed to three members of the Naval Postgraduate School Computer Center Operations Staff, Mr. Ed Donnellan, Mrs. Kris Butler and Mr. M. Anderson, whose professional services contributed extensively to the efficient use of the time necessary in performing the large amount of computer work associated with the development, debugging, and testing of the program presented.

Finally, to my wife and children goes a special debt of gratitude for their patience, understanding, and encouragement during the months of development and research of this thesis. It would not have been possible to accomplish this undertaking without their continued help throughout this period.

I. INTRODUCTION

Concurrent with the proliferation of the digital computer as a viable, indeed often indispensable, engineering tool, the development of state-space methods resulted in the application of the digital computer in the area of modern control theory over the past decade and a half. Initially this application was primarily of an analysis nature, but as the theory of optimal control came into wider publication and acceptance, the role of the digital computer began to take on aspects of the design function. This situation is graphically illustrated by the myriad of industrial proprietary and university developed computer programs for computer aided control system design based upon state variable and optimal control theories. However, as pointed out by Mitchell [11] comparatively little work has been done to extend and apply the computational power, speed, and accuracy of modern digital computers to the proven classical theory and methods of control system design. After this essential abandonment of the classical methods (particularly by the academic and theoretical factions of the control engineering community) in favor of the modern theory and techniques, recent publications indicate a realization of the need that both approaches to system design be used in a complimentary rather than mutually exclusive manner. In particular the works of Coffey, MacFarlane, Mitchell, Page, Rosenbrock, Stear, [5,8,11,14] and others show a renewed interest in the tried and proven classical theory of design coupled with the new techniques and approaches developed by proponents of modern control theory, as applicable to the development of usable computer aided control system design algorithms.

Control system design based on frequency domain specifications, for linear time invariant parameter systems using classical methods, has traditionally been done essentially using trial and error techniques. That is, once the form and location of the compensator has been decided upon, the design process reduces to one of iterative analysis until the specifications are met or it becomes obvious to the designer that the compensator form chosen will not satisfy the specifications. When systems of any complexity are involved this method quite often transforms into an extremely tedious and time consuming process.

The availability of a good analysis program coupled with a digital computer operating in an interactive mode, to provide minimal turn around time, can be used to somewhat minimize the tedium of this type of design procedure. By varying the compensator parameters the designer may then observe the effects upon system performance without having to expend considerable effort in repeated calculations. This design by iterative analysis may be continued allowing the computer to perform the calculations until the designer is satisfied with the response obtained. With the recent refinements in mathematical programming techniques, it is possible to relegate some of the design decisions directly to the computer program, thus even further minimizing the time and efforts required of the design engineer. The ultimate goal, which is of course at present a long way down the road, would be to input only the plant dynamics and the desired specifications and have the computer perform the design process completely, without any interaction with the design engineer.

In this thesis an investigation of the automation of the design procedure is undertaken. In particular, given the desired open loop response in the frequency domain a minimization routine is used in conjunction with an analysis algorithm to systematically vary the compensator parameters to approximate the desired response in accordance with a performance measure which indicates the deviation of the actual open loop frequency response from the desired open loop frequency response.

II. THE AUTOMATED DESIGN PROBLEM

A. GENERAL AUTOMATED DESIGN PHILOSOPHY

The word design, as used in many engineering texts of various disciplines, runs the gamut of describing a process dealing essentially with scaling already existing configurations and components to one dealing with a highly creative activity allowing a maximum amount of freedom in satisfying a set of requirements or specifications. On a day to day basis the majority of the design work accomplished by engineers lies somewhere between these two extremes, combining elements of relatively routine, straight forward calculations with those of creativity and imagination.

The overall scope of the design process in general is indeed so formidable that it often defies complete description in other than general terms, much less quantification in precise mathematical language. This latter is unfortunately a necessity in achieving the automation of this process within the limitations of the presently available hardware and software accessable by the engineer. Largely for this reason, when considering the automatic design problem it must by necessity be viewed, for the near future at least, in terms of a limited scope. That is, such items as creativity, experience, and intuition, which are undoubtedly elements of the total design process as viewed in its entirety, cannot be achieved by a computer as the device exists at the present time.

One of the key elements of any design process is decision making. In the overall design process the engineer makes decisions based upon the information available to him at the time of the design undertaking. Quite often these decisions are based on a comparison of numerical values of various elements of the design configuration. It is in this specific area of the overall design process that the modern day computer can serve as a significant tool for the engineer involved in design, particularly when a large number of comparisons (expressable in numerical terms) are involved. In other words, the computer can be used in a combination analysis and elementary decision making role to aid the engineer in the overall design procedure. By means of analysis numerical values may be obtained, forming a basis from which comparisons can be drawn. The analysis portion is generally essential for exact design and forms the basis for any optimization. The speed available in the digital computer allows many calculations and comparisons, over a wide range of possible solutions, to be performed which would be prohibitive if peformed by hand.

The employment of the digital computer as an aid in the aggregate design process is perhaps as much an art as the design process itself. While the use of computer aided design programs can free the engineer from many burdensome and time consuming calculations, the individual engineer must still provide the framework in which to interpret the results in the hard light of physical significance. The engineer must also be capable and willing to define the problem in a form with which the computer can work. More often than not, this latter requires a much sharper, clearer definition (and subsequently understanding) of the problem than if the same problem were being designed long hand. Unlike the analysis problem, which generally possesses a specific answer, the design problem may not have a unique solution which further complicates any computer aided design algorithm. Another aspect that should not be overlooked is the fact that when using a computer aided design program the engineer often receives, if nothing else, an indication of the solutions which are not feasible. This in itself is often not a trivial result especially in dealing with systems of any complexity.

Another important factor which should not be overlooked is that once an answer is obtained using a computer aided design algorithm some type of sensitivity study will more than likely be necessary. This is largely due to the fact that component values in physical hardware cannot generally be controlled to the precision with which a computer does its calculations. Therefore, while a particular algorithm for use in automated design may generate what is indeed a theoretically plausible solution the implementation of such a solution may not be possible if it is extremely sensitive to the parameter values involved.

B. SOME ASPECTS OF COMPUTERIZING THE CLASSICAL DESIGN OF CONTROL SYSTEMS

Classically the purpose of the design problem in feedback control systems has been to make a given plant satisfy a set of performance specifications imposed upon the system. Design of linear, time invariant compensators for the control of linear systems has formed a major portion of the design efforts of conventional servomechanisms. In the process of investigating the automation of this design procedure, certain key elements were found to be generally necessary in implementing a basic practical computer algorithm for automated design, regardless of whether it is based on time or frequency domain approaches. These elements consisted of:

1. A complete description of the plant or process to be controlled.

2. Specifications describing the response desired from the system.

3. Some type of criterion function or performance measure relating the actual to the desired response.

4. The form or structure of the compensator or control to be implemented.

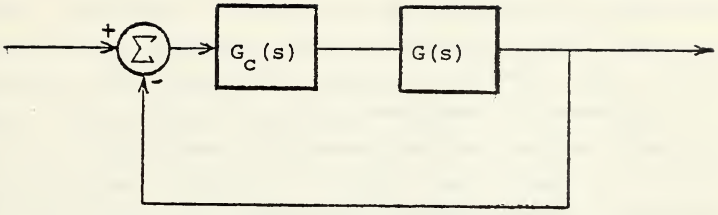
These items appear in varying degrees as a common denominator among present day automated design algorithms [1,5,11,13] regardless of the specific methods employed or systems being considered. That is to say, these elements appear in algorithms designed for time domain, transform domain, stochastic processes, and nonlinear systems. Generally items 1. and 4. are employed in some form of analysis algorithm while items 2. and 3. form the basis for a minimization (or maximization) routine that provides a foundation for some type of decision making within the algorithm. This idea of using a performance measure upon which to base some type of decision making routine is a direct carryover from developments in the computerized design of optimal control systems which relies heavily upon the idea of minimizing some criterion function. Unlike the techniques of optimal control however, the cost functional as used in this context is not necessarily a function of the states and the applied control, but rather a measure of the deviation of the actual response from some desired response which has been specified by the designer. This is perhaps a fine distinction but one that must be recognized and kept in mind. One of the many consequences of this is that there is no longer a strong motivation to restrict the performance measure to a quadratic form in order that the second method of Liapunov may be used to prove that a stable system will result. Rather, the form of the cost function in computerized design based upon classical control theory is chosen to have properties which aid in the convergence of the numerical minimization routine used in the algorithm or which has some inherent physical significance in the design process.

C. A FORMULATION OF THE AUTOMATIC DESIGN PROBLEM IN THE FREQUENCY DOMAIN

In this thesis the automatic design problem based on theory developed for classical design methods using frequency domain procedures is investigated. Several approaches based on frequency domain techniques have been presented by various people to automate the control system design process [5,11,13,14,20]. These algorithms range from methods designed primarily for use with multivariable systems to specialized interpretations of frequency domain data such as vector frequency domain techniques and inverse Nyquist methods. The suitability of these various methods have been demonstrated by their respective authors and others working in the area of computer aided control system design. While each represents a step forward in the line of progress toward automating the design process using frequency domain techniques the suitability of each algorithm is dictated to some extent by the particular type of design problem being solved. Many of these methods require gradient calculations, and those that do not rely primarily on simplex nonlinear programming algorithms. The former methods restrict the form of the cost function to be minimized in the sense that it must be differentiable and problems inherent with numerical differentiating algorithms are obviously involved. The latter methods involving simplex algorithms have difficulty in handling constrained minimization problems, particularly when solutions exist near or on the constraint boundaries.

One of the advantages of the frequency response approach lies in the fact that it lends itself easily to the use of open-loop frequency response data in designing closed loop systems. The use of frequency response methods provides a graphical portrayal of the system response that lends itself well to quick interpretation by the designer. Also since approaches such as those developed by Bode were intended to give simple approximate methods for use in hand calculations, adaptation of these methods in a computer algorithm allows the engineer to plan his design using such techniques and then employ the aid of the computer without the requirement of significant problem restructuring in order to cast the problem in some particular format not generally familiar to the designer, but necessary for solution by a computer algorithm. With systems involving complex transfer functions such restructuring can itself become quite burdensome thus defeating one of the primary reasons for turning to computer aided design. Perhaps it should be pointed out at this point that even using a computer algorithm to aid in the design process does not totally relieve the engineer of all hand calculations. Time must be taken to carefully map out a plan of attack on a particular problem. In essence the designer must throughly understand the problem and this is generally achieved by some preliminary rough sketches and calculations before the computer ever makes its appearance in the design process.

Throughout this thesis, the vehicle used for presenting the automated design algorithm is the single-input single-output unity feedback system of the form shown in figure II-1.



General System, Block Diagram

figure II-1

The plant is considered to be linear, time invariant, and completely known. Thus the plant may be described by a rational transfer function in the Laplace variable s of the form

$$G(s) = \frac{\prod_{i=0}^{n} (s + z_i)}{\prod_{i=0}^{m} (s + p_i)}$$

where z_i and p_i may be complex constants. The compensator $G_c(s)$ is also considered to be a rational transfer function in s.

The specifications are given in terms of the system open loop frequency response, and the desired response is achieved by reshaping the system open loop frequency response through the use of the compensator in the feed forward path. This basic procedure is well known to most control engineers and is by no means a novel approach to the design problem [3,10,17,18], but through the years it has found extensive use in the practical design of control systems.

The method proposed and under investigation here is a relatively straight forward adaptation of this technique with a computer algorithm harnessed to do the compensator parameter variation in order to systematically reshape the system open loop frequency response with a minimum amount of calculation and iteration required on the part of the designer. The desired open loop response is specified by means of a magnitude and phase profile over the frequency range of interest. This is nothing more than a set of desired magnitude and corresponding phase values at discrete frequency points selected by the designer. In addition a specific form or structure is postulated for the compensator transfer function $G_c(s)$. At this point of the automated design process the algorithm relies heavily on the experience and intuition of the individual designer in choosing a structure or form of compensator that will have some reasonable probability of success in reshaping the frequency response curves in order to satisfy the design specification. Since no firm analytical basis exists for the determination of a specific structure for the compensator, the computer algorithm must be given this information or an iterative method based on some type of criterion must be used in order to establish the form. This latter method was felt to be much too time consuming and restrictive, because of the wide range of variations in possible structures that might have to be considered, and essentially beyond the scope of this investigation. Thus the choice of the compensator form is relegated to the program user as a design decision that must be undertaken prior to using the algorithm to determine specific values for the coefficients of the compensator transfer function that will yield the desired response. This procedure should not present a major stumbling block for the designer provided that he possesss a thorough understanding of the problem to be solved. In fact, by inputing the specific form of the compensator the designer may investigate compensator configurations which might otherwise be considered unfeasible for the solution of the problem if some type of iterative algorithm were used in determining the structure. With these items forming the basic inputs to the algorithm a nonlinear performance measure coupled with a minimization routine is then used to minimize the difference between the desired and actual frequency response of the open loop transfer function given by,

$$T(s) = G_c(s) \bullet G(s)$$

The performance measure (or cost function) selected is of a normalized form primarily as a matter of convenience in interpreting the algorithm's results regardless of the particular problem being considered. That is , the cost function has a general form given by

$$J(j\underline{\omega}) = f(1.0 - \frac{A(j\underline{\omega})}{D(j\underline{\omega})})$$

where $D(j\underline{\omega})$ represents the desired open loop frequency response of the entire system, $A(j\underline{\omega})$ is the actual open loop response, and $\underline{\omega}$ represents the set of discrete frequency values at which the cost function is to be evaluated. As mentioned previously this general form of normalized cost function was selected primarily as a matter of convenience in that a particular cost function can be constructed in such a manner that the optimal or smallest value of $J(j\underline{\omega})$ under ideal solution of the problem will be zero.

By inputing the desired response in the form of the magnitude and phase profile described above, normal frequency domain specifications such as gain margin, phase margin, and bandwidth (here defined as the open loop gain crossover frequency) may be supplied to the algorithm in one type of format. With these specifications forming part of the gain and phase profiles they are automatically incorporated into the overall cost function each time the algorithm evaluates the frequency response of the open loop system over the range of frequencies specified by the designer, thus also eliminating the requirement of specialized computation within the algorithm to check for satisfaction of these particular specifications as the compensator parameters are varied.

III. THE MINIMIZATION TECHNIQUE USED

A. GENERAL

A variety of functional minimization techniques have come into use in recent years are discussed in ref. [8]. The minimization algorithm used in this work was developed originally by M. J. Box, [4], and is capable of finding the minimum of a general nonlinear cost function, composed of several variables, within a constrained region. The complex method of Box has been implemented in program form as part of the standard subroutine library at the Naval Postgraduate School W. R. Church Computer Center. The subroutine (BCXPLX), was originally programmed by R. R. Hilleary of the staff of the Naval Postgraduate School Computer Center and has subsequently been modified slightly by the author in order to render it more suitable for use in the scheme of the compensator optimization program.

In addition to its availability as a standard subroutine, several other features of this particular minimization algorithm made it an attractive choice for use in minimizing a cost function based on frequency domain specifications. While admittedly more efficient and sophisticated algorithms for function minimization are available, the flexibility and ease of programming of the complex method of Box are among considerations that should not be overlooked. This method does not require that the derivatives of the cost function or implicit constraints be calculated, thus avoiding the sundry problems typically associated with numerical methods for the calculation of derivatives, and eliminating the restriction that the derivative of the cost function exist over the range of interest. In implementing the minimization method the cost function need not be expressed explicitly in terms of the variables to be manipulated by the program in the minimization process. Unlike many of the other minimization programs available, BOXPLX, also provides a restart capability. That is, once a possible minimum value of the cost functional is reached the program automatically restarts from randomly generated initial values of the variables. While this method by no means guarantees that a global minimum will be achieved for every conceivable problem the restart capability does at least somewhat attempt to avoid the problem of the minimization terminating on a local minima within the search area. This method is also to some extent scale independent in that the size of the initial geometric figure used in the search for the minimum value of the objective function is roughly scaled to the order of the problem variables, through the use of the difference between the lower and upper bounds on the variables.

As is the case with other minimization algorithms, this particular one is not void of inherent disadvantages and difficulties. Present in any minimization algorithm of course is the recurring problem that no guarantee can be made of achieving a global minimum for all classes of problems. The complex method cannot handle equality constraints without modification of the algorithm. Also as the number of variables increases the method rapidly becomes inefficient. Finally the unconstrained problem is apparently more efficiently handled via gradient techniques.

B. DESCRIPTION OF THE ALGORITHM

The minimization method developed by Box has its basis in the sequential simplex techniques of linear programming, but is specifically designed to obtain solutions to problems of constrained optimization and to avoid such problems as the inability of constructing the first simplex. Thus the resulting solution of the minimization problem is a set of variables which may lie at the extreme edges or boundaries of their permissible ranges. The problems generally associated with constraints on the variables is essentially attacked by using a flexible geometric figure having n+1, or more, vertices and capable of expanding or contracting, in any or all directions. Therefore, unlike many of the simplex algorithms no provision is made to maintain a regular geometry between the various points in an n-dimensional space, but rather a changing structure (dubbed a complex), capable of flattening, rounding corners, and eventually collapsing on the point representing the minimum of the objective function, is employed rendering the method more flexible in handling constrained minimization problems. The number of independent variables in the objective function and constraint equations dictate either directly or implicitly the dimensionality of the space to be searched. The vertices making up the complex are continually rejected and generated as a search for the minimum of the objective function is carried out with each new vertex generated being required to satisfy all the imposed constraints.

The complex method searches for the minimum of an objective function $J(\underline{x})$, where the vector $\underline{x}$ represents a set of n variables, within a region of space bounded by upper and lower limits on the variables (explicit constraints) given by,

$$L_i \leq x_i \leq U_i \ , \ i=1,2,3,\ldots,n$$

and in addition subject to the restrictions imposed by the constraint functions (implicit constraints) of the form,

$$h_j(\underline{x}) \geq 0 \quad j=1,2,3,\ldots,m.$$

As with the simplex methods this is an iterative algorithm. It uses a set of $k \geq n+1$ points which form the vertices of the complex and simultaneously satisfy all the imposed constraints. Initially only one point or vertex must satisfy all the constraints and serves as a means of generating the other k-1 vertices through the use of pseudorandom numbers r_i, uniformly distributed over the closed interval from zero to one. The additional k-1 vertices of the first complex are established by the following formula:

$$x_i = L_i + r_i \bullet (U_i - L_i)$$

As can be seen from the formula, the vertices generated in this fashion will always satisfy the explicit constraints, however, each one must be checked to insure compliance with the implicit constraints. If an implicit constraint violation is indeed found to occur at this stage then the trial vertex is simply moved, repeatedly if necessary, halfway in toward the centroid of the other already accepted vertices, until ultimately a feasible point is found. Continued application of this procedure will result in the generation of the k-1 additional points required for the initial complex within the feasible region defined by the constraints.

Once the initial complex has been determined the iteration proceeds by evaluating the objective function $J(\underline{x})$ at each vertex. The vertex at which the objective function has the largest value is then designated as the current worst vertex and this point is reflected through the centroid of the remaining vertices establishing a new complex. If this worst vertex is designated as $\underline{x}^W$ then the overreflection is accomplished according to the formula

$$\underline{x}^N = (1.0 + \text{alpha}) \bullet \underline{x}^{CR} - \text{alpha} \bullet \underline{x}^W$$

where alpha ≥ 1.0 is the overreflection coefficient, $\underline{x}^{CR}$ represents the centroid of the remaining vertices, and $\underline{x}^N$ is the new vertex. If the overreflection of the current worst vertex results in a violation of the constraints or if the new point still has the worst value in the set of vertices then the point is moved halfway toward the centroid used in the overreflection process. This retraction is repeated (if necessary) until the overreflection coefficient is reduced to some small value, delta, or a suitable new vertex is established. If the overreflection coefficient is reduced to the value delta and the objective function value at this new point is still the worst in the set of vertices then the projected vertex is replaced by its original value and the second worst vertex will be overreflected instead. This process keeps the complex moving toward the minimum of the objective function provided the figure has not collapsed into its centroid. Once assured that the new complex satisfies all the constraints and results in an improvement of the objective function, vertices of the complex are rejected and generated in a systematic manner aimed at minimization of the objective function. Thus essentially the complex moves over the feasible region eventually straddling and collapsing upon the point that renders a minimum value for the objective function.

This process continues and is eventually terminated when the complex shrinks to a predetermined acceptable small value epsilon given by:

$$\left\{ \frac{1}{k} \sum_{l=1}^{k} [J(\underline{x}^{C}) - J(\underline{x}^{V})]^{2} \right\}^{0.5} < \text{epsilon}$$

Where $J(\underline{x}^{C})$ represents the value of the objective function at the centroid and $J(\underline{x}^{V})$ is the value at the various vertices.

As pointed out by Box [4], the exact values of the overreflection coefficient, alpha, and the exact number of vertices, k, do not appear to be critical to the functioning of the algorithm provided that they are greater than unity and n+1 respectively. The main concern here is that a value of alpha larger than unity keeps the complex from shrinking prematurely as it traverses the feasible region in search of the minimum objective function value. Values of alpha = 1.3 and k = 2n as given in ref. 4 are used here since the exact values chosen are apparently not critical and good results have been reported, by the originator of the algorithm, using these particular values. For a detailed example of the iteration process and a more sophisticated explanation of the algorithm the interested reader is directed to refs. 2, 4, and 6.

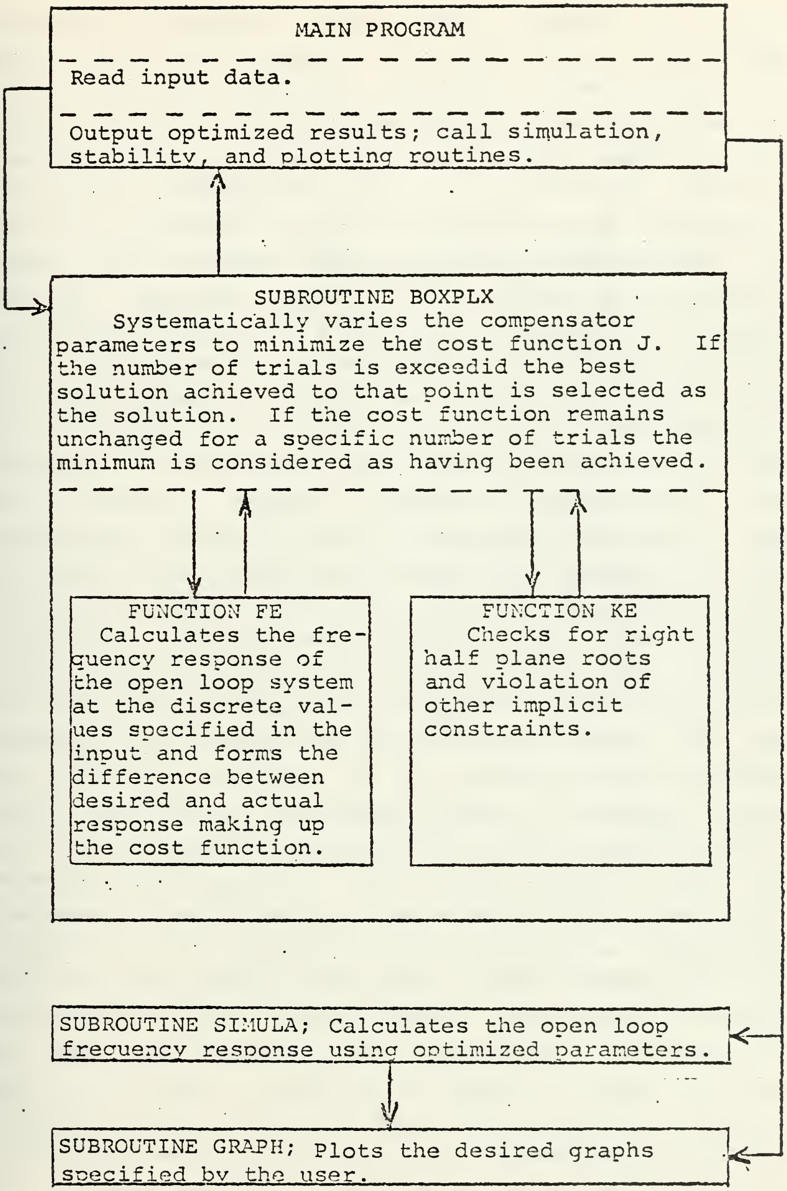
IV. PROGRAM DESCRIPTION AND UTILIZATION

A. DESCRIPTION OF ALGORITHM IMPLEMENTATION

The frequency domain algorithm described in the latter part of Chapter II has been implemented in program form using FORTRAN IV coding and configured for use on the IBM 360-67 computer system available at the Naval Postgraduate School Computer Center. With the exception of the graphing subroutine the remainder of the program is self contained and may be used on any computer system having a standard FORTRAN IV compiler. Without the overhead routines required of the graphing subroutine, the basic program requires approximately 180k bytes of core memory, and including the core requirements of the plotting routines roughly 210k bytes of memory are required by the program. While admitedly less core would be necessary if such techniques as array overlaying were employed when programming the algorithm, it was felt that for simplicity in following the flow of the program by future users these techniques were best left unused.

The computer aided linear time invariant compensator optimization program (CALICO) is essentially composed of four main parts with numerous additional small subroutines functioning in supportive roles. Since often a program user may wish to perform certain modifications to programs of this nature in order that they may better serve his particular purpose, Appendix A provides a description of the supportive subroutines used within the program to facilitate accomplishment of any program modifications that may be desired. A block diagram of the major components of the overall program are shown in figure IV-1. The main program serves primarily to input data describing the system under consideration and to output the values of the roots and the gain of the compensator that will best achieve the desired response in accordance with the cost function selected. During program execution, the main program also serves to direct the sequence of execution of the other major sections of the CAICC algorithm.

Since the plant and compensator are assumed to be linear time-invariant, they may be completely described by transfer functions in the independent Laplace variable s. Once the plant and assumed form of the series compensation are input along with the profiles forming the desired phase and magnitude response program control proceeds to the minimization subroutine, BOXPLX. From the point of view of computer aided control system design, associated with this subroutine are two key function sub-programs, FE(X) and KE(X). The function sub-program FE(X) performs the calculation of the system open loop frequency response, at the discrete frequencies provided in the input data, and calculates the value of the cost function based upon the difference between the actual and desired frequency response at the discrete frequency values. Six separate cost functions, from which the program user may select one, are incorporated into this function sub-program. The value of the cost function is used by the minimization routine to vary the compensator parameters in order to minimize this quantity. The function sub-program KE(X) checks for any violation of the implicit constraints, which in this particular case are concerned mainly with the presence of right half plane roots. The implicit constraints used, also impose limits on the real and imaginary parts of the numerator and denominator polynomials evaluated at any



Program Block Diagram

Figure IV-1

particular frequency. These latter limits are a direct result of the finite computer word size and the particular method used to represent numerical values within the machine's memory. Both of these factors have an effect in determining the maximum and minimum size numerical value that may be represented within any particular computer. While the coefficient values determined by the minimization routine will not exceed these maximum or minimum values, the real and imaginary parts of the numerator and denominator polynomials evaluated at a particular frequency are squared during the process of determining the magnitude of the frequency response. Without the implicit constraints these squared values can exceed the size of the numerical capability of the machine, particularly in the case of high order systems, causing premature termination of the algorithm and hence failure in obtaining a solution. Once the minimization subroutine determines the coefficients that result in a minimum value of the cost function or the minimization process is terminated for other reasons, program control is transfered back to the main program. Here the roots and gain of the compensator which resulted in a minimum cost function are calculated and output. The open loop frequency response of the sytem is then calculated using a more densely distributed number of frequency values than would normally be selected by the designer for use in the minimization process. This is done largely to show the existence of any resonant peaks which might otherwise be straddled and not apparent if only the discrete frequencies selected by the designer were used in the plotting routines. The stability of the entire system is then checked by adding the open loop numerator and denominator polynomials and applying the Routh stability criterion to the resulting polynomial which is the system characteristic equation. Finally the results at the discrete frequencies specified by the designer and the "continuous" curve generated by the SIMULA subroutine are output on Bode and gain versus phase plots. In addition, plots of the difference between the desired and actual results at the discrete frequencies specified by the designer are output in order to aid the designer in making any modifications to the form of the compensator that may be necessary in order to better satisfy the system specifications.

As stated in the previous paragraph six separate cost functions have been incorporated into the function sub-program FE(X). The first three of these are of the error squared type and the latter three are of the absolute error type. These six cost functions represent various combinations of one minus the actual response divided by the desired response at the discrete frequencies under consideration. These pre-defined cost functions are provided as a matter of convenience in using the program and certainly do not prevent the user from defining his own cost function provided this is placed within the FE(X) function sub-program. The first cost function combines the deviation of both the desired magnitude and phase responses from the actual magnitude and phase responses of the open loop system. That is the first cost function is of the form

$$J(\underline{\omega}) = \sum \left| \left(1.0 - \frac{AMAG(\underline{\omega})}{DMAG(\underline{\omega})}\right)^2 + \left(1.0 - \frac{APHASE(\underline{\omega})}{DPHASE(\underline{\omega})}\right)^2 \right|$$

where AMAG and APHASE are the actual magnitude and phase response of the open loop system and DMAG and DPHASE are the desired magnitude and phase respectively. These quantities are evaluated in the cost function and summed over the discrete frequency values in the specified range. The second cost function which the user may choose, uses solely the deviation of the actual magnitude response desired magnitude response evaluated at the discrete frequency values. Thus this cost function is of the general form

$$J(\underline{\omega}) = \sum \left(1.0 - \frac{AMAG(\underline{\omega})}{DMAG(\underline{\omega})}\right)^2$$

If a minimum phase system is being considered the magnitude and phase plots as a function of frequency are uniquely related by the Bode theorems [3]. That is, determining the magnitude ratio of a system over a given frequency range will also result in a unique phase relationship for the system. Thus if the system design being considered is of the minimum phase type the use of the type two cost function described should result in both the magnitude and phase specifications being satisfied. The third pre-defined cost function uses the deviation of the actual open loop phase response form the desired phase response in a function of the form

$$J(\underline{\omega}) = \sum \left(1.0 - \frac{APHASE(\underline{\omega})}{DPHASE(\underline{\omega})}\right)^2$$

While for minimum phase systems the magnitude and phase responses are uniquely related, it should be noted that the fixed loss or gain of the system cannot be determined from the phase characteristics alone. Consequently the use of the type three cost function may result in a constant gain error for the magnitude response of the system. This error however is readily observable from the output produced by the program and correction of this may be accomplished by adjustment of the compensator gain value accordingly. The other three pre-defined cost functions are of the same general form as those just presented with the exception that the squared differences are replaced by the absolute value of the differences. The general rationale in selecting this latter type of cost function was that on occasion the squared type cost function will have a tendency to flatten near the minimum. With the absolute value cost function the user may avoid the excessive time required in finding the minimum values when the case arises where the squared cost function becomes flat.

In utilizing this design program the user should also be aware that the need for frequency scaling of the problem may arise. This is particularly true when high order systems involving high frequencies are being considered. The need for frequency scaling is primarily a result of the finite word size available in the computer system. It must be kept in mind that the magnitude response of the system is computed as the ratio of two complex numbers and while the resulting magnitude may be relatively small the value of the magnitudes of the individual complex numbers forming this ratio may be large. The situation is further complicated by the fact that in computing the magnitudes of the complex numbers involved the real and imaginary parts must be squared, thus effectively reducing the maximum power of the numbers which can be allowed by a factor of two. If the magnitudes of the resulting numbers become too large an error message will be printed recommending that the problem be frequency scaled.

B. DATA INPUT AND OUTPUT

In order to simplify use of the program, an attempt has been made to maintain the input information necessary for execution of the algorithm to a minimum, and at the same time preserve a certain amount of flexibility within the program which may be controlled by the user. In so doing some of the program control variables for various options have been set internal to the program and may not be manipulated by the user in the input data deck. These internally set variables do not detract from the general functioning of the program, but mention of this is made to bring it to the user's attention in the event that specialized problems might more effectively be investigated by resetting some of these normally preset program variables. Many of the preset items are concerned with control of the minimization routine. Such items as the upper and lower bounds of the search area, the integer programming option, and the seed for the random number generator are set to specific values within the main program just prior to calling the minimization subroutine. For example for the minimum phase case, the lower and upper bounds on the search areas of the compensator numerator and denominator polynomial coefficients are set at 0.0 and $1.6x10^6$ respectively. For the non-minimum phase case the bounds on the search areas of the numerator polynomial coefficients only are extended to $-1.6x10^6$ and $1.6x10^6$ and the implicit constraints are adjusted to allow right half plane roots for the numerator polynomial. The integer programming option for BOXPLX is also preset to a value of zero indicating that values for the polynomial coefficients need not be integers. In general, selection of the integer programming option results in considerably more execution time for the program and for the purpose of varying the compensator parameters the selection of this option provided nothing to enhance the solution of problems.

The input data required for program execution and control may be divided into general areas as follows: 1.) a title card; 2.) a series of cards giving the gain and coefficients of the plant transfer function; 3.) a series of cards giving the assumed gain and coefficient values of the compensator transfer function; 4.) a control card describing the plots desired, the type of cost function to be used, whether the system contains a zero order hold, and the number and range of the discrete frequency points being considered; 5.) cards containing the discrete frequency values, the desired magnitude, and the desired phase; 6.) finally a card describing the number of iterative trials to be allowed in the minimization and whether any output is desired from the minimization routine in order to keep track of the manner in which the coefficients are being varied. The specific input formats of the above necessary data are described in detail in the following paragraphs. These paragraphs are numbered so as to indicate into which of the six general areas a particular data card belongs. This description coupled with the program listing given in Appendix B should provide sufficient information for setting up the data deck necessary for execution of the program. Figure IV-2 illustrates the data deck used in executing the first example problem presented in section C of this chapter. In this figure each line of data represents an individual data card. The column numbers for the data cards are also shown above and below the data deck section of the figure.

1. Title card.

This must be the first card of the data deck and provides a means of problem identification if several separate problems are run under one job name. Columns 1 through 48 may be used to input whatever alpha-numeric characters the user desires for identifying any particular problem. This title will also be printed on the graphical output requested by the user.

2A. Plant gain

The second card contains the gain of the plant in columns 1 through 10. If the gain is not explicitly present in the plant transfer function then the value input on this card should be unity. The format used to read the plant gain is an E10.0 type format. Thus, the gain may be input in either floating point or scientific notation type formats.

2B. Input form and order of the plant transfer function numerator.

This card specifies whether the numerator of the plant transfer function is to be read in factored or polynomial form and what the order of the numerator is. A P or an F in column 1 indicates whether the plant numerator data is to be read in polynomial or factored form respectively. Columns 2 and 3 contain the order of the numerator. Thus an input of the form F03 starting in column 1 indicates that the polynomial is of third order and is to be read in factored form.

2C. Plant transfer function numerator.

The following card(s) contain either the polynomial coefficients or the factors of the plant transfer function numerator, depending on which form has been chosen for input. If the polynomial form of input has been selected then the polynomial coefficients are input in ascending powers of s using an 8E10.0 format. If more than eight coefficients are needed they are simply continued on successive cards. The coefficient of the highest order term is assumed to be normalized to unity and is not read in explicity. Thus the number of coefficients actually read corresponds to the order of the numerator polynomial. If the factored form of input is chosen then each factor is read in on a separate card with columns 1 through 10 being used for the real part of the factor and columns 11 through 20 for the imaginary part. Again the format used is E10.0, thus allowing the factors to be input in either floating point or scientific notation form. Since complex factors will occur in conjugate pairs the values of these factors need only be input once. Thus the number of cards used will vary depending upon how many purely real and how many complex factors are involved.

2D. Input form and order of the plant transfer function denominator.

Again the form of input is specified as in card number 2B along with the order of the plant transfer function denominator.

2E. Plant transfer function denominator.

The format used to input the denominator is identical to that used for input of the numerator, contingent upon the desired form specified for the input. That is to say, it is possible to input the numerator in polynomial form and the denominator in factored form, or vice versa, since each time a polynomial is input as data the input form is uniquely specified.

3A. Assumed compensator transfer function gain.

Following the input of the information about the plant the description of the form of the compensator must be supplied. This is begun by inputing an assumed compensator gain using the same format as was used to specify the plant gain. Due to the manner in which the compensator transfer function numerator coefficients are handled within the program, if the assumed compensator gain is set to zero, for lack of any better value, whis will result in the starting guess of all the numerator coefficients being zero. This will not cause the program to terminate prematurely, but the initial guess of zero for all the coefficients may not be what is desired by the user. If doubt exists as to what value gain should be assumed, a value of 1 is suggested as a convinient arbitrary starting point.

3B. Input form and order of the assumed compensator numerator.

The method for specifying the form of the input for the compensator initial numerator parameters is identical as that used in specifying the plant numerator. A P indicates polynomial form and an F indicates factored form.

3C. Assumed initial values of compensator numerator parameters.

In order to maintain continuity in the input format the scheme and format used to input the compensator numerator parameters is identical to that used in specifying the plant transfer function numerator.

3D. Input form and order of the assumed compensator denominator.

As in previous cases either polynomial or factored form is specified along with the order of the compensator denominator.

3E. Assumed initial values of compensator denominator parameters.

Once again the same format as used to input the plant data is used in order to maintain the continuity of the input formating.

4A. Program control card.

The program control card is used to input various quantities controlling both the range and number of discrete points used in the minimization of the cost function, the output desired from the program, and also whether a zero order hold is present in the error channel of the system. As many as twelve separate quantities may be input on this single control card in order to direct execution of the program. Columns 1 through 10 specify the minimum frequency (WMIN), in radians, of the total range being considered. Columns 11 through 20 specify the maximum value (WMAX), in radians, of the range of frequencies being considered. Both of these values are read using a 2E10.0 format. Following the maximum and minimum frequency values, up to ten input quantities are read using a 10I3 format. Columns 21 through 23 specify the number of discrete frequency points (NOMEG) over the given range at which the open loop frequency response is to be determined and the cost function evaluated. The next input quantity (KNOW), in columns 24 through 26 specifies whether the discrete frequencies of interest are to be read from data cards or automatically computed. Normally this quantity is set to unity indicating that the discrete frequency points of interest will be read from data cards. A value of zero indicates that the discrete frequency values are to be incremented linearly by the program starting at the minimum frequency value. A value of two indicates that the discrete frequency points will be incremented logarithmically begining with the minimum frequency value. Columns 27 through 29 indicate whether a Bode plot of the results is desired. If this variable (NBODE) is set equal to one no Bode plot will be output. A value of zero for NBODE will result in both magnitude and phase plots. Columns 30 through 32 control the output of the Nyquist plot. If NYQST is set to unity then the Nyquist plot will be suppressed. A value of zero for NYQST will result in a Nyquist plot output, however this is plotted only over the range of frequencies between WMIN and WMAX. Columns 33 through 35 indicate the presence of a zero order hold (IZOH) in the error channel of the unity feedback system. With IZOH set to zero the system is treated as being of a continuous nature. With IZOH set equal to one a zero order hold is considered to be present in the error channel and the open loop magnitude and phase computations are modified accordingly. Columns 36 through 38 (NICHCL) specify whether a magnitude versus phase plot of the results are desired. As with the other plot options, a value of zero will result in a plot being produced while a value of one will suppress the magnitude versus phase plot as part of the output. Columns 39 through 41 contain the variable indicating whether non-minimum phase solutions are to be considered. With this variable (NMINPS) set equal to zero only minimum phase solutions will be allowed. A value of one indicates that a non-minimum phase solution will be allowed provided that it is the one which minimizes the cost function. Columns 42 through 44 indicate if the desired magnitude profile values to be input are in decibels. If this variable (IDB) is zero the magnitude values to be read are dimensionless gain values, while a value of one indicates that the magnitude values to be input will be in decibels. Columns 45 through 47 contain the variable IPLOT. This quantity is used to choose between printer plots or Calcomp plots for the graphical output. A value of zero will result in printer plots and a value of unity will provide Calcomp output of the graphs desired. It is recommended that the printer plot option be used in initially solving a particular problem since this type of output requires less turn around time than the Calcomp output. Finally columns 48 through 50 specify the type of cost function (ICOST) the user wishes to choose among the six preprogrammed functions available in subroutine FE(X). The value specified should be between 1 and 6 corresponding to the particular cost function desired. If this value is zero the program will default to the type one cost function.

4B. Sampling period.

This card must be inserted only if a zero order hold is specified in the error signal channel. The sampling period T, in seconds, is specified in columns 1 through 10. If the system being considered is of the continuous type then this card must be omitted.

5A. Discrete frequency values considered.

Next follows a set of data cards specifying the values in radians per second of the frequencies at which the desired gain and phase values of the open loop frequency response are specified and at which the cost function is to be calculated. These values are input in an 8E10.0 format using as many cards as necessary to specify all the frequency values at which the program is to execute the algorithm.

5B. Gain profile of the desired open loop response.

This set of data cards contains the desired open loop gain response at the discrete frequency values previously specified. The same 8E10.0 format is used here as was the case in reading the discrete frequency values. The same number of cards should be used as was used in specifying the discrete frequency values.

5C. Phase profile of the desired response.

Here follows a series of cards specifying the desired open loop phase response, in degrees, at the discrete frequency values being considered.

6. Minimization trials card.

Here both the number of trials to be allowed in the minimization process and the print interval for diagnostic purposes are specified. These quantities are input using a 2I5 format. If the number of trials is zero then a default value of 2000 is assumed. If the print interval is specified as zero then no output from the minimization routine will result. Caution should be exercised in specifying the print interval in that if this interval is made too small an excessive amount of printed output may result. The program user is referred to the comment section of subroutine BOXPLX in Appendix A for guidelines in selection of the print interval.

```
(COLUMN NUMBERS, READ VERTICALLY)
         11111111112222222222333333333344444444445555555555666666666677777777778
12345678901234567890123456789012345678901234567890123456789012345678901234567890
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


COMP. OPT. EXAMP. ONE  LEAD COMP. 10PTS
 4.0
P00
F02
 0.0
 1.0
 2.0
 1.0
F01
 0.0
F01
 1.0
  0.2      20.0      010  1                     1  1
 0.200E 00 0.334E 00 0.557E 00 0.928E 00 0.155E 01 0.258E 01 0.431E 01 0.719E 01
 0.120E 02 0.200E 02
 0.995E 01 0.591E 01 0.346E 01 0.195E 01 0.101E 01 0.459E 00 0.179E 00 0.606E-01
 0.176E-01 0.445E-02
-0.969E 02-0.101E 03-0.109E 03-0.120E 03-0.137E 03-0.157E 03-0.178E 03-0.200E 03
-0.221E 03-0.238E 03
 5000


||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
         11111111112222222222333333333344444444445555555555666666666677777777778
12345678901234567890123456789012345678901234567890123456789012345678901234567890
(COLUMN NUMBERS, READ VERTICALLY)
```

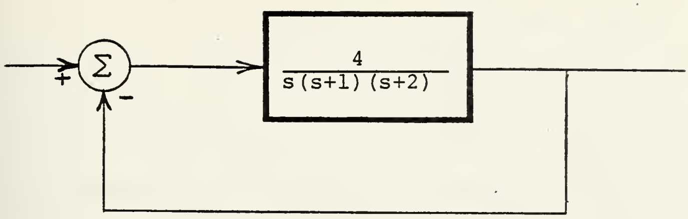
Figure IV-2

C. SERIES COMPENSATOR DESIGN EXAMPLE PROBLEMS

In this section a number of example problems are presented and discussed in order to illustrate the use of the computer aided compensator design algorithm. In the hope of presenting a clear, concise illustration of the algorithm and its use the example problems begin with relatively simple straight forward text book examples and gradually progress to problems of a slightly more difficult nature. Also included are problems that are intended to illustrate some limitations of the program in that these problems were not completely solved by the algorithm or they may have required a certain amount of manipulation of the input data in order to obtain a solution. These latter types of problems have been included since it is important that the designer using any computer aided design algorithm understand the algorithm's limitations and peculiarities as well as its intended capabilities. It is not intended to mislead the reader by implying that the example problems presented here provide a comprehensive presentation of all possible limitations of this automated design program. Rather they represent those limitations which were encountered in the process of verifying the program over a necessarily limited period of time. With this in mind the first example problem which deals with a relatively simple lead compensator is presented.

1. Compensator Design, Example 1.

Consider the uncompensated system shown in figure IV-3.



Design Example 1, Block Diagram

Figure IV-3

The asymptotic Bode diagram for the open loop system is shown in in figure IV-3A. As can be seen from the open loop Bode plot, even though the system is stable it is lightly damped. A single section lead compensator is to be used to increase the phase margin. A complete discussion of the reasons for selection of the lead compensator and a detailed discussion of this particular example is presented by Thaler and Brown [18]. The transfer function form of the single section lead compensator consists of a first order numerator and a first order denominator. A desired magnitude and phase profile, consisting of ten discrete frequency points over the range from 0.2 to 20.0 radians, was selected to give a phase margin of approximately 45 degrees in order to increase the damping of the system. As an initial guess the compensator was assumed to have a gain of 1.0, a zero at the origin and a pole at -1.0. This information was supplied to the program in the prescribed format and the optimized results in the form of magnitude and phase plots are shown in figure IV-3B. Here, as throughout the remainder of the example problems, the diamond shaped symbols represent the desired values specified by the program user, the X symbols represent the values computed by the program at the discrete frequency values specified using the compensator parameters returned as the solution from the minimization routine, and the continuous curve represents the response computed by the program again using the parameters returned as a solution but with a more dense set of frequency values used in order to show any resonant peaks or anomalies in the magnitude and phase curves which might be difficult to visualize from a sparsely selected set of discrete values. As can be seen in figure IV-3B the results match the desired response well over the range of interest. Figure IV-3C shows the same results using the magnitude versus phase plot to portray the open loop frequency response. The numerical values of the compensator parameters returned from the program as a solution along with the starting values used are shown in figures IV-3D, IV-3E, and IV-3F. A comparison with the solution values given in ref. 18 will show that they agree. Figure IV-3G shows plots of the differences between the actual and desired open loop frequency response of the resulting system at the discrete frequencies considered in the minimization process.

This problem as described required approximately 500 iterations within the minimization routine to achieve the solution shown. This however includes two restarts to attempt insuring that a global minimum had been achieved and that termination was not due to a local minimum. From the begining to the first restart required approximately 250 trials with the cost function being reduced from a value of 6.33 to a value of 0.00711 (recall that the minimum achievable under ideal condition is zero which due to machine errors alone in representing numbers would not likely ever be achieved). The entire solution required 32.5 seconds of CPU time, a good 10 seconds of which was used in the production of Calcomp plots. These numbers concerning the time required for solution are presented for what they are worth in that extrapolation of these numbers to more complicated problems is not a straight forward task. As mentioned in Chapt III the minimization routine does become inefficient when the number of variable parameters is allowed to become large. It was felt that a detailed investigation of the efficiency and time required for solution of problems as a function of the number of variable parameters and the number of discrete frequency values to be considered was beyond the scope of the basic development of the program. However, the amount of time required for solution of some of the example problems that follow will be given in order to give the user an "intuitive feel" for the approximate amount of time that may be need to handle certain types of problems.

In order to illustrate that the starting point chosen should have reasonable values, but that the exact values chosen should not effect the final results significantly, the same problem was run a second time with the pole and zero of the compensator both chosen to be at the origin. Again an assumed initial gain of one was chosen for the compensator. The graphical results of the solution are shown in figures IV-3H through IV-3J. The numerical results returned using these new starting values are not shown here since they were identical to within three decimal places with those values shown previously.

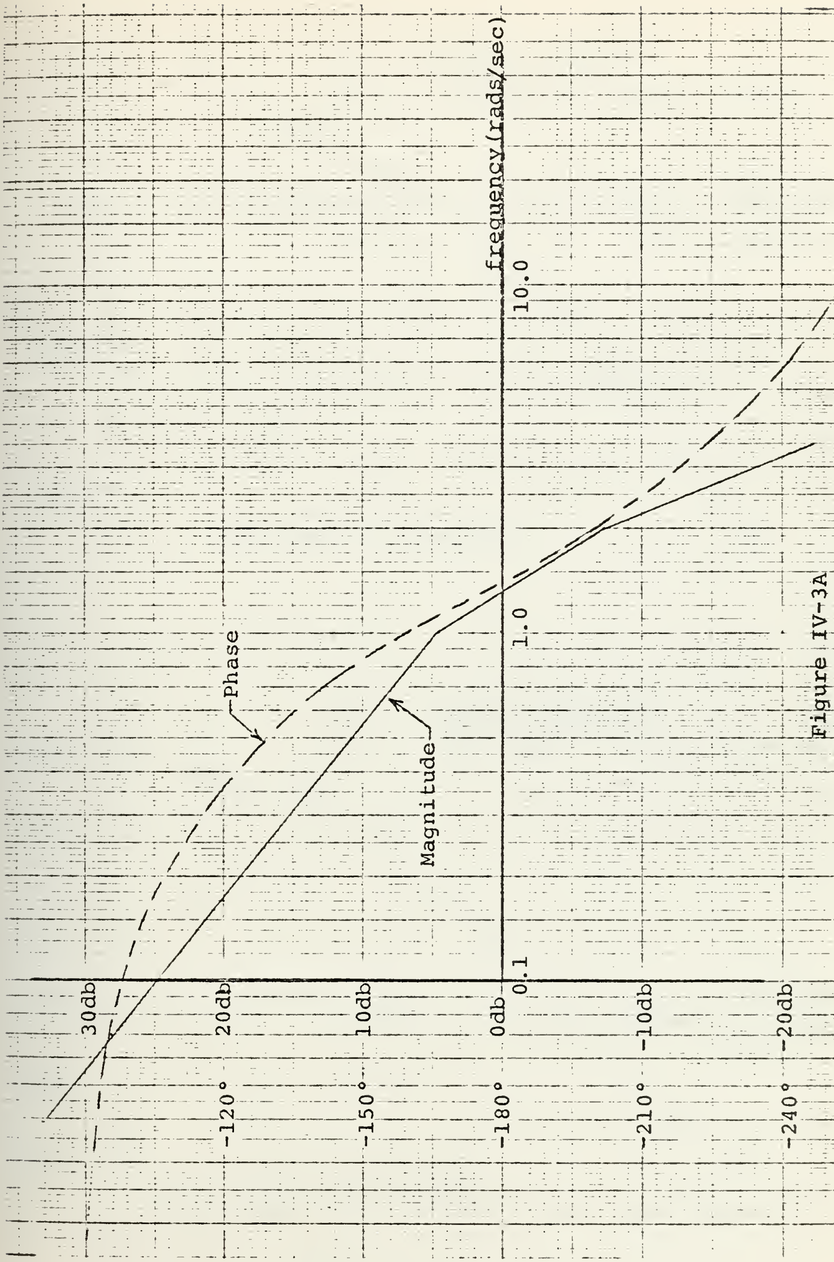
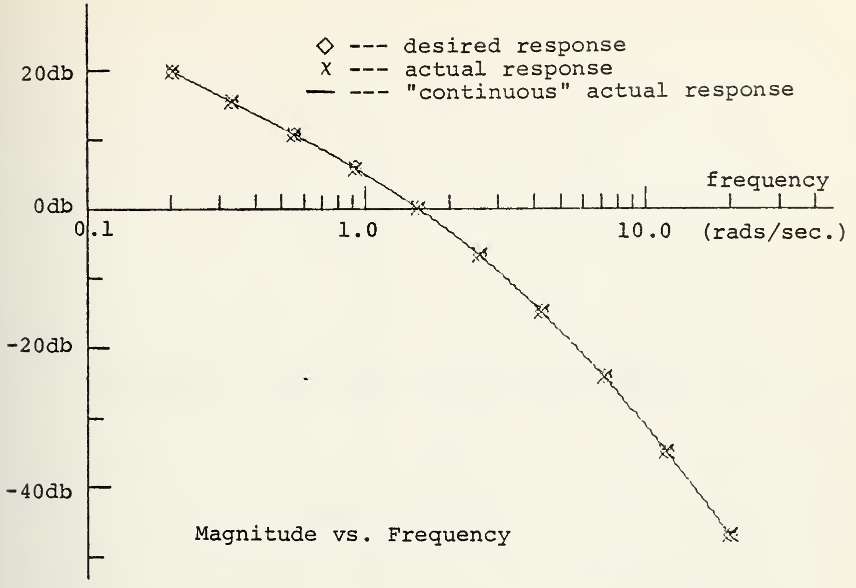
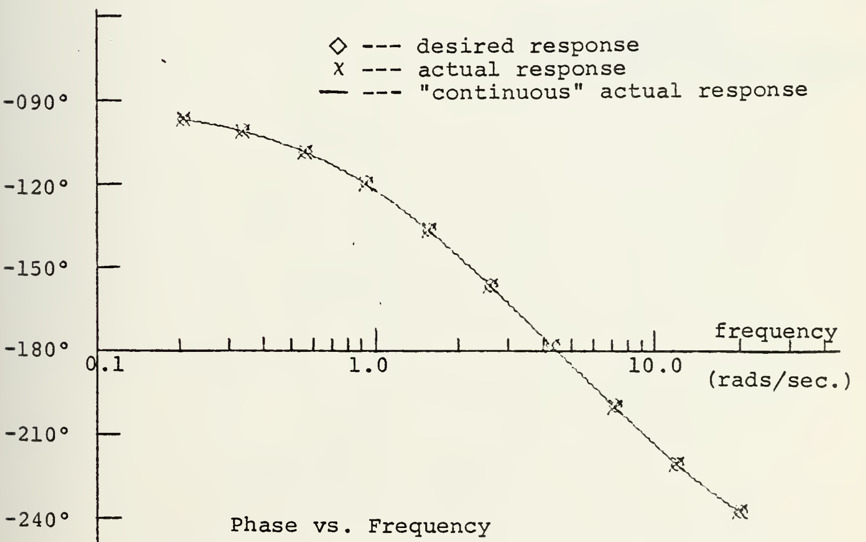


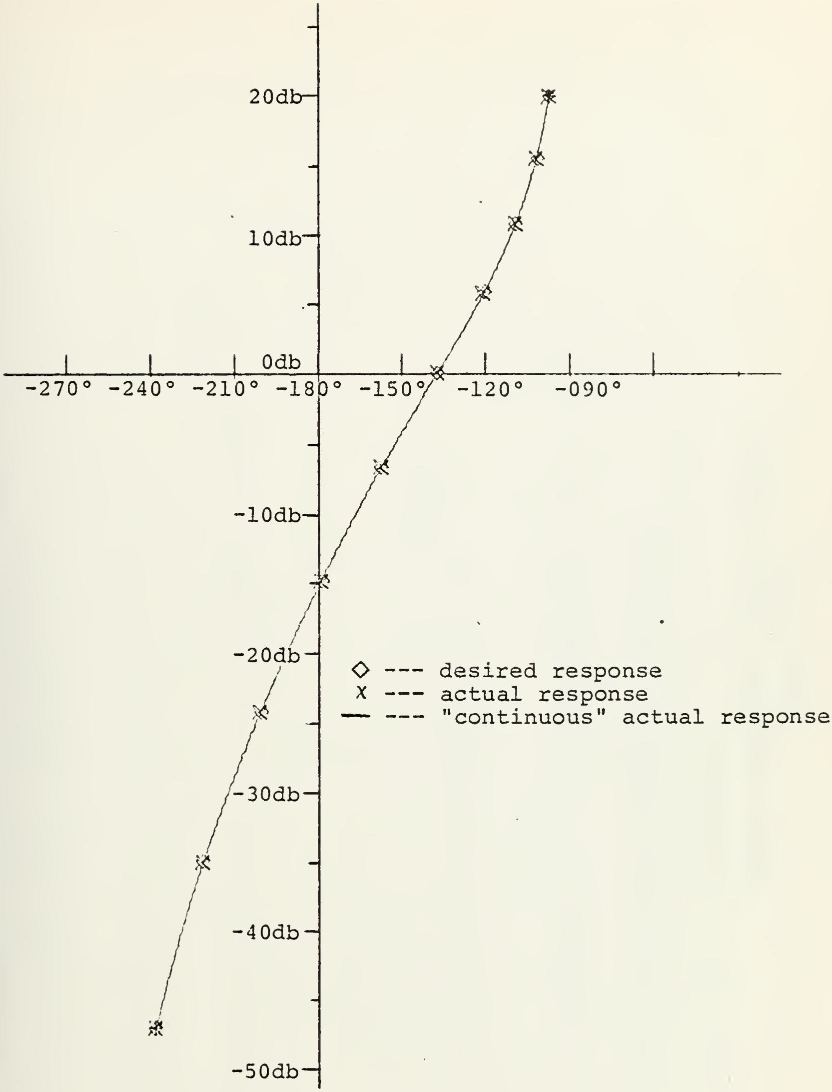
Figure IV-3A
Uncompensated Open Loop System Bode Plot





Compensated System Magnitude and Phase Responses

Figure IV-3B



Magnitude vs. Phase

Figure IV-3C

```
    TITLE --- COMP. OPT. EX. ONE LAD COMP ROOTS
---------------------------------------------------------------------

    UNCOMPENSATED TRANSFER FUNCTION GAIN =  4.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   0.0            2.000000E 00    3.000000E 00    1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
 ARE:      REAL PART            IMAGINARY PART

        0.0                    0.0

       -1.000000E 00           0.0

       -2.000000E 00           0.0
---------------------------------------------------------------------

    COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

    COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   0.0            1.000000E 00

    COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
 ARE:      REAL PART            IMAGINARY PART

        0.0                    0.0

    COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00   1.000000E 00
```

Computer Numerical Output Example 1

Figure IV-3D

```
COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART          IMAGINARY PART

      -1.000000E 00         0.0
------------------------------------------------------------------------
------------------------------------------------------------------------

     THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

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

     THE TOTAL NUMBER OF TRIALS CALLED FOR =  5000

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

     THE COST FUNCTION TO BE USED IS THE TYPE   1

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

THE MINIMUM COST FUNCTION VALUE =  7.010020E-05

THE ERROR RETURN CODE FROM BOXPLX =   0

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

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   9.990060E 05    9.984820E 05

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 ROOTS ARE:      REAL PART          IMAGINARY PART

          -1.001426E 00         0.0
------------------------------------------------------------------------

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   9.985400E 05    9.568900E 04
```

Computer Numerical Output Example 1

Figure IV-3E

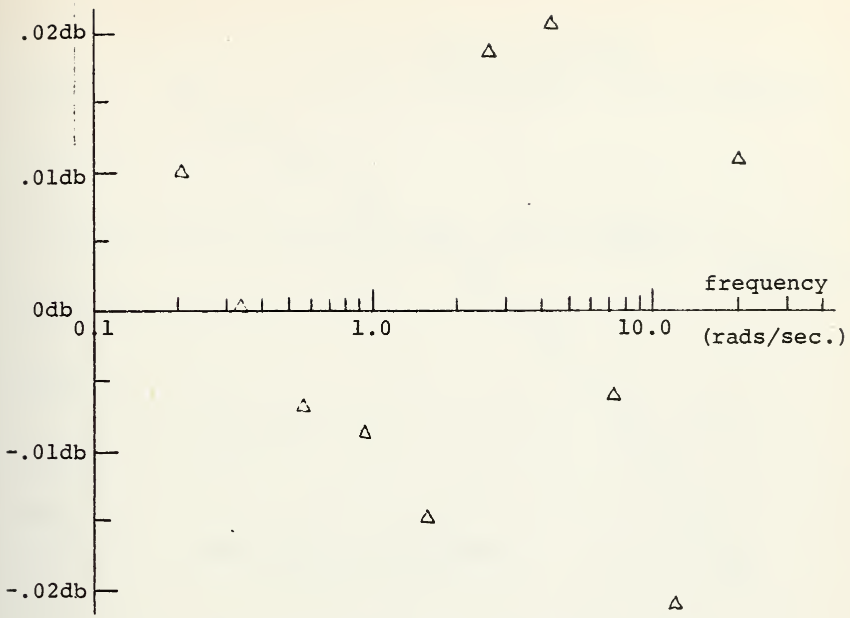
```
OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
ROOTS ARE:      REAL PART          IMAGINARY PART

         -1.001554E 01          0.0
------------------------------------------------------------------------

   OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.001596E 01
```

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
2.000000E-01	1.996645E 01	1.996643E 01	-9.687019E 01	-9.689999E 01
3.340000E-01	1.545201E 01	1.545175E 01	-1.014153E 02	-1.010000E 02
5.570000E-01	1.077461E 01	1.076152E 01	-1.087801E 02	-1.090000E 02
9.280000E-01	5.791651E 00	5.800692E 00	-1.202251E 02	-1.200000E 02
1.549999E 00	7.141680E-02	8.642113E-02	-1.366093E 02	-1.370000E 02
2.580000E 00	-6.743149E 00	-6.763746E 00	-1.568891E 02	-1.570000E 02
4.309999E 00	-1.492242E 01	-1.494294E 01	-1.784067E 02	-1.780000E 02
7.190000E 00	-2.435672E 01	-2.435052E 01	-2.001377E 02	-2.000000E 02
1.200000E 01	-3.511089E 01	-3.508974E 01	-2.206923E 02	-2.210000E 02
2.000000E 01	-4.702190E 01	-4.705279E 01	-2.376905E 02	-2.380000E 02

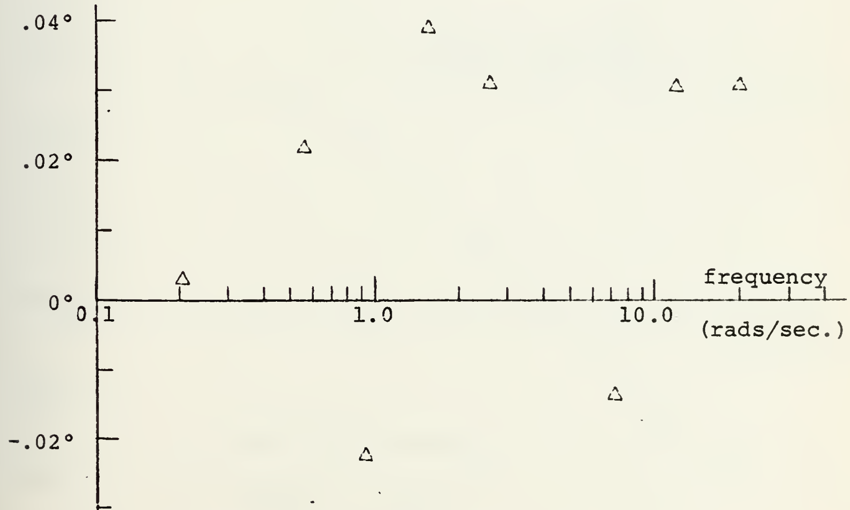
```
------------------------------------------------------------------------
   THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE.
------------------------------------------------------------------------
```

Computer Numerical Output Example 1

Figure IV-3F

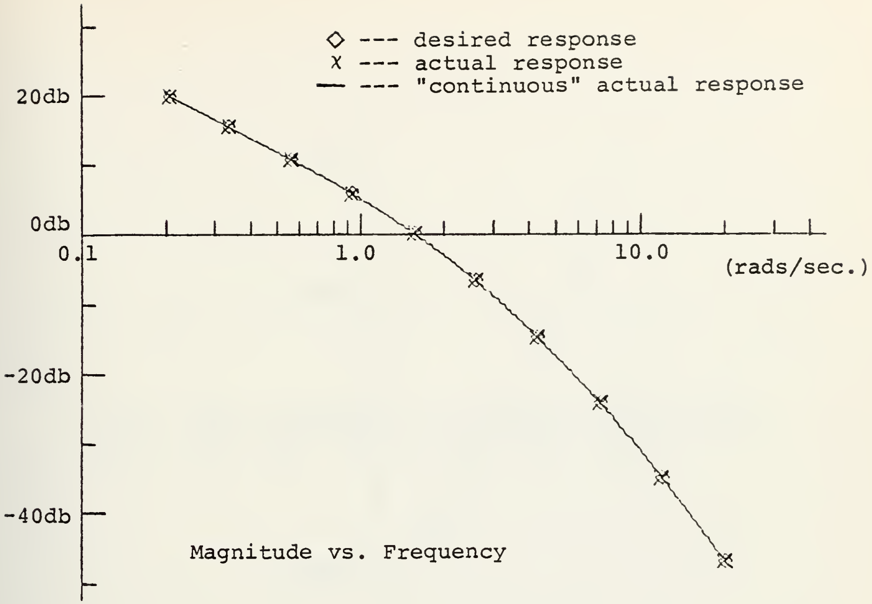


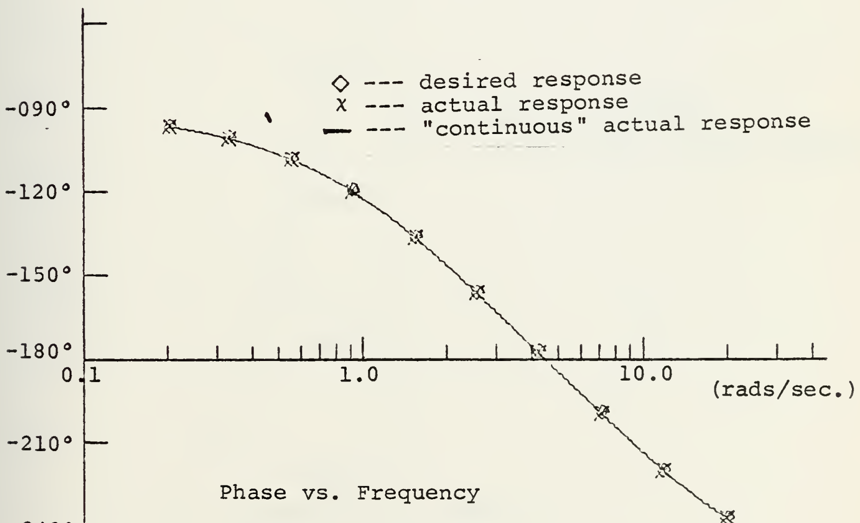
Magnitude Difference vs. Frequency



Phase Difference vs. Frequency

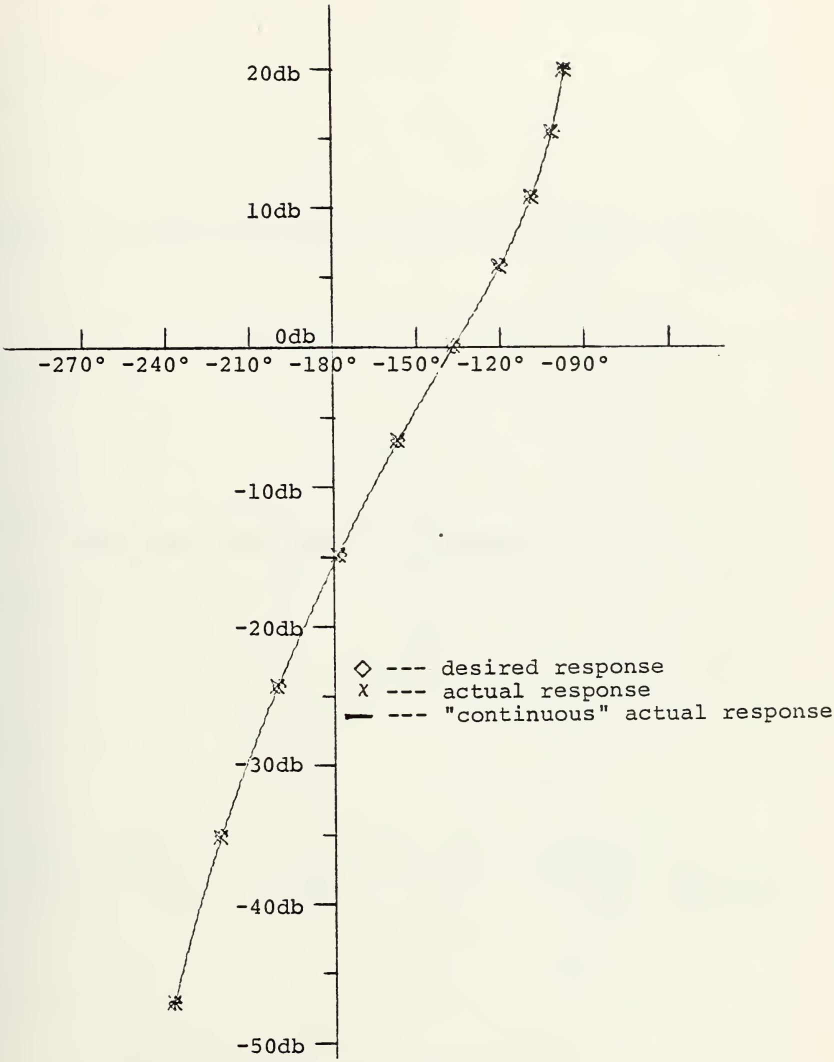
Figure IV-3G





Example 1, Run #2

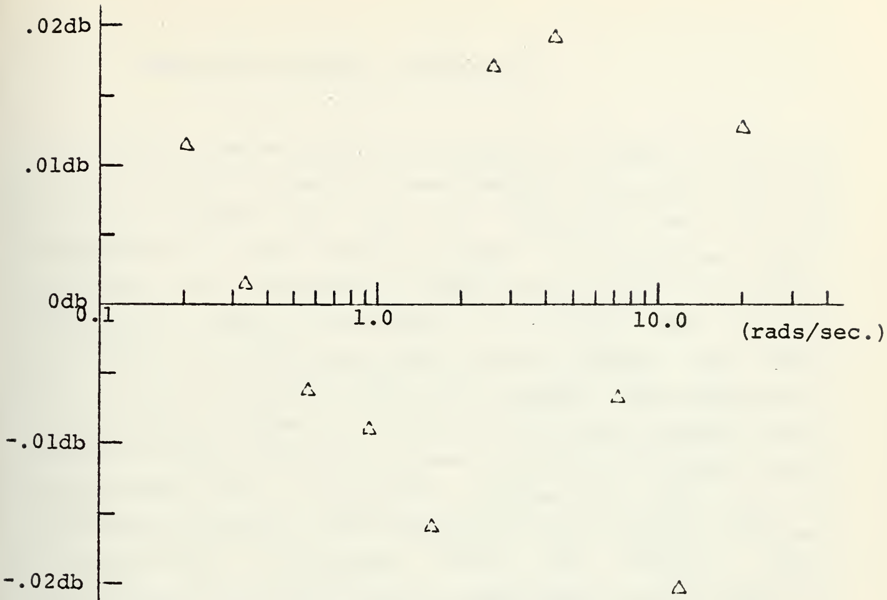
Figure IV-3H



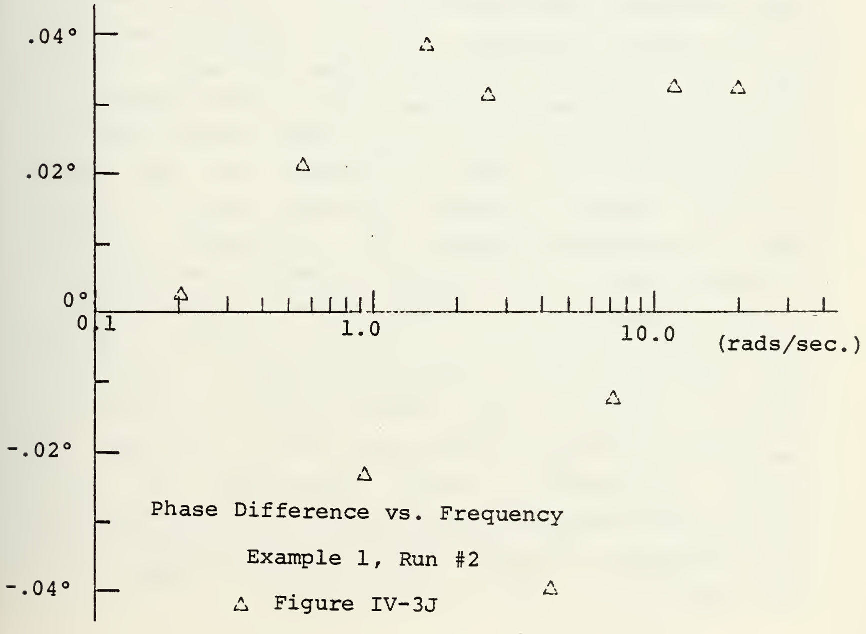
Magnitude vs. Phase

Example 1, Run#2

Figure IV-3I



Magnitude Difference vs. Frequency



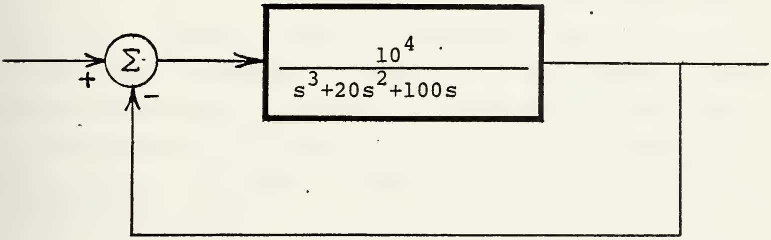
Phase Difference vs. Frequency

Example 1, Run #2

△ Figure IV-3J

2. Compensator Design, Example 2.

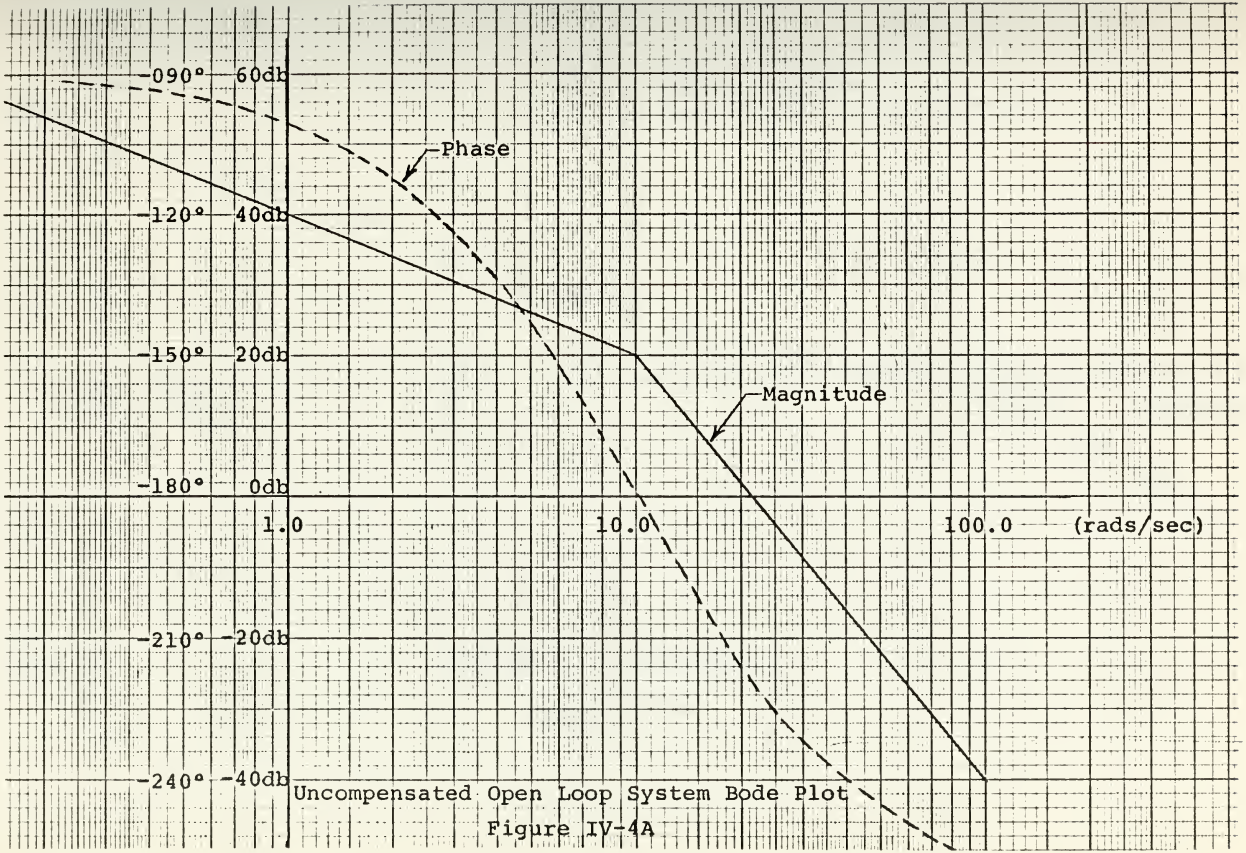
This example problem (discussed in detail in ref.10) has been chosen not only to familiarize the reader with the use of the program, but also to illustrate one of the shortcomings of the design program. In particular, while the program was successful in finding the correct solution, difficulties were encountered in the root finding subroutines. In this particular problem, during the final phases of execution the roots of a fourth order polynomial must be found in order to calculate the phase response using subroutine SIMULA which does these calculations using a more densely spaced set of discrete frequencies than in the minimization routine. Since numerical root finding routines have difficulty in finding roots for fourth order polynomials, the lack of good convergence for the roots of the system in both root finding subroutines caused the phase calculations normally done by subroutine SIMULA to be aborted. It should be emphasized that failure of the root finding subroutines in this case only effects the generation of the "continuous" phase response and in no way created any problems for the minimization part of the program as evidenced by the resulting correct values for the compensator parameters. The original uncompensated system being considered is shown in figure IV-4. A Bode diagram of the uncompensated system (figure IV-4A) reveals that the system as it exists is unstable. If stability were the only consideration, a simple attenuation of the plant gain would be sufficient to provide adequate phase margin to insure this. However, as with many series compensation problems, here we must effectively strike a compromise between effectiveness of control (generally associated with a higher gain) and stability of the system (generally associated with a lower gain). As discussed in ref. 10 a simple gain attenuation would result in stability being achieved, but also has an undesirable effect upon the velocity error constant. Thus as discussed in detail in ref. 10, a lag compensation network appears as the best candidate to accomplish the job.



Design Example 2, Block Diagram

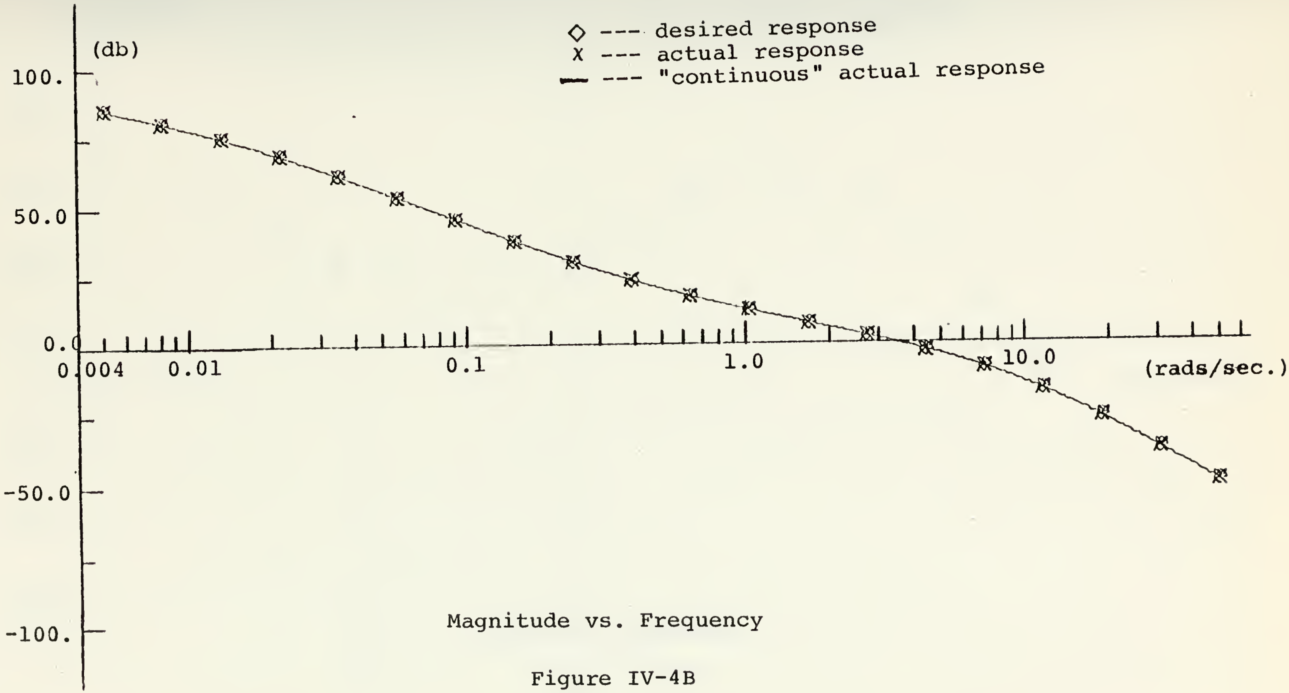
Figure IV-4

In this particular problem twenty discrete frequency values over the range from 0.005 to 50.0 radians were selected and the desired magnitude and phase values specified at these frequencies. As in the previous example problem the form of the compensator is selected as having a first order numerator and first order denominator, that is in this case a single section lag compensator is to be used. A type one form of the cost function was selected for the minimization process. The magnitude and phase profiles of the desired response along with the resulting response with the compensator included in the open loop system are shown in figures IV-4B, IV-4C, and IV-4D. As mentioned, the continuous curve for the phase response was not computed because of the failure of the root finding routines to find the open loop system roots. It can be seen in figure IV-4C that the phase response does agree well at the discrete frequencies specified. Since we are dealing with a minimum phase system and figure IV-4B shows that there exists no resonant peaks in the magnitude curve the user may conclude that the phase response will also be well behaved for the various frequency values between the explicitly specified discrete values plotted. This case should serve to illustrate that the program requires that the user still provide an interpretation of the results. The computer output of figures IV-4E, IV-4F, and IV-4G shows that the compensator parameters are in agreement with the problem solution as presented in ref. 10 if the reader wishes to make the comparison. Figures Iv-4H and IV-4I show plots of the differences between the actual and desired magnitude and phase at the discrete frequencies selected for the minimization process.



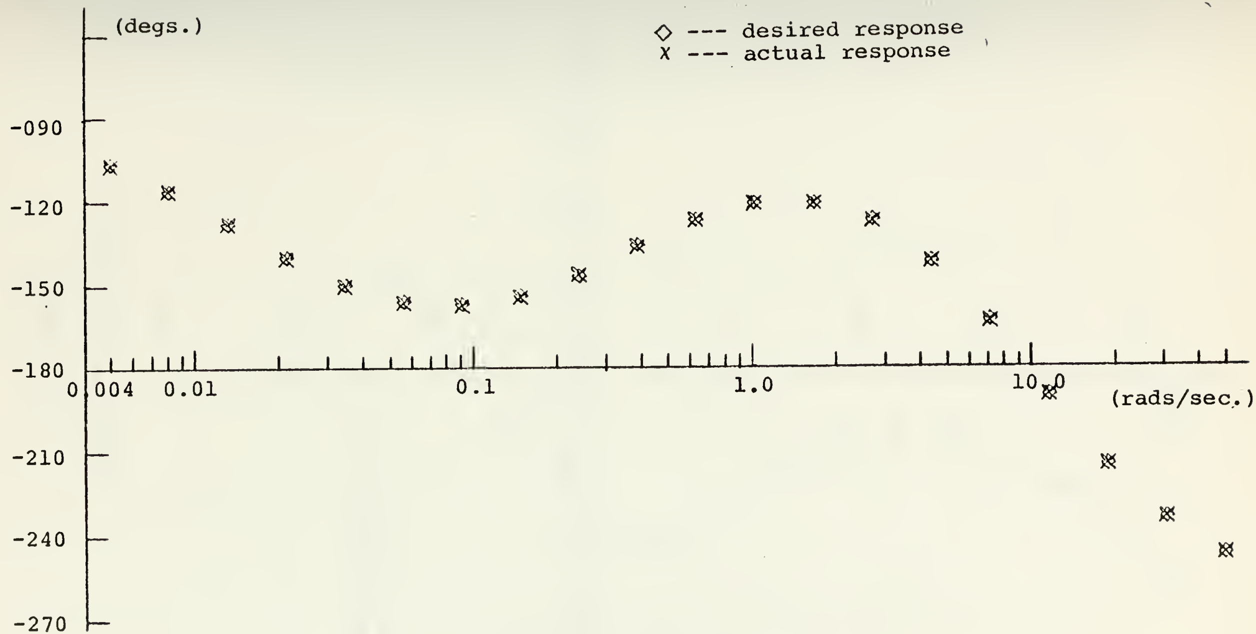
Uncompensated Open Loop System Bode Plot

Figure IV-4A



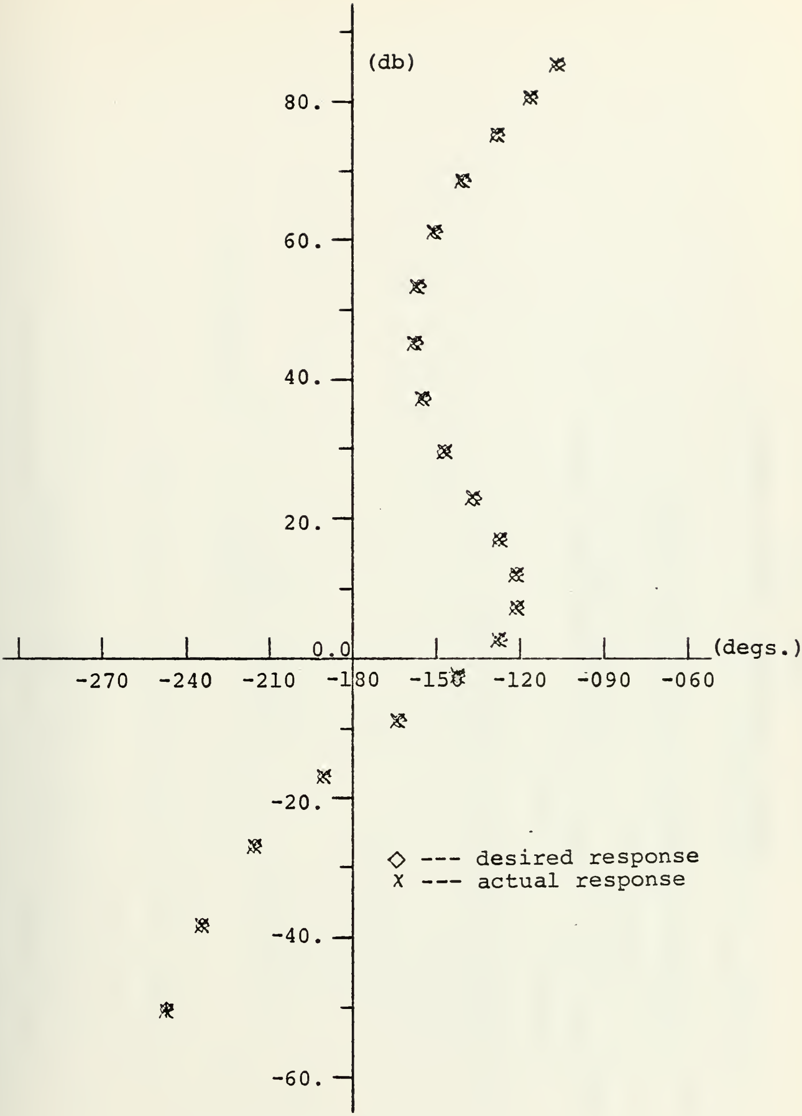
Magnitude vs. Frequency

Figure IV-4B



Phase vs. Frequency

Figure IV-4C



Magnitude vs. Phase

Figure IV-4D

```
TITLE --- COMP. OPTIMIZ. EXAMPLE 2 LAG COMP. 2OPT

UNCOMPENSATED TRANSFER FUNCTION GAIN = 1.0000000E 04

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.0000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

0.0              1.0000000E 02   2.0000000E 01   1.0000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART           IMAGINARY PART

          0.0                 0.0
         -1.0000000E-01       0.0
         -1.0000000E 01       0.0

COMPENSATOR TRANSFER FUNCTION GAIN = 1.0000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.0000000E 00    1.0000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
ARE:      REAL PART           IMAGINARY PART

         -1.0000000E 00       0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.0000000E 00    1.0000000E 00
```

Computer Numerical Output Example 2

Figure IV-4E

```
COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:    REAL PART        IMAGINARY PART

     -1.000000E 00         0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE. THEREFORE NO RIGHT-HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE TOTAL NUMBER OF TRIALS CALLED TCP = 10000

THE COST FUNCTION TO BE USED IS THE TYPE   1

THE MINIMUM COST FUNCTION VALUE =  2.210733E-04

THE ERROR RETURN CODE FROM BOXPLX =    0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  8.830318E 01    2.247941E 02

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
ROOTS ARE:      REAL PART          IMAGINARY PART

          -3.929228E-01          0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  8.836658E 01    5.727121E 03
```

Computer Numerical Output Example 2

Figure IV-4F

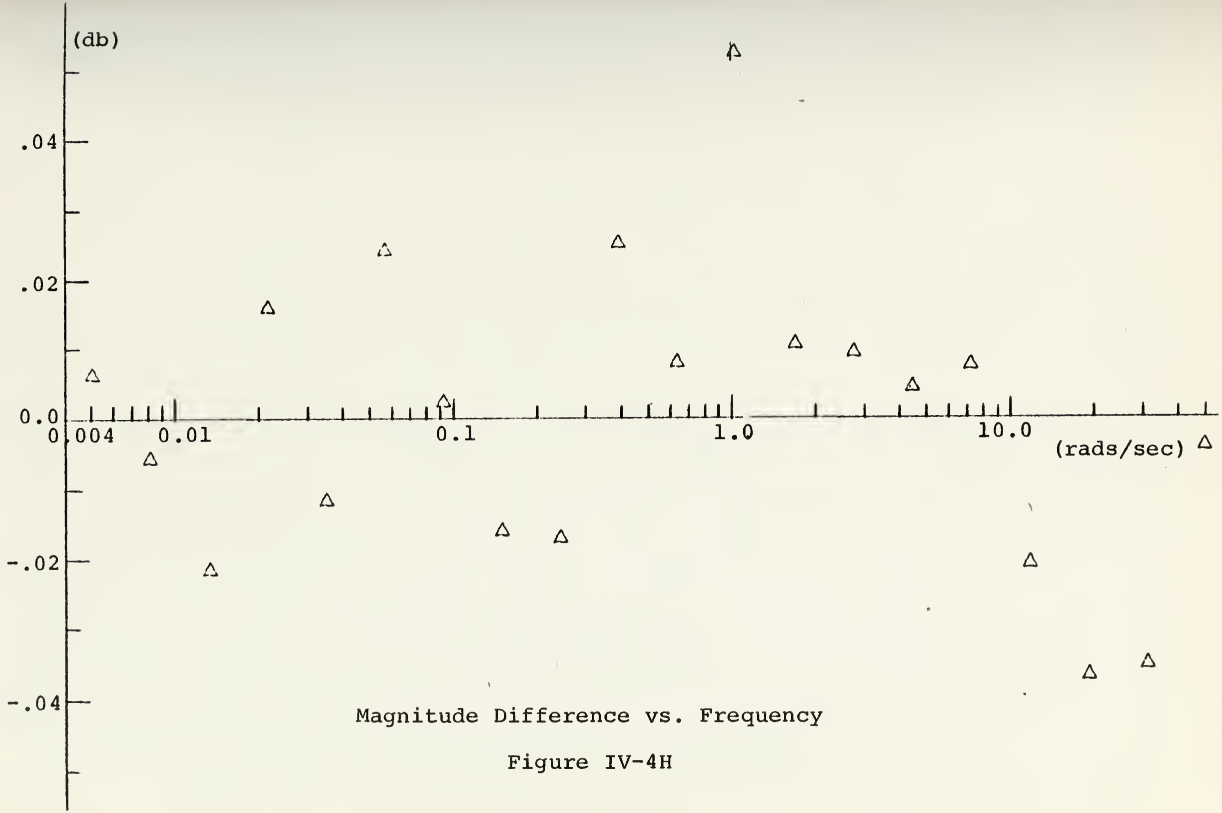
```
OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
ROOTS ARE:          REAL PART            IMAGINARY PART

                -1.542549E-02              0.0
-----------------------------------------------------------------------

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =   3.924033E-02

-----------------------------------------------------------------------
*WARNING**WARNING**WARNING**WARNING**WARNING**WARNING**WARNING**WARNING*

THE PHASE SIMULATION OF THE SYSTEM WAS ABANDONED
SINCE THE CALLS TO THE SEPARATE ROOT FINDING SUBROUTINES
FAILED TO FIND THE SYSTEM DENOMINATOR ROOTS

*WARNING**WARNING**WARNING**WARNING**WARNING**WARNING**WARNING**WARNING*
-----------------------------------------------------------------------
```

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
4.999999E-03	8.553139E 01	8.557966E 01	-1.072834E 02	-1.070000E 02
8.119997E-03	8.071274E 01	8.074354E 01	-1.166654E 02	-1.170000E 02
1.320000E-02	7.520157E 01	7.522351E 01	-1.287743E 02	-1.290000E 02
2.140000E-02	6.873960E 01	6.872325E 01	-1.413359E 02	-1.410000E 02
3.480000E-02	6.135204E 01	6.135371E 01	-1.514261E 02	-1.510000E 02
5.640000E-02	5.346347E 01	5.346042E 01	-1.571777E 02	-1.570000E 02
9.159994E-02	4.535258E 01	4.533026E 01	-1.583655E 02	-1.580000E 02
1.490000E-01	3.735859E 01	3.738461E 01	-1.550280E 02	-1.550000E 02
2.420000E-01	2.976214E 01	2.979915E 01	-1.474957E 02	-1.480000E 02
3.920000E-01	2.300933E 01	2.298436E 01	-1.375030E 02	-1.370000E 02
6.370000E-01	1.715445E 01	1.714654E 01	-1.275698E 02	-1.280000E 02
1.030000E 00	1.211532E 01	1.208287E 01	-1.217841E 02	-1.220000E 02
1.679999E 00	7.357646E 00	7.347118E 00	-1.217110E 02	-1.220000E 02
2.720000E 00	2.615990E 00	2.666670E 00	-1.284057E 02	-1.280000E 02
4.429999E 00	-2.575865E 00	-2.580223E 00	-1.426557E 02	-1.430000E 02
7.190000E 00	-8.366469E 00	-8.375550E 00	-1.644373E 02	-1.640000E 02
1.170000E 01	-1.697510E 01	-1.695422E 01	-1.908168E 02	-1.910000E 02
1.900000E 01	-2.697255E 01	-2.695573E 01	-2.156210E 02	-2.160000E 02
3.079999E 01	-3.830315E 01	-3.827280E 01	-2.347276E 02	-2.350000E 02
5.000000E 01	-5.040393E 01	-5.031986E 01	-2.478126E 02	-2.480000E 02

```
-----------------------------------------------------------------------
  THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE
-----------------------------------------------------------------------
```

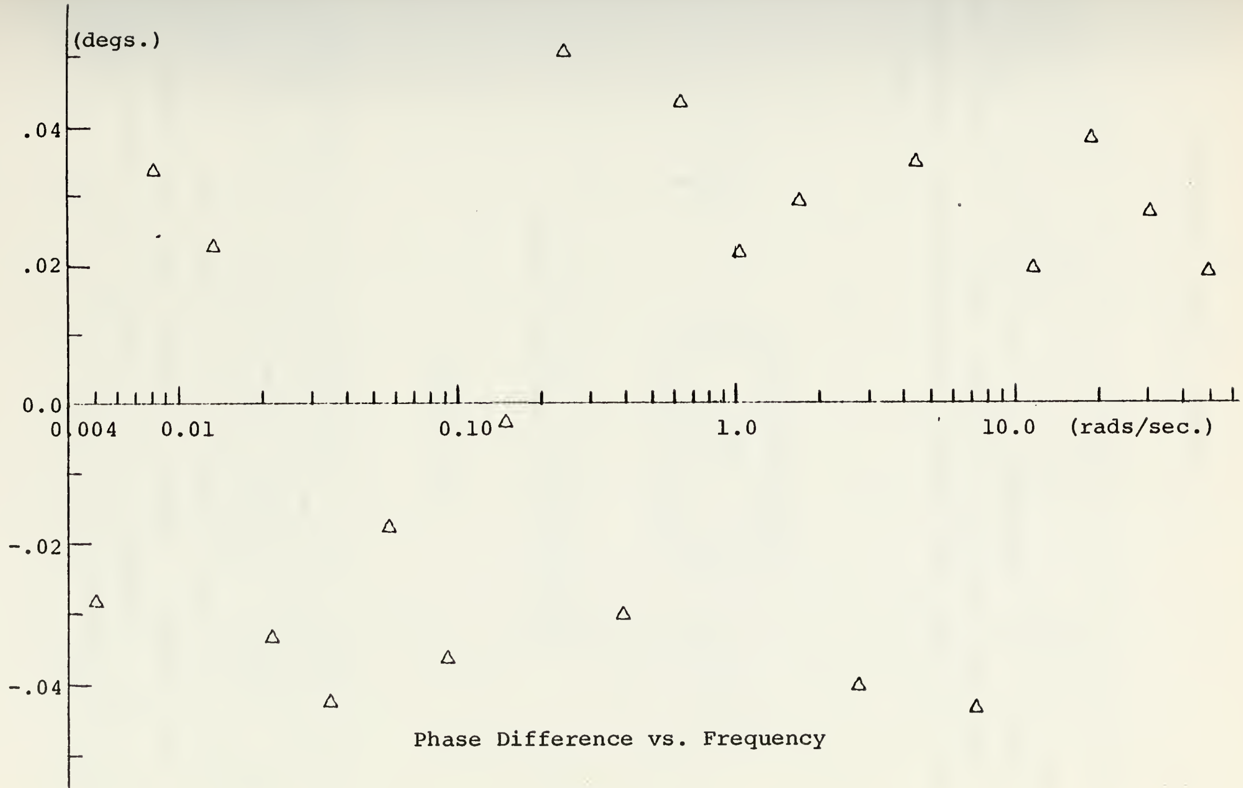
Computer Numerical Output Example 2

Figure IV-4G



Magnitude Difference vs. Frequency

Figure IV-4H

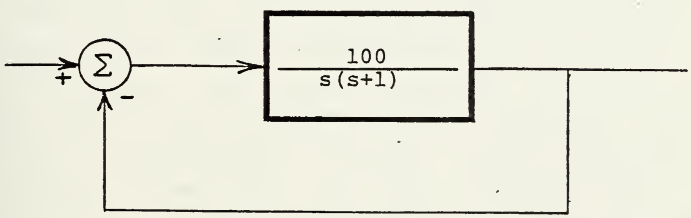


Phase Difference vs. Frequency

Figure IV-4I

3. Compensator Design, Example 3.

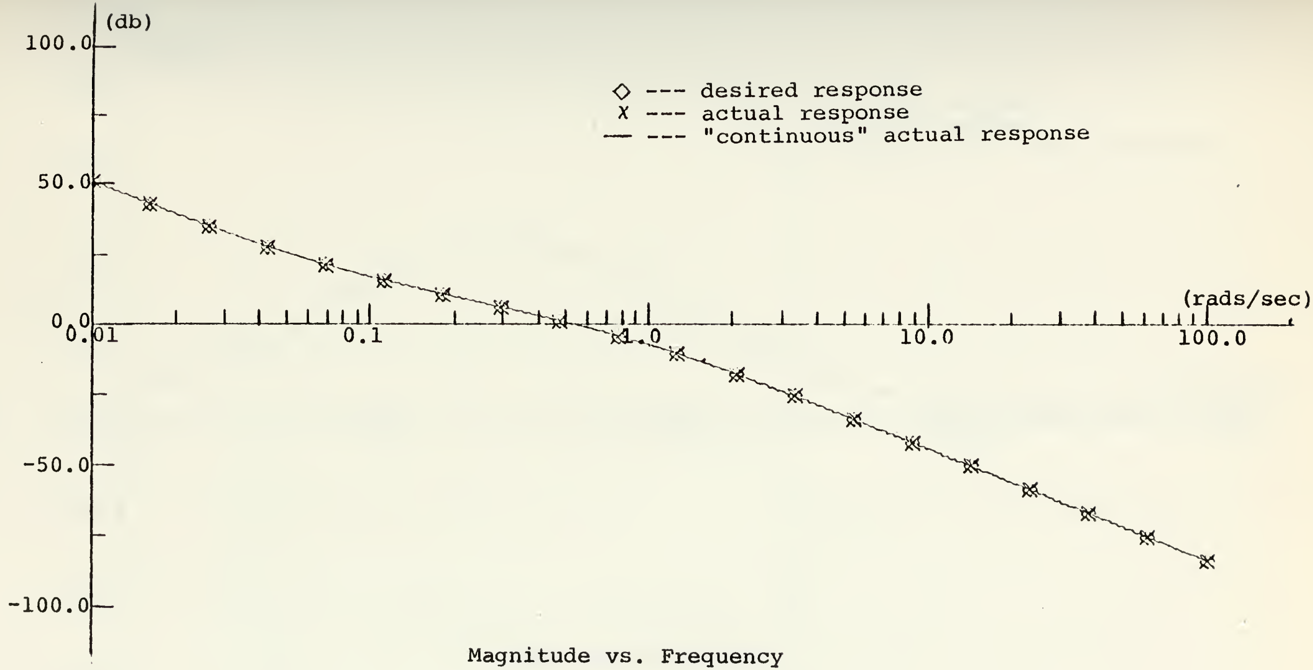
Another lag type compensation problem is presented in this example. This time, however, the existence of a fourth order polynomial for either the open loop system transfer function numerator or denominator has purposely been avoided to insure that the program will run to completion. The uncompensated system block diagram is shown in figure IV-5.



Design Example 3, Block Diagram

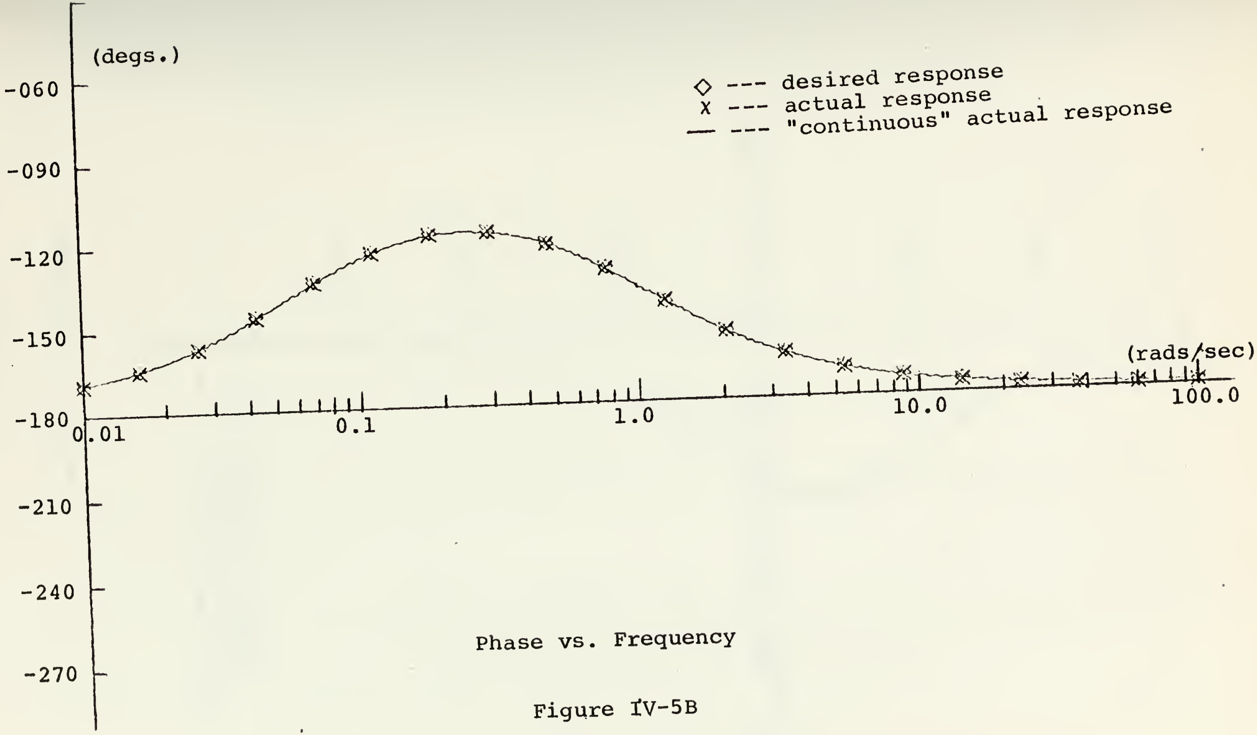
Figure IV-5

A Bode diagram for the uncompensated system along with a detailed discussion of the problem may be found in ref. 17. The uncompensated system does not possess an adequate phase margin to insure proper system performance. Thus, a single section lag compensator is to be used in order to achieve the desired open loop frequency response. Desired gain and phase profiles are selected for the compensated open loop frequency response using 20 discrete frequency points over the range from 0.01 to 100.0 radians. The profiles selected along with the results achieved are shown in figures IV-5A, IV-5B, and IV-5C. The numerical values assumed as initial estimates of the compensator parameters and the numerical results of the output are shown in figures IV-5D, IV-5E, and IV-5F. Plots of the differences between the actual magnitude and phase, and the desired magnitude and phase are shown in figures IV-5G and IV-5H. In this particular problem approximately 1200 iterations of the minimization routine and 1 minute and 25 seconds were required for solution of the problem. In order to illustrate the idea that for minimum phase systems the compensator parameters may be determined to within only a fixed loss or gain when using only the phase profile, this same problem was solved again using the type 3 cost function which considers only the difference between the desired and actual phase at the discrete frequencies. The results are shown in figures IV-5I, IV-5J, and IV-5K. As can be seen from the magnitude plot in figure IV-5I, there is a constant gain error in the resulting magnitude curve. The numerical values assumed at the start and those returned after the minimization of the cost function are shown in the computer output of figures IV-5L, IV-5M, and IV-5N. The resultant magnitude response is higher than desired. The necessary correction that must be applied to obtain the desired response may be read directly from figure IV-5O, which shows the difference between the actual and desired magnitude response. That is, as can be seen from figure IV-5C, the resultant magnitude of the open loop system after compensation is approximately 36.8 db higher than what is desired. Since the program does not alter any of the plant parameters this means that the compensator gain value returned from the program when the type 3 cost function was used must be decreased by approximately a factor of 69.2. A comparison of the compensator gain values shown in figures IV-5F and IV-5N will show that they due indeed differ by this value. Also the astute observer may notice that there is a slight difference in the compensator pole value returned as the solution when the problem was solved using the type 3 cost function. While the difference in values is small it is suspected that in this particular case it is due to the fact that only a finite frequency range has been considered and the lower frequencies have been "slighted" somewhat in that the nonlinear phase curve has not yet flattened out at 0.1 radians. This is also the same situation that existed the first time the program was executed, but in this case the magnitude profile was also taken into account in the cost function and any deviations of the resultant magnitude values dominated the cost function value. In other words, it appears that in this particular problem the cost function is more sensitive to variations in the magnitude values than in the variations in the phase values.



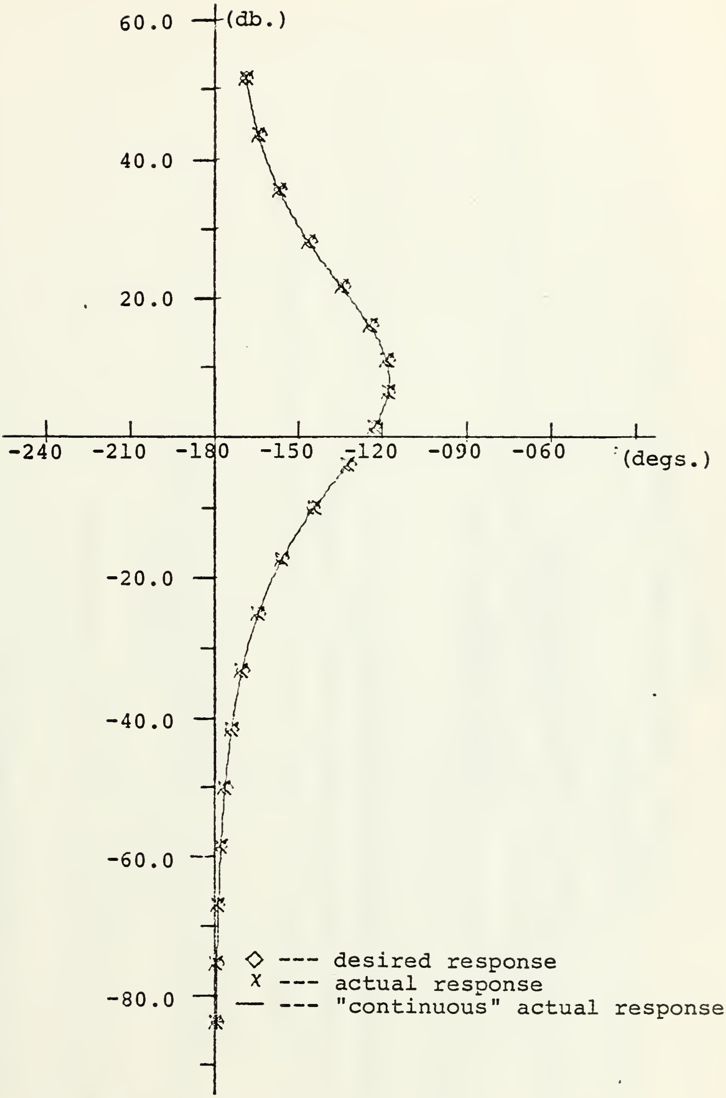
Magnitude vs. Frequency

Figure IV-5A



Phase vs. Frequency

Figure IV-5B



Magnitude vs. Phase

Figure IV-5C

```
TITLE --- COMPEN. OPTIMIZ. EXAMPLE 3   20PTS
---------------------------------------------------------------------
          UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 02

          UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
      COEFFICIENTS IN ASCENDING POWERS OF S

        1.000000E 00

          UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
      COEFFICIENTS IN ASCENDING POWERS OF S

        0.0                 1.000000E 00     1.000000E 00

          UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
    ARE:      REAL PART            IMAGINARY PART

                0.0                  0.0

               -1.000000E 00         0.0
---------------------------------------------------------------------
          COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

          COMPENSATOR TRANSFER FUNCTION NUMERATOR
      COEFFICIENTS IN ASCENDING POWERS OF S

        1.000000E 00     1.000000E 00

          COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
    ARE:      REAL PART            IMAGINARY PART

               -1.000000E 00         0.0

          COMPENSATOR TRANSFER FUNCTION DENOMINATOR
      COEFFICIENTS IN ASCENDING POWERS OF S

        1.000000E 00     1.000000E 00
```

Computer Numerical Output Example 3, Run #1

Figure IV-5D

```
    COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
 ARE:        REAL PART            IMAGINARY PART

           -1.000000E 00          0.0
-----------------------------------------------------------------------
-----------------------------------------------------------------------

     THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

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

     THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

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

     THE COST FUNCTION TO BE USED IS THE TYPE    1

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

THE MINIMUM COST FUNCTION VALUE =  2.522382E-04

THE ERROR RETURN CODE FROM ECXPLX =    0

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

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

     1.832424E 00    3.058981E 01

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 ROOTS ARE:         REAL PART           IMAGINARY PART

                  -5.990308E-02          0.0
-----------------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

     2.086647E 00    4.892434E 03
```

Computer Numerical Output Example 3, Run #1

Figure IV-5E

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR

ROOTS ARE:	REAL PART	IMAGINARY PART
	-4.265048E-04	0.0

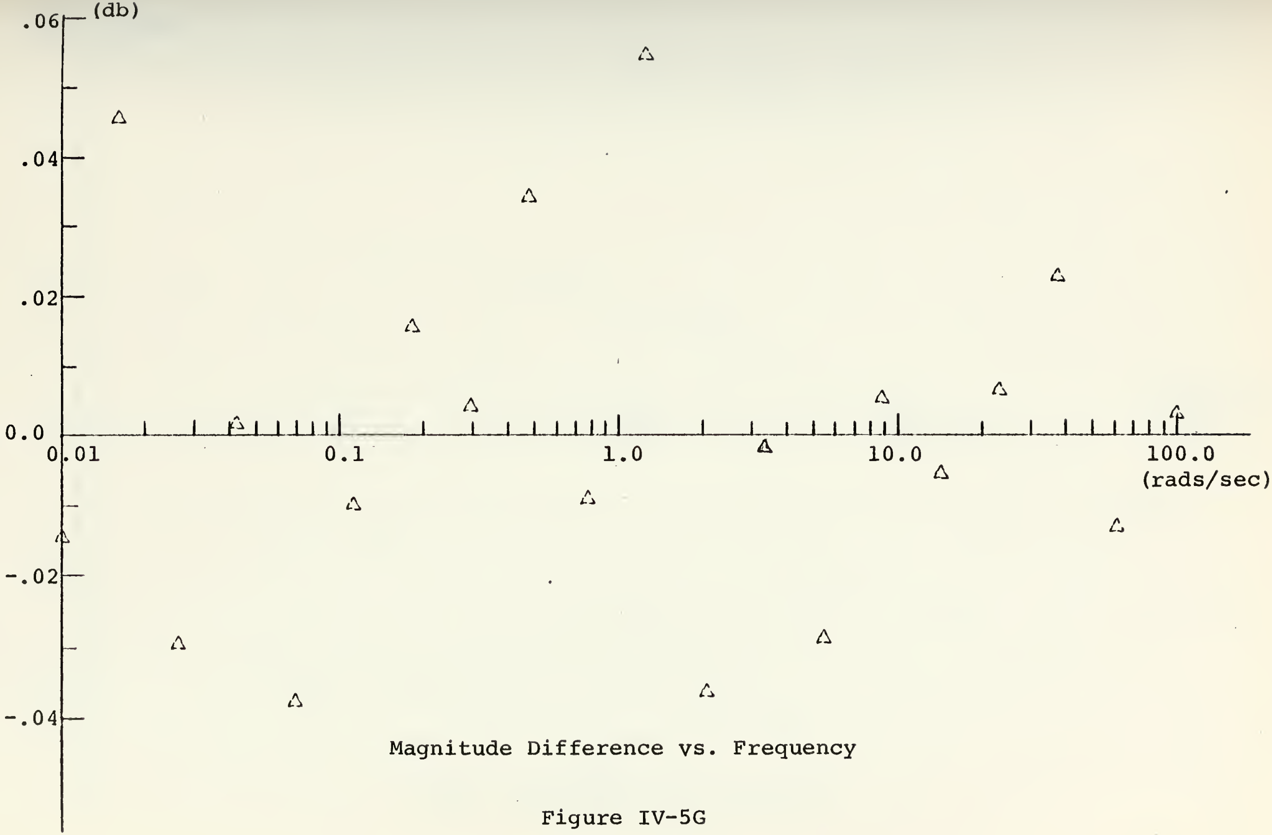
OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 6.252471E-03

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999998E-03	5.158107E 01	5.159566E 01	-1.686534E 02	-1.690000E 02
1.620000E-02	4.339182E 01	4.334634E 01	-1.642871E 02	-1.640000E 02
2.640000E-02	3.537256E 01	3.540230E 01	-1.568031E 02	-1.570000E 02
4.280000E-02	2.799507E 01	2.799347E 01	-1.463345E 02	-1.460000E 02
6.949997E-02	2.147296E 01	2.151093E 01	-1.343827E 02	-1.340000E 02
1.130000E-01	1.587987E 01	1.588976E 01	-1.241597E 02	-1.240000E 02
1.830000E-01	1.097100E 01	1.095549E 01	-1.183622E 02	-1.180000E 02
2.980000E-01	6.239247E 00	6.235075E 00	-1.178781E 02	-1.180000E 02
4.830000E-01	1.397734E 00	1.363709E 00	-1.227599E 02	-1.230000E 02
7.850000E-01	-4.036157E 00	-4.026987E 00	-1.324646E 02	-1.320000E 02
1.270000E 00	-1.031660E 01	-1.037114E 01	-1.444644E 02	-1.450000E 02
2.070000E 00	-1.762509E 01	-1.758852E 01	-1.558610E 02	-1.560000E 02
3.360000E 00	-2.549973E 01	-2.549309E 01	-1.644401E 02	-1.640000E 02
5.459999E 00	-3.370943E 01	-3.368059E 01	-1.702453E 02	-1.700000E 02
8.860000E 00	-4.203107E 01	-4.203645E 01	-1.739451E 02	-1.740000E 02
1.440000E 01	-5.043427E 01	-5.042867E 01	-1.762640E 02	-1.760000E 02
2.339999E 01	-5.885547E 01	-5.886189E 01	-1.776985E 02	-1.780000E 02
3.789999E 01	-6.722755E 01	-6.725021E 01	-1.785784E 02	-1.790000E 02
6.159999E 01	-7.566331E 01	-7.565030E 01	-1.791251E 02	-1.790000E 02
1.000000E 02	-8.407938E 01	-8.408238E 01	-1.794612E 02	-1.790000E 02

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES THAT THE SYSTEM IS STABLE

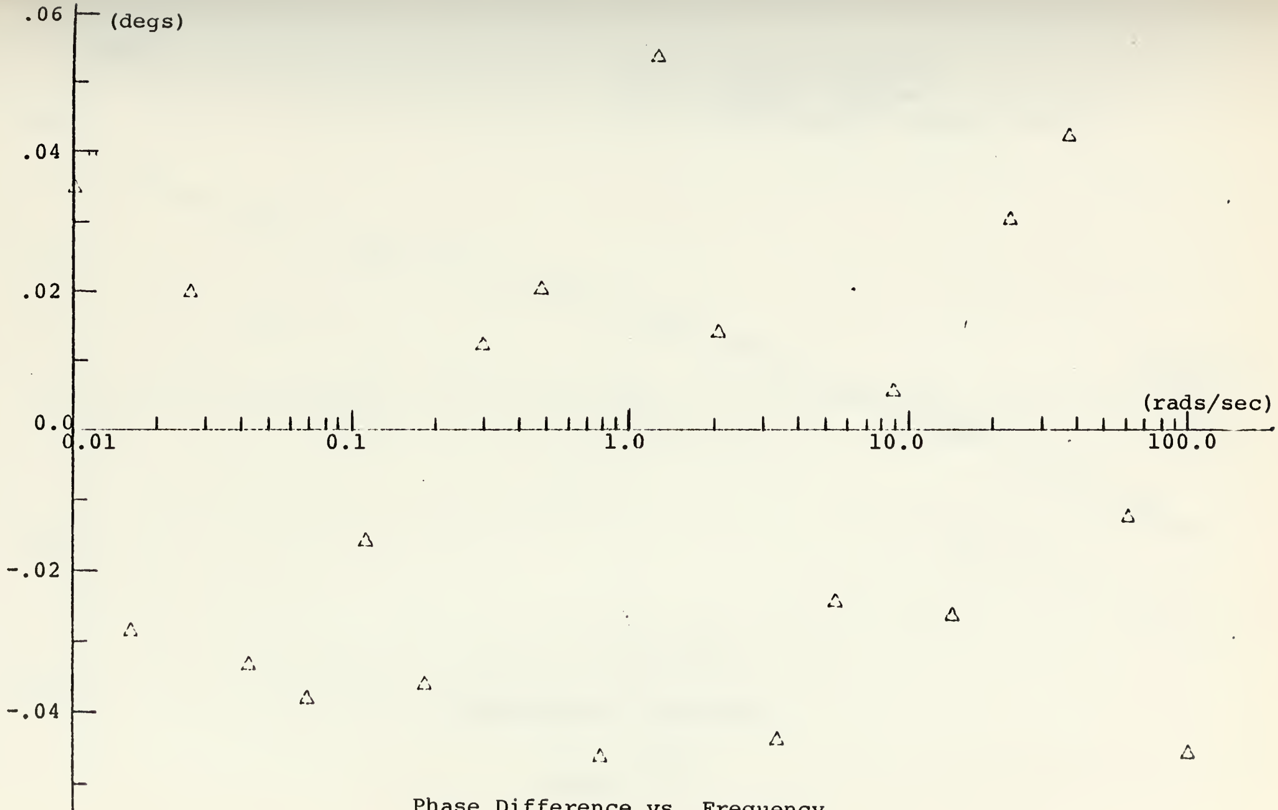
Computer Numerical Output Example 3, Run #1

Figure IV-5F



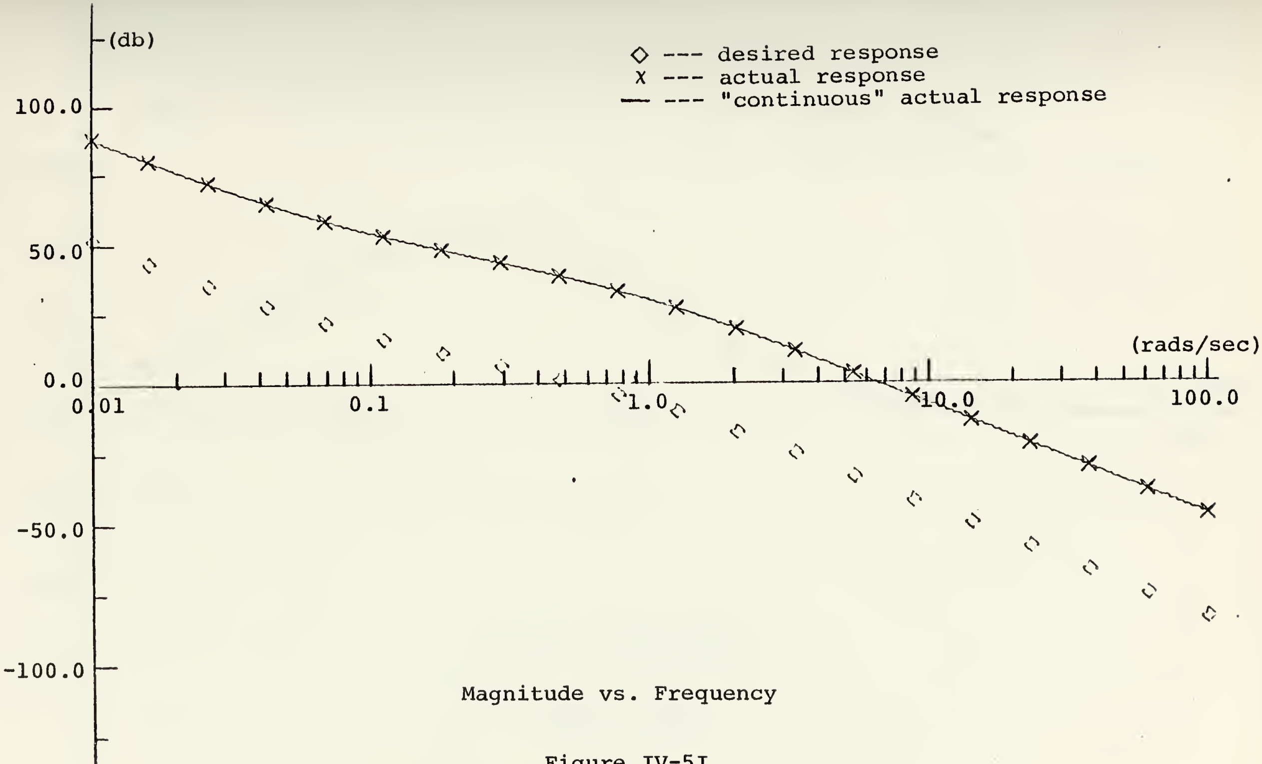
Magnitude Difference vs. Frequency

Figure IV-5G



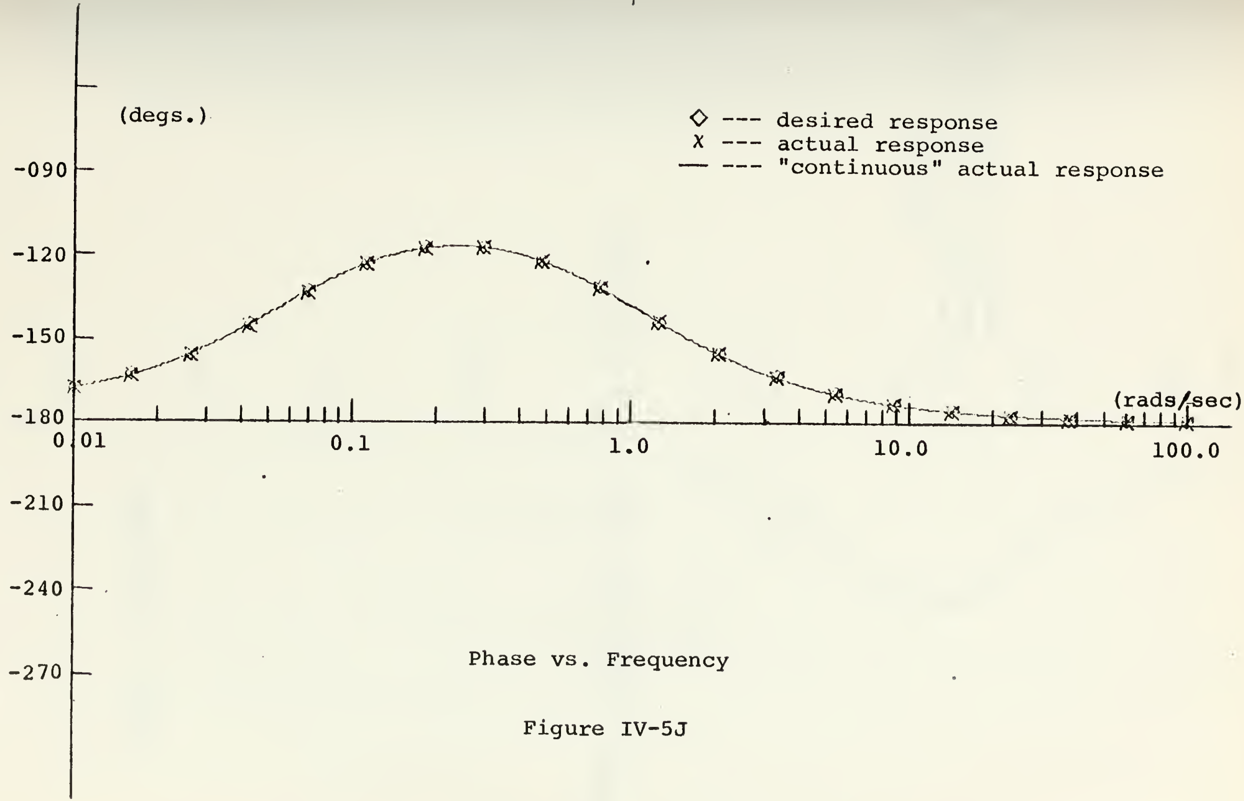
Phase Difference vs. Frequency

Figure IV-5H



Magnitude vs. Frequency

Figure IV-5I



Phase vs. Frequency

Figure IV-5J

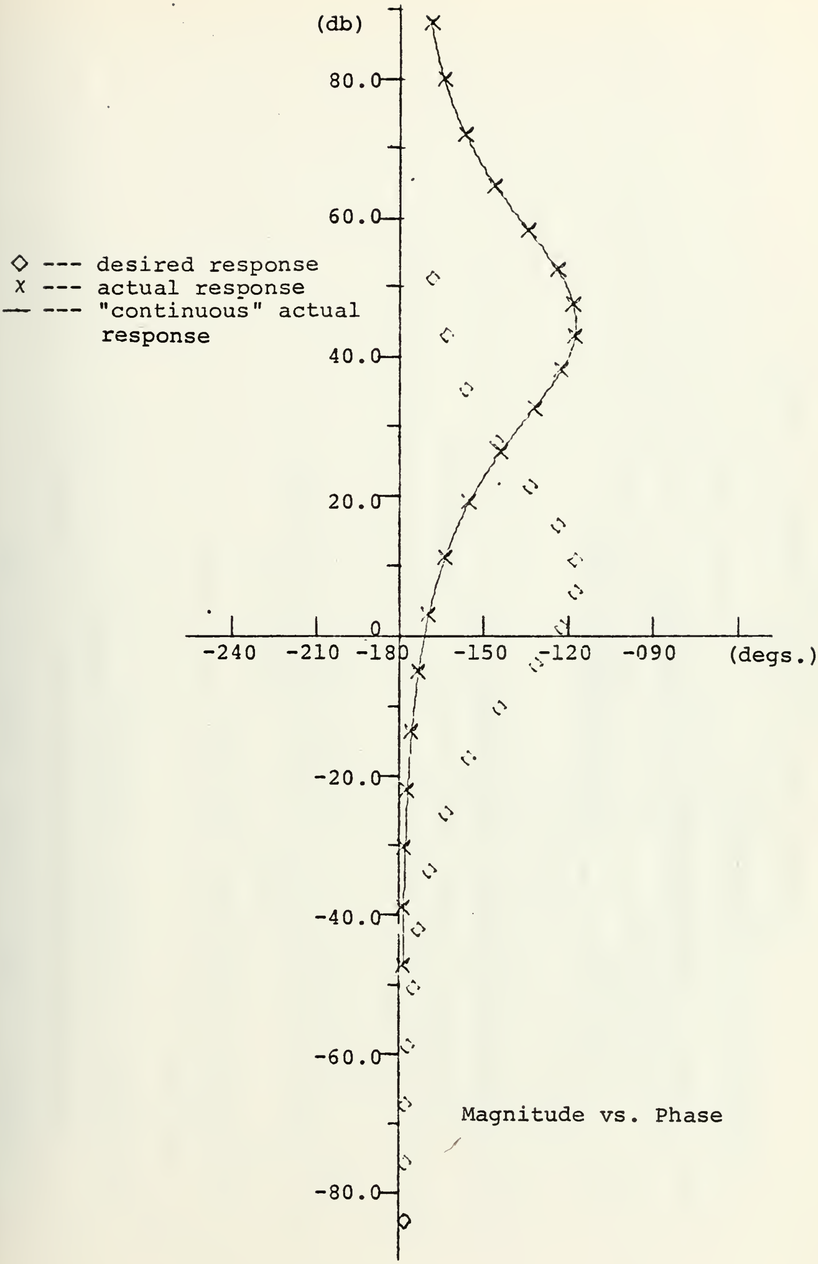

(db)
80.0
60.0
40.0
20.0
0
-20.0
-40.0
-60.0
-80.0
◇ --- desired response
X --- actual response
— --- "continuous" actual response
-240
-210
-180
-150
-120
-090
(degs.)
Magnitude vs. Phase

Figure IV-5K

```
TITLE --- COMPEN. OPTIMIZ. EXAMPLE 3B  20PTS

UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 02

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

0.0              1.000000E 00    1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:     REAL PART          IMAGINARY PART

          0.0                  0.0

         -1.000000E 00         0.0

COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00    1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
ARE:     REAL PART          IMAGINARY PART

         -1.000000E 00         0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00    1.000000E 00
```

Computer Numerical Output Example 3, Run #2

Figure IV-5L

```
      COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:        REAL PART           IMAGINARY PART

         -1.000000E 00          0.0
-----------------------------------------------------------------------
-----------------------------------------------------------------------

      THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION
-----------------------------------------------------------------------
-----------------------------------------------------------------------

      THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000
-----------------------------------------------------------------------
-----------------------------------------------------------------------

      THE COST FUNCTION TO BE USED IS THE TYPE    3
-----------------------------------------------------------------------

THE MINIMUM COST FUNCTION VALUE =  7.442613E-05

THE ERROR RETURN CODE FROM BOXPLX =    0
-----------------------------------------------------------------------
-----------------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   7.943527E 03    1.341418E 05

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
ROOTS ARE:        REAL PART           IMAGINARY PART

                 -5.921741E-02          0.0
-----------------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.132473E 02    3.089048E 05
```

Computer Numerical Output Example 3, Run #2
Figure IV-5M

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
ROOTS ARE: REAL PART IMAGINARY PART

-3.666091E-04 0.0

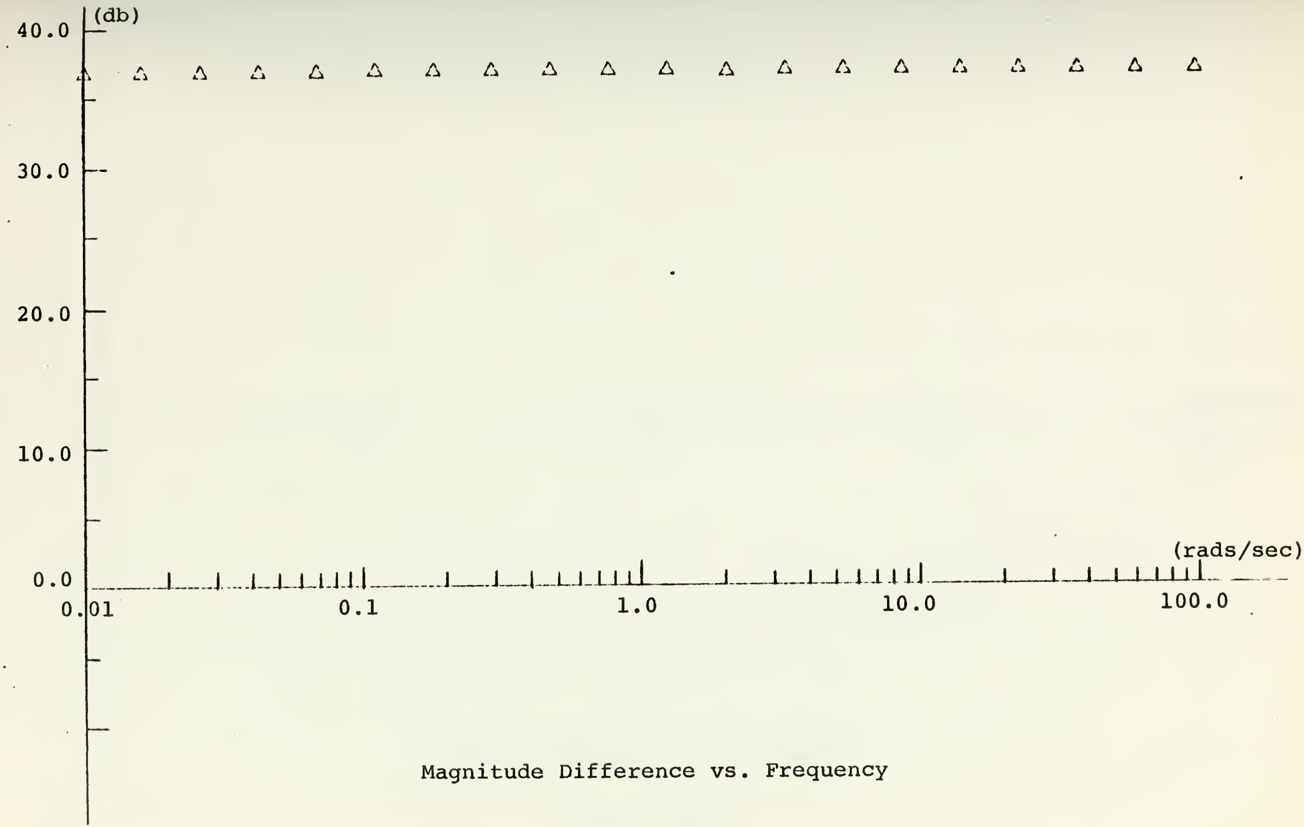
OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 4.342495E-01

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999998E-03	8.831963E 01	5.159566E 01	-1.688882E 02	-1.690000E 02
1.620000E-02	8.013324E 01	4.334634E 01	-1.643318E 02	-1.640000E 02
2.640000E-02	7.212305E 01	3.540230E 01	-1.566887E 02	-1.570000E 02
4.280000E-02	6.476300E 01	2.799347E 01	-1.461021E 02	-1.460000E 02
6.949997E-02	5.826442E 01	2.151093E 01	-1.341061E 02	-1.340000E 02
1.130000E-01	5.269189E 01	1.588976E 01	-1.239180E 02	-1.240000E 02
1.830000E-01	4.779518E 01	1.095549E 01	-1.181869E 02	-1.180000E 02
2.980000E-01	4.306915E 01	6.235075E 00	-1.177628E 02	-1.180000E 02
4.830000E-01	3.823000E 01	1.363709E 00	-1.227269E 02	-1.230000E 02
7.850000E-01	3.279701E 01	-4.026987E 00	-1.324192E 02	-1.320000E 02
1.270000E 00	2.651669E 01	-1.037114E 01	-1.444362E 02	-1.450000E 02
2.070000E 00	1.920857E 01	-1.758852E 01	-1.558436E 02	-1.560000E 02
3.360000E 00	1.133397E 01	-2.549809E 01	-1.644295E 02	-1.640000E 02
5.459999E 00	3.124302E 00	-3.363059E 01	-1.702287E 02	-1.700000E 02
8.860000E 00	-5.197330E 00	-4.203645E 01	-1.739409E 02	-1.740000E 02
1.440000E 01	-1.360053E 01	-5.042867E 01	-1.762615E 02	-1.760000E 02
2.339999E 01	-2.202173E 01	-5.886189E 01	-1.776968E 02	-1.780000E 02
3.789999E 01	-3.039377E 01	-6.725021E 01	-1.785774E 02	-1.790000E 02
6.159999E 01	-3.882957E 01	-7.565030E 01	-1.791245E 02	-1.790000E 02
1.000000E 02	-4.724564E 01	-8.408233E 01	-1.794607E 02	-1.790000E 02

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE

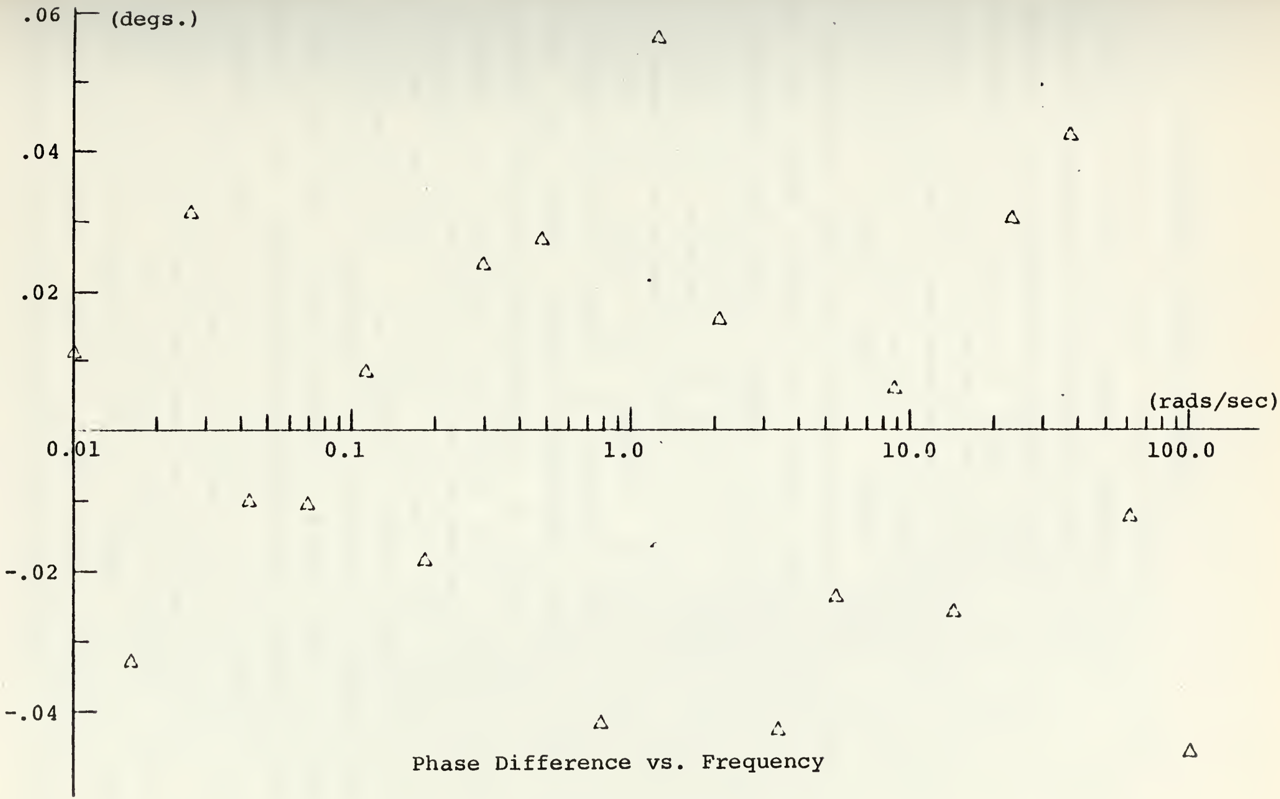
Computer Numerical Output Example 3, Run #2

Figure IV-5N



Magnitude Difference vs. Frequency

Figure IV-50



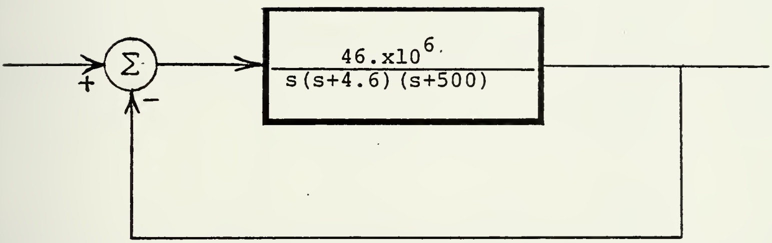
Phase Difference vs. Frequency

Figure IV-5P

4. Compensator Design, Example 4

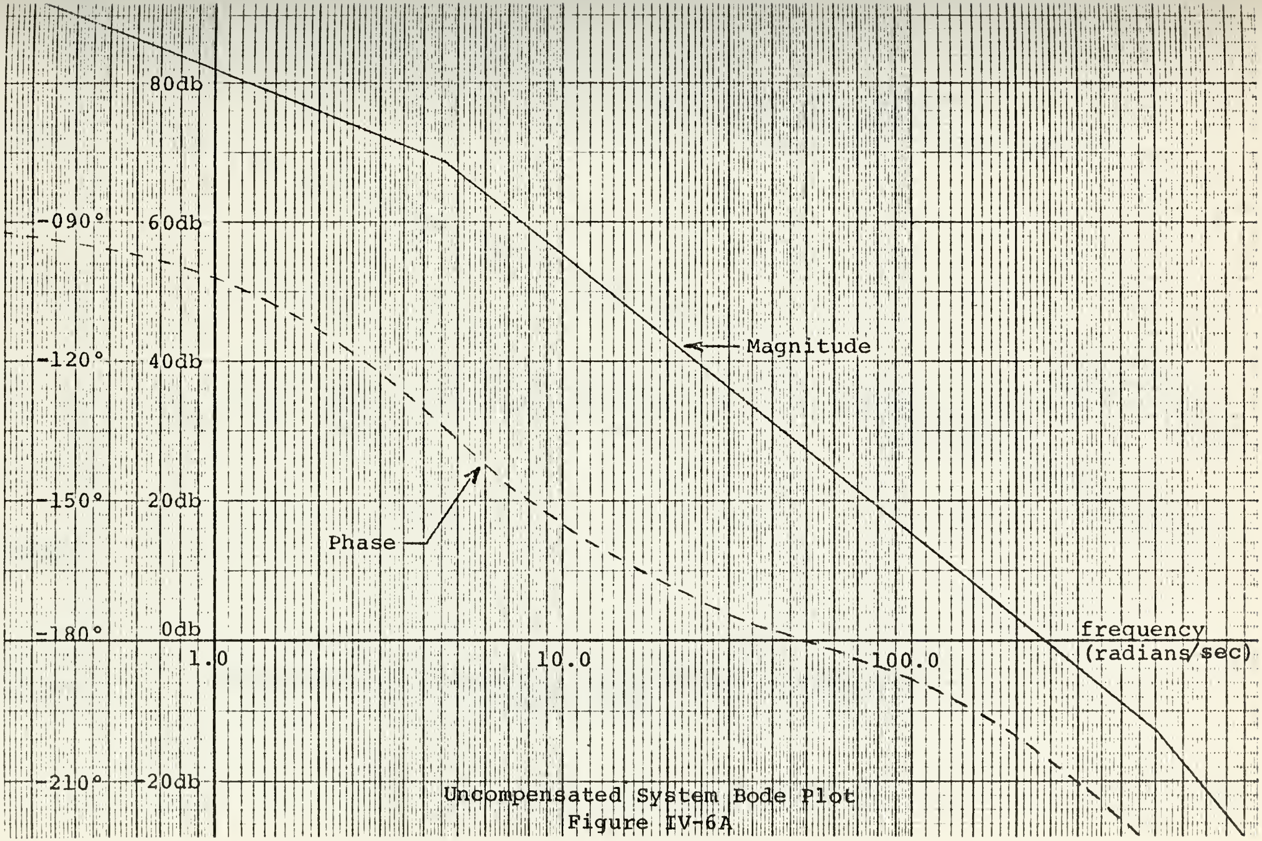
In this example a slightly more complicated problem is presented, in that the final solution requires a double section compensator of the form of a notch filter. A block diagram of the uncompensated system is shown in figure IV-6. As discussed by Thaler and Brown [18], from whom this example was taken, the uncompensated system is unstable and the Bode diagram of the uncompensated system shown in figure IV-6A indicates a phase margin of approximately -30 degrees. With such a large negative phase margin for the uncompensated system and the rapid drop in slope of both the gain and phase curves the designer might suspect that compensation using only a single section compensator will be difficult to accomplish. Again desired magnitude and phase profiles are selected and initially a single section of compensation is assumed to observe how close to the desired response the system will perform. As can be seen in the resultant graphical output of figures IV-6A, IV-6B, and IV-6C, the single section compensator fails to meet the required frequency specifications. In fact, while it is possible to stabilize ths system with a single section compensator the bandwidth will be excessively large and the desired magnitude profile prevents the program from accomplishing this. The numerical results returned from the single section compensator run are shown in figures IV-6D, IV-6E, and IV-6F. As can be seen in figure IV-6F, the Routh test of the characteristic equation also indicates system instability which is to be expected. The magnitude and phase difference curves of figures IV-6G and IV-6H, which indicate the difference between the specified magnitude and phase profiles and the values actually achieved, indicate that additional compensation may be needed at the higher frequency ranges. Thus a double section compensator was assumed and the results achieved are illustrated in figures

IV-6I, IV-6J, and IV-6K. The program is able to satisfy the specifications with a double section of compensation and the parameters required to accomplish this are shown in figures IV-6L, IV-6M, and IV-6N. The differences between the desired and specified magnitude and phase values for the double section of compensation inserted in the system are shown in figures IV-6O and IV-6P.

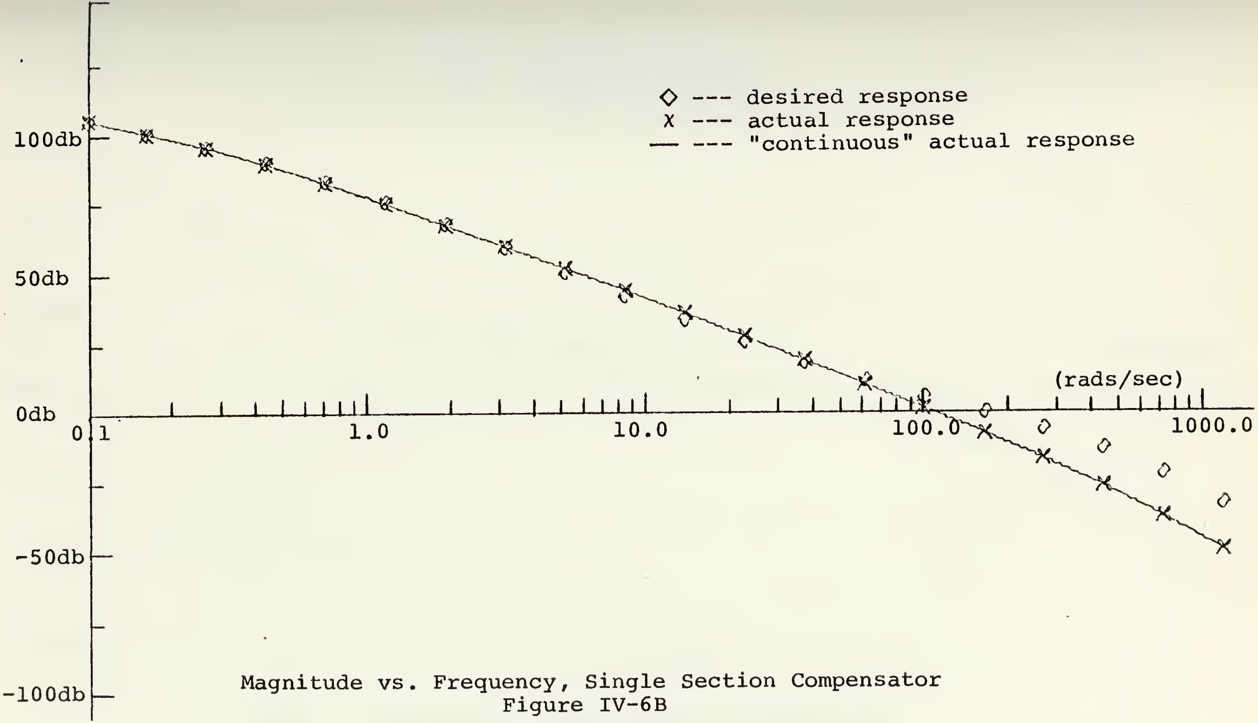


Design Example 4, Block Diagram

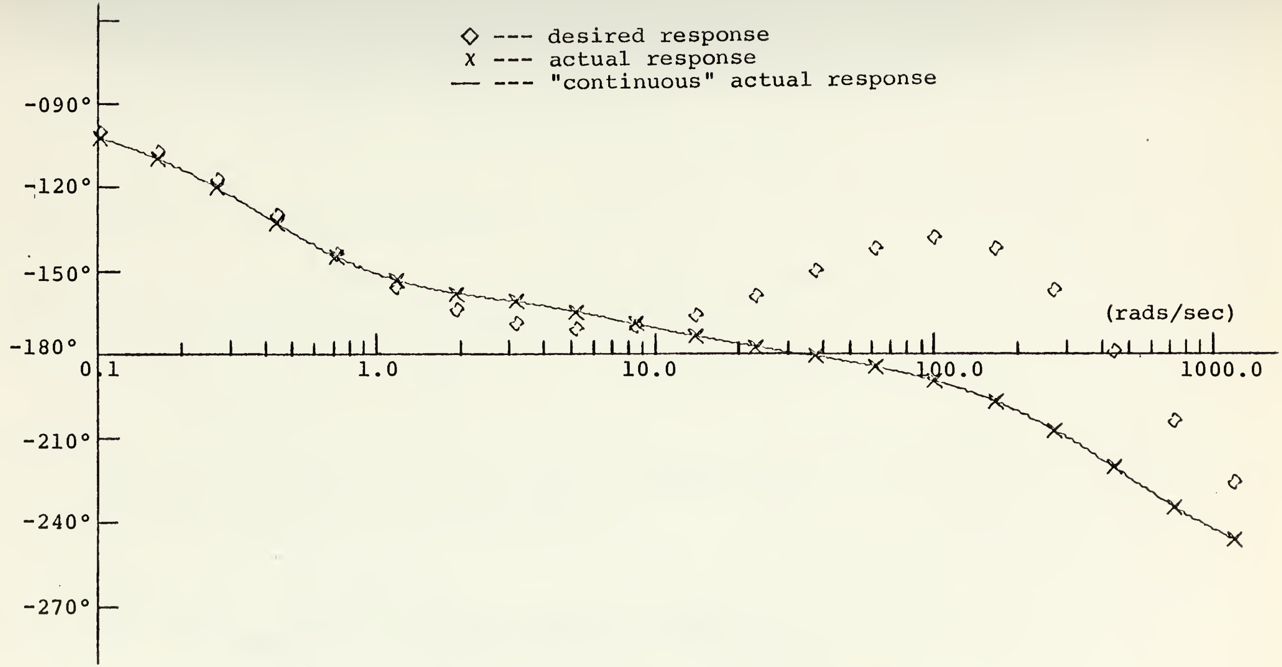
Figure IV-6



Uncompensated System Bode Plot
Figure IV-6A



Magnitude vs. Frequency, Single Section Compensator
Figure IV-6B



Phase vs. Frequency, Single Section Compensator

Figure IV-6C

```
TITLE --- COMP. OPT. EXP. 4  2 PTS TYPE 1 CST SOL

UNCOMPENSATED TRANSFER FUNCTION GAIN =  4.600000E 07

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

0.0          2.300000E 03   5.045999E 02   1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART          IMAGINARY PART

          0.0                0.0
          -4.599999E 00      0.0
          -5.000000E 02      0.0

COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00    1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
ARE:      REAL PART          IMAGINARY PART

          -1.000000E 00      0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00    1.000000E 00
```

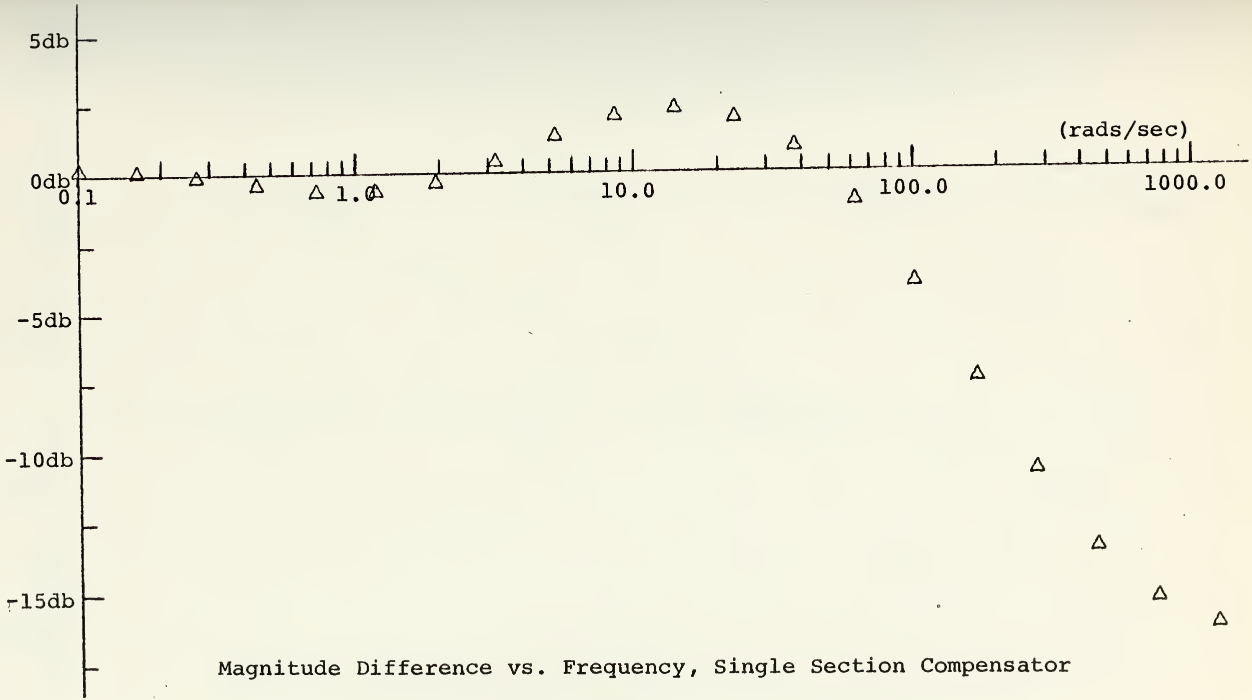
Computer Numerical Output
Figure IV-6D

```
COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART              IMAGINARY PART

        -1.000000E 00          0.0
------------------------------------------------------------------------
------------------------------------------------------------------------

    THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM-PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

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

    THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

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

    THE COST FUNCTION TO BE USED IS THE TYPE    1

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


THE FIRST VERTEX VALUES ARE:

   3.39[illegible] 05   1.08[illegible]201E 05   3.292[illegible] 05   7.[illegible]01[illegible] 05


THE MINIMUM COST FUNCTION VALUE = 3.[illegible]21[illegible]01E 00

THE ERROR RETURN CODE FROM BOXPLX = 2

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

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   3.39[illegible]76E 05    1.08[illegible]201E 05
```

Computer Numerical Output
Figure IV-6E

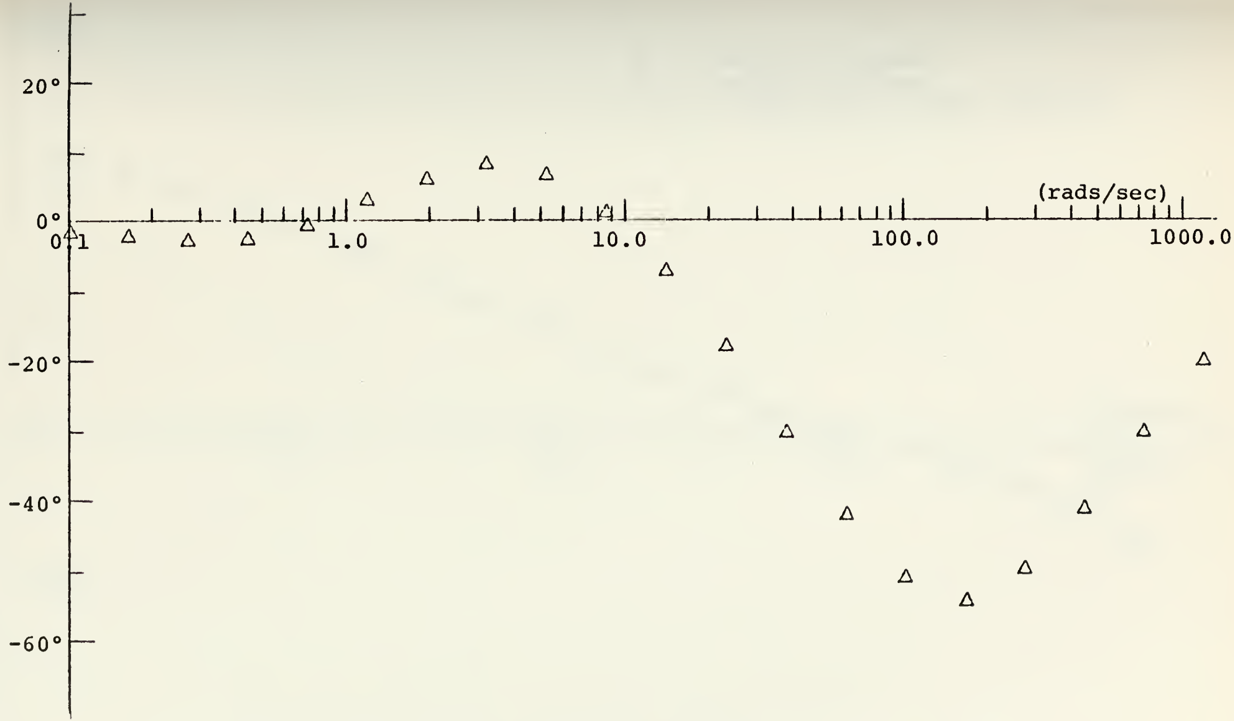
```
      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 ROOTS ARE:       REAL PART          IMAGINARY PART

             -3.128681E 00        0.0
---------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S.

   3.292767E 05    7.801555E 05

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 ROOTS ARE:       REAL PART          IMAGINARY PART

             -4.220654E-01        0.0
---------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =  1.392288E-01


---------------------------------------------------------------
********  W A R N I N G  ********   ********  W A R N I N G  ********

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES SYSTEM
INSTABILITY

********  W A R N I N G  ********   ********  W A R N I N G  ********
---------------------------------------------------------------
```

Computer Numerical Output
Figure IV-6F



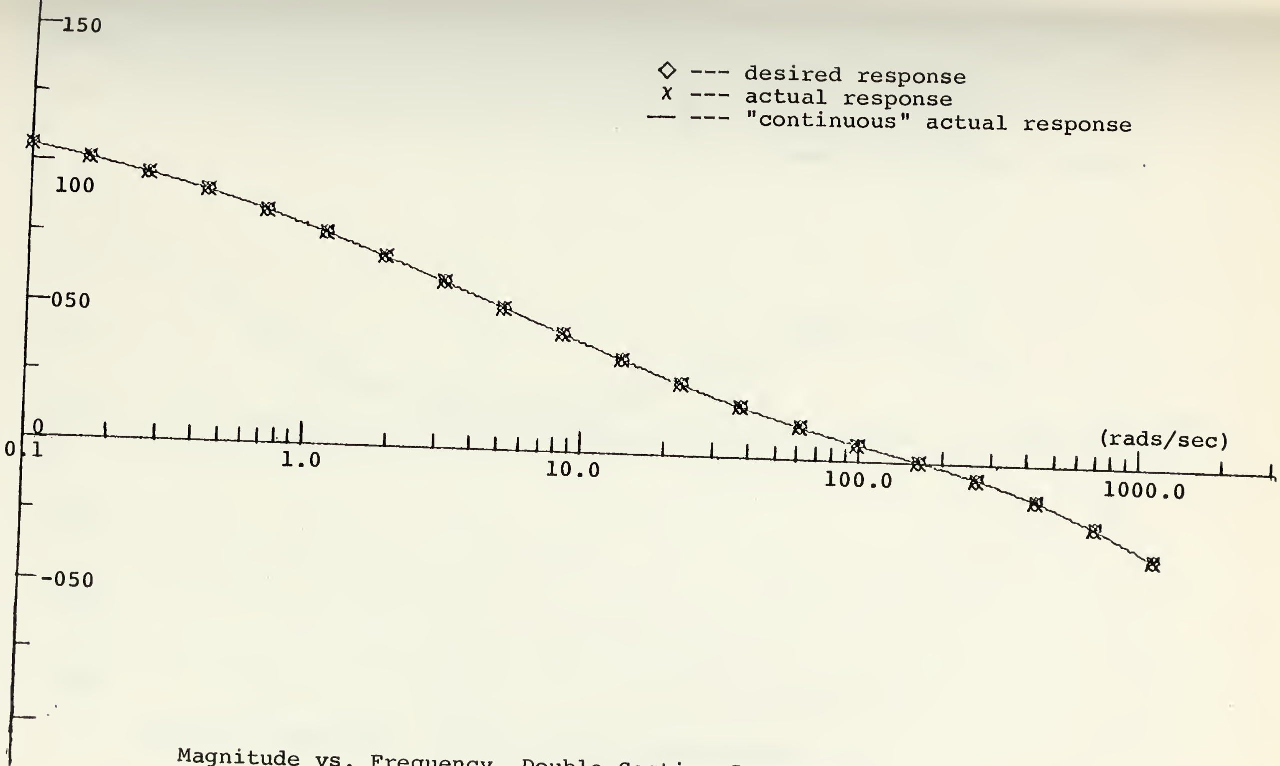
Magnitude Difference vs. Frequency, Single Section Compensator

Figure IV-6G



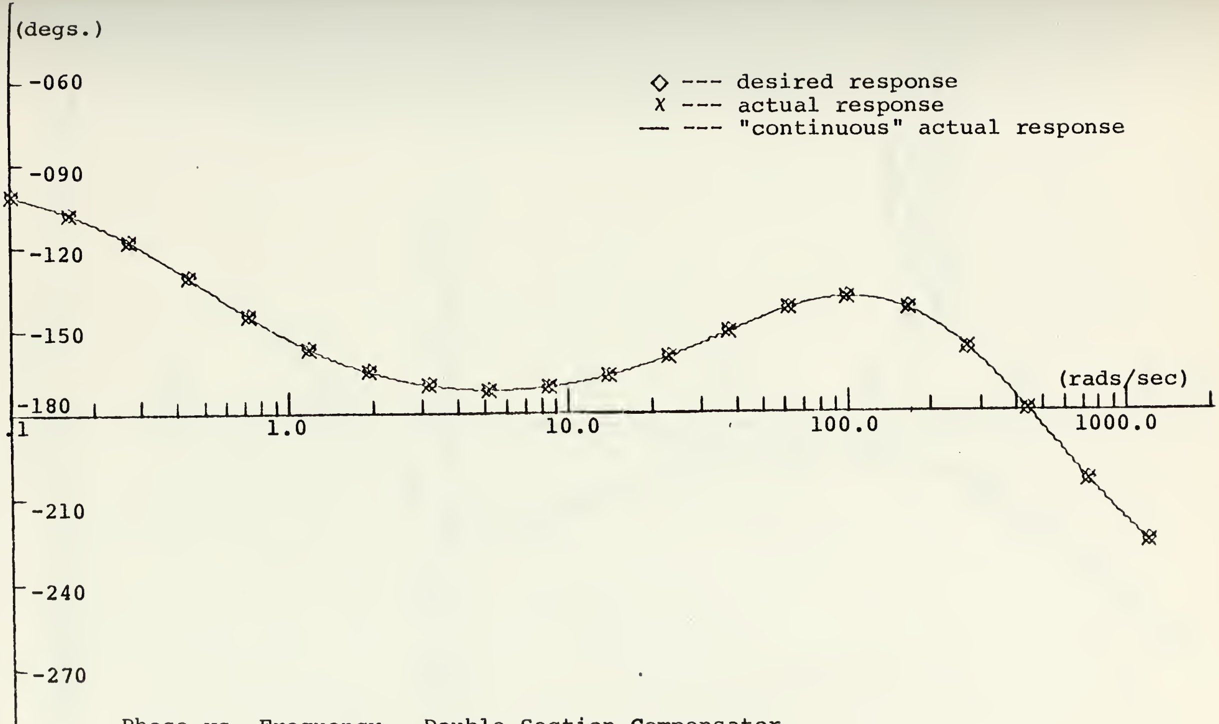
Phase Difference vs. Frequency, Single Section Compensator

Figure IV-6H



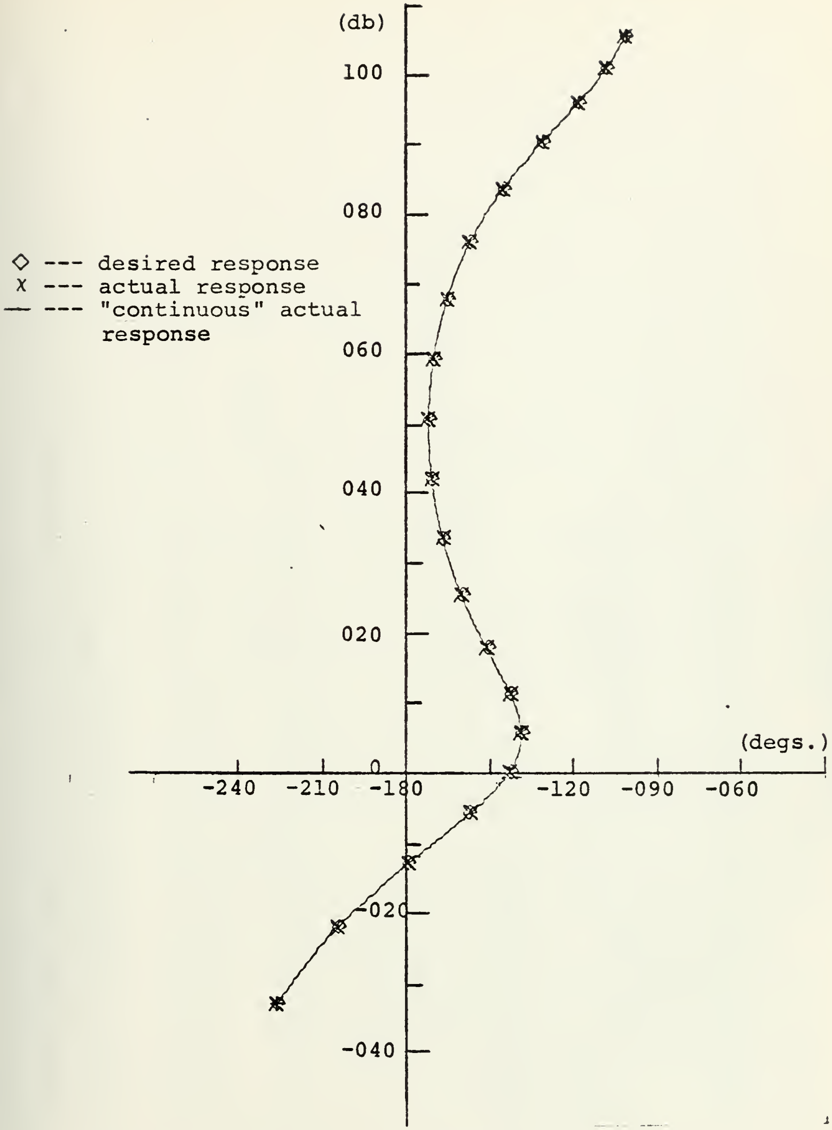
Magnitude vs. Frequency, Double Section Compensator

Figure IV-6I



Phase vs. Frequency, Double Section Compensator

Figure IV-6J



Magnitude vs. Phase, Double Section Compensator
Figure IV-6K

```
TITLE --- COMP. OPT. EXP. 4 ROOTS. TYPE 1 CSF.

UNCOMPENSATED TRANSFER FUNCTION GAIN =  4.600000E 07

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

0.0           2.300000E 03   5.045999E 02   1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART              IMAGINARY PART

        0.0                      0.0
       -4.599999E 00             0.0
       -5.000000E 02             0.0

COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00   2.000000E 00   1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
ARE:      REAL PART              IMAGINARY PART

       -1.000000E 00             0.0
       -1.000000E 00             0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00   2.000000E 00   1.000000E 00
```

Computer Numerical Output
Figure IV-6L

```
COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART            IMAGINARY PART

     -1.000000E 00          0.0

     -1.000000E 00          0.0
------------------------------------------------------------------------
------------------------------------------------------------------------

     THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

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

     THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

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

     THE COST FUNCTION TO BE USED IS THE TYPE   1

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


THE BEST VERTEX VALUES ARE:

   4.906714E 05   1.082894E 05   1.964592E 03   4.907928E 05   9.853190E 05
  1.963274E 03


THE MINIMUM COST FUNCTION VALUE =  1.532074E-04

THE ERROR RETURN CODE FROM BOXPLX =   0

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

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
  COEFFICIENTS IN ASCENDING POWERS OF S

   4.906714E 05   1.082894E 05   1.964592E 03
```

Computer Numerical Output

Figure IV-6M

OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR

ROOTS ARE:	REAL PART	IMAGINARY PART
	-5.013924E 01	0.0
	-4.981477E 00	0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

4.907928E 05 9.853190E 05 1.963274E 04

OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR

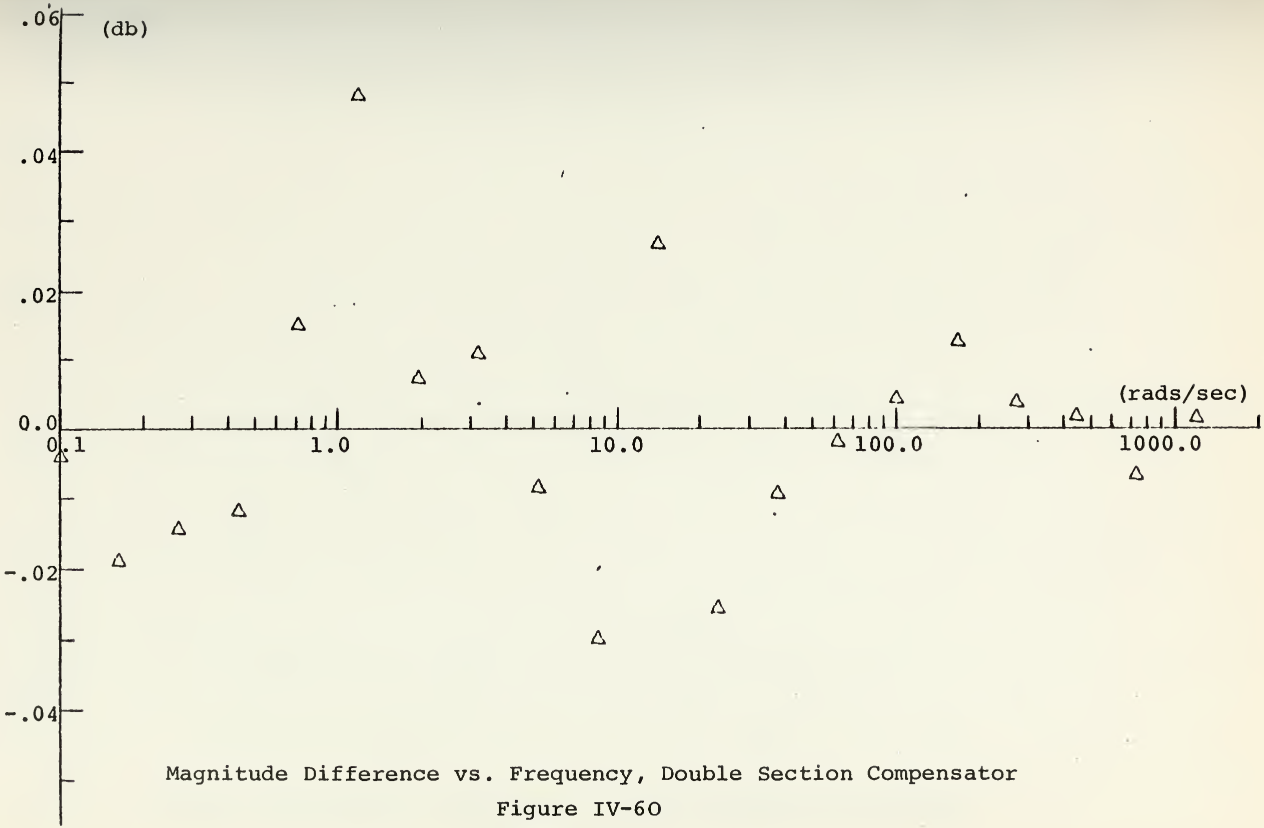
ROOTS ARE:	REAL PART	IMAGINARY PART
	-5.013762E 02	0.0
	-4.985962E-01	0.0

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.000670E 00

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999996E-02	1.058469E 02	1.053451E 02	-1.013448E 02	-1.010000E 02
1.640000E-01	1.012747E 02	1.012492E 02	-1.082134E 02	-1.080000E 02
2.690000E-01	9.631134E 01	9.632481E 01	-1.183571E 02	-1.180000E 02
4.410000E-01	9.061436E 01	9.062053E 01	-1.315053E 02	-1.310000E 02
7.219999E-01	8.392397E 01	8.301300E 01	-1.453849E 02	-1.450000E 02
1.179999E 00	7.634747E 01	7.631154E 01	-1.570761E 02	-1.570000E 02
1.940000E 00	6.865238E 01	6.809666E 01	-1.654027E 02	-1.650000E 02
3.179999E 00	5.957620E 01	5.957272E 01	-1.702902E 02	-1.700000E 02
5.219999E 00	5.089742E 01	5.090614E 01	-1.720669E 02	-1.720000E 02
8.559999E 00	4.225528E 01	4.227885E 01	-1.708802E 02	-1.710000E 02
1.400000E 01	3.382295E 01	3.378616E 01	-1.669525E 02	-1.670000E 02
2.300000E 01	2.565051E 01	2.566400E 01	-1.602865E 02	-1.600000E 02
3.770000E 01	1.813543E 01	1.813747E 01	-1.514848E 02	-1.510000E 02
6.179999E 01	1.152750E 01	1.152682E 01	-1.430150E 02	-1.430000E 02
1.010000E 02	5.803600E 00	5.800692E 00	-1.391436E 02	-1.390000E 02
1.660000E 02	3.495603E-01	3.406665E-01	-1.434512E 02	-1.430000E 02
2.720000E 02	-5.513581E 00	-5.514483E 00	-1.574460E 02	-1.580000E 02
4.460000E 02	-1.272772E 01	-1.272776E 01	-1.797863E 02	-1.800000E 02
7.320000E 02	-2.192239E 01	-2.171650E 01	-2.051648E 02	-2.050000E 02
1.200000E 03	-3.287631E 01	-3.287946E 01	-2.270911E 02	-2.270000E 02

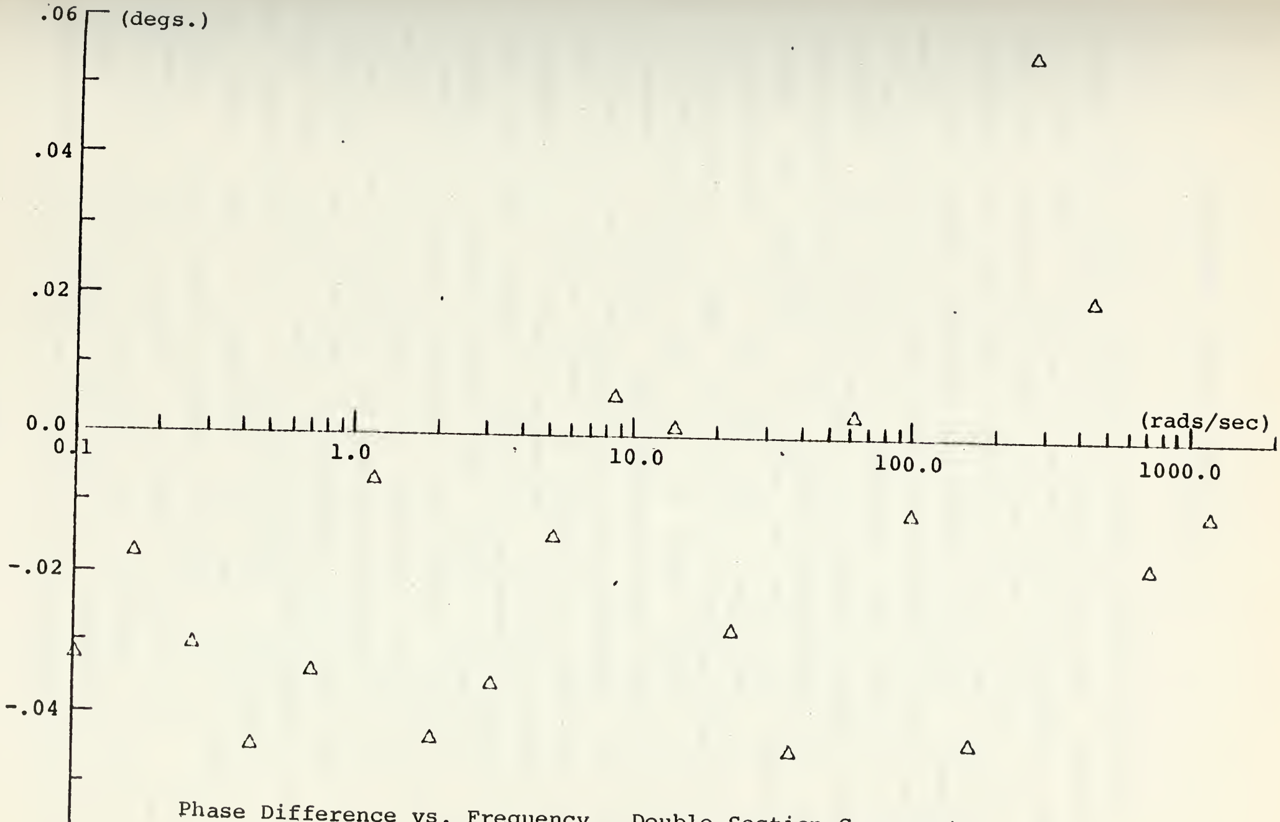
THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE

Figure IV-6N



Magnitude Difference vs. Frequency, Double Section Compensator

Figure IV-60



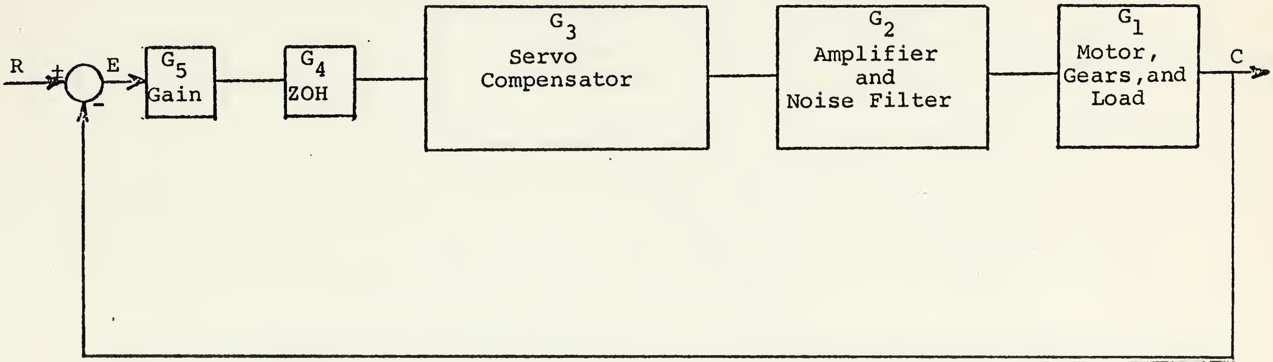
Phase Difference vs. Frequency, Double Section Compensator

Figure IV-6P

5. Compensator Design, Example 5.

This final compensator design example represents a problem of a slightly different nature than those presented to this point. The servo system under consideration here has a zero order hold present in the error channel and in addition, due to the large coefficient values that result in the plant transfer function, the problem requires frequency scaling in order to insure that the numerical limitations of the particular computer that was being used would not be exceeded. The original single loop system is shown in the block diagram of figure IV-7. Series compensation of this system was desired in order to meet the following specifications; 1.) the open loop magnitude at 377 radians/second ≥ 30db, 2.) the gain crossover frequency ≥ 1885 radians/second, and 3.) the phase margin at gain crossover ≥ 45 degrees. These specifications were to be met using only a series compensator and the parameters of that part of the system shown in figure IV-7, other than the compensator, was to remain unchanged. An initial attempt at using the CALICO program to design a compensator resulted in a warning of excessive number size during computation and recommended frequency scaling of the problem. Thus a scale factor of 1000 was selected as a convenient value and the entire system, including the sampling frequency of the zero order hold, was scaled down in frequency by the value of this scale factor. Using Laplace transform notation, this was accomplished by a straight-forward substitution of s = 1000S, where S represents the new scaled frequency values. A block diagram showing the values of the system parameters after scaling is given in figure IV-7A. When this type of scaling is performed the resultant numerical values of the program need only be multiplied by the appropriate constant value in order to be corrected for use in the unscaled system.

Originally, work on this particular system indicated that a fifth order compensator could provide the necessary phase margin and gain crossover frequency, but that the 30 db specification at 377 radians/second was not satisfied. Thus, with this information as a starting point the magnitude profile of the system with the fifth order compensator in the loop was adjusted at the low frequency end to satisfy the 30 db requirement at the scaled frequency value of 0.377 radians/second. The phase profile was left unchanged. The results of the program can be seen in figures IV-7E through IV-7I. As can be seen from the graphical output the desired specifications have been satisfied by the resulting compensated system.



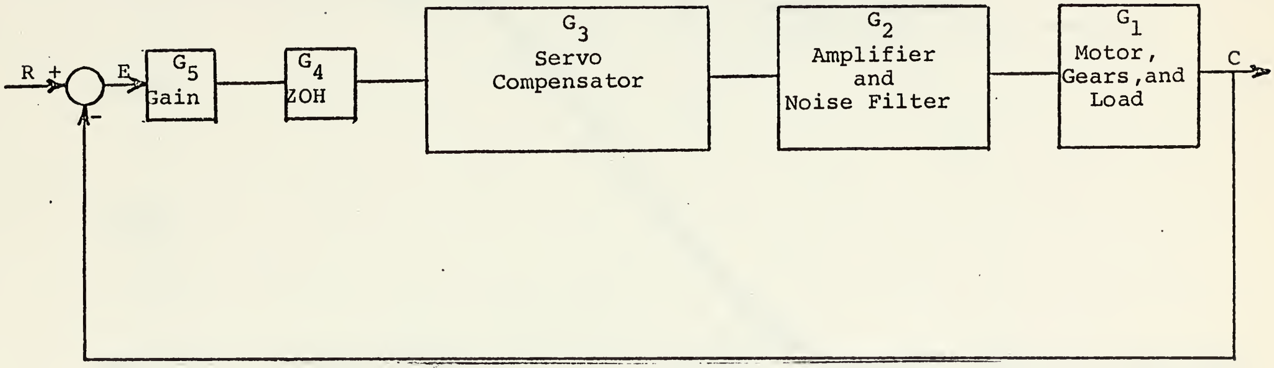
$$G_1 = \frac{558.6}{s(.00125s + 1)}$$

ZOH $T = \frac{1}{1560}$ seconds

$$G_2 = \frac{4.51638\times10^{12}}{(s^2 + 5.0265\times10^3 s + 1.01064\times10^8)}$$

$G_5 = 0.5$

Unscaled System Block Diagram
Figure IV-7



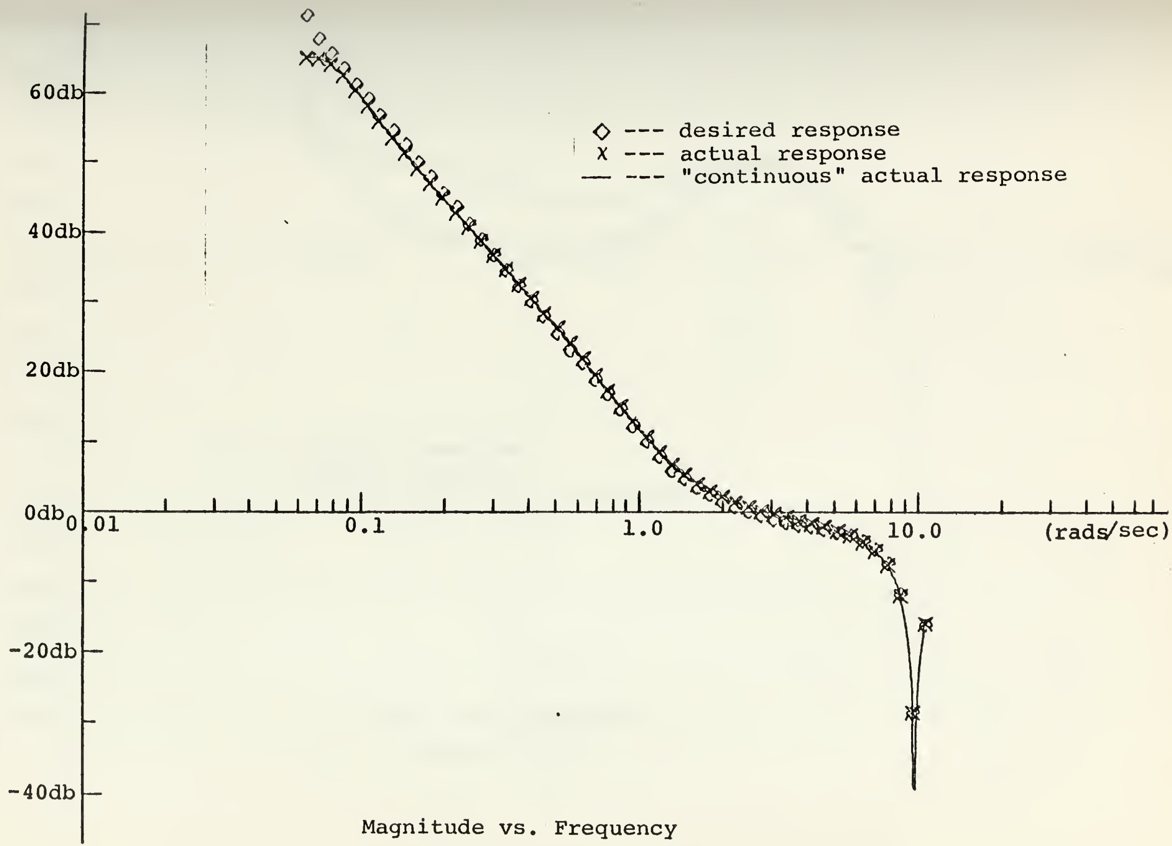
$$G_1 = \frac{4.4688 \times 10^{-1}}{S(S + 8.0 \times 10^{-4})}$$

ZOH $T = \frac{1}{1.560}$ seconds

$$G_2 = \frac{8.085179 \times 10^{3}}{S^2 + 5.02654S + 101.0647}$$

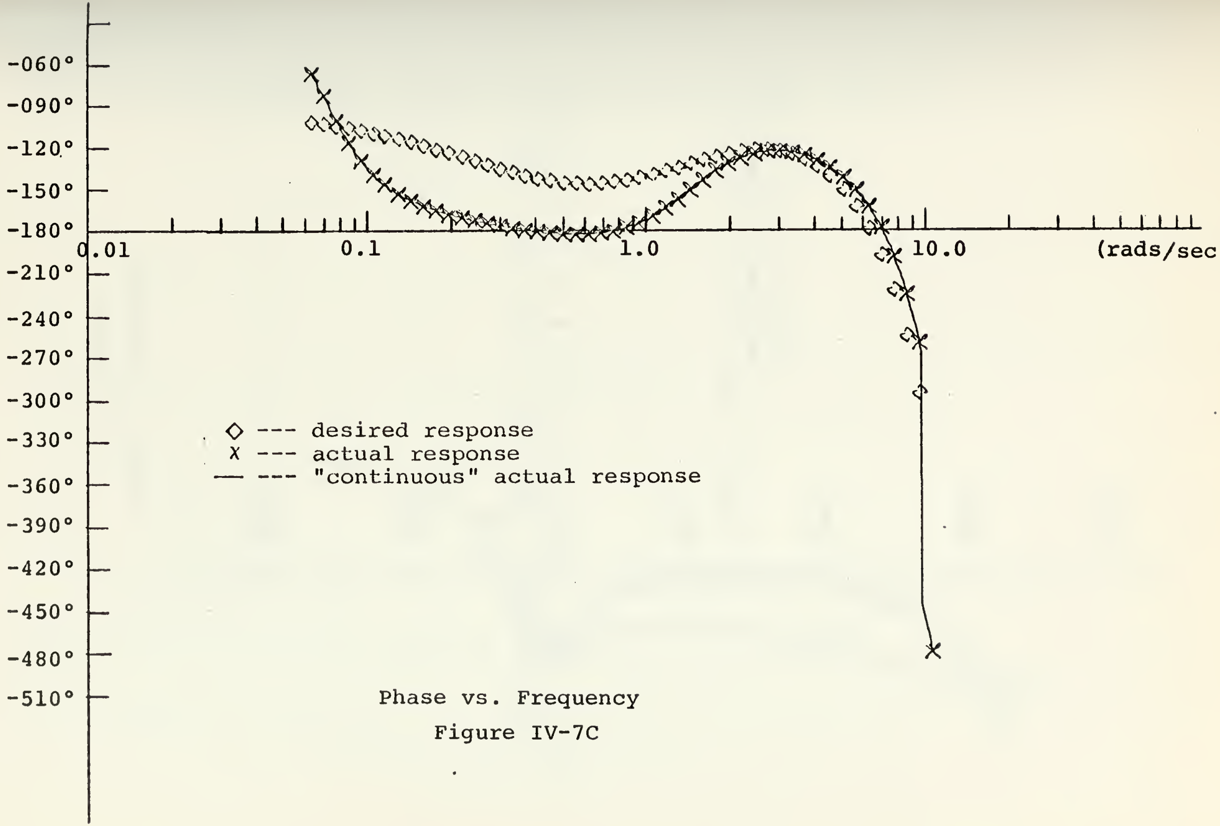
$G_5 = 0.5$

Scaled System Block Diagram
Figure IV-7A



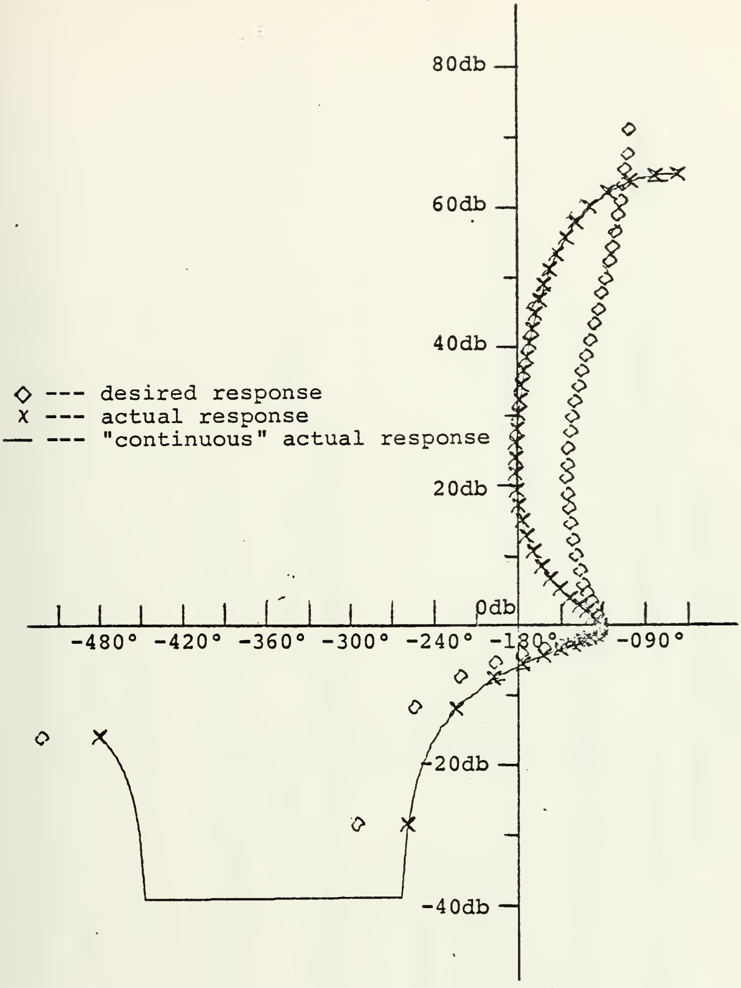
Magnitude vs. Frequency

Figure IV-7B



Phase vs. Frequency

Figure IV-7C



Magnitude vs. Phase

Figure IV-7D

```
TITLE --- COMPENSATOR OPTIMIZATION TEST RUN 9 SCALED 1000S

UNCOMPENSATED TRANSFER FUNCTION GAIN =  8.523400E 04

    UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

 1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

 0.0          3.085178E 01    1.050360E 02    5.826546E 00    1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:     REAL PART          IMAGINARY PART

       0.0                 0.0
      -8.000000E-01        0.0
      -2.513273E 00       -9.133870E 00
      -2.513273E 00        9.133870E 00

    COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

    COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

 1.000000E 00    5.000000E 00    1.000000E 01    1.000000E 01    5.000000E 00
 1.000000E 00
```

Computer Numerical Output
Figure IV-7E

COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS ARE:

REAL PART	IMAGINARY PART
-1.000000E 00	0.0
-1.000000E 00	0.0
-1.000000E 00	0.0
-1.000000E 00	0.0
-1.000000E 00	0.0

COMPENSATOR TRANSFER FUNCTION DENOMINATOR COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 05   5.000000E 04   1.000000E 04   1.000000E 03   5.000000E 01
1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS ARE:

REAL PART	IMAGINARY PART
-1.000000E 01	0.0
-1.000000E 01	0.0
-1.000000E 01	0.0
-1.000000E 01	0.0
-1.000000E 01	0.0

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM-PHASE TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

THE SYSTEM HAS A ZERO ORDER HOLD IN THE LOOP.

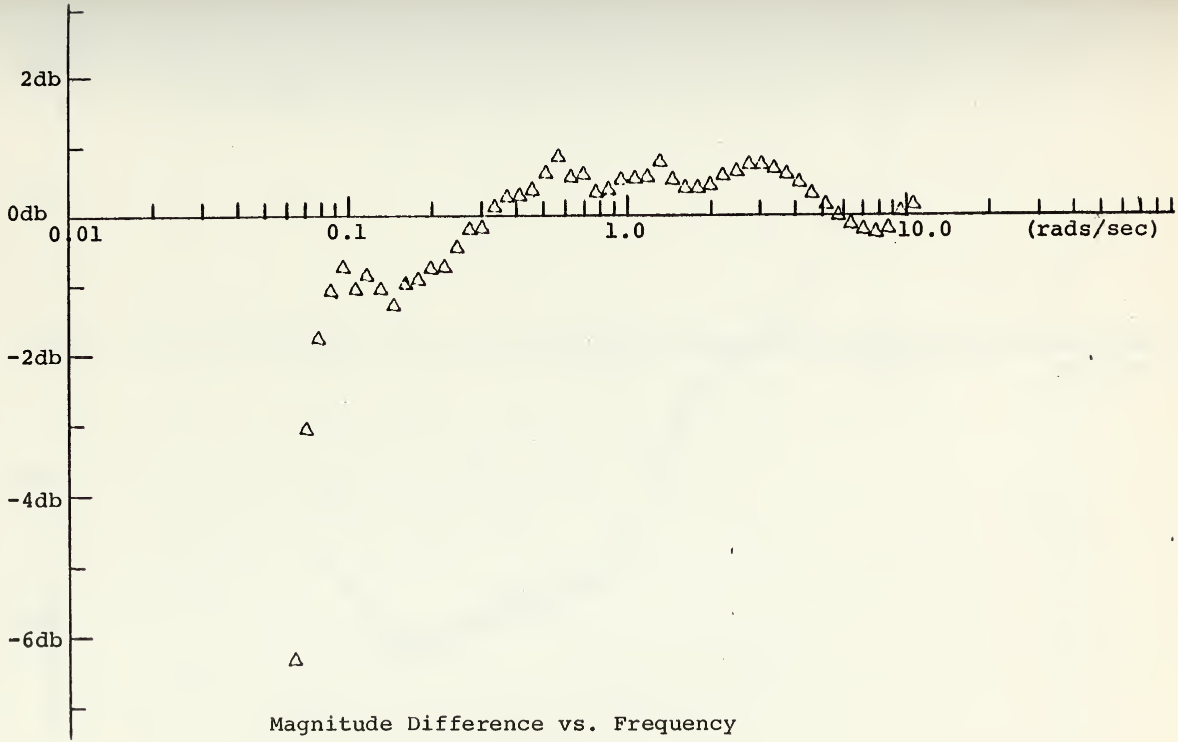
THE SAMPLING PERIOD T = 6.410200E-01 (SECS)

THE SAMPLING FREQUENCY WS = 9.801853E 00 (RADIANS)

Computer Numerical Output
Figure IV-7F

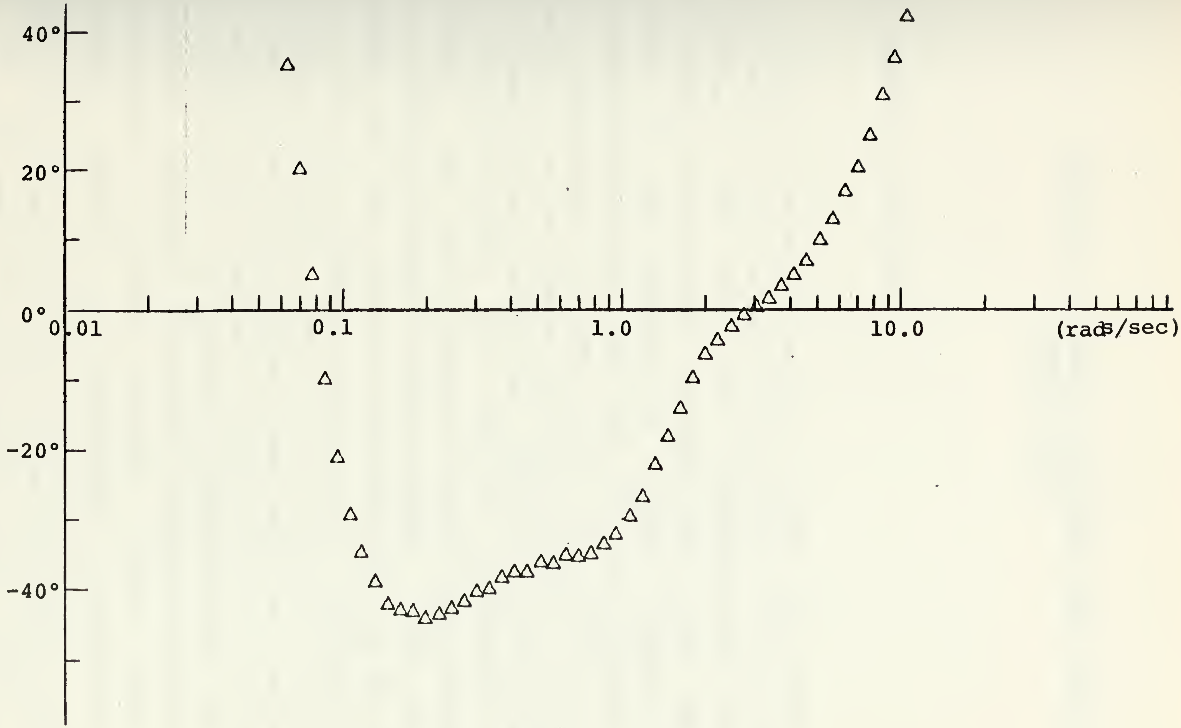
```
THE MINIMUM COST FUNCTION VALUE =  2.973144E 00

THE ERROR RETURN CODE FROM BOXPLX =    2
------------------------------------------------------------------------
------------------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   2.144453E-06    3.712168E 00    4.494462E 00    3.196418E 00    8.601346E-01
  6.835622E-02

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
  ROOTS ARE:       REAL PART              IMAGINARY PART

               -7.228979E 00          0.0

               -3.759962E 00          0.0

               -8.090903E-01          1.164690E 00

               -8.090903E-01         -1.164690E 00

               -5.776758E-07          0.0
------------------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   3.275204E 00    3.180759E 01    6.194180E 02    3.866127E 01    3.137374E-02
  1.956992E-03

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
  ROOTS ARE:       REAL PART              IMAGINARY PART

               -2.559161E-02          6.818771E-02

               -2.559161E-02         -6.818771E-02

               -4.581697E-03          1.405507E-02

               -4.581697E-03         -1.405507E 02

               -1.597125E 01          0.0
------------------------------------------------------------------------

      OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =  3.492921E 01
```

Computer Numerical Output
Figure IV-7G



Magnitude Difference vs. Frequency

Figure IV-7H



Phase Difference vs. Frequency

Figure IV-7I

V. SYNTHESIS OF TRANSFER FUNCTIONS FROM FREQUENCY RESPONSE DATA

A. GENERAL DISCUSSION

In the process of running various test problems to exercise and verify the algorithm presented in the preceding chapters, it was found that in addition to its use as a compensator design program, the basic method could also be applied to the synthesis of transfer functions from frequency response data. That is, if the magnitude and phase measurements of some system are obtained it is possible to use the program (CALICO) in order to obtain an approximate linear analytical expression for the transfer function of the system from which the measurements were obtained. This type of problem is quite often encountered in the modeling process of dynamic systems. That is, the engineer may have measured input-output data of the system in the form of a frequency response and is confronted with obtaining a transfer function which will model this system. This may be the result of the process being too complex to be analyzed and the pertinent equations written on the basis of physical laws or the governing physical laws may not be completely understood in the case of systems involving new technologies or reasearch efforts. Quite often the procedure followed in synthesizing or determining an approximate linear transfer function from frequency response data is to construct a Bode plot for the measured data and to fit the measured characteristic with straight line asymptotic approximations where possible and quadratic factors where sharp peaks in the measured data occur. As in compensator design by classical frequency domain techniques, this procedure for synthesizing transfer functions can involve a considerable amount of time consuming and tedious "cut and try" on the part of the engineer.

The algorithm previously presented is so structured that the use of the program for this type of transfer function synthesis is possible with no modifications to the coding itself. Since the algorithm was not originally developed for the specific purpose of transfer function synthesis there are several precautions that must be taken in order to insure that the transfer function obtained does indeed adequately represent the system from which the measurements were obtained. In particular when using the program in order to synthesize a transfer function, a separate simulation program for time domain response should be used to verify the transfer function obtained. This procedure of checking the time domain response should be followed for any transfer function obtained from frequency response data. The primary concern here is that it is possible to accurately represent the frequency response of some systems, over a finite frequency range, by a transfer function of higher order than the true transfer function, but in some cases the dominant response to standard time domain inputs will be considerably different than that of the actual system. Also, while there is a direct relationship between the time domain and frequency domain responses through the inverse Fourier transform integral, this integration, strictly speaking, is taken over the entire frequency range from $-\infty$ to $+\infty$. In practical applications it is obviously necessary to limit the range of frequencies to be measured to some finite region. If, however, this region is not sufficiently large for a particular system the resulting transfer function obtained from the frequency response data may not accurately describe the system. If this is the case, it should become readily apparent in a time domain simulation of the equation obtained from the frequency response data. The possible pitfalls in the use of this algorithm to accomplish transfer function synthesis, such as the one just mentioned, will be discussed later when some example problems will also be presented.

A linear time invariant system may be expressed as a ratio of two frequency dependent polynomials of the form

$$H(s) = \frac{a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0}{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_1 s + b_0} = \frac{N(s)}{D(s)}$$

where s is the frequency dependent complex variable of the Laplace transform. Thus the value of this transfer function is dependent upon the value of s and the unknown coefficients of the numerator and denominator. Note that this is analogous to the situation that occurs in the design of a cascade compensator for a unity feedback system with the plant represented by a unity gain block. Thus for the purposes of using the program, if we imagine the measured frequency response data to represent some desired system's open loop frequency response profile, and represent the plant of that system by a constant gain of unity, then the transfer function H(s) to be synthesized may be considered the same as a cascade compensator in the system. Thus by representing the problem in this manner the CALICO program may be used to approximate the transfer function of the actual system by varying the parameters of this pseudo compensator (actually the transfer function of the system) in order to minimize the cost function. The cost function, here represents the error between the measured frequency response data (entered as the desired profile) and the frequency response simulation calculated by the program.

Thus in effect the program minimizes the error between the transfer function approximating the system and the measured frequency response of the system at the discrete frequency points which form the profile input into the program.

As is the case when using the program to design compensators the user must assume a specific order for the numerator and denominator representing, in this case, the transfer function to be synthesized. While unfortunately the program will not vary the order of the numerator and denominator polynomials in order to obtain the best match to the measured frequency response, the user can employ the measured data in order to make an initial guess, as it were, of the form of the transfer function. After this, one may scrutinize the results from the program, which graphs the measured and simulated responses, to make a decision if the form of the transfer function should be adjusted in order to more closely represent the system. Here also, a time domain simulation might be extremely useful in guiding decisions regarding the particular form or modification that should be tried in order to obtain a "good" model for the system under investigation. The fact that the user selects the specific orders of the transfer function numerator and denominator and the minimization routine varies the parameters to obtain a minimum cost function while leaving the orders unchanged presents the potential of obtaining a low order model of a higher order system. While for linear systems, with known transfer functions, there are numerous analytical techniques available for accomplishing this, the capability exists here of essentially obtaining a reduced order transfer function without a priori knowledge of the higher order transfer function describing the system. The only data required by the program is the frequency response of the higher order system and nothing concerning the parameters of the high order transfer function is required.

Since some transfer functions may contain roots of the numerator polynomial in the right half of the s-plane (nonminimum phase systems) this type of transfer function should not be overlooked when attempting to model a system. To this end a nonminimum phase option may be chosen in the CALICO computer algorithm. When this option is selected the search area for the numerator coefficients is expanded to include both positive and negative values and the implicit constraints requiring left half plane roots for the numerator polynomial are eliminated. Selection of this option does not guarantee that a nonminimum phase transfer function will result from the CALICO program but it does allow this type of transfer function to be considered as a feasible solution of the problem.

B. TRANSFER FUNCTION SYNTHESIS: EXAMPLE PROBLEMS

In this section several example problems are presented to illustrate the use of the CALICO program in determining transfer functions from frequency response data. These examples illustrate the validity of the basic concept of using the minimization algorithm to determine the transfer function, but they are by no means an exhaustive presentation of all possible variations of cost functions and frequency ranges that may be employed in determining a transfer function representation of a system. For example, as will be shown in the example problems, different transfer function representations may be obtained by considering different ranges of the measured frequency response. That is, while the order of the numerator and denominator are fixed by the user, the parameter values returned from the program may vary depending upon the frequency range being considered. This may be desirable if seperate low frequency and high frequency representations are being determined.

Also in using the type one cost function, which includes both phase and magnitude, the cost surface is different than if the magnitude difference alone were used in the cost function. This has an effect on the resulting parameter values, particularly when lower order transfer functions are chosen to represent higher order systems.

The necessary data and the input format employed are almost identical to those used when the program functions in its originally designed compensator design mode. Figure V-1 illustrates the input data deck used for the first synthesis example problem. As can be seen the only significant change is that the plant has been represented by a unity gain transfer function. This has the effect that the resulting "compensator" represents the transfer function modeling the entire system.

EXAMPLE CATA DECK

```
(COLLMN NUMBERS, READ VERTICALLY)
         11111111112222222222333333333344444444445555555555666666666677777777778
12345678901234567890123456789012345678901234567890123456789012345678901234567890
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||


TRANSFER FUNCTION SYNTHESIS EXAMP 1 20PT
1.0
P00
P00
1.0
P00
F04
1.0
1.0
1.0
1.0
  0.1       30.0    020   1                     1  1
 0.100E 00 0.135E 00 0.182E 00 0.246E 00 0.332E 00 0.449E 00 0.606E 00 0.818E 00
 0.110E 01 0.149E 01 0.201E 01 0.272E 01 0.367E 01 0.495E 01 0.669E 01 0.903E 01
 0.122E 02 0.165E 02 0.222E 02 0.300E 02
 0.125E 01 0.125E 01 0.123E 01 0.121E 01 0.118E 01 0.112E 01 0.102E 01 0.888E 00
 0.719E 00 0.533E 00 0.359E 00 0.219E 00 0.120E 00 0.594E-01 0.266E-01 0.108E-01
 0.401E-02 0.139E-02 0.458E-03 0.145E-03
-0.103E 02-0.139E 02-0.187E 02-0.251E 02-0.335E 02-0.445E 02-0.584E 02-0.755E 02
-0.955E 02-0.118E 03-0.142E 03-0.167E 03-0.193E 03-0.218E 03-0.242E 03-0.264E 03
-0.284E 03-0.301E 03-0.315E 03-0.326E 03
10000


||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
         11111111112222222222333333333344444444445555555555666666666677777777778
12345678901234567890123456789012345678901234567890123456789012345678901234567890
(COLUMN NUMBERS, READ VERTICALLY)
```

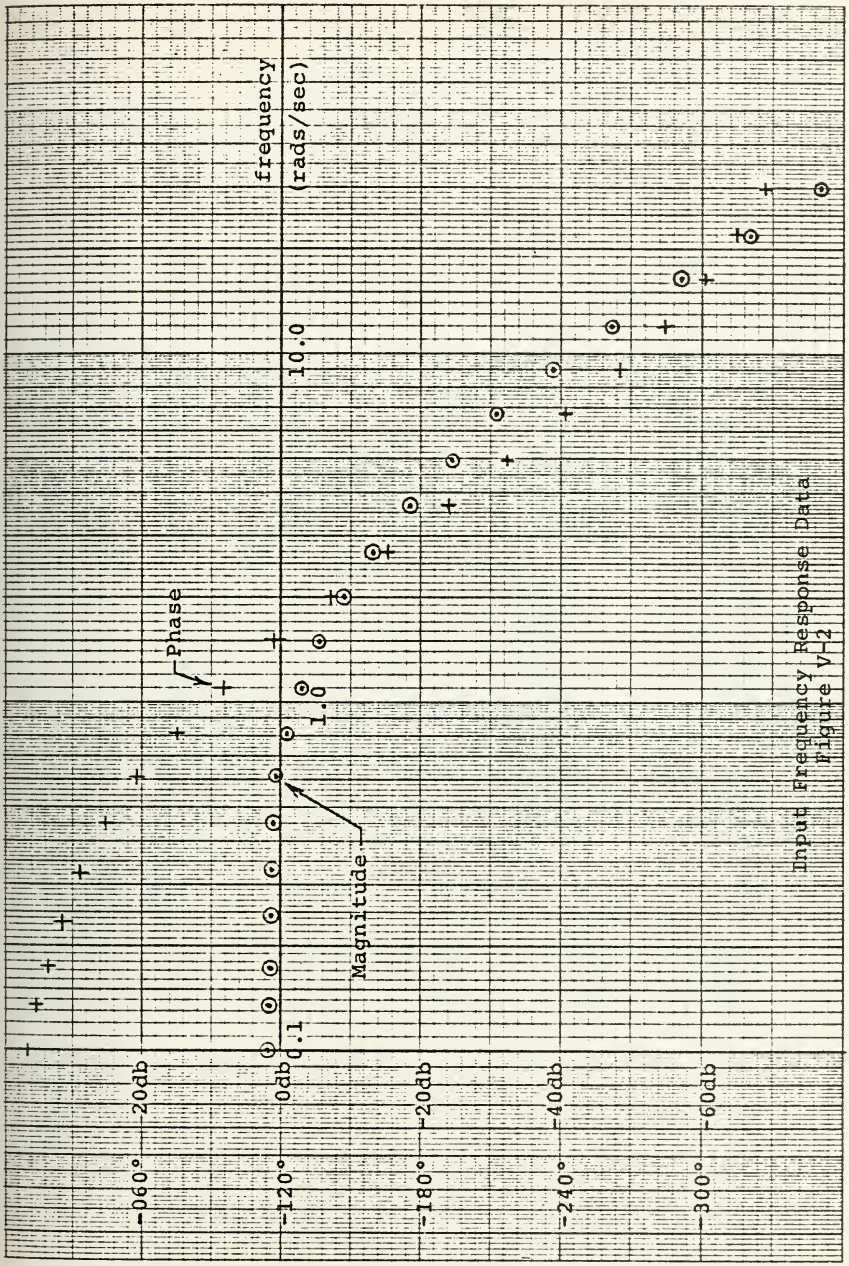
Transfer Function Synthesis, Data Deck
Figure V-1

1. Synthesis Example 1.

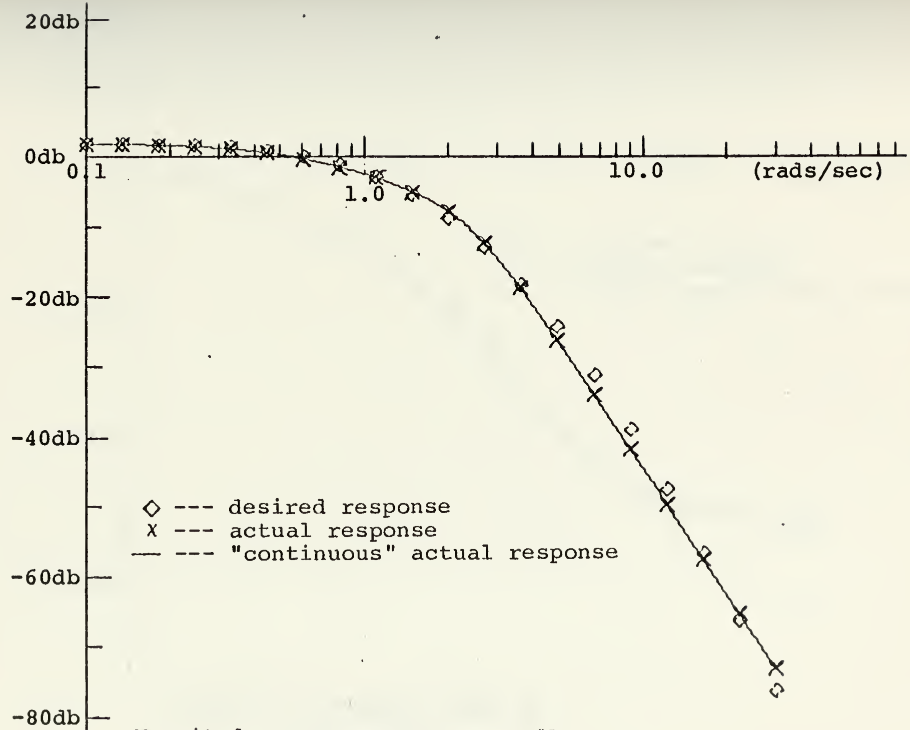
The frequency response data, shown graphically in figure V-2, was generated from a known transfer function using a computer frequency response program. As can be seen, twenty magnitude and phase values were selected at discrete frequencies over the range from 0.1 radians/second to 30 radians/second. Examination of the frequency response data suggests a transfer function of the form of a constant gain over a polynomial of some unknown order. This form will yield a constant magnitude at low frequencies and a decreasing magnitude as the frequency increases. Closer examination of the magnitude curve in the range between 2 and 20 radians/second indicates a slope of approximately 60 db/decade, implying a polynomial of at least third order in the denominator. The phase curve extends from 10 degrees to 330 degrees over the range of frequencies considered. This indicates that the denominator may be of fourth order or higher. As a first guess a third order denominator is assumed. The results from the CALICO program are shown in figure V-2A through V-2H. The initial guess for the denominator roots and the system gain were chosen as 1, as can be seen in figure V-2F. Figures V-2A and V-2D show that the resulting magnitude curve approximates the measured response well at the lower frequencies with a slight deviation begining to appear at the higher frequencies. The phase plot of figure V-2B and the plot of the difference between the actual phase and desired phase shown in figure V-2E indicate a significant amount of deviation from the measured values in the higher frequency region. This suggests that another pole in the transfer function is necessary to more accurately model the system. Thus, the program was again executed, this time using a fourth order denominator. The results illustrated in figures V-2I through V-2F indicate that this second form of the transfer function more accurately represents the system. Since the original transfer function is known in this case, an opportunity exists to evaluate the performance of the program in terms of the accuracy of the transfer function returned after minimization of the cost function based on the input frequency response data. This of course would generally not be the case when trying to determine the transfer function of an actual system that was to be modeled. The parameter values returned from the CALICO program shown in figure V-20 may be compared with the original transfer function given below.

$$T(s) = \frac{126.0}{(s + 1)(s + 2)(s + 5)(s + 10)}$$

As can be seen, the values of the transfer function roots returned from the program for the fourth order denominator are in agreement with the values of the original transfer function used to generate the frequency response data. If for the system under consideration only the magnitude response were of interest, the transfer function consisting of a constant gain over a third order denominator might be of sufficient accuracy to model the system. This judgement, of course, rests with the user who is familiar with the intended use of the transfer function modeling the system.

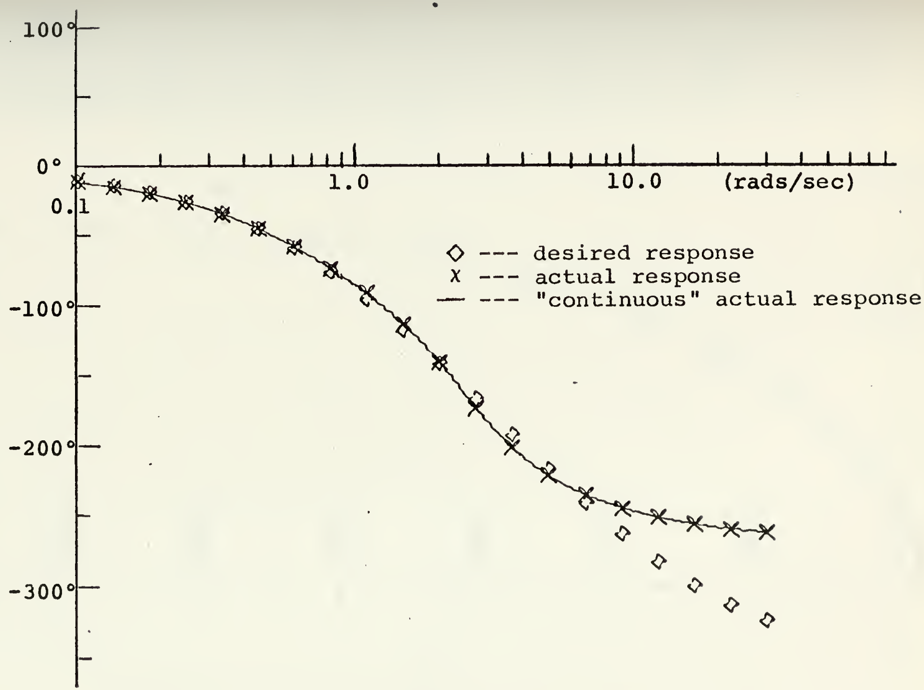


Input Frequency Response Data
Figure V-2



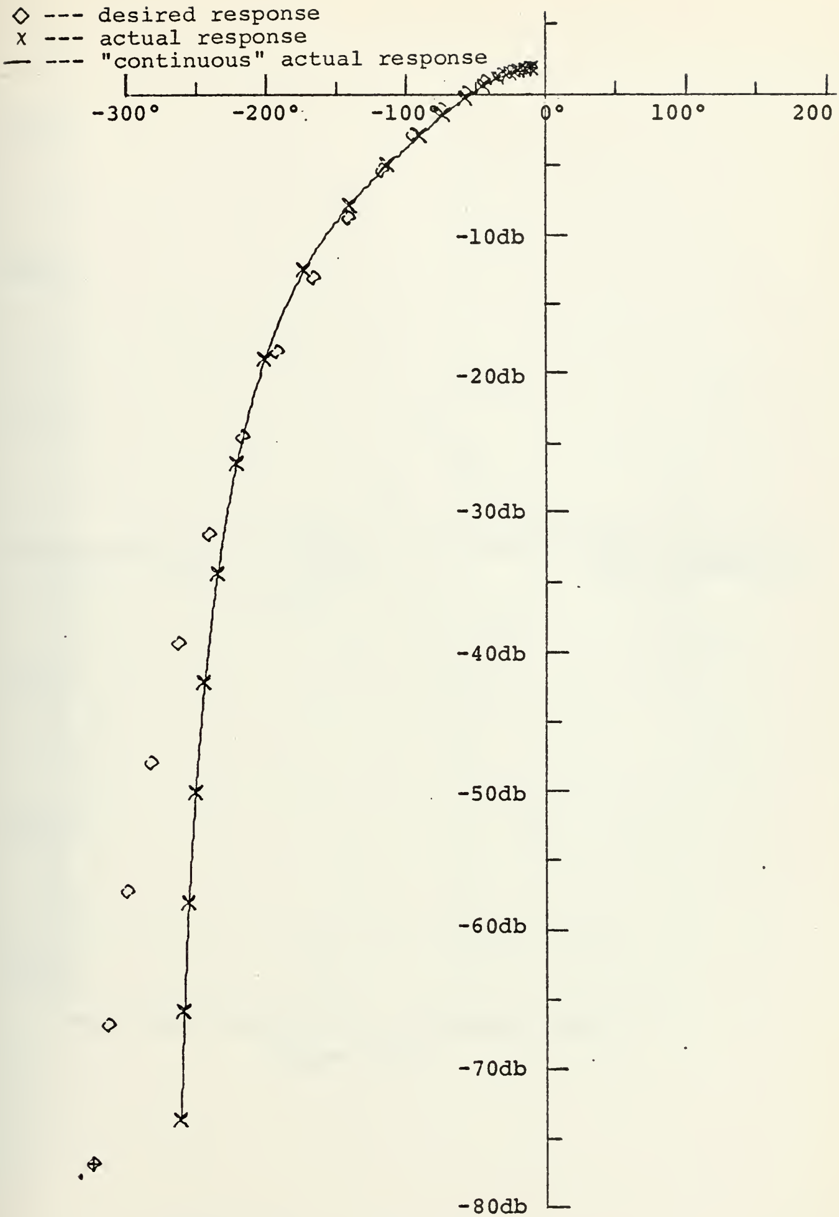
Magnitude vs. Frequency, Run #1

Figure V-2A

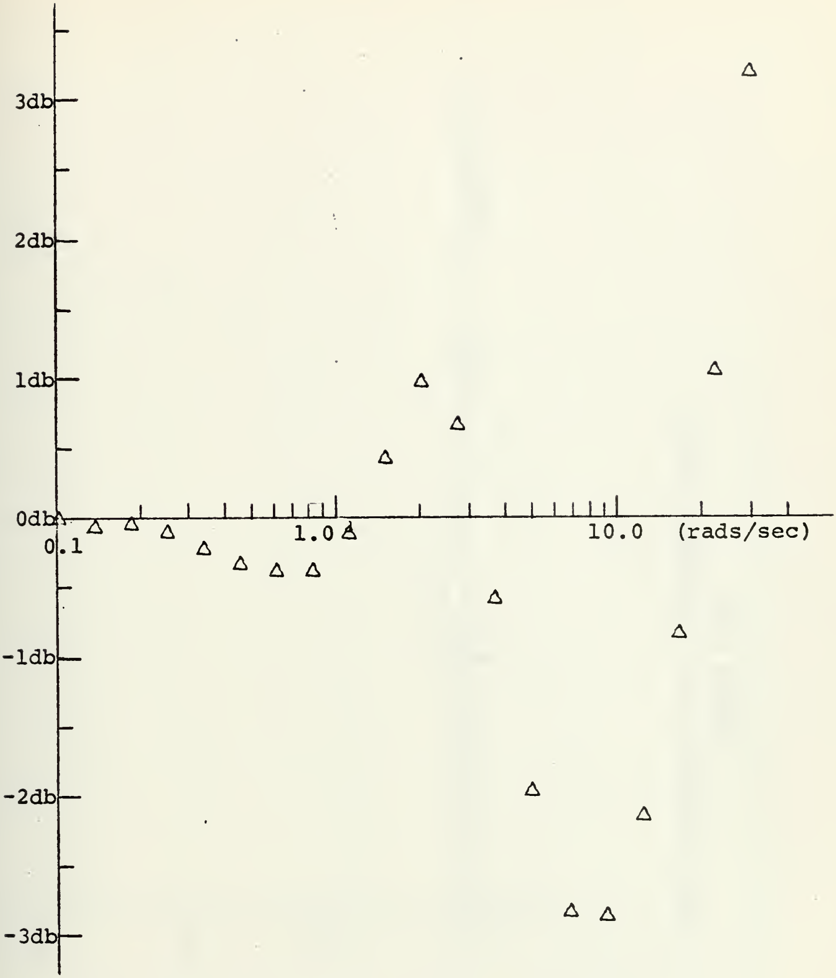


Phase vs. Frequency, Run #1

Figure V-2B

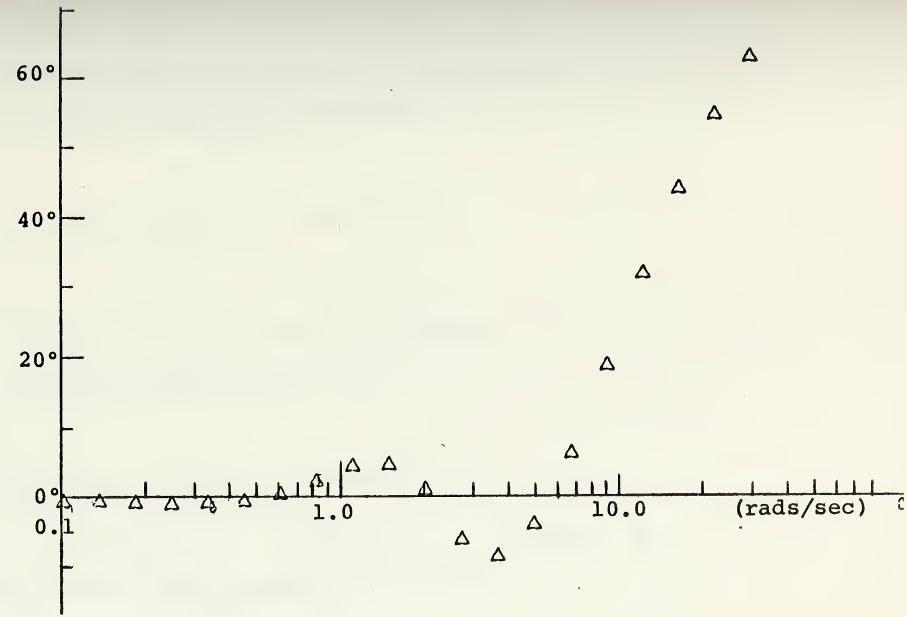


Magnitude vs. Phase, Run #1
Figure V-2C



Magnitude Difference vs. Frequency, Run #1

Figure V-2D



Phase Difference vs. Frequency, Run #1

Figure V-2E

```
TITLE --- TRANSFER FUNCTION SYNTHESIS EXAMP 1 2OPT
--------------------------------------------------------------------------

UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00
--------------------------------------------------------------------------

COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

1.000000E 00    3.000000E 00    3.000000E 00    1.000000E 00

COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:      REAL PART          IMAGINARY PART

     -1.000000E 00       0.0

     -1.000000E 00       0.0

     -1.000000E 00       0.0
--------------------------------------------------------------------------
--------------------------------------------------------------------------

THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION
```

Computer Numerical Output, Run #1

Figure V-2F

```
THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE   1

THE MINIMUM COST FUNCTION VALUE =  6.255069E-01
THE ERROR RETURN CODE FROM BOXPLX =   2

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   2.005749E 05

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   1.590744E 05    3.000009E 05    1.352915E 05    3.551281E 04

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 ROOTS ARE:      REAL PART          IMAGINARY PART

            -1.544968E 00       1.958797E 00
            -1.544968E 00      -1.958797E 00
            -7.197119E-01       0.0

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =  5.647960E 00
```

Computer Numerical Output, Run #1

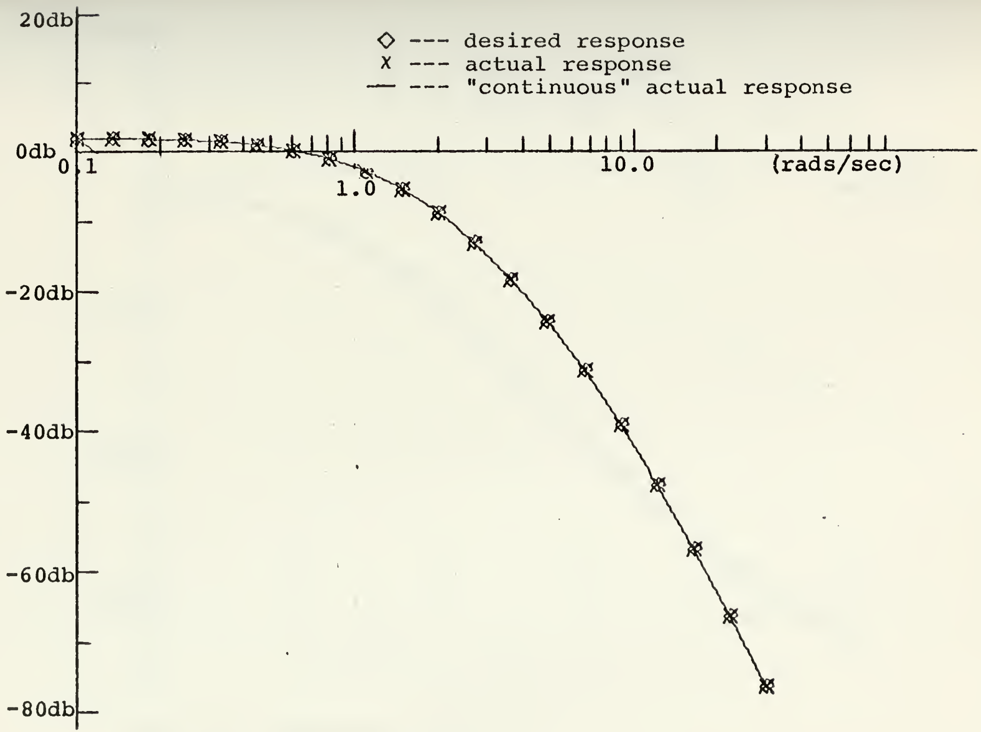
Figure V-2G

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999996E-02	1.933724E 00	1.938200E 00	-1.075708E 01	-1.030000E 01
1.350000E-01	1.869240E 00	1.938200E 00	-1.446946E 01	-1.390000E 01
1.820000E-01	1.754977E 00	1.798098E 00	-1.938194E 01	-1.870000E 01
2.460000E-01	1.552960E 00	1.655700E 00	-2.590143E 01	-2.509999E 01
3.320000E-01	1.210264E 00	1.437634E 00	-3.439147E 01	-3.350000E 01
4.489999E-01	6.472952E-01	9.843595E-01	-4.453173E 01	-4.450000E 01
6.059999E-01	-2.082647E-01	1.719992E-01	-5.782793E 01	-5.839999E 01
8.180000E-01	-1.417542E 00	-1.031741E 00	-7.312437E 01	-7.550000E 01
1.099999E 00	-2.983166E 00	-2.865421E 00	-9.093784E 01	-9.550000E 01
1.490000E 00	-5.045173E 00	-5.465456E 00	-1.132075E 02	-1.180000E 02
2.009999E 00	-7.919121E 00	-8.898106E 00	-1.409275E 02	-1.420000E 02
2.719999E 00	-1.252182E 01	-1.319112E 01	-1.731349E 02	-1.670000E 02
3.669999E 00	-1.899828E 01	-1.841637E 01	-2.014789E 02	-1.930000E 02
4.950000E 00	-2.648900E 01	-2.452426E 01	-2.218055E 02	-2.180000E 02
6.690000E 00	-3.433565E 01	-3.150235E 01	-2.356471E 02	-2.420000E 02
9.030000E 00	-4.219980E 01	-3.933153E 01	-2.451152E 02	-2.640000E 02
1.220000E 01	-5.008112E 01	-4.793710E 01	-2.518177E 02	-2.840000E 02
1.650000E 01	-5.797523E 01	-5.713969E 01	-2.566531E 02	-3.010000E 02
2.220000E 01	-6.572296E 01	-6.678268E 01	-2.601191E 02	-3.150000E 02
3.000000E 01	-7.357813E 01	-7.677264E 01	-2.627043E 02	-3.260000E 02

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE

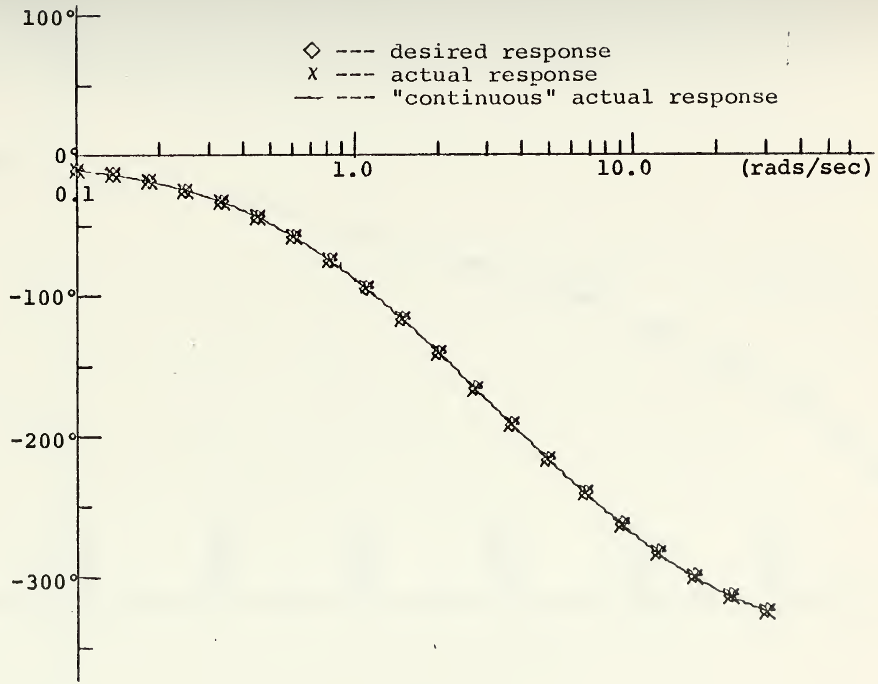
Computer Numerical Output, Run #1

Figure V-2H



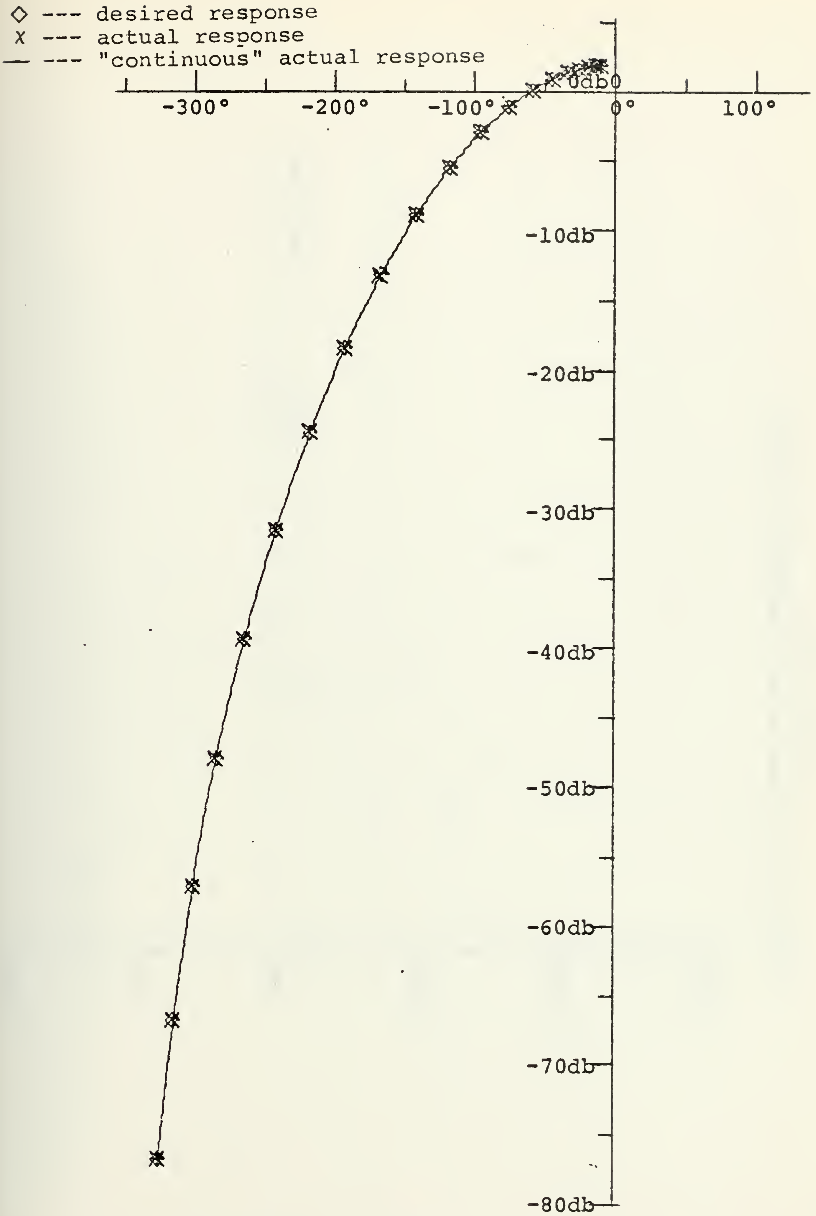
Magnitude vs. Frequency, Run #2

Figure V-2I



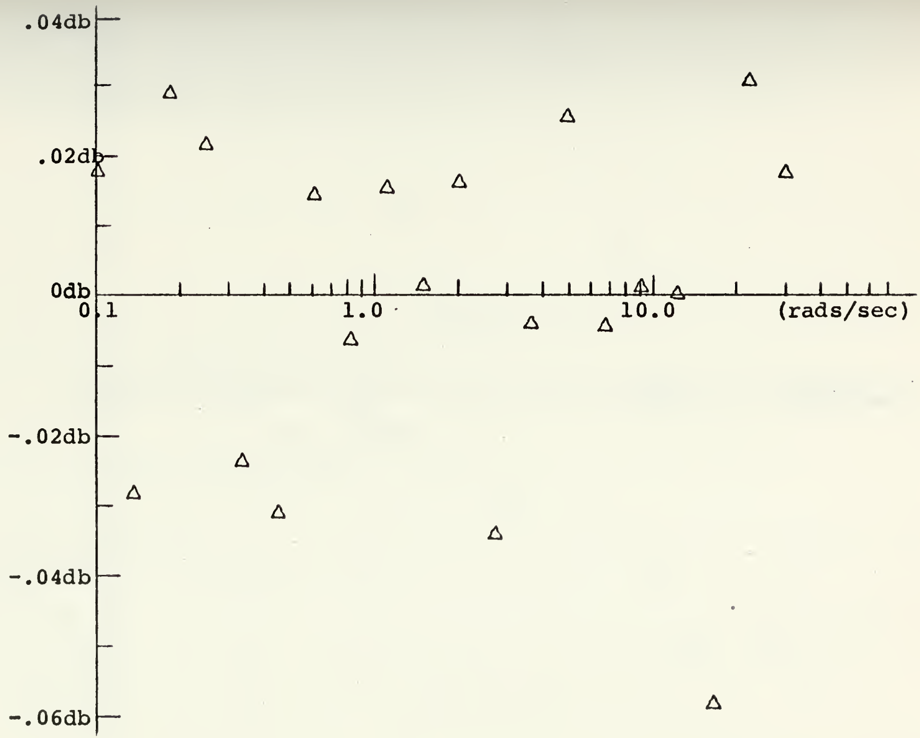
Phase vs. Frequency, Run #2

Figure V-2J



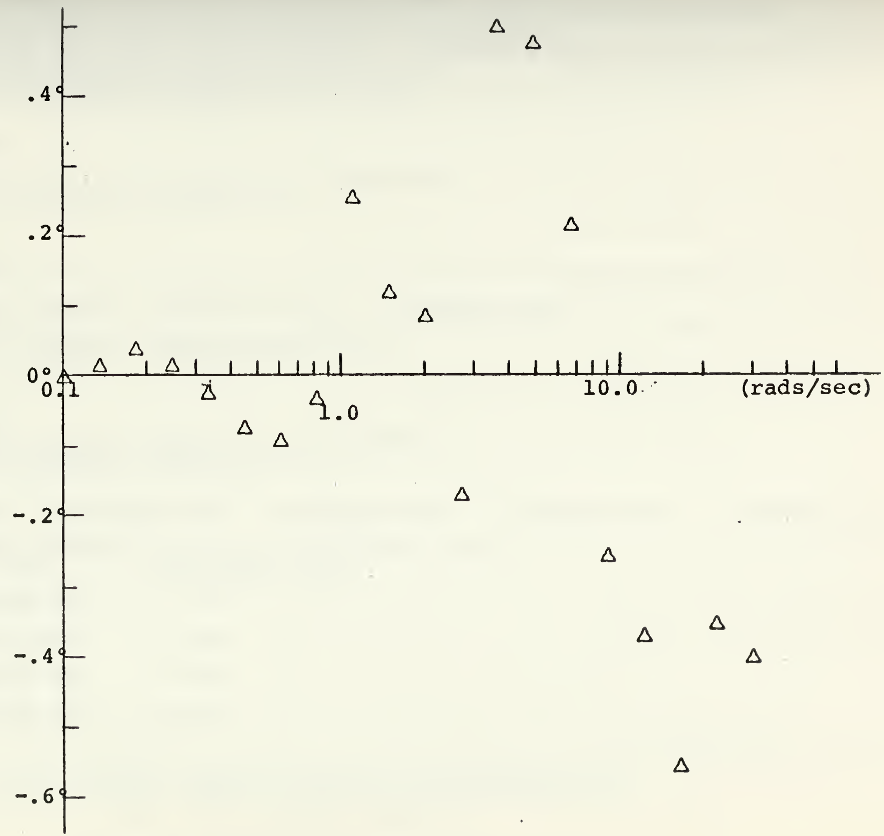
Magnitude vs. Phase, Run #2

Figure V-2K



Magnitude Difference vs. Frequency, Run #2

Figure V-2L



Phase Difference vs. Frequency, Run #2

Figure V-2M

```
TITLE --- TRANSFER FUNCTION SYNTHESIS EXAMP 1 20PT
--------------------------------------------------------------------------------

UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00
--------------------------------------------------------------------------------

    COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

    COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00

    COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00   4.000000E 00   6.000000E 00   4.000000E 00   1.000000E 00

    COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
 ARE:      REAL PART         IMAGINARY PART

         -1.000000E 00        0.0

         -1.000000E 00        0.0

         -1.000000E 00        0.0

         -1.000000E 00        0.0
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------

    THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION
```

Computer Numerical Output, Run #2

Figure V-2N

```
THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

THE COST FUNCTION TO BE USED IS THE TYPE    1

THE MINIMUM COST FUNCTION VALUE =  1.911033E-04
THE ERROR RETURN CODE FROM BOXPLX =  -1

   OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  7.718595E 00

   OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  6.122410E 00   1.103352E 01   5.943789E 00   1.100959E 00   6.122756E-02

   OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 ROOTS ARE:      REAL PART          IMAGINARY PART
              -9.931983E 00        0.0
              -5.054280E 00        0.0
              -1.998310E 00        0.0
              -9.968118E-01        0.0

   OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =  1.260640E 02
```

Computer Numerical Output, Run #2

Figure V-20

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999996E-02	1.955822E 00	1.938200E 00	-1.030384E 01	-1.030000E 01
1.350000E-01	1.909704E 00	1.933200E 00	-1.388632E 01	-1.390000E 01
1.820000E-01	1.826536E 00	1.738098E 00	-1.866319E 01	-1.870000E 01
2.460000E-01	1.677292E 00	1.655700E 00	-2.508612E 01	-2.509999E 01
3.320000E-01	1.413649E 00	1.437634E 00	-3.332655E 01	-3.350000E 01
4.489999E-01	9.531442E-01	9.843595E-01	-4.457687E 01	-4.450000E 01
6.059999E-01	1.862076E-01	1.719992E-01	-5.849570E 01	-5.839999E 01
8.180000E-01	-1.038156E 00	-1.031741E 00	-7.553554E 01	-7.550000E 01
1.099999E 00	-2.850209E 00	-2.865421E 00	-9.524660E 01	-9.550000E 01
1.490000E 00	-5.464069E 00	-5.465456E 00	-1.178840E 02	-1.180000E 02
2.009999E 00	-8.881952E 00	-8.898106E 00	-1.419164 02	-1.420000E 02
2.719999E 00	-1.322533E 01	-1.319112E 01	-1.671723E 02	-1.670000E 02
3.669999E 00	-1.842046E 01	-1.841637E 01	-1.925000E 02	-1.930000E 02
4.950000E 00	-2.449834E 01	-2.452426E 01	-2.175242E 02	-2.180000E 02
6.690000E 00	-3.150691E 01	-3.150235E 01	-2.417867E 02	-2.420000E 02
9.030000E 00	-3.933047E 01	-3.933153E 01	-2.642620E 02	-2.640000E 02
1.220000E 01	-4.793704E 01	-4.793710E 01	-2.843740E 02	-2.840000E 02
1.650000E 01	-5.719827E 01	-5.713969E 01	-3.015610E 02	-3.010000E 02
2.220000E 01	-6.675203E 01	-6.678268E 01	-3.153562E 02	-3.150000E 02
3.000000E 01	-7.675511E 01	-7.677264E 01	-3.264048E 02	-3.260000E 02

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE

Computer Numerical Output, Run #2

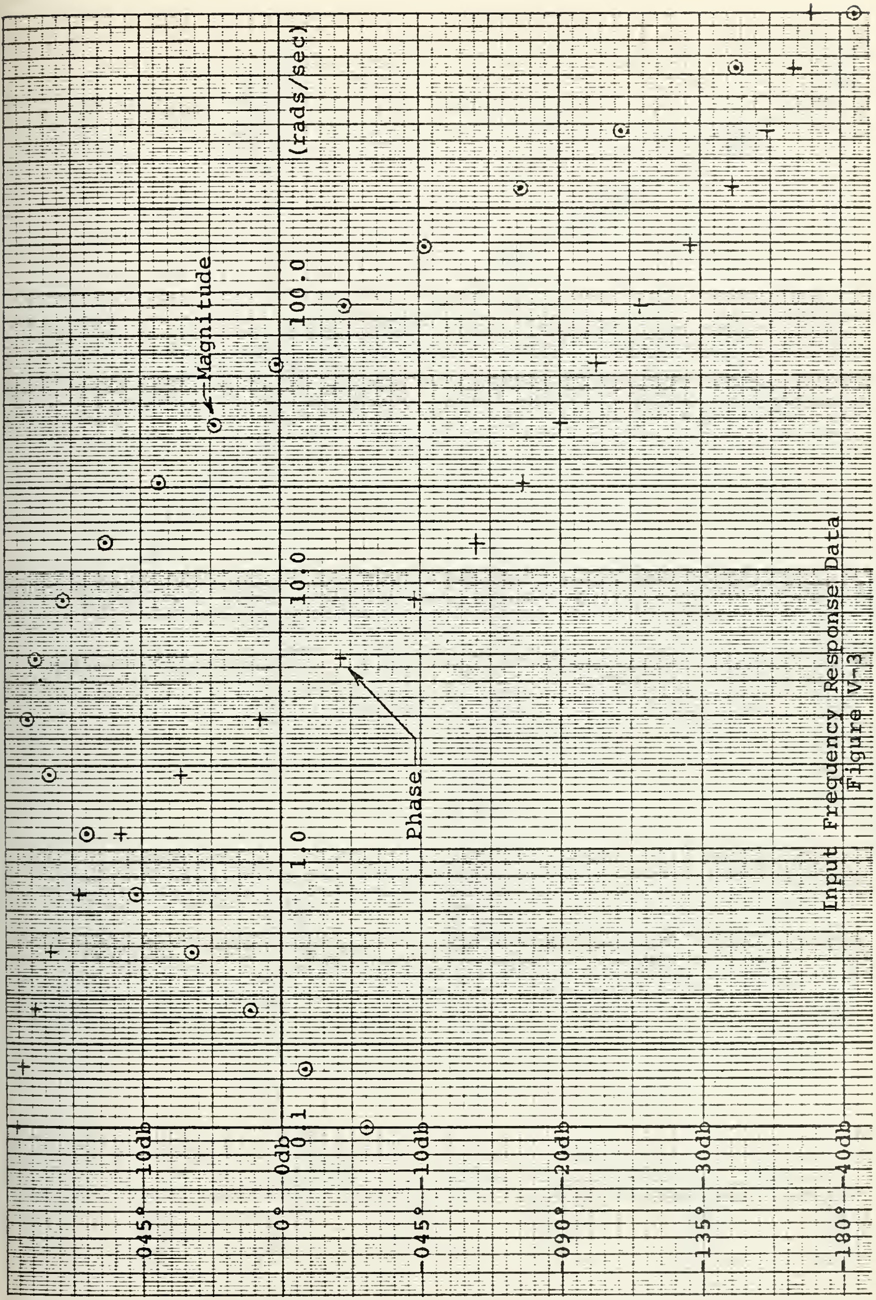
Figure V-2P

2. Synthesis Example 2.

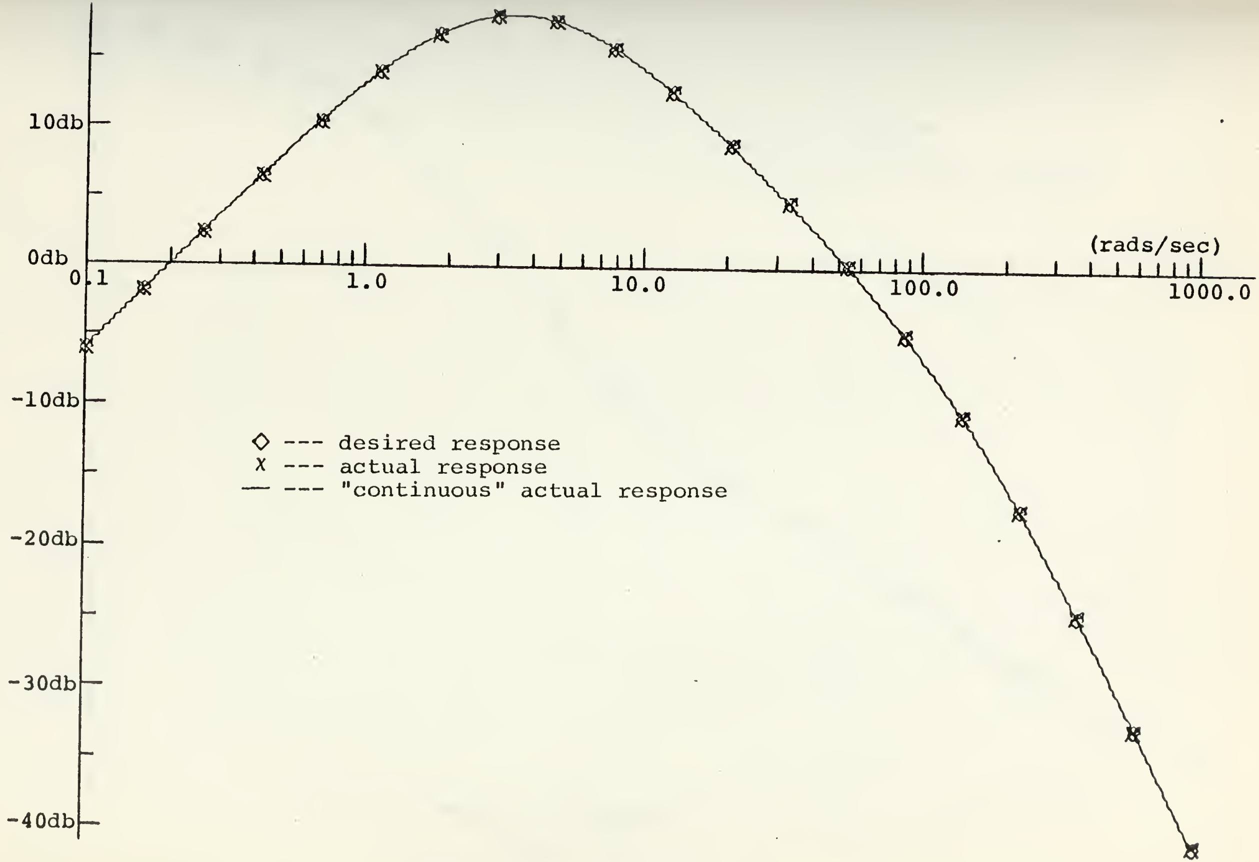
As a second synthesis example, consider the frequency response data shown in figure V-3. Here the magnitude and phase response for a system are shown at twenty distinct frequency values over the range from 0.1 to 1000.0 radians/second. As can be seen from this figure the positive slope of 20 db/decade of the mgnitude curve at the lower frequencies suggests a transfer function with at least one zero. This is also confirmed by the phase response, which at the lower frequencies initially has a value of 90 degrees. At the higher frequencies inspection of the magnitude plot shows a negative slope of approximately 40 db/decade, indicating the dominance of a third order denominator in this area. The phase plot approaching -180 degrees at these higher frequencies also suggests this. Thus, initially a form for the transfer function consisting of a first order numerator and a third order denominator was selected to model the system from which the frequency response data was obtained. The results are shown in figures V-3A through V-3H. As can be seen from the graphical output presented in the figures, this particular form of transfer function appears to match the measured frequency response quite well over the entire range of frequencies considered. The roots and gain of the transfer function returned from the program are shown in figures V-3G and V-3H. As in the previous example it happens that the exact transfer function used to generate the measured data input to the CAIICO program is known. This transfer function is given below in order that the reader may compare the values of the original transfer function parameters with those returned from the program.

$$T(s) = \frac{9000s}{(s + 2.4)(s + 5)(s + 150)}$$

As can be seen from the data presented in figures V-3G and V-3H, the parameter values returned from the program are in close agreement with the original transfer function values. While it is unlikely that they will ever be exactly the same as the original values there is certainly not a significant difference between the two sets of values for most engineering purposes.

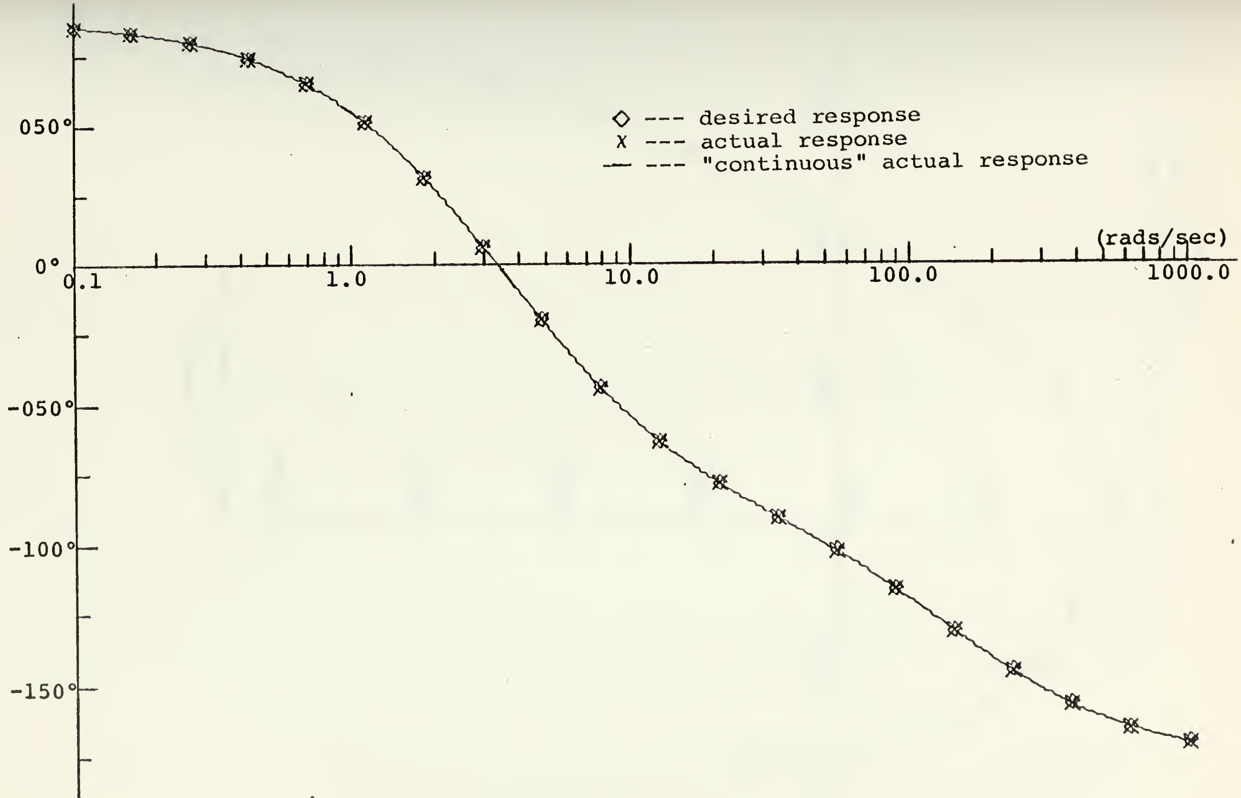


Input Frequency Response Data
Figure V-3



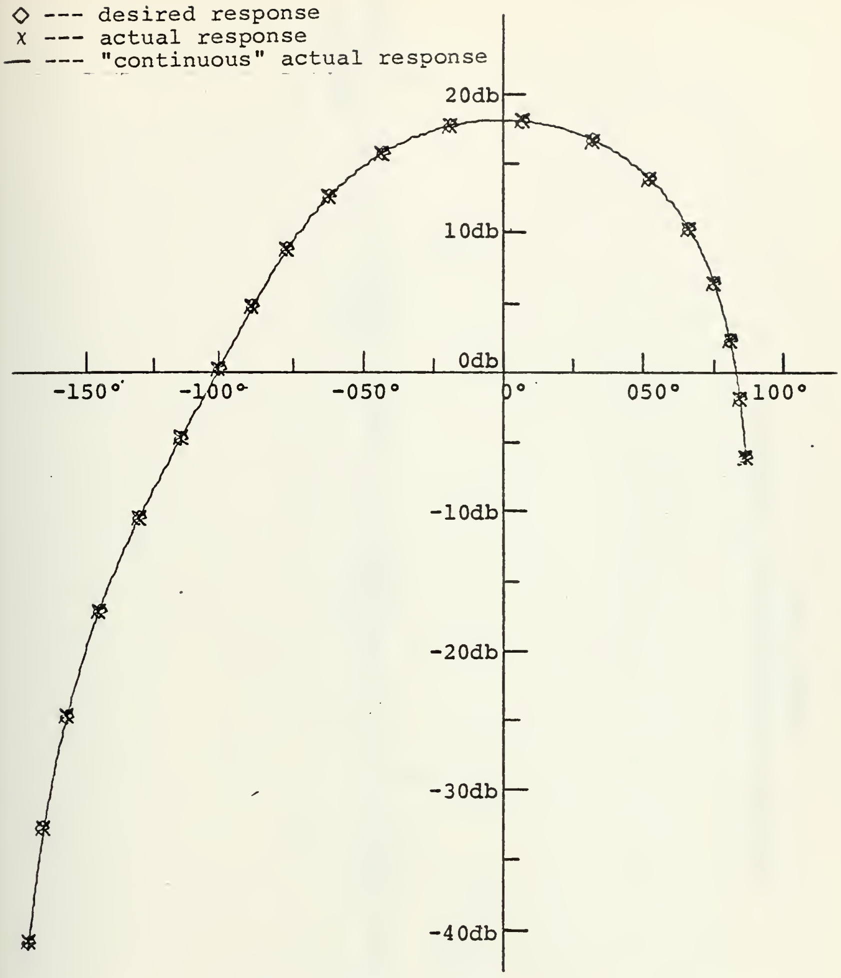
Magnitude vs. Frequency

Figure V-3A



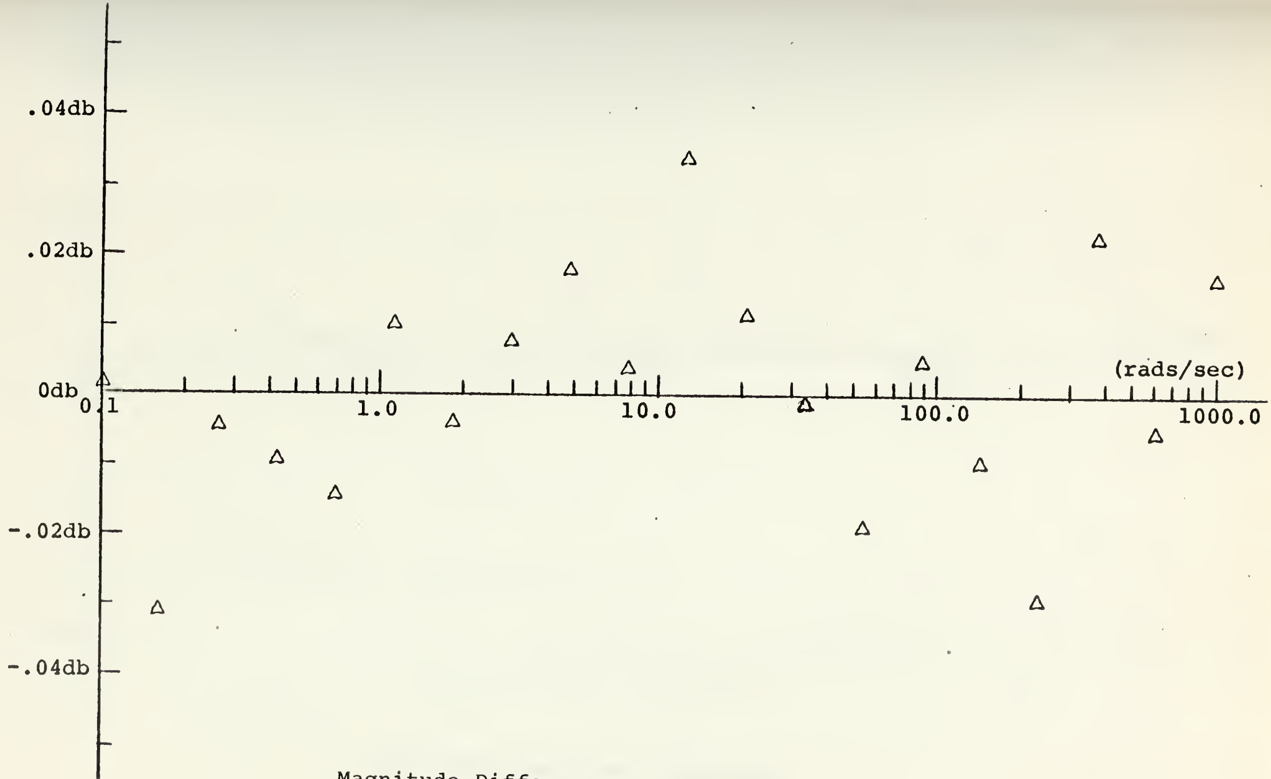
Phase vs. Frequency

Figure V-3B



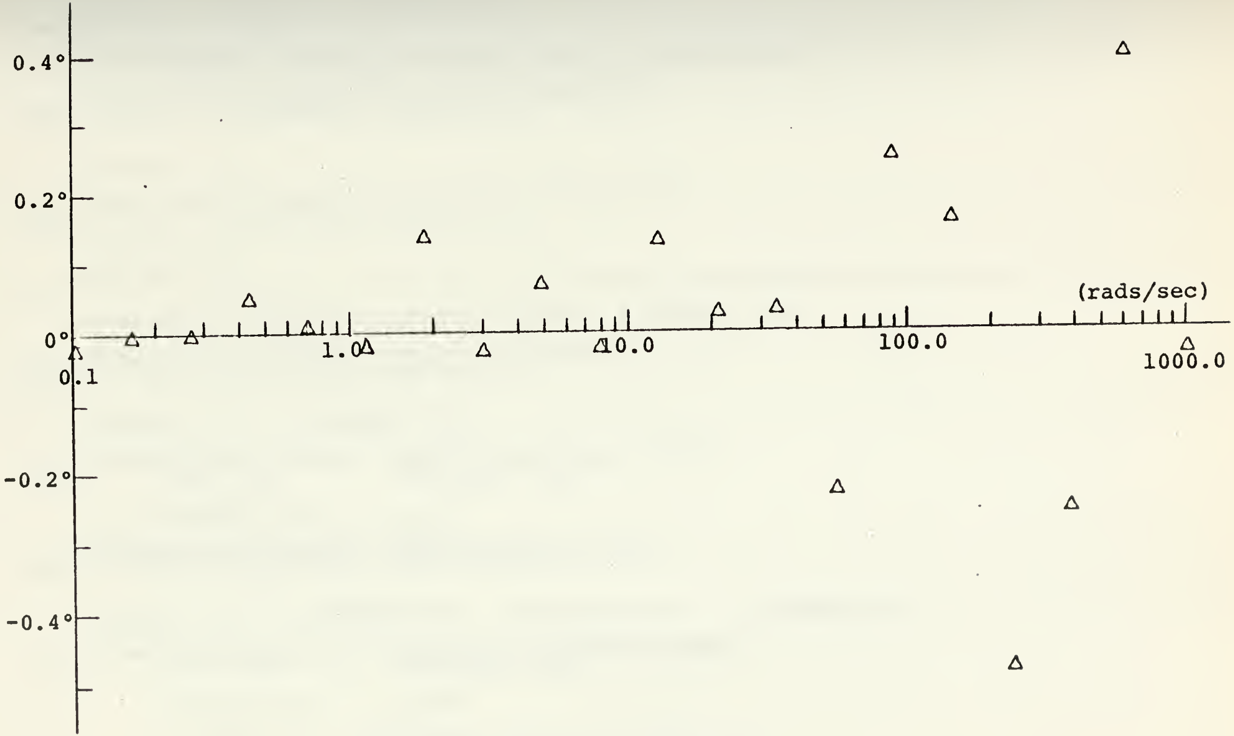
Magnitude vs. Phase

Figure V-3C



Magnitude Difference vs. Frequency

Figure V-3D



Phase Difference vs. Frequency

Figure V-3E

```
TITLE --- SYNTHESIS EXAMPLE TWO
------------------------------------------------------------------------

UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00

    UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00
------------------------------------------------------------------------

    COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

    COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00    1.000000E 00

    COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
 ARE:      REAL PART           IMAGINARY PART

        -1.000000E 00        0.0

    COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  1.000000E 00    3.000000E 00    3.000000E 00    1.000000E 00

    COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
 ARE:      REAL PART           IMAGINARY PART

        -1.000000E 00        0.0

        -1.000000E 00        0.0

        -1.000000E 00        0.0
------------------------------------------------------------------------
```

Computer Numerical Output

Figure V-3F

```
--------------------------------------------------------------------------------
     THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
     THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
     THE COST FUNCTION TO BE USED IS THE TYPE   1
--------------------------------------------------------------------------------
THE MINIMUM COST FUNCTION VALUE =  1.506277E-04
THE ERROR RETURN CODE FROM BOXPLX =   0
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

    2.618764E 01    2.443936E 05
      OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
  ROOTS ARE:      REAL PART         IMAGINARY PART
                -1.071535E-04       0.0
--------------------------------------------------------------------------------
```

Computer Numerical Output

Figure V-3G

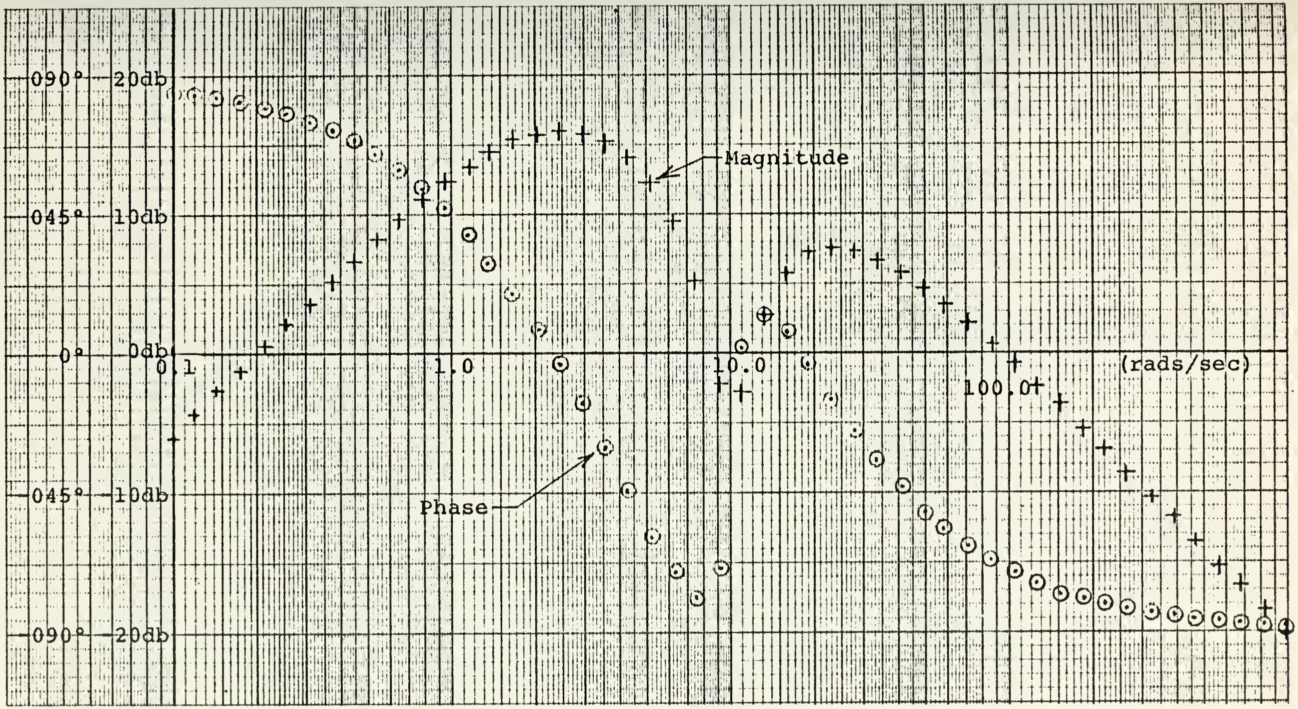
```
    OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  4.891628E 04    3.042503E 04    4.272676E 03    2.710205E 01

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
ROOTS ARE:      REAL PART           IMAGINARY PART

               -1.502601E 02        0.0

               -4.978378E 00        0.0

               -2.412790E 00        0.0
--------------------------------------------------------------------

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =  9.017527E 03

  FREQUENCY      MAGNITUDE(DB)   DESIRED MAG(DB)   PHASE(DEG)      DESIRED PHASE
   9.999996E-02  -6.036484E 00   -6.037990E 00    8.637637E 01    8.639999E 01
   1.619999E-01  -1.861110E 00   -1.830299E 00    8.419530E 01    8.420000E 01
   2.640000E-01   2.340905E 00    2.345422E 00    8.059622E 01    8.059999E 01
   4.280000E-01   6.435030E 00    6.444384E 00    7.484970E 01    7.479999E 01
   6.950000E-01   1.031233E 01    1.039656E 01    6.570982E 01    6.570000E 01
   1.129999E 00   1.395462E 01    1.394458E 01    5.167982E 01    5.170000E 01
   1.830000E 00   1.663714E 01    1.670111E 01    3.193719E 01    3.179999E 01
   2.980000E 00   1.810199E 01    1.809430E 01    6.953293E 00    6.980000E 00
   4.830000E 00   1.777023E 01    1.775233E 01   -1.943172E 01   -1.950000E 01
   7.849999E 00   1.579552E 01    1.579161E 01   -4.352361E 01   -4.350000E 01
   1.270000E 01   1.268311E 01    1.264914E 01   -6.266949E 01   -6.279999E 01
   2.070000E 01   8.861003E 00    8.849588E 00   -7.767279E 01   -7.770000E 01
   3.359999E 01   4.709544E 00    4.710566E 00   -9.006964E 01   -9.009999E 01
   5.459999E 01   2.379835E-01    2.557422E-01   -1.022297E 02   -1.020000E 02
   8.859999E 01  -4.696572E 00   -4.701540E 00   -1.157497E 02   -1.160000E 02
   1.440000E 02  -1.043819E 01   -1.042867E 01   -1.308414E 02   -1.310000E 02
   2.340000E 02  -1.716870E 01   -1.713969E 01   -1.454845E 02   -1.450000E 02
   3.790000E 02  -2.467876E 01   -2.470154E 01   -1.572561E 02   -1.570000E 02
   6.160000E 02  -3.273282E 01   -3.272775E 01   -1.656042E 02   -1.660000E 02
   1.000000E 03  -4.099533E 01   -4.101219E 01   -1.710311E 02   -1.710000E 02
--------------------------------------------------------------------
     THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE
--------------------------------------------------------------------
```

Computer Numerical Output

Figure V-3H

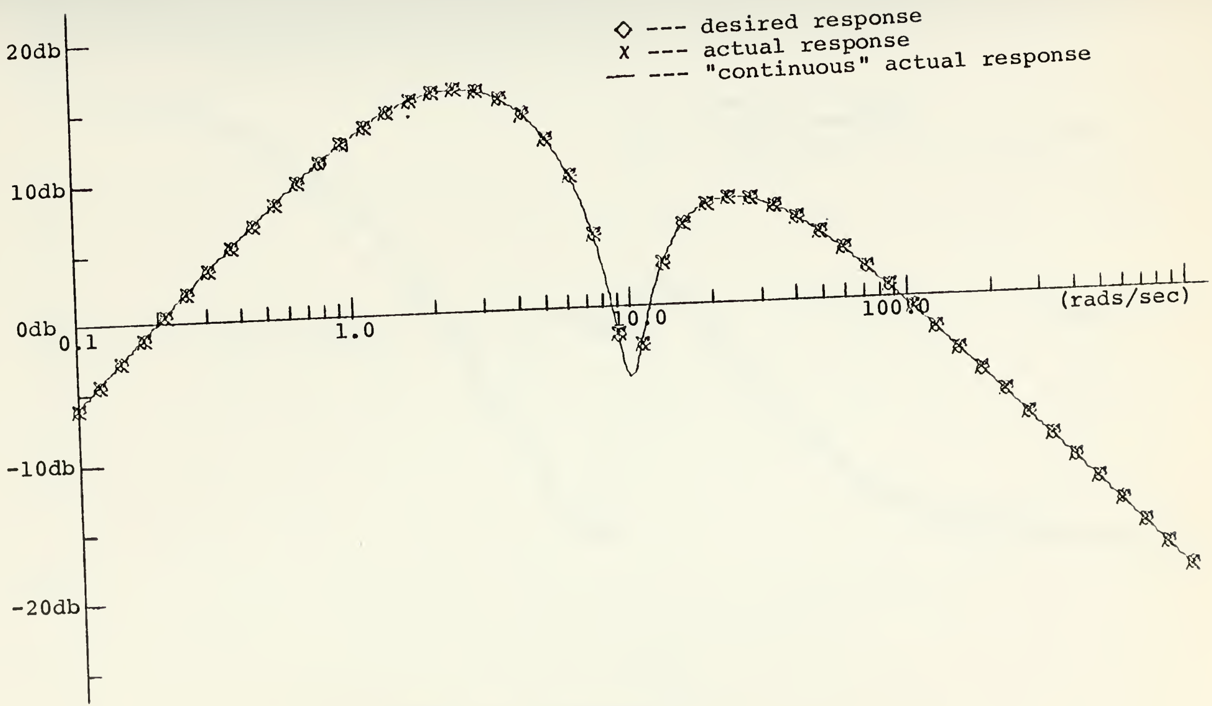
3. Synthesis Example 3.

A frequency response of a system with resonant peaks, shown in figure V-4, is considered in this example. The same example is considered in ref. 19, where a combination of graphical and analytical techniques are used to determine the transfer function. Initially a transfer function consisting of a first order numerator and second order denominator was assumed in trying to model the system. This choice was based on the low and high frequency magnitude curve slopes of approximately +20 db/decade and -20 db/decade respectively and the low and high frequency phase values of approximately +90 degrees and -90 degrees. This form failed to match the dip in the magnitude curve in the region of 10 radians/second and the jump in the phase curve in the same frequency area. After increasing the order of the numerator and denominator, a transfer function consisting of a third order numerator and fourth order denominator was found to provide a phase and gain curve that closely approximates the measured frequency response. The results of this are shown in figures V-4A through V-4H. The numerical values of the transfer function parameters returned from the program closely approximate those of the actual system from which the frequency response measurements were obtained. The interested reader may verify this fact by comparing the values shown in figures V-4G and V-4E with those of the actual system given in ref. 19.

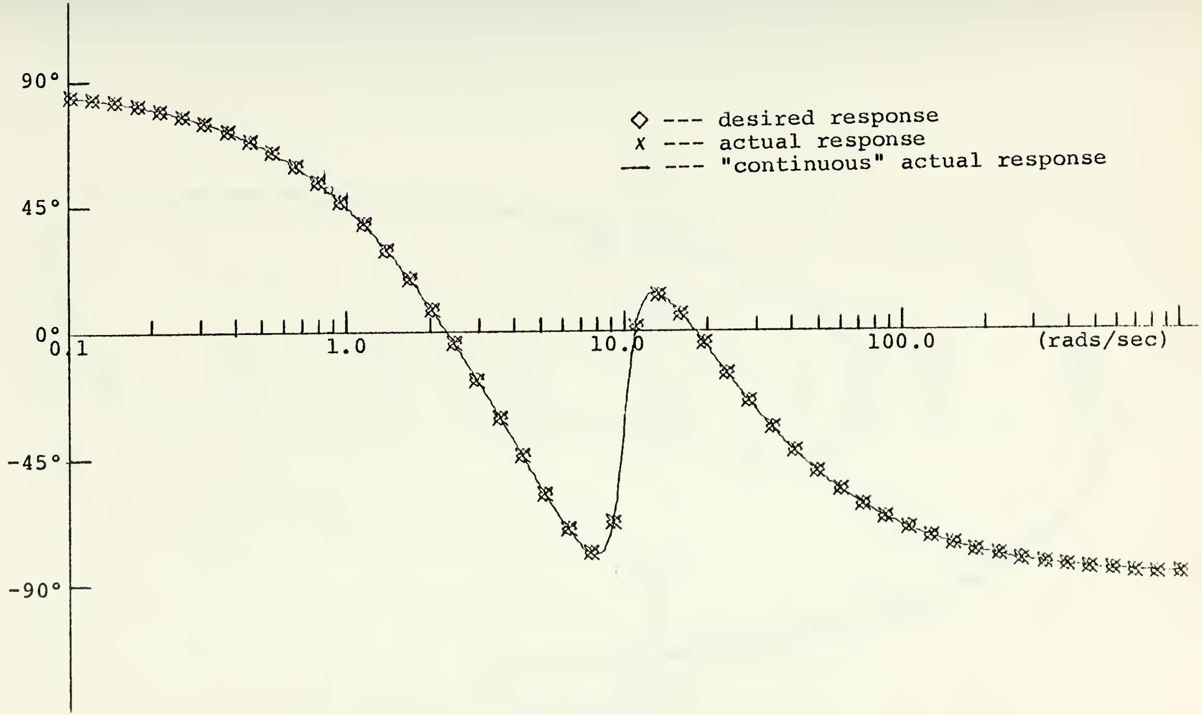


Input Frequency Response Data

Figure V-4

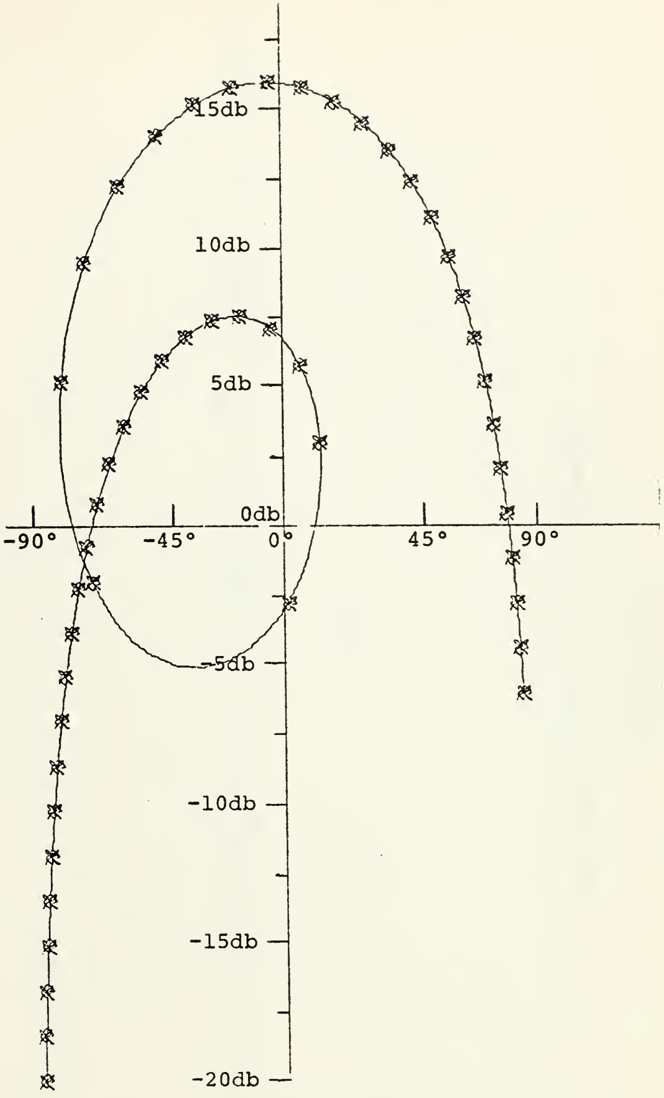


Magnitude vs. Frequency
Figure V-4A

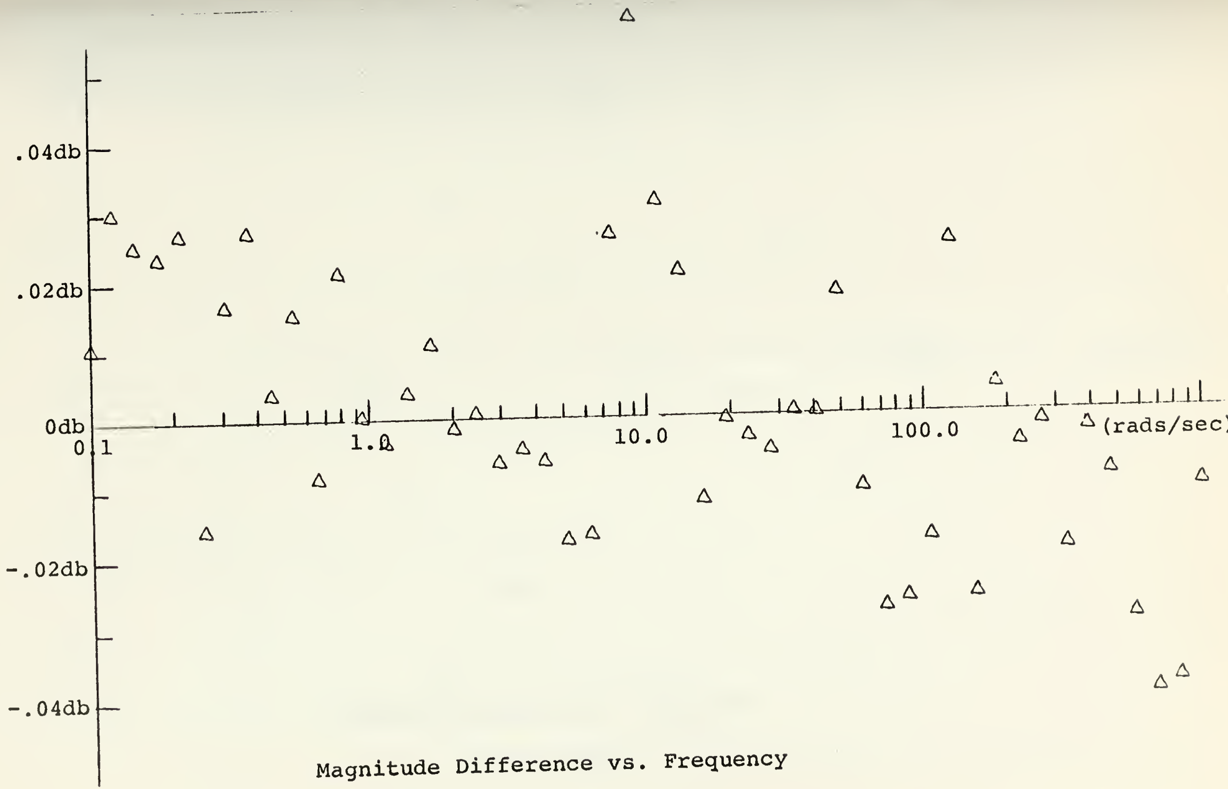


Phase vs. Frequency

Figure V-4B

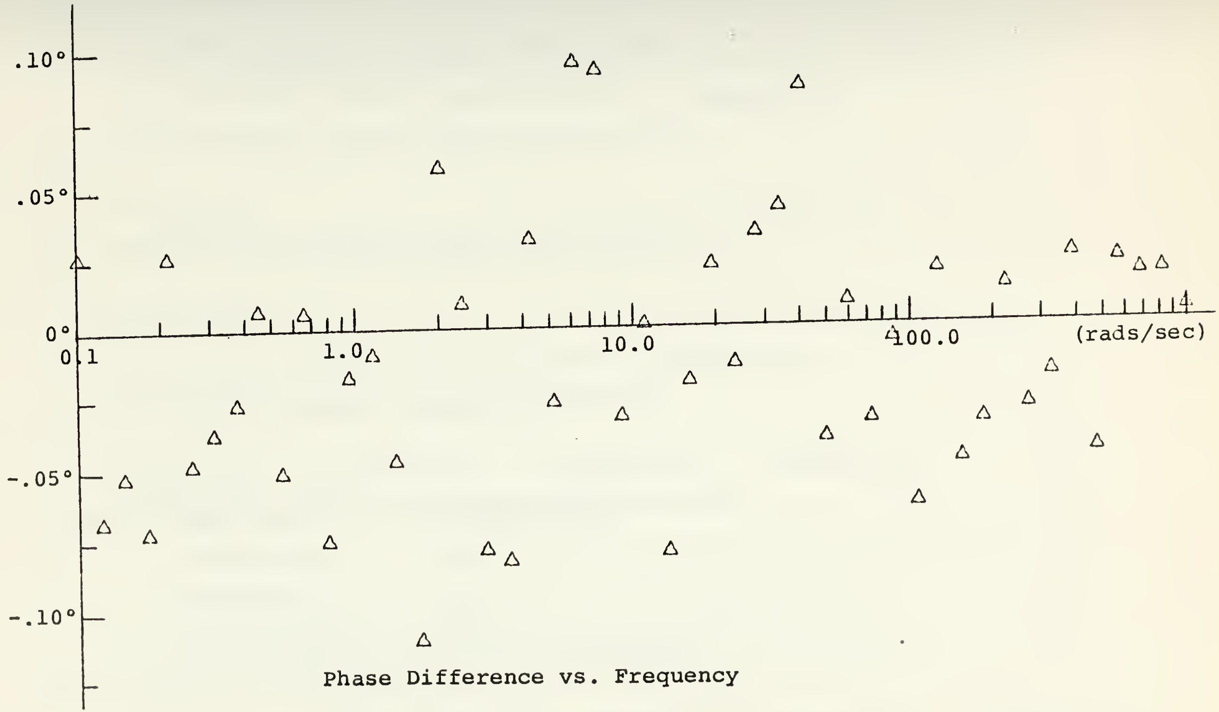


Magnitude vs. Phase
Figure V-4C



Magnitude Difference vs. Frequency

Figure V-4D



Phase Difference vs. Frequency

Figure V-4E

```
     TITLE --- SYNTHESIS EXAMPLE THREE 50 PTS
------------------------------------------------------------------------
     UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

     UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00

     UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00
------------------------------------------------------------------------
     COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

     COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00    3.000000E 00    3.000000E 00    1.000000E 00

     COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
 ARE:     REAL PART          IMAGINARY PART

       -1.000000E 00        0.0

       -1.000000E 00        0.0

       -1.000000E 00        0.0

     COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00    4.000000E 00    6.000000E 00    4.000000E 00    1.000000E 00
```

Computer Numerical Output

Figure V-4F

```
      CCMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
ARE:     REAL PART          IMAGINARY PART

        -1.000000E 00         0.0
        -1.000000E 00         0.0
        -1.000000E 00         0.0
        -1.000000E 00         0.0
--------------------------------------------------------------------------
--------------------------------------------------------------------------

      THE CCMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTICN FOR THE CCMPENSATOR TRANSFER FUNCTION
--------------------------------------------------------------------------
--------------------------------------------------------------------------

      THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000
--------------------------------------------------------------------------
--------------------------------------------------------------------------

      THE CCST FUNCTICN TO BE USED IS THE TYPE  0
--------------------------------------------------------------------------

THE MINIMUM CCST FUNCTICN VALUE =  4.754462E-04

THE ERRCR RETURN CODE FRCM BOXPLX =  -1
--------------------------------------------------------------------------
--------------------------------------------------------------------------

      OPTIMIZED CCMPENSATCR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

    6.788290E 00    2.365316E 05    4.754176E 03    2.362180E 03
```

Computer Numerical Output

Figure V-4G

```
     OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
ROOTS ARE:      REAL PART           IMAGINARY PART

              -1.006292E 00          9.955896E 00

              -1.006292E 00         -9.955896E 00

              -2.869929E-05          0.0
--------------------------------------------------------------

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

  4.726638E 04   4.021491E 04   9.944633E 03   8.747952E 02   2.364240E 01
     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
ROOTS ARE:      REAL PART           IMAGINARY PART
              -1.991696E 01          0.0
              -1.012778E 01          0.0
              -4.956751E 00          0.0
              -1.999504E 00          0.0
--------------------------------------------------------------

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN =  9.991237E 01
```

```
--------------------------------------------------------------
    THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE
--------------------------------------------------------------
```

Computer Numerical Output

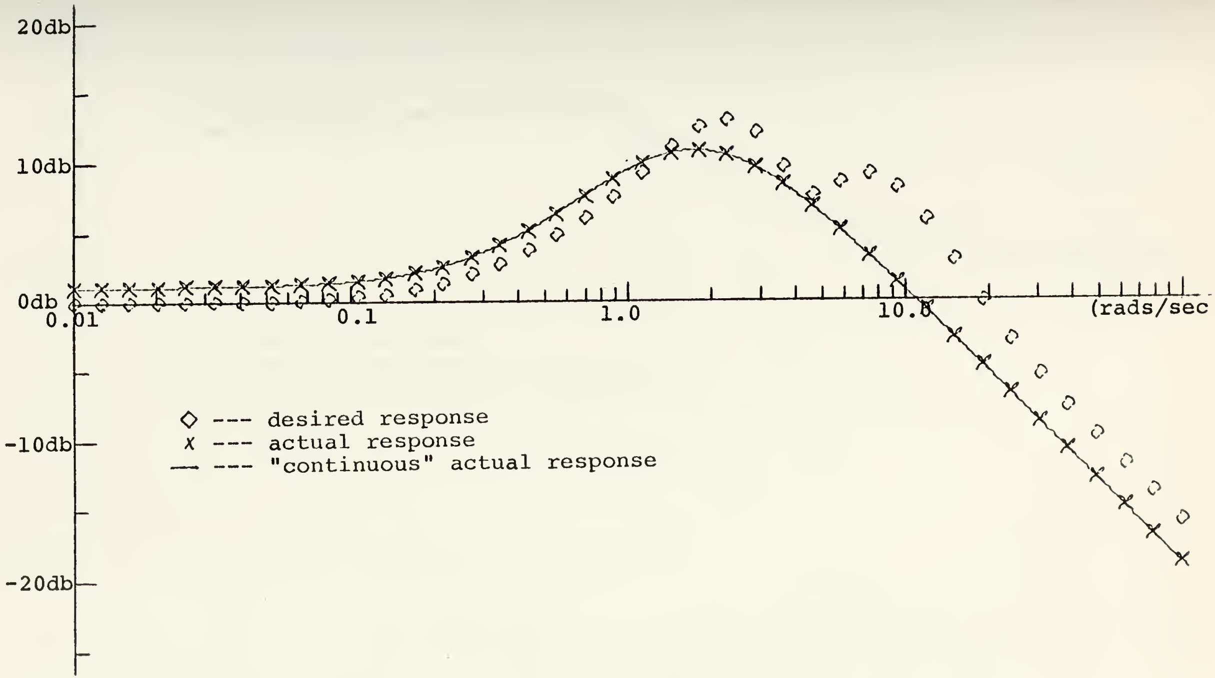
Figure V-4H

4. *Synthesis Example 4.*

In this final example problem a low order approximation to a known higher order system is illustrated. The original system transfer function under consideration consists of a seventh order numerator and an eight order denominator given by

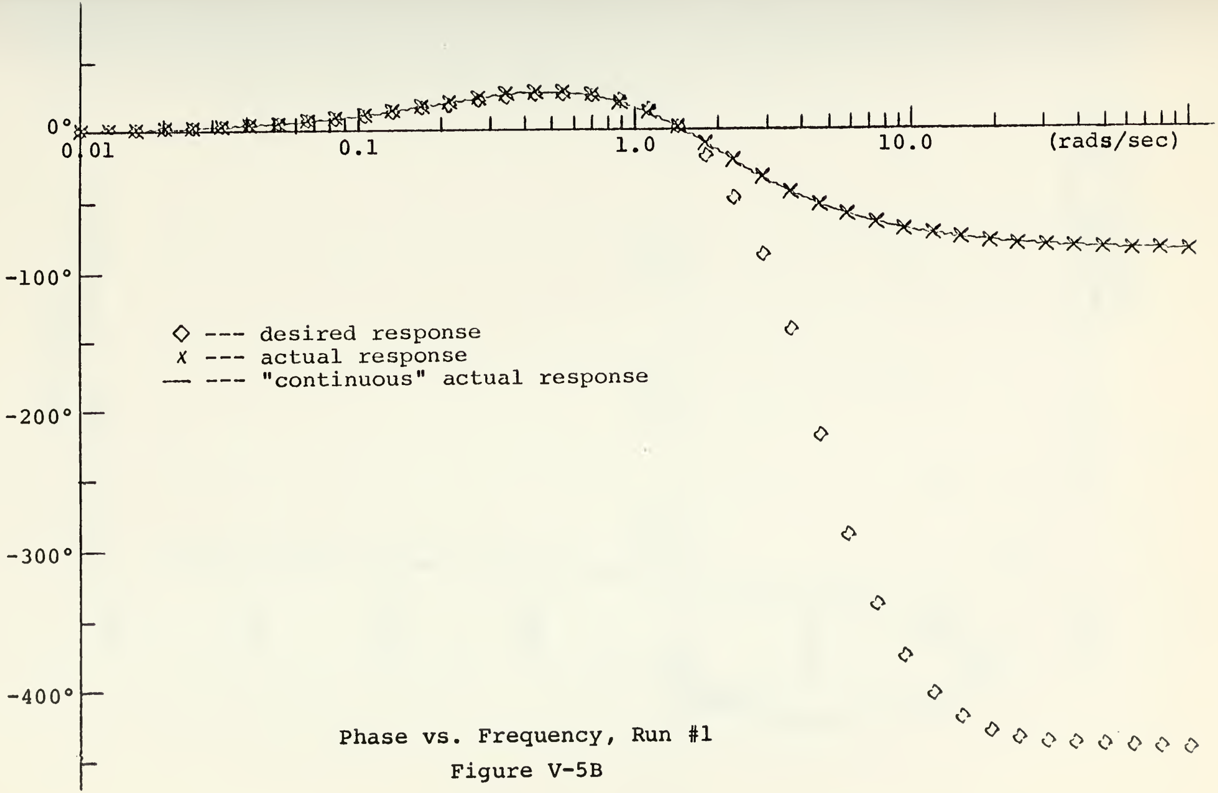
$$\frac{16s^7+483s^6+6010s^5+36380s^4+122664s^3+222088s^2+185760s+40320}{s^8+36s^7+546s^6+4536s^5+22449s^4+67284s^3+118124s^2+109584s+40320}$$

An analytical method of approximating this high order transfer function is discussed in ref. 21. Initially 40 discrete points, representing the frequency response of the high order transfer function were selected over the frequency range from 0.01 radians/second to 100.0 radians/second. In order to be able to form a comparison with the reduced order model obtained in ref. 21, a transfer function consisting of a first order numerator and a second order denominator was assumed. The results returned from the program using this form of transfer function are shown in figures V-5A through V-5H. It is not surprising that there is a significant amount of deviation in the magnitude curve in the vicinity of the resonant effects of the higher order system nor that the phase response of this reduced order model does not match the higher order phase response in the higher frequency range. In order to compare these results with analytical methods for forming reduced order models, the frequency response for the reduced order model of ref. 21 is shown in figures V-5I through V-5M. The numerical values returned from the CALICO program are of the same order of magnitude as those found by the analytical technique, but as can be seen from the magnitude curves of the two approaches (figuresV-5A and V-5I), the analytical method matches the higher order transfer function extremely well at the lower and higher frequencies and makes little attempt to match the higher order system in the range of frequencies at which the resonant peaks occur. The results returned from the program, on the other hand because they are a function of the deviation of the response from the desired values over the entire range of frequencies, tend to distribute any error between the responses over the entire range of frequencies being considered. A slightly different representation of the system may be obtained by varying the frequency range under consideration and the type of cost function used in the program. For example, figures V-5N through V-5U illustrate the effects of using a type two cost function and limiting the frequency range from 0.1 to 100.0 radians/second. Again the decision as to which model is best must be made by the user in light of the intended use of the reduced order representation.



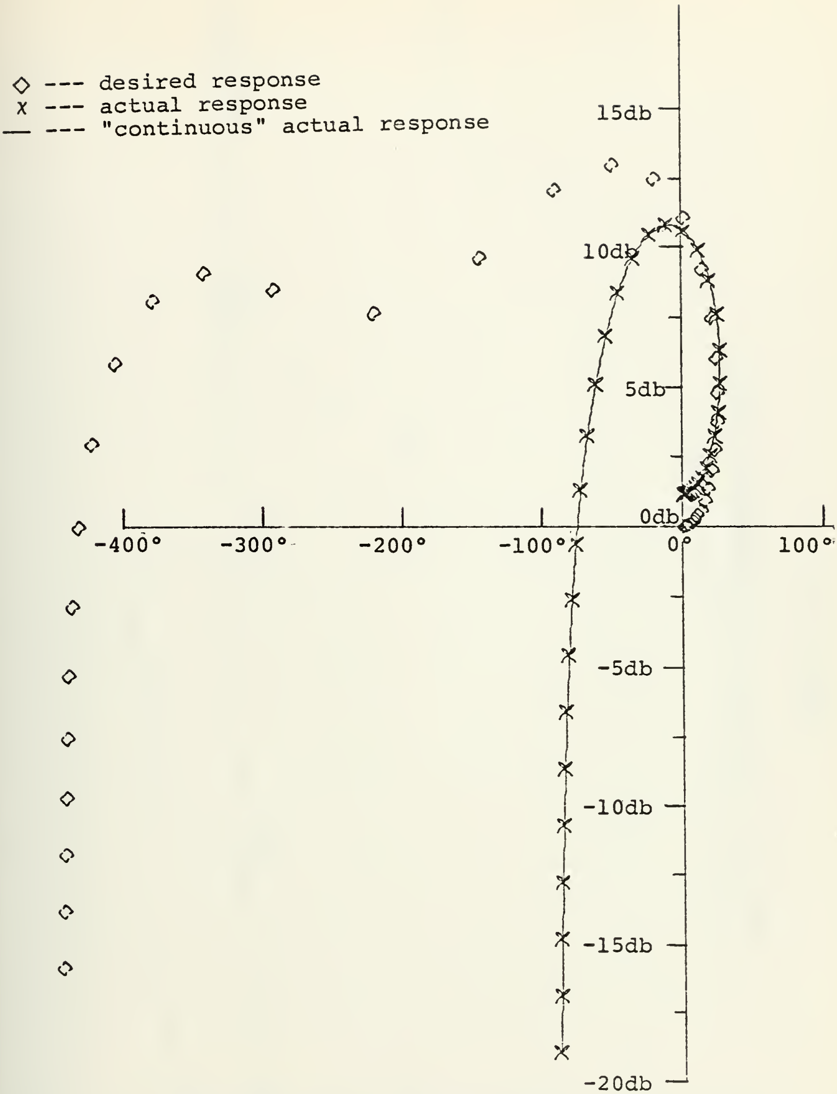
Magnitude vs. Frequency, Run #1

Figure V-5A



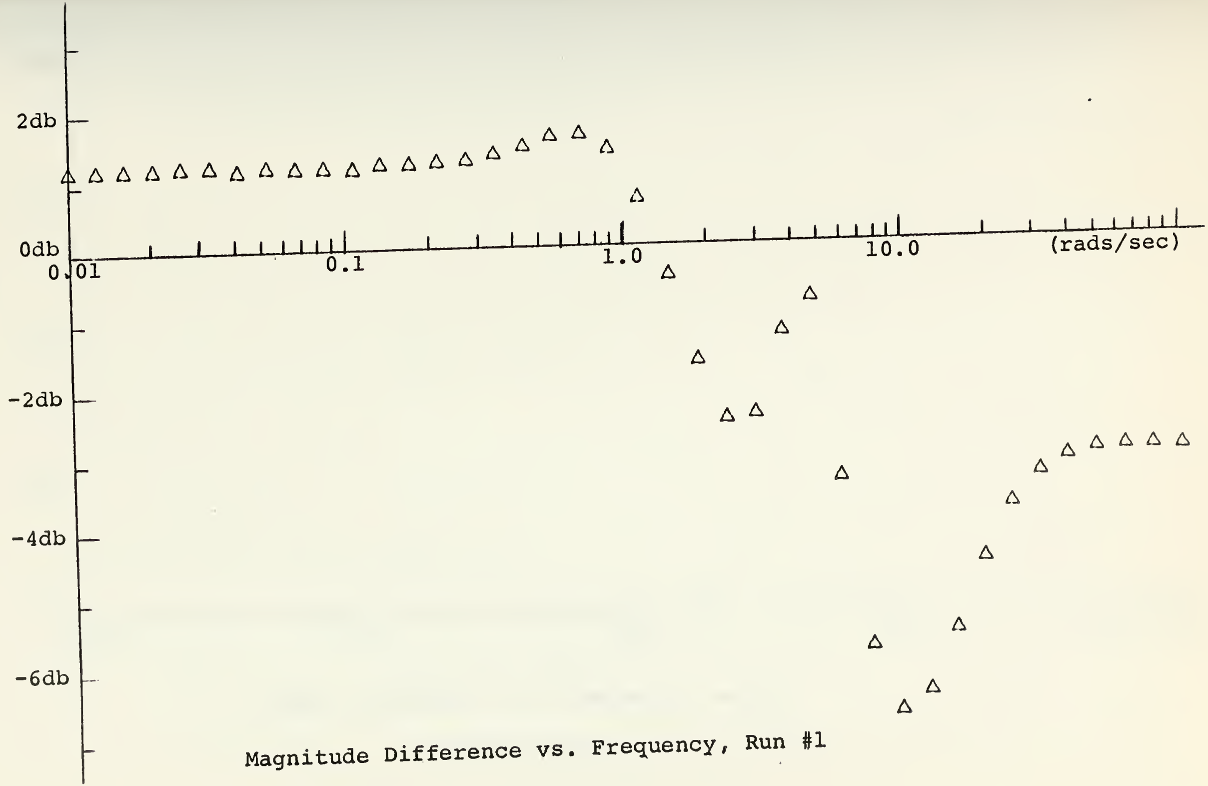
Phase vs. Frequency, Run #1

Figure V-5B



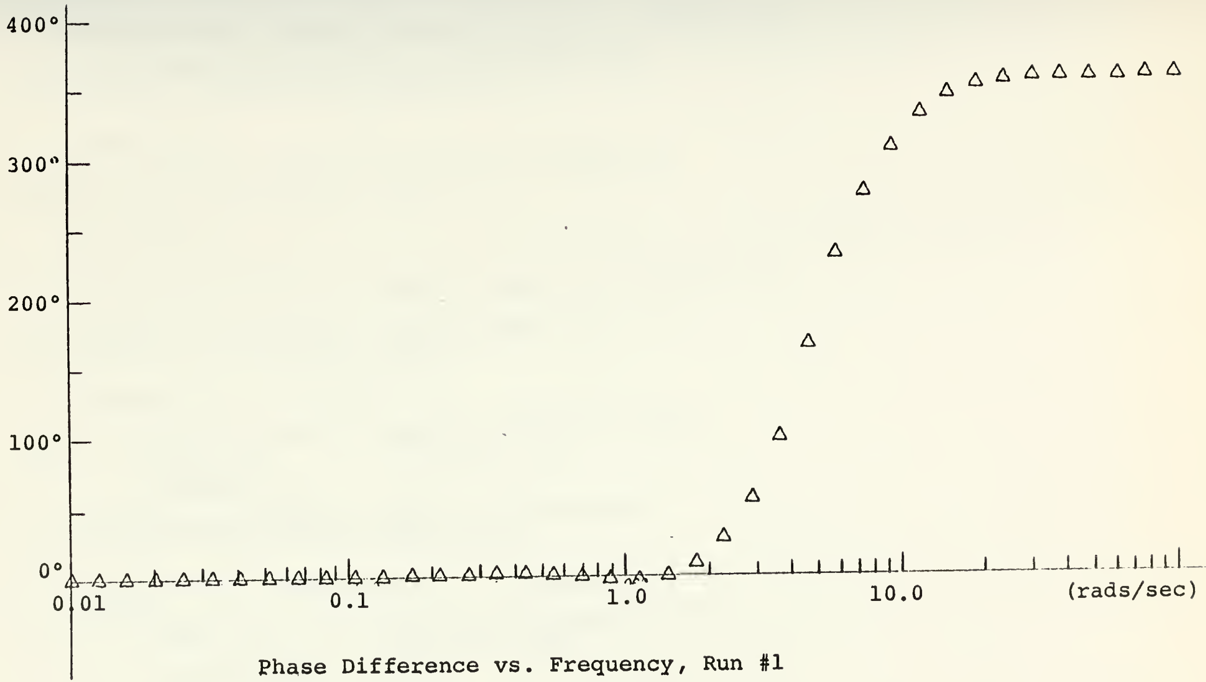
Magnitude vs. Phase, Run #1

Figure V-5C



Magnitude Difference vs. Frequency, Run #1

Figure V-5D



Phase Difference vs. Frequency, Run #1

Figure V-5E

```
TITLE --- LOW CFC APPROX EXAMP 40PTS
--------------------------------------------------------------------------------
     UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

     UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00

     UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00
--------------------------------------------------------------------------------
     COMPENSATOR TRANSFER FUNCTION GAIN =  1.000000E 00

     COMPENSATOR TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00    1.000000E 00

     COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS
 ARE:      REAL PART           IMAGINARY PART

        -1.000000E 00        0.0

     COMPENSATOR TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00    2.000000E 00    1.000000E 00

     COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS
 ARE:      REAL PART           IMAGINARY PART

        -1.000000E 00        0.0

        -1.000000E 00        0.0
--------------------------------------------------------------------------------
```

Computer Numerical Output, Run #1

Figure V-5F

```
     THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION

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

     THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

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

     THE COST FUNCTION TO BE USED IS THE TYPE    1

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

THE MINIMUM COST FUNCTION VALUE =  1.321693E 01

THE ERROR RETURN CODE FROM BOXPLX =    0

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

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   2.146659E 05    6.417439E 05

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
  ROOTS ARE:       REAL PART          IMAGINARY PART

                 -3.348156E-01        0.0
-----------------------------------------------------------------

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

   1.876209E 05    1.869405E 05    5.706169E 04

     OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
  ROOTS ARE:       REAL PART          IMAGINARY PART

                 -1.638056E 00        7.776945E-01

                 -1.638056E 00       -7.776945E-01
-----------------------------------------------------------------
```

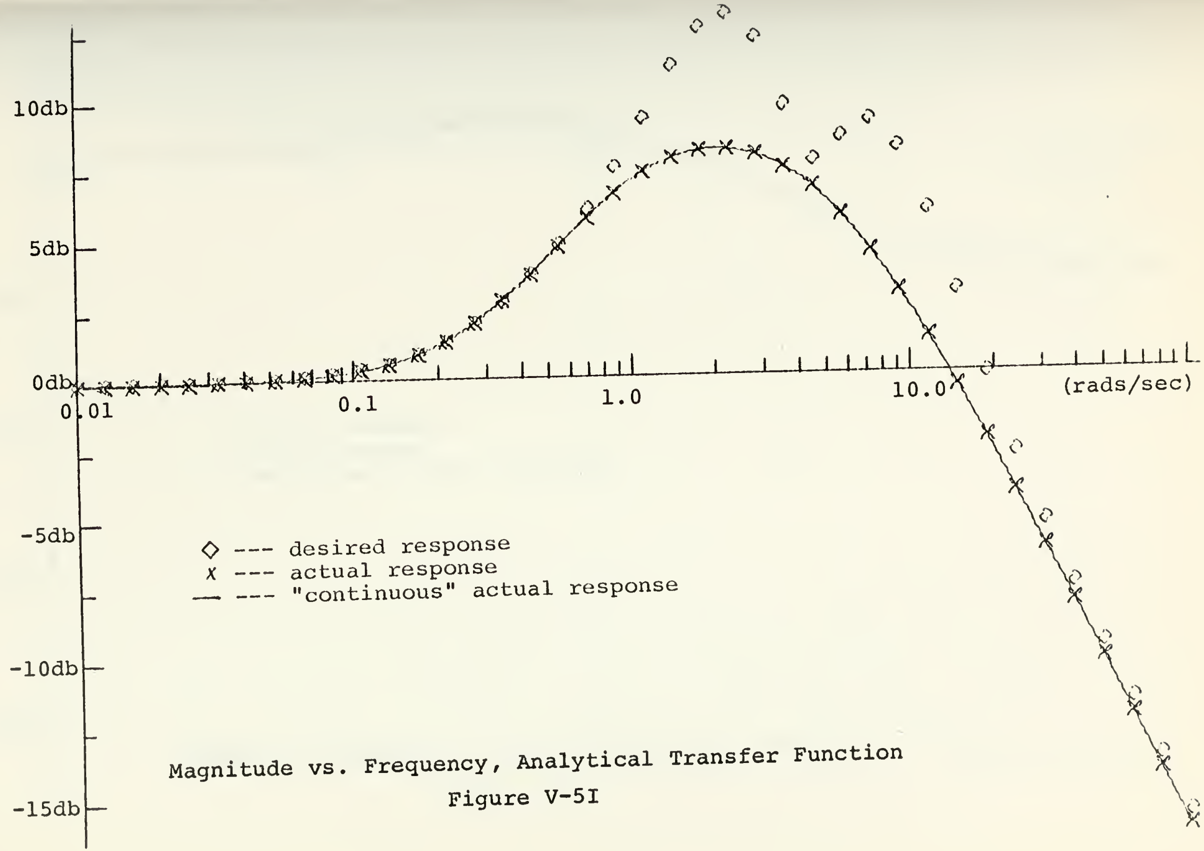
Computer Numerical Output, Run #1
Figure V-5G

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.124649E 01

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999998E-03	1.181429E 00	0.0	1.139874E 00	1.080000E 00
1.270000E-02	1.183698E 00	0.0	1.447248E 00	1.370000E 00
1.600000E-02	1.187197E 00	0.0	1.822537E 00	1.730000E 00
2.030000E-02	1.192967E 00	0.0	2.310744E 00	2.190000E 00
2.570000E-02	1.202132E 00	0.0	2.922201E 00	2.770000E 00
3.260000E-02	1.216921E 00	0.0	3.700172E 00	3.509999E 00
4.120000E-02	1.240154E 00	8.642113E-02	4.663218E 00	4.429999E 00
5.220000E-02	1.277468E 00	8.642113E-02	5.881681E 00	5.580000E 00
6.609994E-02	1.336473E 00	1.719992E-01	7.394758E 00	7.000000E 00
8.379996E-02	1.429868E 00	2.567422E-01	9.268644E 00	8.750000E 00
1.059999E-01	1.573788E 00	4.237797E-01	1.151788E 01	1.090000E 01
1.339999E-01	1.792952E 00	5.876731E-01	1.416620E 01	1.330000E 01
1.700000E-01	2.125519E 00	9.064573E-01	1.722145E 01	1.609999E 01
2.150000E-01	2.599790E 00	1.363709E 00	2.044739E 01	1.889999E 01
2.729999E-01	3.266556E 00	2.007405E 00	2.364001E 01	2.170000E 01
3.460000E-01	4.132125E 00	2.797576E 00	2.625536E 01	2.389999E 01
4.380000E-01	5.186727E 00	3.750413E 00	2.773952E 01	2.529999E 01
5.540000E-01	6.385768E 00	4.810984E 00	2.751895E 01	2.559999E 01
7.020000E-01	7.663657E 00	6.063919E 00	2.505437E 01	2.459999E 01
8.890000E-01	8.855056E 00	7.531536E 00	1.997882E 01	2.159999E 01
1.129999E 00	9.954495E 00	9.277854E 00	1.200836E 01	1.470000E 01
1.429999E 00	1.064944E 01	1.110188E 01	1.683441E 00	1.549999E 00
1.799999E 00	1.086059E 01	1.254731E 01	-1.007033E 01	-1.979999E 01
2.290000E 00	1.051939E 01	1.304492E 01	-2.293156E 01	-5.000000E 01
2.899999E 00	9.678394E 00	1.214909E 01	-3.474916E 01	-9.039999E 01
3.669999E 00	8.401459E 00	9.635990E 00	-4.546928E 01	-1.450000E 02
4.639999E 00	6.861783E 00	7.676304E 00	-5.432191E 01	-2.210000E 02
5.879999E 00	5.118869E 00	8.530223E 00	-6.163776E 01	-2.930000E 02
7.440000E 00	3.268962E 00	9.127317E 00	-6.749031E 01	-3.430000E 02
9.429999E 00	1.321501E 00	8.131801E 00	-7.219646E 01	-3.800000E 02
1.190000E 01	-6.150827E-01	5.889321E 00	-7.587126E 01	-4.070000E 02
1.510000E 01	-2.636412E 00	2.984381E 00	-7.835545E 01	-4.240000E 02
1.909999E 01	-4.648530E 00	-4.353884E-02	-8.118462E 01	-4.340000E 02
2.420000E 01	-6.685974E 00	-2.865421E 00	-8.304007E 01	-4.390000E 02
3.070000E 01	-8.741075E 00	-5.352125E 00	-8.451234E 01	-4.420000E 02
3.889999E 01	-1.079027E 01	-7.618134E 00	-8.566870E 01	-4.430000E 02
4.920000E 01	-1.282622E 01	-9.762332E 00	-8.657521E 01	-4.440000E 02
6.239999E 01	-1.488786E 01	-1.183520E 01	-8.729959E 01	-4.450000E 02
7.900000E 01	-1.693501E 01	-1.389297E 01	-8.786689E 01	-4.460000E 02
1.000000E 02	-1.898141E 01	-1.591760E 01	-8.831483E 01	-4.470000E 02

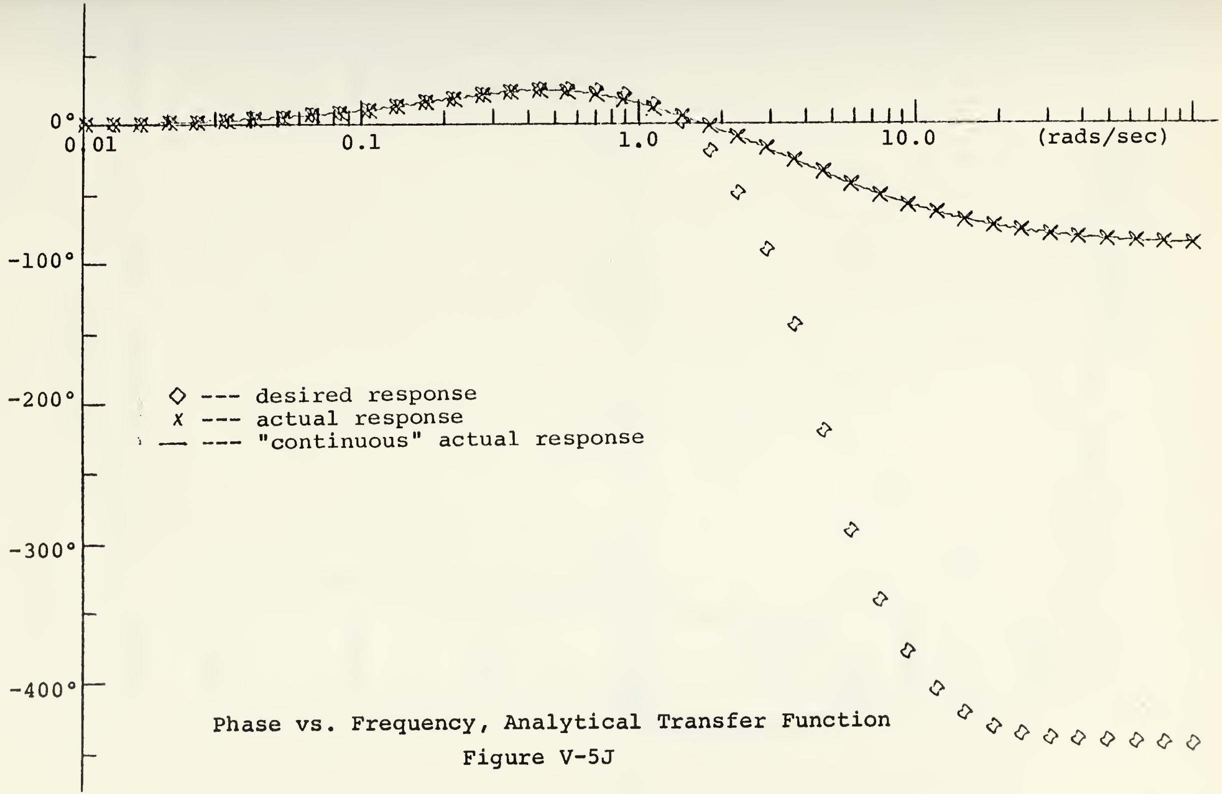
THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE

Computer Numerical Output, Run #1
Figure V-5H



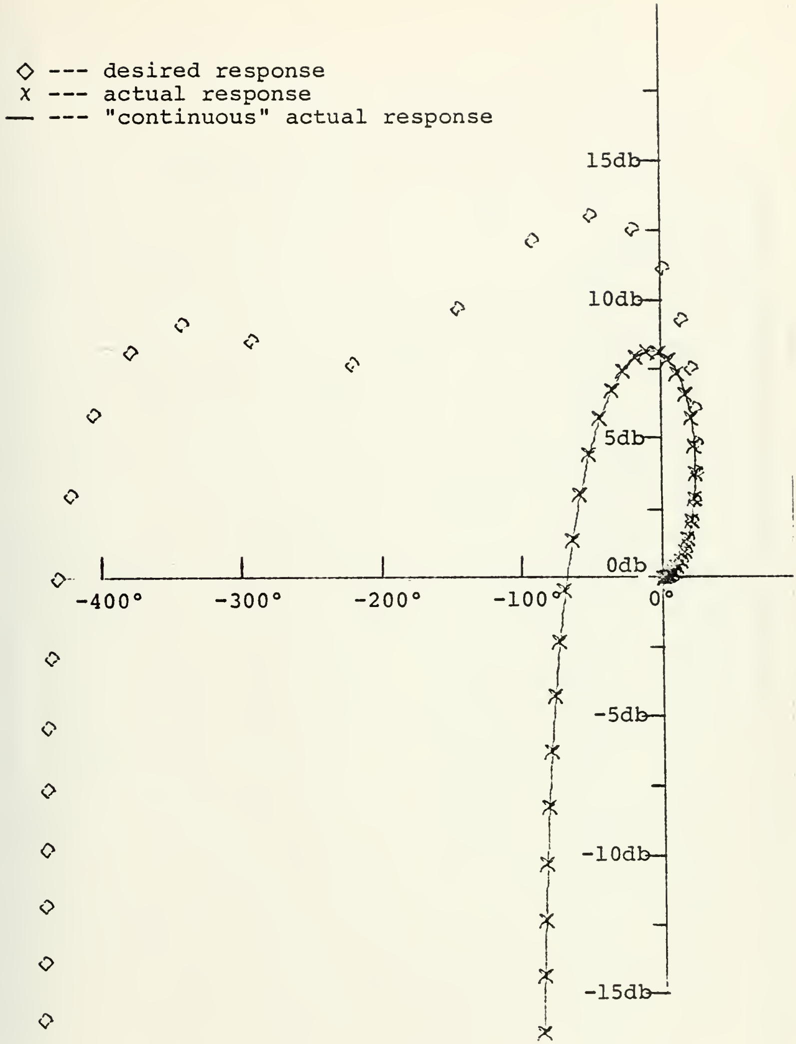
Magnitude vs. Frequency, Analytical Transfer Function

Figure V-5I



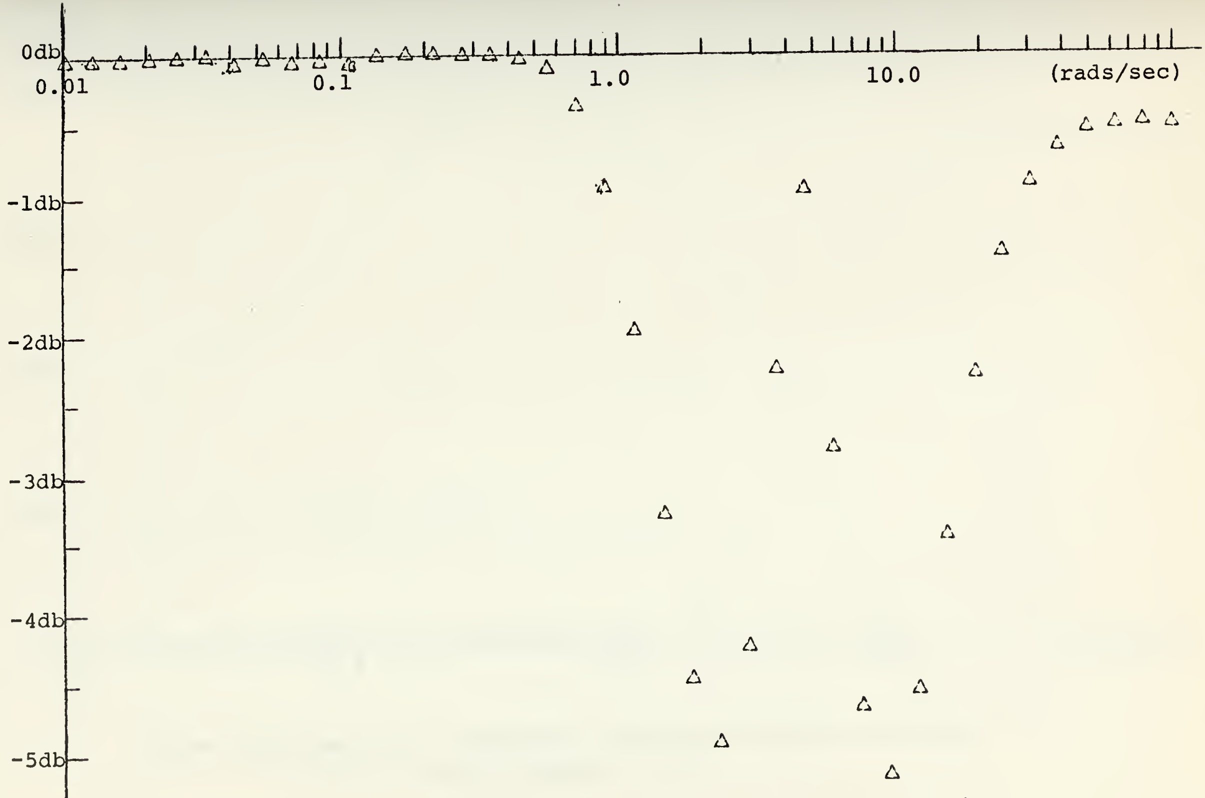
Phase vs. Frequency, Analytical Transfer Function

Figure V-5J



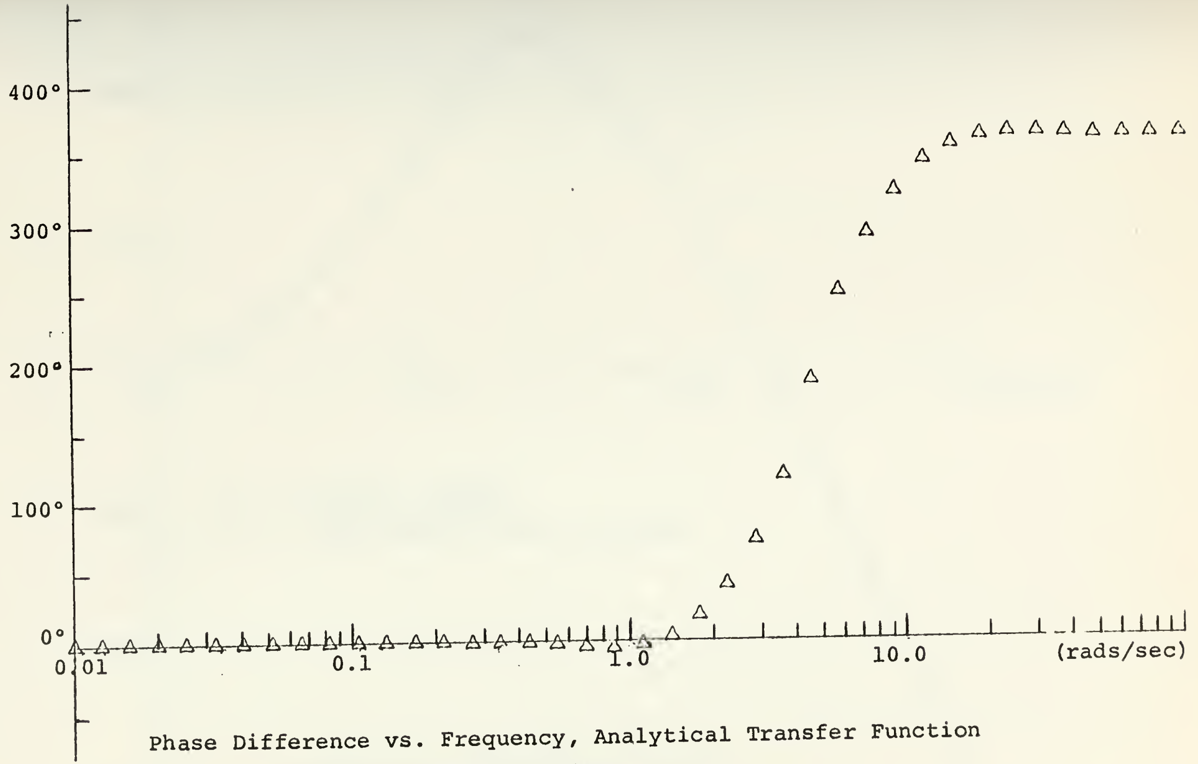
Magnitude vs. Phase, Analytical Transfer Function

Figure V-5K



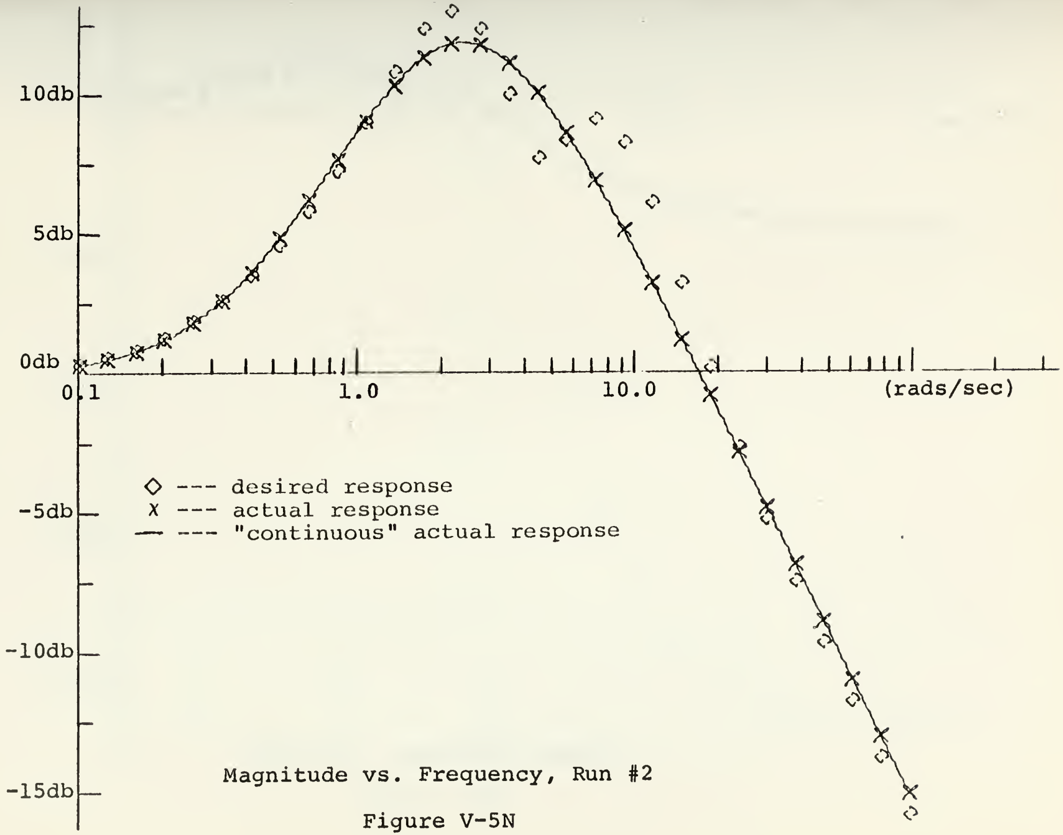
Magnitude Difference vs. Frequency, Analytical Transfer Function

Figure V-5L



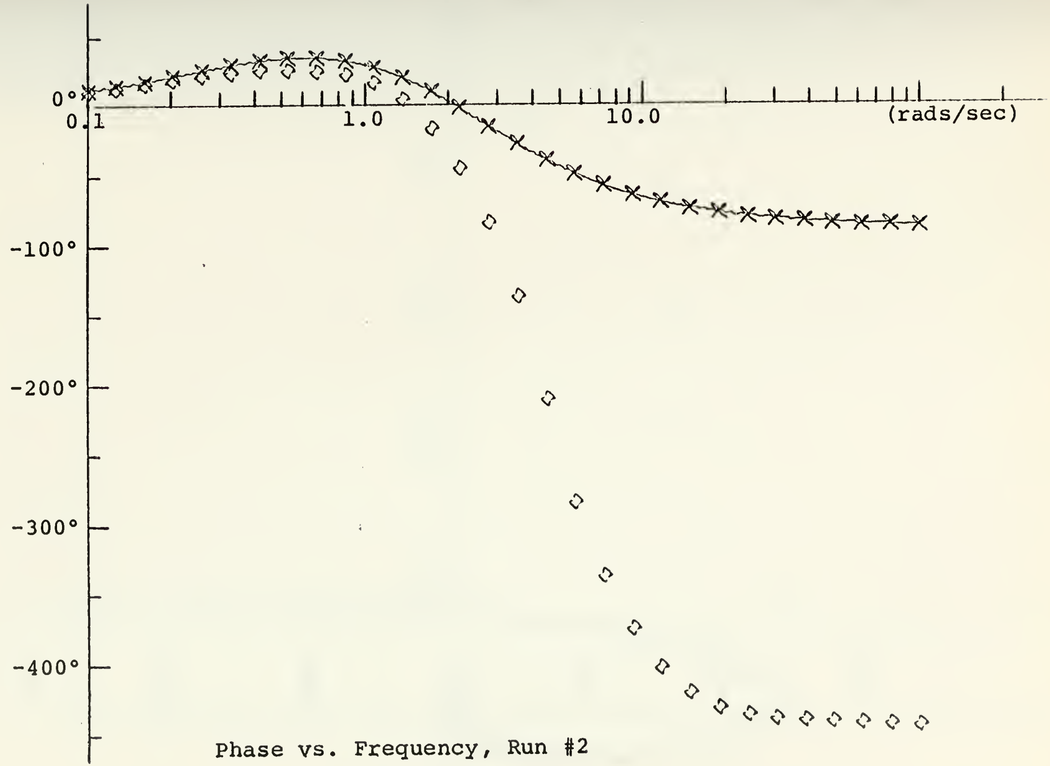
Phase Difference vs. Frequency, Analytical Transfer Function

Figure V-5M



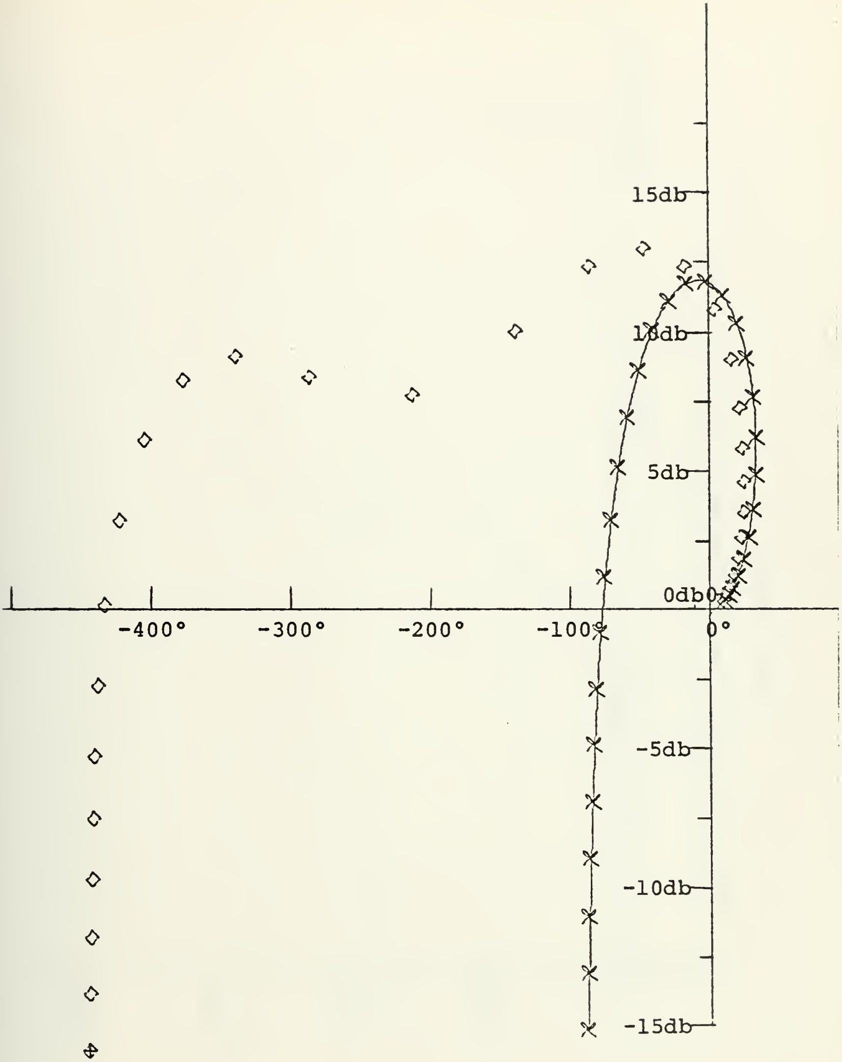
Magnitude vs. Frequency, Run #2

Figure V-5N



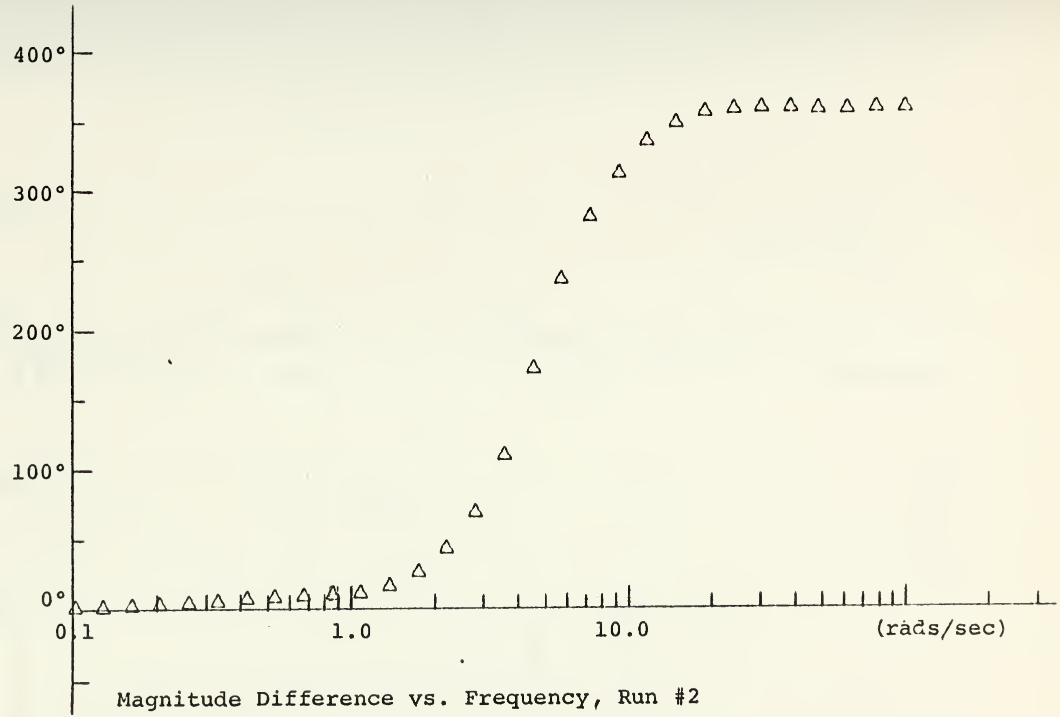
Phase vs. Frequency, Run #2

Figure V-50



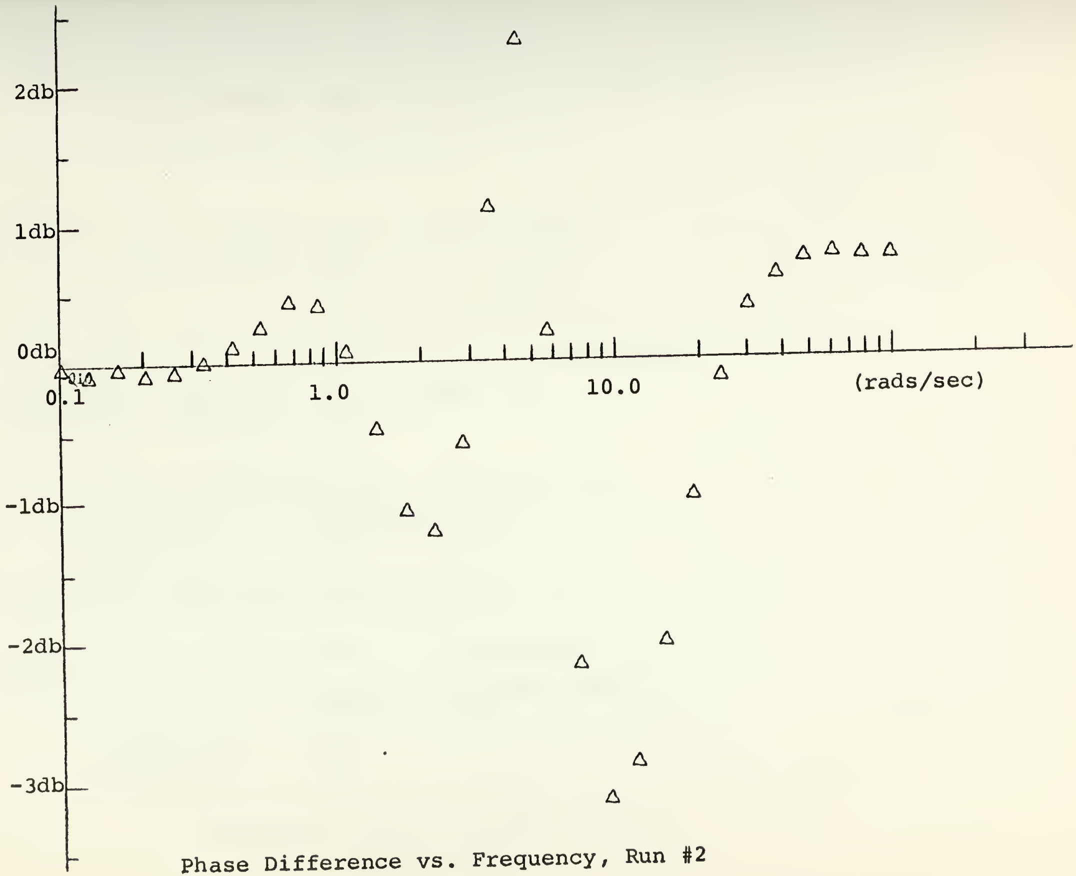
Magnitude vs. Phase, Run #2

Figure V-5P



Magnitude Difference vs. Frequency, Run #2

Figure V-5Q



Phase Difference vs. Frequency, Run #2

Figure V-5R

```
     TITLE --- LOW ORD APPROX DAMP ROOTS
--------------------------------------------------------------------------------

     UNCOMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

     UNCOMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00

     UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00
--------------------------------------------------------------------------------

     COMPENSATED TRANSFER FUNCTION GAIN =  1.000000E 00

     COMPENSATED TRANSFER FUNCTION NUMERATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00     1.000000E 00

     COMPENSATED TRANSFER FUNCTION NUMERATOR ROOTS
 ARE:      REAL PART            IMAGINARY PART

         -1.000000E 00          0.0

     COMPENSATED TRANSFER FUNCTION DENOMINATOR
COEFFICIENTS IN ASCENDING POWERS OF S

   1.000000E 00     2.000000E 00     1.000000E 00

     COMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS
 ARE:      REAL PART            IMAGINARY PART

         -1.000000E 00          0.0

         -1.000000E 00          0.0
--------------------------------------------------------------------------------
```

Computer Numerical Output, Run #2

Figure V-5S

```
    THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUM PHASE
TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLOWED IN
THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION
-----------------------------------------------------------------------
-----------------------------------------------------------------------

    THE TOTAL NUMBER OF TRIALS CALLED FOR = 10000

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

    THE COST FUNCTION TO BE USED IS THE TYPE   2

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

THE MINIMUM COST FUNCTION VALUE =  4.723787E-01

THE ERROR RETURN CODE FROM BOXPLX =   2

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

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

  3.427396E 05    9.702760E 05

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR
 ROOTS ARE:      REAL PART          IMAGINARY PART

           -3.532414E-01        0.0
-----------------------------------------------------------------------

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 COEFFICIENTS IN ASCENDING POWERS OF S

  3.429794E 05    2.506541E 05    5.576800E 04

    OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOR
 ROOTS ARE:      REAL PART          IMAGINARY PART

           -2.241913E 00        1.060160E 00

           -2.241913E 00       -1.060160E 00
-----------------------------------------------------------------------
```

Computer Numerical Output, Run #2

Figure V-5T

OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = 1.739832E 01

FREQUENCY	MAGNITUDE(DB)	DESIRED MAG(DB)	PHASE(DEG)	DESIRED PHASE
9.999996E-02	3.197653E-01	3.416665E-01	1.162979E 01	1.030000E 01
1.270000E-01	5.074022E-01	5.876731E-01	1.447106E 01	1.270000E 01
1.610000E-01	7.908712E-01	8.278493E-01	1.777969E 01	1.510000E 01
2.040000E-01	1.266574E 00	1.289158E 00	2.149046E 01	1.829999E 01
2.589999E-01	1.812137E 00	1.868431E 00	2.544092E 01	2.109999E 01
3.290004E-01	2.669186E 00	2.606670E 00	2.924416E 01	2.350000E 01
4.180000E-01	3.639315E 00	3.521825E 00	3.238510E 01	2.529999E 01
5.300000E-01	4.861073E 00	4.608976E 00	3.427291E 01	2.559999E 01
6.720000E-01	6.231177E 00	5.808692E 00	3.446308E 01	2.459999E 01
8.520000E-01	7.678854E 00	7.272227E 00	3.230791E 01	2.229999E 01
1.080000E 00	9.081659E 00	9.004979E 00	2.771144E 01	1.620000E 01
1.370000E 00	1.034261E 01	1.083158E 01	2.036575E 01	4.099999E 00
1.740000E 00	1.130731E 01	1.238186E 01	1.033685E 01	-1.609999E 01
2.209999E 00	1.181641E 01	1.304492E 01	-1.800491E 00	-4.529999E 01
2.809999E 00	1.176266E 01	1.236096E 01	-1.505461E 01	-8.450000E 01
3.559999E 00	1.114894E 01	1.004854E 01	-2.789549E 01	-1.330000E 02
4.520000E 00	1.005291E 01	7.747795E 00	-3.963762E 01	-2.120000E 02
5.740000E 00	8.604420E 00	8.399112E 00	-4.967783E 01	-2.860000E 02
7.280000E 00	6.930445E 00	9.127317E 00	-5.791054E 01	-3.390000E 02
9.240000E 00	5.102574E 00	8.265992E 00	-6.458289E 01	-3.770000E 02
1.170000E 01	3.201303E 00	6.107025E 00	-6.986574E 01	-4.050000E 02
1.490000E 01	1.199996E 00	3.227359E 00	-7.416071E 01	-4.230000E 02
1.869999E 01	-8.126724E-01	1.719992E-01	-7.749942E 01	-4.340000E 02
2.400000E 01	-2.852070E 00	-2.721653E 00	-8.014937E 01	-4.390000E 02
3.029999E 01	-4.883443E 00	-5.256147E 00	-8.222666E 01	-4.420000E 02
3.859999E 01	-6.943992E 00	-7.535015E 00	-8.387128E 01	-4.430000E 02
4.869999E 01	-8.989978E 00	-9.739044E 00	-8.516147E 01	-4.440000E 02
6.209999E 01	-1.106032E 01	-1.180132E 01	-8.618961E 01	-4.450000E 02
7.879999E 01	-1.312574E 01	-1.335009E 01	-8.699693E 01	-4.460000E 02
1.000000E 02	-1.519318E 01	-1.591760E 01	-8.763351E 01	-4.470000E 02

THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES
THAT THE SYSTEM IS STABLE

Computer Numerical Output, Run #2
Figure V-5U

VI. CONCLUSION AND SUGGESTED FUTURE STUDIES

The investigations carried out in this thesis illustrate the feasibility of automating the classical design of compensators by specifying a set of desired values for the magnitude and phase response of the open loop system at a number of discrete frequency values over the range of interest. The results achieved show that the complex minimization method of M. J. Box is quite capable of handling the constrained minimization problem and conveniently provides a means of avoiding the numerical differentiation of the cost function, required with gradient search techniques. The incorporation of stability criterion in the implicit constraints of the minimization algorithm also insures that the program will not design an unstable compensator. By specifying the desired response in terms of the magnitude and phase profiles, common frequency domain specifications such as open loop bandwidth, phase margin, and gain margin have been incorporated into one common format and the necessity for specialized routines to check these specifications have been avoided. Those example problems presented for which known results had been established by other techniques show that the algorithm presented provides accuracy that is certainly within the necessary tolerances for most engineering applications of compensator design.

Also, while not specifically designed for the purpose, the basic premise of the algorithm's minimizing the difference between some desired frequency response and the frequency response of the system as calculated by the computer program while the transfer function parameters are varied has been shown to be effective in both determining a transfer function from measured frequency response data and in forming low order models of known higher order linear systems. In the formulation of low order models the choice of cost function and the frequency range specified may be used to emphasize different aspects of the frequency response which the low order model is to approximate.

While this work has shown that the basic algorithm presented can be used to effectively design series compensators based on frequency response specifications, considerable work remains to be done in this area. Specifically, an option to select that only real poles and zeros be considered as viable solutions to the compensator design algorithm would not be a difficult addition to the program and should prove quite useful. Also, the incorporation of a provision for handling pure time delays would not require significant modification of the program coding and would extend the capability of the algorithm to the solution of problems involving transport delays. Investigation of the method presented should be further pursued in terms of multiloop systems and systems involving compensation in the feedback loop. It may also prove worthwhile to investigate extending this method to nonlinear components using describing function techniques.

In the areas of transfer function synthesis and reduced order modeling, a more detailed investigation of both the effects of different cost functions and considering different frequency ranges should be conducted. In addition, it appears feasible to weight the cost function over a particular frequency range by varying the density of the discrete frequencies at which the desired magnitude and phase profiles are specified. Time did not permit a detailed investigation of this idea and further investigation into this area may be applicable to the program utilization in both the areas of compensator design and transfer function synthesis.

APPENDIX A

DESCRIPTION OF SUBROUTINES

In this section a brief description and background of various subroutines is presented to aid the interested reader in better understanding the internal logic of the program. Also these descriptions are intended to facilitate the task of anyone desiring to make modifications or additions to the program in order to handle systems of a configuration other than the unity feedback form assumed throughout this work.

SUBROUTINE CONORM

The purpose of this subroutine sub-program is to normalize the coefficient of the highest order term to unity. This is accomplished in a simple, straight forward manner by dividing all the coefficients of the polynomial by the coefficient of the highest order term. Thus a polynomial given by

$$P(s) = \sum_{i=0}^{n} a_i \cdot s^i$$

becomes on return from this subroutine

$$P(s) = \sum_{i=0}^{n} \frac{a_i}{a_n} \cdot s^i$$

where a_n is the coefficient of the s^n term. The normalizing factor a_n is also returned to the calling program.

It should be noted that the coefficients of the normalized polynomial are returned under the same variable name as the input coefficients.

Definitions of the input and output parameters are:

Input Variables

V ----------- A real one-dimensional array containing the coefficients of the polynomial whose highest order coefficient is to be normalized to unity. Upon return from the subroutine this array will contain the normalized coefficients of the polynomial.

N ----------- The dimension of the array containing the coefficients to be normalized.

VNORM ------- The normalizing factor, returned to the calling program, by which all coefficients have been divided.

SUBROUTINE PHASE

This subroutine sub-program is used to calculate the phase angle of a polynomial at a particular frequency. In order to circumvent the difficulty associated with standard arctangent routines, which return values limited to the range $-\pi \leq \theta \geq \pi$, the phase contributions from the individual factors are calculated and then summed to form the total phase angle at a particular frequency. That is, if the polynomial is factored and represented as the product of first order terms of the form

$$P(s) = \prod_{i=0}^{n} (s + z_i)$$

where in general z_i may be a complex number of the form $z_i = a_i + jb_i$, then at a particular frequency $s = j\omega$ the phase contribution of the individual factors may be evaluated as

$$\phi_i = \tan^{-1} \frac{(b_i + \omega)}{a_i}$$

and the total phase angle at the particular frequency is the sum of these individual phase contributions. The phase contribution of each of these first order terms will be within the principal values of the arctangent routine of the computer and the problems generally associated with determination of the phase for higher than first order terms are avoided.

Definitions of the input and output parameters are:

Input Variables

RR ---------- A real one-dimensional array containing the real parts of the roots of the polynomial.

RI ---------- A real one-dimensional array containing the imaginary parts of the roots of the polynomial.

N ----------- An integer value specifying the number of roots of the polynomial.

OMEGA ------- The frequency in radians/sec at which the value of the phase is to be computed.

Output Variables

TANGD ------- The total phase contribution of the polynomial given in degrees.

SUBROUTINE PINVRT

The purpose of this subroutine sub-program is to rearrange the coefficients of a polynomial in inverse order from that fed into the subroutine. Consider for example a polynomial arranged in descending powers of s

$$P(s) = a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0$$

Upon return from this subroutine the coefficients would be arranged in ascending powers of s. That is the polynomial would be arranged as follows

$$P(s) = a_0 + a_1 s + \ldots + a_{n-1} s^{n-1} + a_n s^n$$

While both representations are mathematically equivalent the sequence of the coefficients must be taken into account when using various polynomial manipulation techniques within the computer. The method used in reversing the sequence of the input coefficients consists of temporarily storing the first coefficient and sequentially shifting the remaining coefficients. The coefficient in temporary storage is then placed in the last coefficient location. This process is continued with the use of nested do loops and each time the shifting operation occurs the coefficient in temporary storage is placed one location lower than the previous number retrieved from temporary storage.

This subroutine requires no other subroutines or functions to carray out the procedure. The size of the one dimensional array of coefficients that may be rearranged is determined by the dimension of the array in the calling program.

Definitions of the input and output parameters are:

Input Variables

X(I) -------- A real one dimensional array containing the coefficients to be rearranged. Upon return from the subroutine this array contains the input coefficients in reverse sequence from that entered.

N ----------- The dimension of the array containing the coefficients to be rearranged.

SUBROUTINE PMPY

The purpose of this subroutine sub-program is to multiply the coefficients of one polynomial X(s) times the coefficients of another polynomial Y(s) to obtain the coefficients of the resulting polynomial Z(s). That is, if it is desired to multiply two polynomials of the form

$$P_1(s) = a_0 + a_1 s + a_2 s^2 + \ldots + a_{n-1} s^{n-1} + a_n s^n$$

and

$$P_2(s) = b_0 + b_1 s + b_2 s^2 + \ldots + b_{m-1} s^{m-1} + b_m s^m$$

together in order to obtain a third resultant polynomial of the general form

$$P_3(s) = c_0 + c_1 s + c_2 s^2 + \ldots + c_{i-1} s^{i-1} + c_i s^i$$

this subroutine is used in accomplishing this. The order of the resulting polynomial is calculated by forming the sum of the orders of the two input polynomials as

$$i = n + m$$

The various coefficients of the resulting polynomial, which are returned in ascending powers of the independent variable, are calculated as the sums of products of the input coefficients by means of two nested do-loops. The arrangement of the two do-loops is such that the appropriate product terms are summed in forming the respective coefficients of the resultant polynomial. All input and output coefficients must be real numbers. The size of the polynomials that may be multiplied are controlled by their respective dimension statements within the calling program.

Definitions of the input and output parameters are:

Input Variables

X ----------- A real one dimensional array containing the coefficients of the first polynomial to be multiplied. These must be arranged in ascending powers of the independent variable.

IDIMX ------- The dimension of the array X containing the coefficients of the first polynomial.

Y ----------- A real one dimensional array containing the coefficients of the second polynomial to be multiplied. These must be arranged in ascending powers of the independent variable.

IDIMY ------- The dimension of the array Y containing the coefficients of the second polynomial.

Output Variables

Z ----------- A real one dimensional array containing the coefficients of the resultant polynomial. These coefficients are arranged in ascending powers of the independent variable.

IDIMZ ------- The calculated dimension of the array Z containing the coefficients of the resultant polynomial.

SUBROUTINE PVAL

The purpose of this subroutine sub-program is to evaluate a polynomial having the complex variable $s = \sigma + j\omega$ and real coefficients a_i. For a given polynomial of order n of the form

$$P(s) = \sum_{i=0}^{n} a_i \bullet s^i$$

s is declared as a complex variable within the subroutine and the indicated operations of summation and multiplication are carried out by a simple loop operation taking advantage of the associative and distributive laws of multiplication and addition. This subroutine requires no other sub-programs to perform its function.

Definitions of the input and output parameters are:

Input Variables

A(I) -------- A real one-dimensional array containing the coefficients of the polynomial that is to be evaluated. The coefficients are assumed to be arranged in ascending order according to the powers of s with zero explicitly input for the coefficient of any missing powers of s occuring in the polynomial. The maximum size of this array is controlled by the dimension of the array within the main program. The values of the coefficients in this array remain unchanged after execution of the subroutine.

NN ---------- The order of the polynomial to be evaluated. This value is one less than the dimension of A(I).

PR ---------- The value of the real part of the complex variable s at which the polynomial is to be evaluated.

PI ---------- The value of the imaginary part of the complex variable s at which the polynomial is to be evaluated.

Output Variables

VR ---------- The real part of the complex resultant value of the polynomial evaluated at $s = \sigma + j\omega$.

VI ---------- The imaginary part of the complex resultant value of the polynomial evaluated at $s = \sigma + j\omega$.

SUBROUTINE ROUTH

The purpose of this subroutine sub-program is to determine if a given input polynomial possesses right half plane roots by using the Routh stability criterion. The polynomial coefficients are input in descending powers of the independent variable. From these coefficients the Routh array is calculated and the first column of this array is searched for an indication of the existence of right half plane roots. Thus from an input polynomial of the form

$$P(s) = b_n s^n + b_{n-1} s^{n-1} + \dots + b_1 s + b_0$$

the Routh array

1^{st} row	b_n	b_{n-2}	b_{n-4}	......
2^{nd} row	b_{n-1}	b_{n-3}	b_{n-5}	......
3^{rd} row	a_{31}	a_{32}	a_{33}	....
4^{th} row	a_{41}	a_{42}	a_{43}	....
	$\vdots$	$\vdots$	$\vdots$	
n^{th} row	a_{n1}	a_{n2}	a_{n3}	....
$(n+1)^{st}$ row	$a_{(n+1)1}$	$a_{(n+1)2}$	$a_{(n+1)3}$	...

is formed, where in general the entries from the third row onward are given by

$$a_{ri} = \frac{\left[a_{(n-1)\,1}\right]\left[a_{(n-2)\,(i+1)}\right] - \left[a_{(n-2)\,1}\right]\left[a_{(n-1)\,(i+1)}\right]}{a_{(n-1)\,1}}$$

The first column of this array is then tested to determine if any of the elements are less than zero. Note that since all coefficients are limited to positive values by the search area restrictions in other parts of the program no provision has been made to handle the case of the coefficients of the polynomial being all negative. Thus a polynomial with all negative coefficients should be multiplied by -1.0 prior to using this particular subroutine for testing for right half plane roots.

Definitions of input and output parameters are:

Input Variables

Y ----------- A real one dimensional array containing the coefficients of the polynomial in descending powers of the independent variable.

N ----------- The dimension of the array Y.

Output Variables

ISTABL ------ Integer output indicating the absence or presence of right half plane roots

0---Indicates the polynomial has no right half plane roots.

1---Indicates the polynomial has right half plane roots.

SUBROUTINE SEMBL

The purpose of this subroutine sub-program is to calculate the coefficients of a polynomial from the roots of the polynomial. The calculation is done using complex arithmetic in order to be able to compute the polynomial coefficients when complex roots are present. The resulting polynomial coefficients are, however assumed to be real. Thus, if complex roots are present they must be in conjugate pairs. Also the coefficient of the highest order term is set equal to unity. Consider a polynomial of degree n having n roots. This may be expressed as

$$P(s) = \prod_{i=0}^{n} (s + z_i)$$

where in general the z_i's may be complex. The coefficients of the polynomial that results when the above indicated multiplication is performed may be computed by the formula

$$A_i = (-1)^{n-i} \cdot \sum (\text{product of roots taken n-i at a time})$$

Thus the coefficients of the polynomial are given by

$$a_0 = z_1 \cdot z_2 \cdot z_3 \cdot z_4 \cdot \ldots \cdot z_n$$

$$a_1 = z_1 \cdot z_2 \cdot z_3 \cdot \ldots \cdot z_{n-1} + z_2 \cdot z_3 \cdot z_4 \cdot \ldots \cdot z_n$$

$$\vdots$$

$$a_{n-1} = z_1 + z_2 + z_3 + z_4 + z_5 + \ldots + z_n$$

$$a_n = 1$$

The largest order polynomial that this particular subroutine will handle is a 20th order. If larger polynomials are to be considered the dimension of R within the subroutine must be changed.

Definitions of the input and output parameters are:

Input Variables

RR ---------- A real one-dimensional array containing the real parts of the polynomial roots.

RI ---------- A real one-dimensional array containing the imaginary parts of the polynomial roots.

N ----------- An integer specifying the order of the polynomial or equivalently the dimension of the arrays containing the real and imaginary parts of the roots.

Output Variables

CF ---------- A real one-dimensional array containing the resulting coefficients of the polynomial arranged in ascending powers of the independent variable. The dimension of this array must be one larger than the dimensions of the arrays containing the real and imaginary parts of the roots.

APPENDIX B

PROGRAM LISTING

```
C COMPUTER AIDED COMPENSATOR OPTIMIZATION PROGRAM
C
C
      REAL *4NTITLE
      DIMENSION CO(20), RTR(20), RTI(20), X(24), C(12), D(12), CCF(20),
     1XS(24), XL(27), XU(24)
      COMMON NCN,NCD,NTFN,NTFD,NOMEG,A(20),B(20),CMAG(500),DQPHSD(500),W
     1FREQ(500),NSN,NSD,COFASN(40),COFASD(40),QMAG(500),CR(500),CI(500),
     2CPHSR(500),QPHSD(500),N1,N2,N3,N4,PI,T,WS,IZCH,ADMAG(500),NTITLE(1
     32),FREQ(500),AMAG(500),NMINFS,COFWRK(40),IPLOT,WFREQS(500),SAPHSD(
     4500),SAMAG(500),DIFF1(500),DIFF2(500),ICOST,CFWRK1(40)
      DATA IPP/1HP/
      PI = 3.1415926
      PHICHK = 0.0
      IDELNU = 0
      IDELDE = 0
      PHI = 0.0
      IPHI = 0
      PSICHK = 0.0
      PSI = 0.0
      IPSI = 0
      IEREXP = 0
C
C   READ IN PROBLEM TITLE
C
      READ (5,60,END=59) (NTITLE(I),I=1,12)
      WRITE (6,61) NTITLE
      WRITE (6,62)
      READ (5,64) GAIN
      WRITE (6,65) GAIN
      KOUNT = 0
    1 KOUNT = KOUNT+1
C
C   BEGIN SECTION TO READ IN COEFFICIENTS IN FACTORED OR POLYNOMIAL
C   FORM.  IF POLYNOMIAL FORM IS USED COEFFICIENT OF HIGHEST
C   ORDER TERM IS ASSUMED TO BE UNITY, AND ROOTS ARE CALCULATED.
C
      READ (5,66) INFORM,N
      NP = N+1
      IF (N.EQ.0) GO TO 6
      IF (INFORM.EQ.IPP) GO TO 5
      I = 0
    2 I = I+1
      READ (5,64) RR,RI
      RTR(I) = -RR
      RTI(I) = RI
C
C   IF ROOTS OCCUR IN COMPLEX CONJUGATE PAIRS THE REAL AND IMAGINARY
```

```
C     VALUES OF THE FACTORS NEED BE ENTERED ONLY ONCE
C
      IF (RI) 3,4,3
    3 I = I+1
      RTR(I) = -RR
      RTI(I) = -RI
    4 IF (I.LT.N) GO TO 2
      CALL SEMBL (N,RTR,RTI,CO)
      GO TO 8
    5 READ (5,64) (CO(I),I=1,N)
    6 CO(NP) = 1.0
      IF (N) 8,8,7
    7 CALL PRCD (CO,NP,RTR,RTI,COF,NRCCT1,IERRT1)
      WRITE (6,67) NROOT1,IERRT1
      IF (IERRT1.EQ.0) GO TO 8
      CALL PRBM (CO,NP,RTR,RTI,COF,NRCCTA,IERRTA)
      WRITE (6,62)
      WRITE (6,63)
      WRITE (6,68) NROOTA,IERRTA
      WRITE (6,63)
      WRITE (6,62)
    8 GO TO (9,12,16,19), KOUNT
C
C     MULTIPLY PLANT NUMERATOR COEFFICIENTS BY PLANT GAIN AND LOAD
C     PLANT NUMERATOR ARRAY.
C
    9 DO 10 I=1,NP
      A(I) = CO(I)*GAIN
   10 CONTINUE
C
      NTFN = N
      N3 = NP
      WRITE (6,69)
C
C     WRITE PLANT NUMERATOR COEFFICIENTS
C
      WRITE (6,70) (CO(I),I=1,NP)
      IF (N.LE.0) GO TO 1
      WRITE (6,71)
C
C     WRITE PLANT NUMERATOR ROOTS.
C
      DO 11 I=1,N
      WRITE (6,72) RTR(I),RTI(I)
   11 CONTINUE
C
      GO TO 1
C
```

```
C     LOAD PLANT DENOMINATOR ARRAY WITH COEFFICIENTS.
C
   12 DO 13 I=1,NP
      E(I) = CO(I)
   13 CONTINUE
C
      WRITE (6,73)
C
C     WRITE COEFFICIENTS OF PLANT DENOMINATOR.
C
      WRITE (6,70) (B(I),I=1,NP)
      IF (N.LE.0) GO TO 15
      WRITE (6,74)
C
C     WRITE ROOTS OF PLANT DENOMINATOR.
C
      DO 14 I=1,N
      WRITE (6,72) RTR(I),RTI(I)
   14 CONTINUE
C
   15 NTFD = N
      N4 = NP
      WRITE (6,62)
C
C     READ IN ASSUMED COMPENSATOR GAIN.
C
      READ (5,64) GAINCO
      WRITE (6,75) GAINCO
      GO TO 1
C
C     MULTIPLY COMPENSATOR NUMERATOR COEFFICIENTS BY COMPENSATOR GAIN.
C
   16 DO 17 I=1,NP
      C(I) = CO(I)*GAINCO
   17 CONTINUE
C
      NCN = N
      N1 = NP
      WRITE (6,76)
C
C     WRITE ASSUMED COMPENSATOR NUMERATOR COEFFICIENTS
C
      WRITE (6,70) (CO(I),I=1,NP)
      IF (N.LE.0) GO TO 1
      WRITE (6,77)
C
C    WRITE ROOTS OF ASSUMED COMPENSATOR NUMERATOR.
C
```

```
      DO 18 I=1,N
      WRITE (6,72) RTR(I),RTI(I)
   18 CONTINUE
C
      GO TO 1
C
C     LOAD COMPENSATOR DENOMINATOR COEFFICIENT ARRAY WITH ASSUMED STARTING
C     VALUES.
C
   19 DO 20 I=1,NP
      D(I) = CO(I)
   20 CONTINUE
C
      WRITE (6,78)
C
C     WRITE COEFFICIENTS OF ASSUMED COMPENSATOR DENOMINATOR POLYNOMIAL.
C
      WRITE (6,70) (D(I),I=1,NP)
      WRITE (6,79)
C
C     WRITE ROOTS OF ASSUMED COMPENSATOR DENOMINATOR POLYNOMIAL.
C
      DO 21 I=1,N
      WRITE (6,72) RTR(I),RTI(I)
   21 CONTINUE
C
      NCD = N
      N2 = NP
      WRITE (6,62)
C
C     READ IN PROGRAM CONTROL DATA.
C
      READ (5,80) WMIN,WMAX,NOMEG,KNOW,NBODE,NYQST,IZOH,NICHOL,NMINFS,IC
     1B,IPLOT,ICOST
      WRITE (6,62)
      IF (NMINFS.EQ.0) WRITE (6,81)
      IF (NMINFS.EQ.1) WRITE (6,82)
      WRITE (6,62)
      IF (IZOH.NE.1) GO TO 22
      READ (5,64) T
      WS = (2.0*PI)/T
      WRITE (6,62)
      WRITE (6,83) T,WS
      WRITE (6,62)
   22 KNOW = KNOW+1
      WFREQ(1) = WMIN
      IF (NOMEG-2) 31,31,23
   23 XOMEG = NOMEG
```

```
      GO TO (29,26,24), KNOW
   24 DWFQ = (WMAX-WMIN)/(XOMEG-1.0)
C
      DO 25 I=2,NOMEG
   25 WFREQ(I) = WFREQ(I-1)+DWFQ
C
C
C     READ DISCRETE FREQUENCY VALUES AND DESIRED GAIN AND PHASE.
      GO TO 31
   26 READ (5,64) (WFREQ(I),I=1,NOMEG)
      READ (5,64) (DMAG(I),I=1,NOMEG)
      IF (IDB.NE.1) GO TO 28
C
C     IF DESIRED GAIN IS READ IN IN DB CONVERT TO A MAGNITUDE.
C
      DO 27 I=1,NOMEG
      DMAG(I) = DMAG(I)/20.0
      DMAG(I) = 10.0**DMAG(I)
   27 CONTINUE
C
   28 READ (5,64) (DQPHSD(I),I=1,NOMEG)
      GO TO 31
   29 Y = (ALOG10(WMAX)-ALOG10(WMIN))/(XOMEG-1.0)
      Z = 10.0**Y
C
      DO 30 I=2,NOMEG
      J = I-1
   30 WFREQ(I) = WFREQ(J)*Z
C
   31 R = 9./13.
C
C     SET UP INITIAL GUESS, UPPER, & LOWER BOUNDS FOR SEARCH AREA.
C
      DO 32 J=1,N1
      XS(J) = C(J)
      XU(J) = 1.E+06
      XL(J) = 0.0
      IF (NMINFS.EQ.1) XL(J)=-XU(J)
   32 CONTINUE
C
C
      DO 33 J=1,N2
      XS(N1+J) = D(J)
      XU(N1+J) = 1.E+06
      XL(N1+J) = 0.0
   33 CONTINUE
C
      READ (5,84) NTA,NPR
```

```
      WRITE (6,62)
      WRITE (6,85) NTA
      WRITE (6,62)
      WRITE (6,62)
      WRITE (6,86) ICOST
      WRITE (6,62)
      NAV = 0
      NV = N1+N2
      IF = 0
C
C   CALL OPTIMIZATION ROUTINE.
C
      CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMIN,IERBXP)
      WRITE (6,62)
      WRITE (6,87)
      WRITE (6,70) (XS(I),I=1,NV)
      WRITE (6,88) YMIN,IERBXP
      WRITE (6,62)
      WRITE (6,62)
C
C   LOAD COMPENSATOR NUMERATOR ARRAY WITH THE OPTIMIZED POLYNOMIAL
C   COEFFICIENTS.
C
      DO 34 I=1,N1
      C(I) = XS(I)
   34 CONTINUE
C
      WRITE (6,89)
C
C   WRITE THE OPTIMIZED COMPENSATOR NUMERATOR COEFFICIENTS IN
C   UN-NORMALIZED FORM.
C
      WRITE (6,70) (C(I),I=1,N1)
C
      DO 35 I=1,N1
      CO(I) = C(I)/C(N1)
   35 CONTINUE
C
      IF (NCN.EQ.0) GO TO 37
C
C   COMPUTE THE ROOTS OF THE OPTIMIZED COMPENSATOR NUMERATOR.
C
      CALL PRQD (CO,N1,RTR,RTI,COF,NROOT2,IERRT2)
      IF (IERRT2.EQ.0) GO TO 36
      CALL PRBM (CO,N1,RTR,RTI,COF,NROOTB,IERRTB)
      WRITE (6,62)
      WRITE (6,63)
      WRITE (6,90) NROOTB,IERRTB
```

```
      WRITE (6,63)
      WRITE (6,62)
   36 WRITE (6,91)
C
C     WRITE THE ROOTS OF THE OPTIMIZED COMPENSATOR NUMERATOR.
C
      DO 37 I=1,NCN
      WRITE (6,92) RTR(I),RTI(I)
   37 CONTINUE
C
      WRITE (6,62)
C
C     LOAD COMPENSATOR DENOMINATOR ARRAY WITH THE OPTIMIZED POLYNOMIAL
C     COEFFICIENTS.
C
      DO 38 I=1,N2
      D(I) = XS(N1+I)
   38 CONTINUE
C
      WRITE (6,93)
C
C     WRITE OPTIMIZED DENOMINATOR COEFFICIENTS IN UN-NORMALIZED FORM.
C
      WRITE (6,70) (D(I),I=1,N2)
C
      DO 39 I=1,N2
      CO(I) = D(I)/D(N2)
   39 CONTINUE
C
      IF (NCC.EQ.0) GO TO 41
C
C     COMPUTE THE ROOTS OF THE OPTIMIZED COMPENSATOR DENOMINATOR.
C
      CALL PRCD (CO,N2,RTR,RTI,COF,NROOT3,IERRT3)
      IF (IERRT3.EQ.0) GO TO 40
      CALL PRBM (CO,N2,RTR,RTI,COF,NROOTC,IERRTC)
      WRITE (6,62)
      WRITE (6,63)
      WRITE (6,94) NROOTC,IERRTC
      WRITE (6,63)
      WRITE (6,62)
   40 WRITE (6,95)
C
C     WRITE THE ROOTS OF THE OPTIMIZED COMPENSATOR DENOMINATOR.
C
      DO 41 I=1,NCD
      WRITE (6,92) RTR(I),RTI(I)
   41 CONTINUE
```

```
C
      WRITE (6,62)
C
C     COMPUTE AND WRITE THE OPTIMIZED COMPENSATOR GAIN.
C
      GAINC = C(N1)/D(N2)
      WRITE (6,96) GAINC
      WRITE (6,97) NROOT2,IERRT2,NROOT3,IERRT3
      CALL PMPY (COFASN,N7,A,N3,C,N1)
      CALL PMPY (COFASD,N8,B,N4,D,N2)
      NSN = N7-1
      NSD = N8-1
C
C     SET UP AND EVALUATE SYSTEM OPEN LOOP FREQUENCY RESPONSE AT THE
C     DISCRETE FREQUENCIES SPECIFIED USING THE OPTIMIZED COMPENSATOR
C     COEFFICIENT VALUES.
C
      DO 52 K=1,NOMEG
      CALL PVAL (COFASN,NSN,0.0,WFREQ(K),QNUMR,QNUMI)
      CALL PVAL (COFASD,NSD,0.0,WFREQ(K),QDENR,QDENI)
      IF (SQRT(QDENI**2+QDENR**2)) 43,42,43
   42 QMAG(K) = 1.E+20
      QR(K) = 1.E+20
      QI(K) = 1.E+20
      GO TO 44
   43 QMAG(K) = SQRT(QNUMI**2+QNUMR**2)/SQRT(QDENI**2+QDENR**2)
      IF (IZOH.EQ.1) QMAG(K)=QMAG(K)*ABS((SIN(PI*WFREQ(K)/WS))/(PI*WFREQ
     1(K)/WS))
      QR(K) = (QNUMR*QDENR+QNUMI*QDENI)/(QDENR**2+QDENI**2)
      QI(K) = (QNUMI*QDENR-QNUMR*QDENI)/(QDENR**2+QDENI**2)
   44 ANGNUR = ATAN2(QNUMI,QNUMR)
      ANGNUD = ANGNUR*57.29578
      IF (ANGNUD.LT.0.0) ANGNUD=360.0+ANGNUD
      DELPHI = ANGNUD-PHI
      IF (IDELNU.EQ.0) GO TO 47
      IF (DELPHI.GE.0.0.AND.IDELNU.EQ.1) GO TO 47
      IF (DELPHI.LT.0.0.AND.IDELNU.EQ.1) GO TO 45
      IF (DELPHI.GE.0.0.AND.IDELNU.EQ.-1) GO TO 46
      IF (DELPHI.LT.0.0.AND.IDELNU.EQ.-1) GO TO 47
   45 IF (ABS(DELPHI).LT.180.0) GO TO 47
      DELPHI = ANGNUD+360.0-PHI
      GO TO 47
   46 IF (ABS(DELPHI).LT.180.0) GO TO 47
      DELPHI = ANGNUD-360.0-PHI
   47 PHICHK = PHICHK+DELPHI
      IPHI = PHICHK/360.0
      PHI = ANGNUD
      ANGNUD = ANGNUD+IPHI*360.0
```

```
      ANGDER = ATAN2(QDENI,QDENR)                                       MAIN3850
      ANGDED = ANGDER*57.29578                                          MAIN3860
      IF (ANGDED.LT.0.0) ANGDED=360.0+ANGDED                            MAIN3870
      DELPSI = ANGDED-PSI                                               MAIN3880
      IF (IDELDE.EQ.0) GO TO 50                                         MAIN3890
      IF (DELPSI.GE.0.0.AND.IDELDE.EQ.1) GO TO 50                       MAIN3900
      IF (DELPSI.LT.0.0.AND.IDELDE.EQ.1) GO TO 48                       MAIN3910
      IF (DELPSI.GE.0.0.AND.IDELDE.EQ.-1) GO TO 49                      MAIN3920
      IF (DELPSI.LT.0.0.AND.IDELDE.EQ.-1) GO TO 50                      MAIN3930
   48 IF (ABS(DELPSI).LT.180.0) GO TO 50                                MAIN3940
      DELPSI = ANGDED+360.0-PSI                                         MAIN3950
      GO TO 50                                                          MAIN3960
   49 IF (ABS(DELPSI).LT.180.0) GO TO 50                                MAIN3970
      DELPSI = ANGDED-360.0-PSI                                         MAIN3980
   50 PSICHK = PSICHK+DELPSI                                            MAIN3990
      IPSI = PSICHK/360.0                                               MAIN4000
      PSI = ANGDED                                                      MAIN4010
      ANGDED = ANGDED+IPSI*360.0                                        MAIN4020
      IF (DELPHI.GE.0.0) IDELNU=1                                       MAIN4030
      IF (DELPHI.LT.0.0) IDELNU=-1                                      MAIN4040
      IF (DELPSI.GE.0.0) IDELDE=1                                       MAIN4050
      IF (DELPSI.LT.0.0) IDELDE=-1                                      MAIN4060
      QPHSD(K) = ANGNUD-ANGDED                                          MAIN4070
      QPHSR(K) = ANGNUR-ANGDER                                          MAIN4080
      IF (IZOH.NE.1) GO TO 51                                           MAIN4090
      IWS = WFREQ(K)/WS                                                 MAIN4100
      QPHSD(K) = QPHSD(K)-(PI*WFREQ(K)/WS)*57.29578-IWS*180.0           MAIN4110
C                                                                       MAIN4120
C     CALCULATE THE DIFFERENCE BETWEEN THE ACTUAL AND DESIRED RESPONSE AT MAIN4130
C     THE DISCRETE FREQUENCY VALUES ORIGINALLY SPECIFIED.               MAIN4140
C                                                                       MAIN4150
   51 DIFF1(K) = (20.0*ALOG10(QMAG(K)))-(20.0*ALOG10(DMAG(K)))          MAIN4160
      DIFF2(K) = QPHSD(K)-DQPHSD(K)                                     MAIN4170
   52 CONTINUE                                                          MAIN4180
C                                                                       MAIN4190
      CALL SIMULA (WMIN,WMAX)                                           MAIN4200
      IF (NBODE) 55,53,55                                               MAIN4210
C                                                                       MAIN4220
C     PREPARE DISCRETE FREQUENCY POINT DATA FOR GRAPHING.               MAIN4230
C                                                                       MAIN4240
   53 DO 54 MA=1,NOMEG                                                  MAIN4250
      FREQ(MA) = ALOG10(WFREQ(MA))                                      MAIN4260
   54 CONTINUE                                                          MAIN4270
C                                                                       MAIN4280
   55 IF (NBODE.EQ.1.AND.NICHOL.EQ.1) GO TO 57                          MAIN4290
C                                                                       MAIN4300
      DO 56 MA=1,NOMEG                                                  MAIN4310
      AMAG(MA) = 20.0*ALOG10(QMAG(MA))                                  MAIN4320
```

```
      ADMAG(MA) = 20.0*ALOG10(DMAG(MA))
   56 CONTINUE
C
C
C  OUTPUT RESULTS ON GRAPHS SPECIFIED.
C
   57 CALL GRAPH (NBODE,NYQST,NICHOL)
      WRITE (6,98)
      WRITE (6,99)
C
      DO 58 IJK=1,NOMEG
      WRITE (6,100) WFREQ(IJK),AMAG(IJK),ADMAG(IJK),CPHSD(IJK),DCPHSD(IJ
     1K),DIFF1(IJK),DIFF2(IJK)
   58 CONTINUE
C
C  SYSTEM STABILITY CHECK USING ROUTH CRITERION ON CHARACTERISTIC EQUA.
C
      CALL PADD (COFWRK,IWRK,COFASN,N7,COFASD,N8)
      CALL PINVRT (COFWRK,IWRK)
      CALL ROUTH (COFWRK,IWRK,ISYS)
      CALL PINVRT (COFWRK,IWRK)
      IF (ISYS.EQ.1) WRITE (6,101)
      IF (ISYS.EQ.0) WRITE (6,102)
   59 STOP
C
   60 FORMAT (12A4)
   61 FORMAT ('1',6X,'TITLE --- ',12A4)
   62 FORMAT (' ',72('-'))
   63 FORMAT (' ',8('*WARNING*'))
   64 FORMAT (8E10.0)
   65 FORMAT ('0',6X,'UNCOMPENSATED TRANSFER FUNCTION GAIN = ',1PE13.6)
   66 FORMAT (A1,I2)
   67 FORMAT ('0','NROOT1 = ',I3,10X,'IERRT1 = ',I3)
   68 FORMAT ('0','THE FIRST CALL TO PRQD IN THE INPUT SECTION RESULTED'
     1,/,' IN AN UNSATISFACTORY DETERMINATION OF POLYNOMIAL ROOTS.',/,
     2 ' SUBROUTINE PRBM WAS CALLED TO TRY RESOLVING PROBLEM.',/,' NROOT
     3A = ',I3,10X,'IERRTA = ',I3)
   69 FORMAT ('0',6X,'UNCOMPENSATED TRANSFER FUNCTION NUMERATOR',/,2X,
     1 'COEFFICIENTS IN ASCENDING POWERS OF S',/)
   70 FORMAT ('0',5(3X,1PE13.6))
   71 FORMAT ('0',6X,'UNCOMPENSATED TRANSFER FUNCTION NUMERATOR ROOTS',
     1 /,3X,'ARE:',5X,'REAL PART',8X,'IMAGINARY PART')
   72 FORMAT ('0',9X,1PE13.6,6X,1PE13.6)
   73 FORMAT ('0',6X,'UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR',/,2X,
     1 'COEFFICIENTS IN ASCENDING POWERS OF S',/)
   74 FORMAT ('0',6X,'UNCOMPENSATED TRANSFER FUNCTION DENOMINATOR ROOTS'
     1,/,3X,'ARE:',5X,'REAL PART',8X,'IMAGINARY PART')
   75 FORMAT ('0',6X,'COMPENSATOR TRANSFER FUNCTION GAIN = ',1PE13.6)
```

```
76 FORMAT ('0',6X,'COMPENSATOR TRANSFER FUNCTION NUMERATOR',/2X,      MAIN4810
   1 'COEFFICIENTS IN ASCENDING POWERS OF S',/)                          MAIN4820
77 FORMAT ('0',6X,'COMPENSATOR TRANSFER FUNCTION NUMERATOR ROOTS',/,3MAIN4830
   1X'ARE:',5X,'REAL PART',8X,'IMAGINARY PART')                         MAIN4840
78 FORMAT ('0',6X,'COMPENSATOR TRANSFER FUNCTION DENOMINATOR',/2X,    MAIN4850
   1 'COEFFICIENTS IN ASCENDING POWERS OF S',/)                          MAIN4860
79 FORMAT ('0',6X,'COMPENSATOR TRANSFER FUNCTION DENOMINATOR ROOTS', MAIN4870
   1 /,3X,'ARE:',5X,'REAL PART',8X,'IMAGINARY PART')                    MAIN4880
80 FORMAT (2E10.0,10I3)                                                  MAIN4890
81 FORMAT ('0',5X,'THE COMPENSATOR TRANSFER FUNCTION IS OF THE MINIMUMAIN4900
   1M PHASE',/,' TYPE, THEREFORE NO RIGHT HALF PLANE ZEROS WILL BE ALLMAIN4910
   2OWED IN',/,' THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION',/MAIN4920
   3)                                                                    MAIN4930
82 FORMAT ('0',5X,'THE COMPENSATOR TRANSFER FUNCTION IS OF THE NON-MIMAIN4940
   1NIMUM',/,' PHASE TYPE, THEREFORE RIGHT HALF PLANE ZEROS WILL BE ALMAIN4950
   2LOWED IN',/,' THE SOLUTION FOR THE COMPENSATOR TRANSFER FUNCTION',MAIN4960
   3/)                                                                   MAIN4970
83 FORMAT ('0',5X,'THE SYSTEM HAS A ZERO ORDER HOLD IN THE LOOP.',//,MAIN4980
   1 ' THE SAMPLING PERIOD  T = ',1PE15.6,' (SECS)',//,                 MAIN4990
   2 ' THE SAMPLING FREQUENCY  WS = ',1PE15.6,' (RADIANS)')             MAIN5000
84 FORMAT (2I5)                                                          MAIN5010
85 FORMAT ('0',5X,'THE TOTAL NUMBER OF TRIALS CALLED FOR = ',I5,/)     MAIN5020
86 FORMAT ('0',5X,'THE COST FUNCTION TO BE USED IS THE TYPE',2X,I2,/)MAIN5030
87 FORMAT (///,' THE BEST VERTEX VALUES ARE:')                          MAIN5040
88 FORMAT (///,' THE MINIMUM COST FUNCTION VALUE = ',1PE13.6,//,' THEMAIN5050
   1 ERROR RETURN CODE FROM BOXPLX = ',I3,/)                            MAIN5060
89 FORMAT ('0',6X,'OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR'MAIN5070
   1,/,2X,'COEFFICIENTS IN ASCENDING POWERS OF S'/)                     MAIN5080
90 FORMAT ('0','THE CALL TO PRCD FOR FINDING OPTIMIZED COMPENSATOR',/MAIN5090
   1,' NUMERATOR ROOTS RESULTED IN UNSATISFACTORY SOLUTION. ',/,        MAIN5100
   2 ' SUBROUTINE PRBM WAS CALLED TO TRY RESOLVING PROBLEM.',/,' NROOTMAIN5110
   3B = ',I3,10X,'IERRTB = ',I3)                                         MAIN5120
91 FORMAT ('0',6X,'OPTIMIZED COMPENSATOR TRANSFER FUNCTION NUMERATOR'MAIN5130
   1,/,3X,'ROOTS ARE:',5X,'REAL PART',8X,'IMAGINARY PART')             MAIN5140
92 FORMAT ('0',15X,1PE13.6,6X,1PE13.6)                                  MAIN5150
93 FORMAT ('0',6X,'OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOMAIN5160
   1R',/,2X,'COEFFICIENTS IN ASCENDING POWERS OF S',/)                  MAIN5170
94 FORMAT ('0','THE CALL TO PRCD FOR FINDING OPTIMIZED COMPENSATOR',/MAIN5180
   1,' DENOMINATOR ROOTS RESULTED IN UNSATISFACTORY SOLUTION.',/,      MAIN5190
   2 ' SUBROUTINE PRBM WAS CALLED TO ATTEMPT RECTIFYING PROBLEM.',/,   MAIN5200
   3 ' NROOTC = ',I3,10X,'IERRTC = ',I3)                                MAIN5210
95 FORMAT ('0',6X,'OPTIMIZED COMPENSATOR TRANSFER FUNCTION DENOMINATOMAIN5220
   1R',/,3X,'ROOTS ARE:',5X,'REAL PART',8X,'IMAGINARY PART')           MAIN5230
96 FORMAT ('0',6X,'OPTIMIZED COMPENSATOR TRANSFER FUNCTION GAIN = ', MAIN5240
   1 1PE13.6)                                                            MAIN5250
97 FORMAT ('0','NROOT2 = ',I3,10X,'IERRT2 = ',I3,///,' ','NROOT3 = ',MAIN5260
   1I3,10X,'IERRT3 = ',I3)                                               MAIN5270
98 FORMAT ('1')                                                          MAIN5280
```

```
99 FORMAT (' ',3X,'FRECUENCY',7X,'MAGNITUDE(DB)',3X,'DESIRED MAG(DB)'MAIN5290
  1,2X,'PHASE(DEG)',6X,'DESIRED PHASE',4X,'MAG DIFF(DB)',4X,'PHASE DIMAIN5300
  2FF(DEG)')                                                           MAIN5310
100 FORMAT (' ',8(3X,1PE13.6))                                          MAIN5320
101 FORMAT ('0',72('-'),/,2(' ********* W A R N I N G ********* '),//,MAIN5330
  1' THE ROUTH TEST OF THE CHARACTERISTIC EQUATION INDICATES SYSTEM',MAIN5340
  2/,' INSTABILITY',//,2(' ********* W A R N I N G ********* '),/,' 'MAIN5350
  3,72('-'))                                                           MAIN5360
102 FORMAT ('0',72('-'),/,5X,' THE ROUTH TEST OF THE CHARACTERISTIC EQMAIN5370
  1UATION INDICATES',/,' THAT THE SYSTEM IS STABLE',/,' ',72('-'))    MAIN5380
    END                                                                MAIN5390
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPVAL
C++++++++++++++++++++          SUBROUTINE PVAL         +++++++++++++++++++++++CPVAL
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPVAL
C++++                                                              ++++CPVAL
C++++ PURPOSE: TO EVALUATE A POLYNOMIAL A(S) WITH REAL COEFFICIENTS +++CPVAL
C++++          FOR WHICH THE SUBSTITUTION S=PR + JRI HAS BEEN MADE  ++++CPVAL
C++++          FOR THE COMPLEX VARIABLE.                           ++++CPVAL
C++++                                                              ++++CPVAL
C++++ CALLING SEQUENCE:                                            ++++CPVAL
C++++ CALL PVAL(A,NN,PR,PI,VR,VI)                                  ++++CPVAL
C++++                                                              ++++CPVAL
C++++ DESCRIPTION OF PARAMETERS:                                   ++++CPVAL
C++++ A      = VECTOR OF REAL COEFFICIENTS, ORDERED FROM LOWEST TO ++++CPVAL
C++++          HIGHEST POWER OF THE INDEPENDENT VARIABLE.          ++++CPVAL
C++++ NN     = THE ORDER OF THE POLYNOMIAL.                        ++++CPVAL
C++++ PR     = THE REAL PART OF THE COMPLEX VARIABLE.              ++++CPVAL
C++++ PI     = THE IMAGINARY PART OF THE COMPLEX VARIABLE.         ++++CPVAL
C++++ VR     = THE REAL PART OF THE RESULTANT VALUE OF THE         ++++CPVAL
C++++          POLYNOMIAL.                                         ++++CPVAL
C++++ VI     = THE IMAGINARY PART OF THE RESULTANT VALUE OF THE    ++++CPVAL
C++++          POLYNOMIAL.                                         ++++CPVAL
C++++                                                              ++++CPVAL
C++++ SUBROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:              ++++CPVAL
C++++     NONE                                                     ++++CPVAL
C++++                                                              ++++CPVAL
C++++ METHOD:                                                      ++++CPVAL
C++++ EVALUATION IS PERFORMED BY MEANS OF NESTED MULTIPLICATION    ++++CPVAL
C++++                                                              ++++CPVAL
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPVAL
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPVAL
      SUBROUTINE PVAL (A,NN,PR,PI,VR,VI)                                PVAL
      DIMENSION A(1)                                                    PVAL
      COMPLEX S,P                                                       PVAL
      S = CMPLX(PR,PI)                                                  PVAL
      P = CMPLX(A(NN+1),0.)                                             PVAL
      IF (NN) 4,3,1                                                     PVAL
C                                                                      CPVAL
C DETERMINE THE COMPLEX VALUE OF THE POLYNOMIAL                        CPVAL
C                                                                       PVAL
    1 DO 2 J=1,NN                                                       PVAL
    2 P = P*S+A(NN+1-J)                                                 PVAL
C                                                                       PVAL
C FIND THE REAL AND IMAGINARY PARTS AND RETURN TO CALLING PROGRAM      CPVAL
C                                                                      CPVAL
    3 VR = REAL(P)                                                      PVAL
      VI = AIMAG(P)                                                     PVAL
    4 RETURN                                                            PVAL
      END                                                               PVAL
```

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSEMB
C++++++++++++++++++++       SUBROUTINE SEMBL      ++++++++++++++++++++++++CSEMB
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSEMB
C++++                                                                  ++++CSEMB
C++++ PURPOSE TO EVALUATE THE REAL COEFFICIENTS OF A POLYNOMIAL        ++++CSEMB
C++++         FROM THE REAL AND COMPLEX ROOTS OF THE POLYNOMIAL.       ++++CSEMB
C++++                                                                  ++++CSEMB
C++++ CALLING SEQUENCE:                                                ++++CSEMB
C++++ CALL SEMBL(N,RR,RI,CF)                                           ++++CSEMB
C++++                                                                  ++++CSEMB
C++++ DESCRIPTION OF PARAMETERS:                                       ++++CSEMB
C++++ N       = INTEGER GIVING THE ORDER OF THE POLYNOMIAL.  THIS      ++++CSEMB
C++++           IS THE SAME AS THE NUMBER OF ROOTS.                    ++++CSEMB
C++++ RR      = ARRAY OF DIMENSION N CONTAINING THE REAL PART OF       ++++CSEMB
C++++           THE POLYNOMIAL ROOTS.                                  ++++CSEMB
C++++ RI      = ARRAY OF DIMENSION N CONTAINING THE IMAGINARY PART     ++++CSEMB
C++++           OF THE POLYNOMIAL ROOTS.                               ++++CSEMB
C++++ CF      = RESULTANT OUTPUT ARRAY CONTAINING THE REAL             ++++CSEMB
C++++           COEFFICIENTS OF THE POLYNOMIAL ORDERED IN ASCENDING    ++++CSEMB
C++++           POWERS OF THE INDEPENDENT VARIABLE.                    ++++CSEMB
C++++                                                                  ++++CSEMB
C++++ SUBROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:                  ++++CSEMB
C++++     NONE                                                         ++++CSEMB
C++++                                                                  ++++CSEMB
C++++ METHOD:                                                          ++++CSEMB
C++++ THE POLYNOMIAL COEFFICIENTS ARE CALCULATED BY FORMING THE        ++++CSEMB
C++++ SUMS OF THE PRODUCTS OF THE ROOTS USING COMPLEX ARITHMETIC       ++++CSEMB
C++++ IN ORDER TO HANDLE ANY COMPLEX ROOTS INVOLVED.                   ++++CSEMB
C++++                                                                  ++++CSEMB
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSEMB
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSEMB
C                                                                           SEMB
C                                                                           SEMB
      SUBROUTINE SEMBL (N,RR,RI,CF)                                         SEMB
C   THIS SUBROUTINE DETERMINES THE COEFFICIENTS OF A                        SEMB
C   POLYNOMIAL FROM ITS ROOTS                                               SEMB
      COMPLEX R(30),C(31),PR,SUM                                            SEMB
      DIMENSION RR(30), RI(30), J(31), CF(31)                               SEMB
      NN = N+1                                                              SEMB
C                                                                           SEMB
      DO 1 I=1,N                                                            SEMB
    1 R(I) = CMPLX(RR(I),RI(I))                                             SEMB
C                                                                           SEMB
      CF(NN) = 1.0                                                          SEMB
C                                                                           SEMB
      DO 9 M=1,N                                                            SEMB
      SUM = CMPLX(0.0,0.0)                                                  SEMB
      L = 1                                                                 SEMB
```

```
      J(1) = 1
      GO TO 3
    2 J(L) = J(L)+1
    3 IF (L-M) 4,6,10
    4 MM = M-1
C
      DO 5 I=L,MM
      II = I+1
    5 J(II) = J(I)+1
C
    6 PR = CMPLX(1.0,0.0)
C
      DO 7 I=1,M
      ICK = J(I)
    7 PR = -PR*R(ICK)
C
      SUM = SUM+PR
C
      DO 8 I=1,M
      L = M-I+1
      IF (J(L)-N+M-L) 2,8,10
    8 CONTINUE
C
      NP = N-M+1
    9 CF(NP) = REAL(SUM)
C
      RETURN
   10 WRITE (6,11)
      RETURN
C
   11 FORMAT (/,4X,14HERROR IN SEMBL)
      END
```

```
C     ..................................................................
C
C        SUBROUTINE PRQD
C
C        PURPOSE
C           CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN POLYNOMIAL
C           WITH REAL COEFFICIENTS.
C
C        USAGE
C           CALL PRQD(C,IC,Q,E,POL,IR,IER)
C
C        DESCRIPTION OF PARAMETERS
C           C      - COEFFICIENT VECTOR OF GIVEN POLYNOMIAL
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH
C                    THE GIVEN COEFFICIENT VECTOR GETS DIVIDED BY THE
C                    LAST NONZERO TERM
C           IC     - DIMENSION OF VECTOR C
C           Q      - WORKING STORAGE OF DIMENSION IC
C                    ON RETURN Q CONTAINS REAL PARTS OF ROOTS
C           E      - WORKING STORAGE OF DIMENSION IC
C                    ON RETURN E CONTAINS COMPLEX PARTS OF ROOTS
C           POL    - WORKING STORAGE OF DIMENSION IC
C                    ON RETURN POL CONTAINS THE COEFFICIENTS OF THE
C                    POLYNOMIAL WITH CALCULATED ROOTS
C                    THIS RESULTING COEFFICIENT VECTOR HAS DIMENSION IR+1
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH
C           IR     - NUMBER OF CALCULATED ROOTS
C                    NORMALLY IR IS EQUAL TO DIMENSION IC MINUS ONE
C           IER    - RESULTING ERROR PARAMETER. SEE REMARKS
C
C        REMARKS
C           THE REAL PART OF THE ROOTS IS STORED IN Q(1) UP TO Q(IR)
C           CORRESPONDING COMPLEX PARTS ARE STORED IN E(1) UP TO E(IR).
C           IER = 0 MEANS NO ERRORS
C           IER = 1 MEANS NO CONVERGENCE WITH FEASIBLE TOLERANCE
C           IER = 2 MEANS POLYNOMIAL IS DEGENERATE (CONSTANT OR ZERO)
C           IER = 3 MEANS SUBROUTINE WAS ABANDONED DUE TO ZERO DIVISOR
C           IER = 4 MEANS THERE EXISTS NO S-FRACTION
C           IER =-1 MEANS CALCULATED COEFFICIENT VECTOR REVEALS POOR
C                   ACCURACY OF THE CALCULATED ROOTS.
C                   THE CALCULATED COEFFICIENT VECTOR HAS LESS THAN
C                   3 CORRECT DIGITS.
C           THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED
C           COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE BEEN
C           CALCULATED.
C           THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT VECTOR IS
C           RECORDED IN Q(IR+1).
C
```

```
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF
C           THE QUOTIENT-DIFFERENCE ALGORITHM WITH DISPLACEMENT.
C           REFERENCE
C           H.RUTISHAUSER, DER QUOTIENTEN-DIFFERENZEN-ALGORITHMUS,
C           BIRKHAEUSER, BASEL/STUTTGART, 1957.
C
C     ..................................................................
C
      SUBROUTINE PRQD (C,IC,Q,E,POL,IR,IER)
C
C      DIMENSIONED DUMMY VARIABLES
      DIMENSION E(1), Q(1), C(1), POL(1)
C
C        NORMALIZATION OF GIVEN POLYNOMIAL
C           TEST OF DIMENSION
C        IR CONTAINS INDEX OF HIGHEST COEFFICIENT
      IER = 0
      IR = IC
      EPS = 1.E-6
      TOL = 1.E-3
      LIMIT = 10*IC
      KOUNT = 0
    1 IF (IR-1) 79,79,2
C
C        DROP TRAILING ZERO COEFFICIENTS
    2 IF (C(IR)) 4,3,4
    3 IR = IR-1
      GO TO 1
C
C           REARRANGEMENT OF GIVEN POLYNOMIAL
C        EXTRACTION OF ZERO ROOTS
    4 O = 1./C(IR)
      IEND = IR-1
      ISTA = 1
      NSAV = IR+1
      JBEG = 1
C
C        Q(J)=1.
C        Q(J+I)=C(IR-I)/C(IR)
C        Q(IR)=C(J)/C(IR)
C        WHERE J IS THE INDEX OF THE LOWEST NONZERO COEFFICIENT
C
      DO 9 I=1,IR
      J = NSAV-I
```

```
      IF (C(I)) 7,5,7                                                   PRQD 970
    5 GO TO (6,8), JBEG                                                 PRQD 980
    6 NSAV = NSAV+1                                                     PRQD 990
      C(ISTA) = 0.                                                      PRQD1000
      E(ISTA) = 0.                                                      PRQD1010
      ISTA = ISTA+1                                                     PRQD1020
      GO TO 9                                                           PRQD1030
    7 JBEG = 2                                                          PRQD1040
    8 Q(J) = C(I)*0                                                     PRQD1050
      C(I) = Q(J)                                                       PRQD1060
    9 CONTINUE                                                          PRQD1070
C                                                                       PRQD1080
C                                                                       PRQD1090
C           INITIALIZATION                                              PRQD1100
      ESAV = 0.                                                         PRQD1110
      Q(ISTA) = 0.                                                      PRQD1120
   10 NSAV = IR                                                         PRQD1130
C                                                                       PRQD1140
C        COMPUTATION OF DERIVATIVE                                      PRQD1150
      EXPT = IR-ISTA                                                    PRQD1160
      E(ISTA) = EXPT                                                    PRQD1170
C                                                                       PRQD1180
      DO 11 I=ISTA,IEND                                                 PRQD1190
      EXPT = EXPT-1.0                                                   PRQD1200
      POL(I+1) = EPS*ABS(Q(I+1))+EPS                                    PRQD1210
   11 E(I+1) = Q(I+1)*EXPT                                              PRQD1220
C                                                                       PRQD1230
C                                                                       PRQD1240
C        TEST OF REMAINING DIMENSION                                    PRQD1250
      IF (ISTA-IEND) 12,20,60                                           PRQD1260
   12 JEND = IEND-1                                                     PRQD1270
C                                                                       PRQD1280
C        COMPUTATION OF S-FRACTION                                      PRQD1290
C                                                                       PRQD1300
      DO 19 I=ISTA,JEND                                                 PRQD1310
      IF (I-ISTA) 13,16,13                                              PRQD1320
   13 IF (ABS(E(I))-POL(I+1)) 14,14,16                                  PRQD1330
C                                                                       PRQD1340
C        THE GIVEN POLYNOMIAL HAS MULTIPLE ROOTS, THE COEFFICIENTS OF   PRQD1350
C        THE COMMON FACTOR ARE STORED FROM Q(NSAV) UP TO Q(IR)          PRQD1360
   14 NSAV = I                                                          PRQD1370
C                                                                       PRQD1380
      DO 15 K=I,JEND                                                    PRQD1390
      IF (ABS(E(K))-POL(K+1)) 15,15,80                                  PRQD1400
   15 CONTINUE                                                          PRQD1410
C                                                                       PRQD1420
      GO TO 21                                                          PRQD1430
C                                                                       PRQD1440
```

```
C            EUCLIDEAN ALGORITHM
C
   16 DO 19 K=I,IEND
      E(K+1) = E(K+1)/E(I)
      Q(K+1) = E(K+1)-Q(K+1)
      IF (K-I) 18,17,18
C
C         TEST FOR SMALL DIVISOR
   17 IF (ABS(Q(I+1))-POL(I+1)) 80,80,19
   18 Q(K+1) = Q(K+1)/Q(I+1)
      POL(K+1) = POL(K+1)/ABS(Q(I+1))
      E(K) = Q(K+1)-E(K)
   19 CONTINUE
C
   20 Q(IR) = -Q(IR)
C
C            THE DISPLACEMENT EXPT IS SET TO 0 AUTOMATICALLY.
C            E(ISTA)=0.,Q(ISTA+1),...,E(NSAV-1),Q(NSAV),E(NSAV)=0.,
C            FORM A DIAGONAL OF THE QD-ARRAY.
C         INITIALIZATION OF BOUNDARY VALUES
   21 E(ISTA) = 0.
      NRAN = NSAV-1
   22 E(NRAN+1) = 0.
C
C            TEST FOR LINEAR OR CONSTANT FACTOR
C         NRAN-ISTA IS DEGREE-1
      IF (NRAN-ISTA) 24,23,31
C
C         LINEAR FACTOR
   23 Q(ISTA+1) = Q(ISTA+1)+EXPT
      E(ISTA+1) = 0.
C
C         TEST FOR UNFACTORED COMMON DIVISOR
   24 E(ISTA) = ESAV
      IF (IR-NSAV) 60,60,25
C
C         INITIALIZE QD-ALGORITHM FOR COMMON DIVISOR
   25 ISTA = NSAV
      ESAV = E(ISTA)
      GO TO 10
C
C         COMPUTATION OF ROOT PAIR
   26 P = P+EXPT
C
C            TEST FOR REALITY
      IF (O) 27,28,28
C
C            COMPLEX ROOT PAIR
```

```
   27 C(NRAN) = P                                                    PRQD1930
      C(NRAN+1) = P                                                  PRQD1940
      E(NRAN) = T                                                    PRQD1950
      E(NRAN+1) = -T                                                 PRQD1960
      GO TO 29                                                       PRQD1970
C                                                                    PRQD1980
C              REAL ROOT PAIR                                        PRQD1990
   28 C(NRAN) = P-T                                                  PRQD2000
      C(NRAN+1) = P+T                                                PRQD2010
      E(NRAN) = 0.                                                   PRQD2020
C                                                                    PRQD2030
C              REDUCTION OF DEGREE BY 2 (DEFLATION)                  PRQD2040
   29 NRAN = NRAN-2                                                  PRQD2050
      GO TO 22                                                       PRQD2060
C                                                                    PRQD2070
C          COMPUTATION OF REAL ROOT                                  PRQD2080
   30 C(NRAN+1) = EXPT+P                                             PRQD2090
C                                                                    PRQD2100
C              REDUCTION OF DEGREE BY 1 (DEFLATION)                  PRQD2110
      NRAN = NRAN-1                                                  PRQD2120
      GO TO 22                                                       PRQD2130
C                                                                    PRQD2140
C          START QD-ITERATION                                        PRQD2150
   31 JBEG = ISTA+1                                                  PRQD2160
      JEND = NRAN-1                                                  PRQD2170
      TEPS = EPS                                                     PRQD2180
      TDELT = 1.E-2                                                  PRQD2190
   32 KOUNT = KOUNT+1                                                PRQD2200
      P = C(NRAN+1)                                                  PRQD2210
      R = ABS(E(NRAN))                                               PRQD2220
C                                                                    PRQD2230
C              TEST FOR CONVERGENCE                                  PRQD2240
      IF (R-TEPS) 30,30,33                                           PRQD2250
   33 S = ABS(E(JEND))                                               PRQD2260
C                                                                    PRQD2270
C          IS THERE A REAL ROOT NEXT                                 PRQD2280
      IF (S-R) 38,38,34                                              PRQD2290
C                                                                    PRQD2300
C          IS DISPLACEMENT SMALL ENOUGH                              PRQD2310
   34 IF (R-TDELT) 36,35,35                                          PRQD2320
   35 P = 0.                                                         PRQD2330
   36 O = P                                                          PRQD2340
C                                                                    PRQD2350
      DO 37 J=JBEG,NRAN                                              PRQD2360
      C(J) = C(J)+E(J)-E(J-1)-O                                      PRQD2370
C                                                                    PRQD2380
C              TEST FOR SMALL DIVISOR                                PRQD2390
      IF (ABS(Q(J))-POL(J)) 81,81,37                                 PRQD2400
```

```
   37 E(J) = C(J+1)*E(J)/Q(J)
C
      C(NRAN+1) = -E(NRAN)+C(NRAN+1)-C
      GO TO 54
C
C        CALCULATE DISPLACEMENT FOR DOUBLE ROOTS
C           QUADRATIC EQUATION FOR DOUBLE ROOTS
C           X**2-(Q(NRAN)+Q(NRAN+1)+E(NRAN))*X+C(NRAN)*C(NRAN+1)=0
   38 P = 0.5*(Q(NRAN)+E(NRAN)+Q(NRAN+1))
      O = P*P-C(NRAN)*Q(NRAN+1)
      T = SQRT(ABS(O))
C
C        TEST FOR CONVERGENCE
      IF (S-TEPS) 26,26,39
C
C        ARE THERE COMPLEX ROOTS
   39 IF (O) 43,40,40
   40 IF (P) 42,41,41
   41 T = -T
   42 F = P+T
      R = S
      GO TO 34
C
C        MODIFICATION FOR COMPLEX ROOTS
C        IS DISPLACEMENT SMALL ENOUGH
   43 IF (S-TDELT) 44,35,35
C
C        INITIALIZATION
   44 O = C(JBEG)+E(JBEG)-P
C
C        TEST FOR SMALL DIVISOR
      IF (ABS(O)-POL(JBEG)) 81,81,45
   45 T = (T/O)**2
      U = E(JBEG)*Q(JBEG+1)/(O*(1.+T))
      V = O+U
      KOUNT = KOUNT+2
C
C        THREEFOLD LOOP FOR COMPLEX DISPLACEMENT
C
      DO 53 J=JBEG,NRAN
      O = Q(J+1)+E(J+1)-U-P
C
C        TEST FOR SMALL DIVISOR
      IF (ABS(V)-POL(J)) 46,46,49
   46 IF (J-NRAN) 81,47,81
   47 EXPT = EXPT+P
      IF (ABS(E(JEND))-TOL) 48,48,81
   48 P = 0.5*(V+O-E(JEND))
```

```
      Q = P*P-(V-U)*(O-U*T-O*W*(1.+T)/Q(JEND))                          PRQD2890
      T = SQRT(ABS(O))                                                  PRQD2900
      GO TO 26                                                          PRQD2910
C                                                                       PRQD2920
C        TEST FOR SMALL DIVISOR                                         PRQD2930
   49 IF (ABS(O)-POL(J+1)) 46,46,50                                     PRQD2940
   50 W = U*O/V                                                         PRQD2950
      T = T*(V/O)**2                                                    PRQD2960
      Q(J) = V+W-E(J-1)                                                 PRQD2970
      U = 0.                                                            PRQD2980
      IF (J-NRAN) 51,52,52                                              PRQD2990
   51 U = Q(J+2)*E(J+1)/(O*(1.+T))                                      PRQD3000
   52 V = O+U-W                                                         PRQD3010
C                                                                       PRQD3020
C        TEST FOR SMALL DIVISOR                                         PRQD3030
      IF (ABS(Q(J))-POL(J)) 81,81,53                                    PRQD3040
   53 E(J) = W*V*(1.+T)/Q(J)                                            PRQD3050
C                                                                       PRQD3060
      Q(NRAN+1) = V-E(NRAN)                                             PRQD3070
   54 EXPT = EXPT+P                                                     PRQD3080
      TEPS = TEPS*1.1                                                   PRQD3090
      TDELT = TDELT*1.1                                                 PRQD3100
      IF (KOUNT-LIMIT) 32,55,55                                         PRQD3110
C                                                                       PRQD3120
C        NO CONVERGENCE WITH FEASIBLE TOLERANCE                         PRQD3130
C           ERROR RETURN IN CASE OF UNSATISFACTORY CONVERGENCE          PRQD3140
   55 IER = 1                                                           PRQD3150
C                                                                       PRQD3160
C        REARRANGE CALCULATED ROOTS                                     PRQD3170
   56 IEND = NSAV-NRAN-1                                                PRQD3180
      E(ISTA) = ESAV                                                    PRQD3190
      IF (IEND) 59,59,57                                                PRQD3200
C                                                                       PRQD3210
   57 DO 58 I=1,IEND                                                    PRQD3220
      J = ISTA+I                                                        PRQD3230
      K = NRAN+1+I                                                      PRQD3240
      E(J) = E(K)                                                       PRQD3250
   58 Q(J) = Q(K)                                                       PRQD3260
C                                                                       PRQD3270
   59 IR = ISTA+IEND                                                    PRQD3280
C                                                                       PRQD3290
C        NORMAL RETURN                                                  PRQD3300
   60 IR = IR-1                                                         PRQD3310
      IF (IR) 78,78,61                                                  PRQD3320
C                                                                       PRQD3330
C        REARRANGE CALCULATED ROOTS                                     PRQD3340
C                                                                       PRQD3350
   61 DO 62 I=1,IR                                                      PRQD3360
```

```
      Q(I) = Q(I+1)
   62 E(I) = E(I+1)
C
C
C        CALCULATE COEFFICIENT VECTOR FROM ROOTS
      POL(IR+1) = 1.
      IEND = IR-1
      JBEG = 1
C
      DO 69 J=1,IR
      ISTA = IR+1-J
      Q = 0.
      P = Q(ISTA)
      T = E(ISTA)
      IF (T) 65,63,65
C
C        MULTIPLY WITH LINEAR FACTOR
C
   63 DO 64 I=ISTA,IR
      POL(I) = Q-P*POL(I+1)
   64 Q = POL(I+1)
C
      GO TO 69
   65 GO TO (66,67), JBEG
   66 JBEG = 2
      POL(ISTA) = 0.
      GO TO 69
C
C        MULTIPLY WITH QUADRATIC FACTOR
   67 JBEG = 1
      U = P*P+T*T
      P = P+P
C
      DO 68 I=ISTA,IEND
      POL(I) = Q-P*POL(I+1)+U*POL(I+2)
   68 Q = POL(I+1)
C
      POL(IR) = Q-P
   69 CONTINUE
C
      IF (IER) 78,70,78
C
C        COMPARISON OF COEFFICIENT VECTORS, IE. TEST OF ACCURACY
   70 P = 0.
C
      DO 75 I=1,IR
      IF (C(I)) 72,71,72
   71 Q = ABS(POL(I))
```

```
      GO TO 73                                                          PRQD3850
   72 C = ABS((POL(I)-C(I))/C(I))                                       PRQD3860
   73 IF (P-C) 74,75,75                                                 PRQD3870
   74 P = O                                                             PRQD3880
   75 CONTINUE                                                          PRQD3890
C                                                                       PRQD3900
      IF (P-TOL) 77,76,76                                               PRQD3910
   76 IER = -1                                                          PRQD3920
   77 C(IR+1) = P                                                       PRQD3930
      E(IR+1) = 0.                                                      PRQD3940
   78 RETURN                                                            PRQD3950
C                                                                       PRQD3960
C         ERROR RETURNS                                                 PRQD3970
C             ERROR RETURN FOR POLYNOMIALS OF DEGREE LESS THAN 1        PRQD3980
   79 IER = 2                                                           PRQD3990
      IR = 0                                                            PRQD4000
      RETURN                                                            PRQD4010
C                                                                       PRQD4020
C             ERROR RETURN IF THERE EXISTS NO S-FRACTION                PRQD4030
   80 IER = 4                                                           PRQD4040
      IR = ISTA                                                         PRQD4050
      GO TO 60                                                          PRQD4060
C                                                                       PRQD4070
C             ERROR RETURN IN CASE OF INSTABLE QD-ALGORITHM             PRQD4080
   81 IER = 3                                                           PRQD4090
      GO TO 56                                                          PRQD4100
      END                                                               PRQD4110
```

```
C     ..................................................................
C
C        SUBROUTINE PRBM
C
C        PURPOSE
C           TO CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN
C           POLYNOMIAL WITH REAL COEFFICIENTS.
C
C        USAGE
C           CALL PRBM (C,IC,RR,RC,POL,IR,IER)
C
C        DESCRIPTION OF PARAMETERS
C           C      - INPUT VECTOR CONTAINING THE COEFFICIENTS OF THE
C                    GIVEN POLYNOMIAL. COEFFICIENTS ARE ORDERED FROM
C                    LOW TO HIGH. ON RETURN COEFFICIENTS ARE DIVIDED
C                    BY THE LAST NONZERO TERM.
C           IC     - DIMENSION OF VECTORS C, RR, RC, AND POL.
C           RR     - RESULTANT VECTOR OF REAL PARTS OF THE ROOTS.
C           RC     - RESULTANT VECTOR OF COMPLEX PARTS OF THE ROOTS.
C           POL    - RESULTANT VECTOR OF COEFFICIENTS OF THE POLYNOMIAL
C                    WITH CALCULATED ROOTS. COEFFICIENTS ARE ORDERED
C                    FROM LOW TO HIGH (SEE REMARK 4).
C           IR     - OUTPUT VALUE SPECIFYING THE NUMBER OF CALCULATED
C                    ROOTS. NORMALLY IR IS EQUAL TO IC-1.
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS
C                     IER=0  - NO ERROR,
C                     IER=1  - SUBROUTINE PQFB RECORDS POOR CONVERGENCE
C                              AT SOME QUADRATIC FACTORIZATION WITHIN
C                              50 ITERATION STEPS,
C                     IER=2  - POLYNOMIAL IS DEGENERATE, I.E. ZERO OR
C                              CONSTANT,
C                              OR OVERFLOW IN NORMALIZATION OF GIVEN
C                              POLYNOMIAL,
C                     IER=3  - THE SUBROUTINE IS BYPASSED DUE TO
C                              SUCCESSIVE ZERO DIVISORS OR OVERFLOWS
C                              IN QUADRATIC FACTORIZATION OR DUE TO
C                              COMPLETELY UNSATISFACTORY ACCURACY,
C                     IER=-1 - CALCULATED COEFFICIENT VECTOR HAS LESS
C                              THAN THREE CORRECT SIGNIFICANT DIGITS.
C                              THIS REVEALS POOR ACCURACY OF CALCULATED
C                              ROOTS.
C
C        REMARKS
C           (1) REAL PARTS OF THE ROOTS ARE STORED IN RR(1) UP TO RR(IR)
C               AND CORRESPONDING COMPLEX PARTS IN RC(1) UP TO RC(IR).
C           (2) ERROR MESSAGE IER=1 INDICATES POOR CONVERGENCE WITHIN
C               50 ITERATION STEPS AT SOME QUADRATIC FACTORIZATION
C               PERFORMED BY SUBROUTINE PQFB.
```

```
C           (3) NO ACTION BESIDES ERROR MESSAGE IER=2 IN CASE OF A ZERO
C               OR CONSTANT POLYNOMIAL. THE SAME ERROR MESSAGE IS GIVEN
C               IN CASE OF AN OVERFLOW IN NORMALIZATION OF GIVEN
C               POLYNOMIAL.
C           (4) ERROR MESSAGE IER=3 INDICATES SUCCESSIVE ZERO DIVISORS
C               OR OVERFLOWS OR COMPLETELY UNSATISFACTORY ACCURACY AT
C               ANY QUADRATIC FACTORIZATION PERFORMED BY
C               SUBROUTINE PQFB. IN THIS CASE CALCULATION IS BYPASSED.
C               IR RECORDS THE NUMBER OF CALCULATED ROOTS.
C               POL(1),...,POL(J-IR) ARE THE COEFFICIENTS OF THE
C               REMAINING POLYNOMIAL, WHERE J IS THE ACTUAL NUMBER OF
C               COEFFICIENTS IN VECTOR C (NORMALLY J=IC).
C           (5) IF CALCULATED COEFFICIENT VECTOR HAS LESS THAN THREE
C               CORRECT SIGNIFICANT DIGITS THOUGH ALL QUADRATIC
C               FACTORIZATIONS SHOWED SATISFACTORY ACCURACY, THE ERROR
C               MESSAGE IER=-1 IS GIVEN.
C           (6) THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED
C               COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE
C               BEEN CALCULATED. IN THIS CASE THE NUMBER OF ROOTS IR IS
C               EQUAL TO THE ACTUAL DEGREE OF THE POLYNOMIAL (NORMALLY
C               IR=IC-1). THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT
C               VECTOR IS RECORDED IN RR(IR+1).
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           SUBROUTINE PQFB     QUADRATIC FACTORIZATION OF A POLYNOMIAL
C                               BY BAIRSTOW ITERATION.
C
C        METHOD
C           THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF
C           SUCCESSIVE QUADRATIC FACTORIZATION PERFORMED BY BAIRSTOW
C           ITERATION. X**2 IS USED AS INITIAL GUESS FOR THE FIRST
C           QUADRATIC FACTOR, AND FURTHER EACH CALCULATED QUADRATIC
C           FACTOR IS USED AS INITIAL GUESS FOR THE NEXT ONE. AFTER
C           COMPUTATION OF ALL ROOTS THE COEFFICIENT VECTOR IS
C           CALCULATED AND COMPARED WITH THE GIVEN ONE.
C           FOR REFERENCE, SEE J. H. WILKINSON, THE EVALUATION OF THE
C           ZEROS OF ILL-CONDITIONED POLYNOMIALS (PART ONE AND TWO),
C           NUMERISCHE MATHEMATIK, VOL.1 (1959), PP.150-180.
C
      SUBROUTINE PRBM (C,IC,RR,RC,POL,IR,IER)
C
C
      DIMENSION C(1), RR(1), RC(1), POL(1), Q(4)
C
C        TEST ON LEADING ZERO COEFFICIENTS
      EPS = 1.E-3
      LIM = 50
      IR = IC+1
```

```
    1 IR = IR-1                                                            PRBM 970
      IF (IR-1) 42,42,2                                                    PRBM 980
    2 IF (C(IR)) 3,1,3                                                     PRBM 990
C                                                                          PRBM1000
C        WORK UP ZERO ROOTS AND NORMALIZE REMAINING POLYNOMIAL             PRBM1010
    3 IER = 0                                                              PRBM1020
      J = IR                                                               PRBM1030
      L = 0                                                                PRBM1040
      A = C(IR)                                                            PRBM1050
C                                                                          PRBM1060
      DO 8 I=1,IR                                                          PRBM1070
      IF (L) 4,4,7                                                         PRBM1080
    4 IF (C(I)) 6,5,6                                                      PRBM1090
    5 RR(I) = 0.                                                           PRBM1100
      RC(I) = 0.                                                           PRBM1110
      POL(J) = 0.                                                          PRBM1120
      J = J-1                                                              PRBM1130
      GO TO 8                                                              PRBM1140
    6 L = 1                                                                PRBM1150
      IST = I                                                              PRBM1160
      J = 0                                                                PRBM1170
    7 J = J+1                                                              PRBM1180
      C(I) = C(I)/A                                                        PRBM1190
      POL(J) = C(I)                                                        PRBM1200
      CALL OVERFL (N)                                                      PRBM1210
      IF (N-2) 42,8,8                                                      PRBM1220
    8 CONTINUE                                                             PRBM1230
C                                                                          PRBM1240
C                                                                          PRBM1250
C        START BAIRSTOW ITERATION                                          PRBM1260
      Q1 = 0.                                                              PRBM1270
      Q2 = 0.                                                              PRBM1280
    9 IF (J-2) 33,10,14                                                    PRBM1290
C                                                                          PRBM1300
C        DEGREE OF RESTPOLYNOMIAL IS EQUAL TO ONE                          PRBM1310
   10 A = POL(1)                                                           PRBM1320
      RR(IST) = -A                                                         PRBM1330
      RC(IST) = 0.                                                         PRBM1340
      IR = IR-1                                                            PRBM1350
      Q2 = 0.                                                              PRBM1360
      IF (IR-1) 13,13,11                                                   PRBM1370
C                                                                          PRBM1380
   11 DO 12 I=2,IR                                                         PRBM1390
      Q1 = Q2                                                              PRBM1400
      Q2 = POL(I+1)                                                        PRBM1410
   12 POL(I) = A*Q2+Q1                                                     PRBM1420
C                                                                          PRBM1430
   13 POL(IR+1) = A+Q2                                                     PRBM1440
```

```
      GO TO 34                                                          PRBM1450
C        THIS IS BRANCH TO COMPARISON OF COEFFICIENT VECTORS C AND POL  PRBM1460
C                                                                       PRBM1470
C        DEGREE OF RESTPOLYNOMIAL IS GREATER THAN ONE                   PRBM1480
C                                                                       PRBM1490
   14 DO 22 L=1,10                                                      PRBM1500
      N = 1                                                             PRBM1510
   15 Q(1) = Q1                                                         PRBM1520
      Q(2) = Q2                                                         PRBM1530
      CALL PQFB (POL,J,Q,LIM,I)                                         PRBM1540
      IF (I) 16,24,23                                                   PRBM1550
   16 IF (Q1) 18,17,18                                                  PRBM1560
   17 IF (Q2) 18,21,18                                                  PRBM1570
   18 GO TO (19,20,19,21), N                                            PRBM1580
   19 Q1 = -Q1                                                          PRBM1590
      N = N+1                                                           PRBM1600
      GO TO 15                                                          PRBM1610
   20 Q2 = -Q2                                                          PRBM1620
      N = N+1                                                           PRBM1630
      GO TO 15                                                          PRBM1640
   21 Q1 = 1.+Q1                                                        PRBM1650
   22 Q2 = 1.-Q2                                                        PRBM1660
C                                                                       PRBM1670
C                                                                       PRBM1680
C        ERROR EXIT DUE TO UNSATISFACTORY RESULTS OF FACTORIZATION      PRBM1690
      IER = 3                                                           PRBM1700
      IR = IR-J                                                         PRBM1710
      RETURN                                                            PRBM1720
C                                                                       PRBM1730
C        WORK UP RESULTS OF QUADRATIC FACTORIZATION                     PRBM1740
   23 IER = 1                                                           PRBM1750
   24 Q1 = Q(1)                                                         PRBM1760
      Q2 = Q(2)                                                         PRBM1770
C                                                                       PRBM1780
C        PERFORM DIVISION OF FACTORIZED POLYNOMIAL BY QUADRATIC FACTOR  PRBM1790
      B = 0.                                                            PRBM1800
      A = 0.                                                            PRBM1810
      I = J                                                             PRBM1820
   25 H = -Q1*B-Q2*A+POL(I)                                             PRBM1830
      POL(I) = B                                                        PRBM1840
      B = A                                                             PRBM1850
      A = H                                                             PRBM1860
      I = I-1                                                           PRBM1870
      IF (I-2) 26,26,25                                                 PRBM1880
   26 POL(2) = B                                                        PRBM1890
      POL(1) = A                                                        PRBM1900
C                                                                       PRBM1910
C        MULTIPLY POLYNOMIAL WITH CALCULATED ROOTS BY QUADRATIC FACTOR  PRBM1920
```

```
      L = IR-1                                                          PRBM1930
      IF (J-L) 27,27,29                                                 PRBM1940
C                                                                       PRBM1950
   27 DO 28 I=J,L                                                       PRBM1960
   28 POL(I-1) = POL(I-1)+POL(I)*Q2+POL(I+1)*Q1                         PRBM1970
C                                                                       PRBM1980
   29 POL(L) = POL(L)+POL(L+1)*Q2+Q1                                    PRBM1990
      POL(IR) = POL(IR)+Q2                                              PRBM2000
C                                                                       PRBM2010
C        CALCULATE ROOT-PAIR FROM QUADRATIC FACTOR X*X+Q2*X+Q1          PRBM2020
      H = -.5*Q2                                                        PRBM2030
      A = H*H-Q1                                                        PRBM2040
      B = SQRT(ABS(A))                                                  PRBM2050
      IF (A) 30,30,31                                                   PRBM2060
   30 RR(IST) = H                                                       PRBM2070
      RC(IST) = B                                                       PRBM2080
      IST = IST+1                                                       PRBM2090
      RR(IST) = H                                                       PRBM2100
      RC(IST) = -B                                                      PRBM2110
      GO TO 32                                                          PRBM2120
   31 B = H+SIGN(B,H)                                                   PRBM2130
      RR(IST) = Q1/B                                                    PRBM2140
      RC(IST) = 0.                                                      PRBM2150
      IST = IST+1                                                       PRBM2160
      RR(IST) = B                                                       PRBM2170
      RC(IST) = 0.                                                      PRBM2180
   32 IST = IST+1                                                       PRBM2190
      J = J-2                                                           PRBM2200
      GO TO 9                                                           PRBM2210
C                                                                       PRBM2220
C        SHIFT BACK ELEMENTS OF POL BY 1 AND COMPARE VECTORS POL AND C  PRBM2230
   33 IR = IR-1                                                         PRBM2240
   34 A = 0.                                                            PRBM2250
C                                                                       PRBM2260
      DO 38 I=1,IR                                                      PRBM2270
      Q1 = C(I)                                                         PRBM2280
      Q2 = POL(I+1)                                                     PRBM2290
      POL(I) = Q2                                                       PRBM2300
      IF (Q1) 35,36,35                                                  PRBM2310
   35 Q2 = (Q1-Q2)/Q1                                                   PRBM2320
   36 Q2 = ABS(Q2)                                                      PRBM2330
      IF (Q2-A) 38,38,37                                                PRBM2340
   37 A = Q2                                                            PRBM2350
   38 CONTINUE                                                          PRBM2360
C                                                                       PRBM2370
      I = IR+1                                                          PRBM2380
      POL(I) = 1.                                                       PRBM2390
      RR(I) = A                                                         PRBM2400
```

```
      RC(1) = 0.                                                        PRBM2410
      IF (IER) 39,39,41                                                 PRBM2420
   39 IF (A-EPS) 41,41,40                                               PRBM2430
C                                                                       PRBM2440
C        WARNING DUE TO POOR ACCURACY OF CALCULATED COEFFICIENT VECTOR  PRBM2450
   40 IER = -1                                                          PRBM2460
   41 RETURN                                                            PRBM2470
C                                                                       PRBM2480
C        ERROR EXIT DUE TO DEGENERATE POLYNOMIAL OR OVERFLOW IN         PRBM2490
C        NORMALIZATION                                                  PRBM2500
   42 IER = 2                                                           PRBM2510
      IR = 0                                                            PRBM2520
      RETURN                                                            PRBM2530
      END                                                               PRBM2540
```

```
C      .................................................................
C
C         SUBROUTINE PQFB
C
C         PURPOSE
C            TO FIND AN APPROXIMATION Q(X)=Q1+Q2*X+X*X TO A QUADRATIC
C            FACTOR OF A GIVEN POLYNOMIAL P(X) WITH REAL COEFFICIENTS.
C
C         USAGE
C            CALL PQFB(C,IC,Q,LIM,IER)
C
C         DESCRIPTION OF PARAMETERS
C            C     - INPUT VECTOR CONTAINING THE COEFFICIENTS OF P(X) -
C                    C(1) IS THE CONSTANT TERM (DIMENSION IC)
C            IC    - DIMENSION OF C
C            Q     - VECTOR OF DIMENSION 4 - ON INPUT Q(1) AND Q(2) MUST
C                    CONTAIN INITIAL GUESSES FOR Q1 AND Q2 - ON RETURN Q(1)
C                    AND Q(2) CONTAIN THE REFINED COEFFICIENTS Q1 AND Q2 OF
C                    Q(X), WHILE Q(3) AND Q(4) CONTAIN THE COEFFICIENTS A
C                    AND B OF A+B*X, WHICH IS THE REMAINDER OF THE QUOTIENT
C                    OF P(X) BY Q(X)
C            LIM   - INPUT VALUE SPECIFYING THE MAXIMUM NUMBER OF
C                    ITERATIONS TO BE PERFORMED
C            IER   - RESULTING ERROR PARAMETER (SEE REMARKS)
C                    IER= 0 - NO ERROR
C                    IER= 1 - NO CONVERGENCE WITHIN LIM ITERATIONS
C                    IER=-1 - THE POLYNOMIAL P(X) IS CONSTANT OR UNDEFINED
C                             - OR OVERFLOW OCCURRED IN NORMALIZING P(X)
C                    IER=-2 - THE POLYNOMIAL P(X) IS OF DEGREE 1
C                    IER=-3 - NO FURTHER REFINEMENT OF THE APPROXIMATION TO
C                             A QUADRATIC FACTOR IS FEASIBLE, DUE TO EITHER
C                             DIVISION BY 0, OVERFLOW OR AN INITIAL GUESS
C                             THAT IS NOT SUFFICIENTLY CLOSE TO A FACTOR OF
C                             P(X)
C
C         REMARKS
C            (1)  IF IER=-1 THERE IS NO COMPUTATION OTHER THAN THE
C                 POSSIBLE NORMALIZATION OF C.
C            (2)  IF IER=-2 THERE IS NO COMPUTATION OTHER THAN THE
C                 NORMALIZATION OF C.
C            (3)  IF IER =-3  IT IS SUGGESTED THAT A NEW INITIAL GUESS BE
C                 MADE FOR A QUADRATIC FACTOR.  Q, HOWEVER, WILL CONTAIN
C                 THE VALUES ASSOCIATED WITH THE ITERATION THAT YIELDED
C                 THE SMALLEST NORM OF THE MODIFIED LINEAR REMAINDER.
C            (4)  IF IER=1, THEN, ALTHOUGH THE NUMBER OF ITERATIONS LIM
C                 WAS TOO SMALL TO INDICATE CONVERGENCE, NO OTHER PROB-
C                 LEMS HAVE BEEN DETECTED, AND Q WILL CONTAIN THE VALUES
C                 ASSOCIATED WITH THE ITERATION THAT YIELDED THE SMALLEST
```

```
C             NORM OF THE MODIFIED LINEAR REMAINDER.
C        (5)  FOR COMPLETE DETAIL SEE THE DOCUMENTATION FOR
C             SUBROUTINES PQFB AND DPQFB.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        NONE
C
C     METHOD
C        COMPUTATION IS BASED ON BAIRSTOW'S ITERATIVE METHOD.  (SEE
C        WILKINSON, J.H., THE EVALUATION OF THE ZEROS OF ILL-CON-
C        DITIONED POLYNOMIALS (PART ONE AND TWO), NUMERISCHE MATHE-
C        MATIK, VOL.1 (1959), PP. 150-180, OR HILDEBRAND, F.B.,
C        INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/
C        TORONTO/LONDON, 1956, PP. 472-476.)
C
C     ..................................................................
C
      SUBROUTINE PQFB (C,IC,Q,LIM,IER)
C
C
      DIMENSION C(1), Q(1)
C
C        TEST ON LEADING ZERO COEFFICIENTS
      IER = 0
      J = IC+1
    1 J = J-1
      IF (J-1) 40,40,2
    2 IF (C(J)) 3,1,3
C
C        NORMALIZATION OF REMAINING COEFFICIENTS
    3 A = C(J)
      IF (A-1.) 4,6,4
C
    4 DO 5 I=1,J
      C(I) = C(I)/A
      CALL OVERFL (N)
      IF (N-2) 40,5,5
    5 CONTINUE
C
C
C        TEST ON NECESSITY OF BAIRSTOW ITERATION
    6 IF (J-3) 41,38,7
C
C        PREPARE BAIRSTOW ITERATION
    7 EPS = 1.E-6
      EPS1 = 1.E-3
      L = 0
      LL = 0
```

```
      Q1 = Q(1)                                                         PQFB 970
      Q2 = Q(2)                                                         PQFB 980
      QQ1 = 0.                                                          PQFB 990
      QQ2 = 0.                                                          PQFB1000
      AA = C(1)                                                         PQFB1010
      BB = C(2)                                                         PQFB1020
      CB = ABS(AA)                                                      PQFB1030
      CA = ABS(BB)                                                      PQFB1040
      IF (CB-CA) 8,9,10                                                 PQFB1050
    8 CC = CB+CB                                                        PQFB1060
      CB = CB/CA                                                        PQFB1070
      CA = 1.                                                           PQFB1080
      GO TO 11                                                          PQFB1090
    9 CC = CA+CA                                                        PQFB1100
      CA = 1.                                                           PQFB1110
      CB = 1.                                                           PQFB1120
      GO TO 11                                                          PQFB1130
   10 CC = CA+CA                                                        PQFB1140
      CA = CA/CB                                                        PQFB1150
      CB = 1.                                                           PQFB1160
   11 CD = CC*.1                                                        PQFB1170
C                                                                       PQFB1180
C         START BAIRSTOW ITERATION                                      PQFB1190
C         PREPARE NESTED MULTIPLICATION                                 PQFB1200
   12 A = 0.                                                            PQFB1210
      B = A                                                             PQFB1220
      A1 = A                                                            PQFB1230
      B1 = A                                                            PQFB1240
      I = J                                                             PQFB1250
      QQQ1 = Q1                                                         PQFB1260
      QQQ2 = Q2                                                         PQFB1270
      DQ1 = HH                                                          PQFB1280
      DQ2 = H                                                           PQFB1290
C                                                                       PQFB1300
C         START NESTED MULTIPLICATION                                   PQFB1310
   13 H = -Q1*B-Q2*A+C(I)                                               PQFB1320
      CALL OVERFL (N)                                                   PQFB1330
      IF (N-2) 42,14,14                                                 PQFB1340
   14 B = A                                                             PQFB1350
      A = H                                                             PQFB1360
      I = I-1                                                           PQFB1370
      IF (I-1) 18,15,16                                                 PQFB1380
   15 H = 0.                                                            PQFB1390
   16 H = -Q1*B1-Q2*A1+H                                                PQFB1400
      CALL OVERFL (N)                                                   PQFB1410
      IF (N-2) 42,17,17                                                 PQFB1420
   17 C1 = B1                                                           PQFB1430
      B1 = A1                                                           PQFB1440
```

```
      A1 = H                                                            PQFB1450
      GO TO 13                                                          PQFB1460
C         END OF NESTED MULTIPLICATION                                  PQFB1470
C                                                                       PQFB1480
C         TEST ON SATISFACTORY ACCURACY                                 PQFB1490
   18 H = CA*ABS(A)+CB*ABS(B)                                           PQFB1500
      IF (LL) 19,19,39                                                  PQFB1510
   19 L = L+1                                                           PQFB1520
      IF (ABS(A)-EPS*ABS(C(1))) 20,20,21                                PQFB1530
   20 IF (ABS(B)-EPS*ABS(C(2))) 39,39,21                                PQFB1540
C                                                                       PQFB1550
C         TEST ON LINEAR REMAINDER OF MINIMUM NORM                      PQFB1560
   21 IF (H-CC) 22,22,23                                                PQFB1570
   22 AA = A                                                            PQFB1580
      BB = B                                                            PQFB1590
      CC = H                                                            PQFB1600
      CQ1 = Q1                                                          PQFB1610
      CQ2 = Q2                                                          PQFB1620
C                                                                       PQFB1630
C         TEST ON LAST ITERATION STEP                                   PQFB1640
   23 IF (L-LIM) 28,28,24                                               PQFB1650
C                                                                       PQFB1660
C         TEST ON RESTART OF BAIRSTOW ITERATION WITH ZERO INITIAL GUESS PQFB1670
   24 IF (H-CD) 43,43,25                                                PQFB1680
   25 IF (Q(1)) 27,26,27                                                PQFB1690
   26 IF (Q(2)) 27,42,27                                                PQFB1700
   27 Q(1) = 0.                                                         PQFB1710
      Q(2) = 0.                                                         PQFB1720
      GO TO 7                                                           PQFB1730
C                                                                       PQFB1740
C         PERFORM ITERATION STEP                                        PQFB1750
   28 HH = AMAX1(ABS(A1),ABS(B1),ABS(C1))                               PQFB1760
      IF (HH) 42,42,29                                                  PQFB1770
   29 A1 = A1/HH                                                        PQFB1780
      B1 = B1/HH                                                        PQFB1790
      C1 = C1/HH                                                        PQFB1800
      H = A1*C1-B1*B1                                                   PQFB1810
      IF (H) 30,42,30                                                   PQFB1820
   30 A = A/HH                                                          PQFB1830
      B = B/HH                                                          PQFB1840
      HH = (B*A1-A*B1)/H                                                PQFB1850
      H = (A*C1-B*B1)/H                                                 PQFB1860
      Q1 = Q1+HH                                                        PQFB1870
      Q2 = Q2+H                                                         PQFB1880
C         END OF ITERATION STEP                                         PQFB1890
C                                                                       PQFB1900
C         TEST ON SATISFACTORY RELATIVE ERROR OF ITERATED VALUES        PQFB1910
      IF (ABS(HH)-EPS*ABS(Q1)) 31,31,32                                 PQFB1920
```

```
   31 IF (ABS(H)-EPS*ABS(Q2)) 32,32,33                                   PQFB1930
   32 LL = 1                                                             PQFB1940
      GO TO 12                                                           PQFB1950
C                                                                        PQFB1960
C         TEST ON DECREASING RELATIVE ERRORS                             PQFB1970
   33 IF (L-1) 12,12,34                                                  PQFB1980
   34 IF (ABS(HH)-EPS1*ABS(Q1)) 35,35,12                                 PQFB1990
   35 IF (ABS(H)-EPS1*ABS(Q2)) 36,36,12                                  PQFB2000
   36 IF (ABS(QQQ1*HH)-ABS(Q1*DQ1)) 37,44,44                             PQFB2010
   37 IF (ABS(QQQ2*H)-ABS(Q2*DQ2)) 12,44,44                              PQFB2020
C         END OF BAIRSTOW ITERATION                                      PQFB2030
C                                                                        PQFB2040
C         EXIT IN CASE OF QUADRATIC POLYNOMIAL                           PQFB2050
   38 Q(1) = C(1)                                                        PQFB2060
      Q(2) = C(2)                                                        PQFB2070
      Q(3) = 0.                                                          PQFB2080
      Q(4) = 0.                                                          PQFB2090
      RETURN                                                             PQFB2100
C                                                                        PQFB2110
C         EXIT IN CASE OF SUFFICIENT ACCURACY                            PQFB2120
   39 Q(1) = Q1                                                          PQFB2130
      Q(2) = Q2                                                          PQFB2140
      Q(3) = A                                                           PQFB2150
      Q(4) = B                                                           PQFB2160
      RETURN                                                             PQFB2170
C                                                                        PQFB2180
C         ERROR EXIT IN CASE OF ZERO OR CONSTANT POLYNOMIAL              PQFB2190
   40 IER = -1                                                           PQFB2200
      RETURN                                                             PQFB2210
C                                                                        PQFB2220
C         ERROR EXIT IN CASE OF LINEAR POLYNOMIAL                        PQFB2230
   41 IER = -2                                                           PQFB2240
      RETURN                                                             PQFB2250
C                                                                        PQFB2260
C         ERROR EXIT IN CASE OF NONREFINED QUADRATIC FACTOR              PQFB2270
   42 IER = -3                                                           PQFB2280
      GO TO 44                                                           PQFB2290
C                                                                        PQFB2300
C         ERROR EXIT IN CASE OF UNSATISFACTORY ACCURACY                  PQFB2310
   43 IER = 1                                                            PQFB2320
   44 Q(1) = QQ1                                                         PQFB2330
      Q(2) = QQ2                                                         PQFB2340
      Q(3) = AA                                                          PQFB2350
      Q(4) = BB                                                          PQFB2360
      RETURN                                                             PQFB2370
      END                                                                PQFB2380
```

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C++++                      SUBROUTINE CCNORM                     ++++
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C++++                                                            ++++
C++++ PURPOSE: TO NORMALIZE THE COEFFICIENT FO THE HIGHEST POWER ++++
C++++          TERM IN A POLYNOMIAL TO UNITY.                    ++++
C++++                                                            ++++
C++++ CALLING SEQUENCE:                                          ++++
C++++ CALL XONORM(V,N,XNORM)                                     ++++
C++++                                                            ++++
C++++ DESCRIPTION OF PARAMETERS:                                 ++++
C++++                                                            ++++
C++++ V      = ONE DIMENSIONAL ARRAY CONTAINING THE UNNORMALIZED ++++
C++++          COEFFICIANTS IN ASCENDING POWER OF THE INDEPENDENT ++++
C++++          VARIABLE.  UPON RETURN FROM THE SUBROUTINE THIS   ++++
C++++          VECTOR WILL CONTAIN THE NORMALIZED COEFFICIENTS.  ++++
C++++ N      = DIMENSION OF THE ARRAY V.                         ++++
C++++ XNORM  = NORMALIZING FACTOR BY WHICH ALL COEFFICIENTS HAVE ++++
C++++          BEEN DIVIDED.                                     ++++
C++++                                                            ++++
C++++ SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED:             ++++
C++++     NONE                                                   ++++
C++++                                                            ++++
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE CONORM (V,N,VNORM)
      DIMENSION V(1)
      IF (V(N)) 1,3,1
C
C INSLRE HIGHEST ORDER COEFFICIENT IS NOT UNITY
C
    1 IF (V(N).EQ.1.0) GO TO 4
C
C  SELECT COEFFICIENT OF HIGHEST ORDER TERM AS NORMALIZING FACTOR
C  AND DIVIDE EACH COEFFICIENT OF POLYNOMIAL BY THIS FACTOR.
C
      VNORM = V(N)
C
      DO 2 I=1,N
      V(I) = V(I)/VNORM
    2 CONTINUE
C
      RETURN
    3 WRITE (6,5)
      RETURN
    4 VNORM = 1.0
      RETURN
C
```

```
5 FCRMAT ('0',72('*'),//,' WARNING----COEFFICIENT OF HIGHEST ORCER TCONO 490
 1ERM IS ZERO.',/,' EXECUTION OF SUBROUTINE CCNCRM TERMINATING WITHCCONO 500
 2UT NORMALIZING',/,' THE COEFFICIENT VECTOR.',//,72('*'))            CONO 510
  ENC                                                                  CONO 520
```

```
C++++                                                                  ++++C
C
C
      SUBROUTINE GRAPH (NBODE,NYQST,NICHOL)
      INTEGER *4ITB(12)/12*0/
      REAL *4NTITLE
      REAL *4TITLE1(12),TITLE2(12),TITLE3(12),TITLE4(12)
      REAL *4TITLE5(12),TITLE6(12)
      REAL *4SCALE(2)/2*0.0/,SCALE1(2)/2*-120./,SCALE2(2)/2*0.0/,RTE(28)
     1/28*0.0/
      DATA TITLE1/'BODE',' PLO','T --','- GA','IN(D','B)  ','VS. ',' FRE
     1','QUEN','CY(R','ADIA','NS) '/
      DATA TITLE2/'BODE',' PLO','T --','- PH','ASE(','DEG)',' VS.',' FRE
     1','QUEN','CY(R','ADIA','NS) '/
      DATA TITLE3/'NICH','OLS ','PLOT',' ---',' GAI','N(DB',')  V','S.
     1','PHAS','E(DE','GREE','S)  '/
      DATA TITLE4/'    ','    ','  NY','QUIS','T PL','OT O','F AC','TUAL
     1',' SYS','TEM ','    ','    '/
      DATA TITLE5/'MAGN','ITUD','E DI','FFER','ENCE','(DB)',' VS.',' FRE
     1','QUEN','CY(R','ADIA','NS) '/
      DATA TITLE6/'PHAS','E DI','FFER','ENCE','(DEG','REES',') VS','. FR
     1','EQUE','NCY(','RADI','ANS)'/
      COMMON NCN,NCD,NTFN,NTFD,NOMEG,A(20),B(20),DMAG(500),DQPHSD(500),W
     1FREQ(500),NSN,NSD,COFASN(40),COFASD(40),CMAG(500),CR(500),CI(500),
     2CPHSR(500),QPHSD(500),N1,N2,N3,N4,PI,T,WS,IZCH,ACMAG(500),NTITLE(1
     32),FREQ(500),AMAG(500),NMINFS,CCFWRK(40),IPLCT,WFRECS(500),SAPHSD(
     4500),SAMAG(500),DIFF1(500),DIFF2(500),ICCST,CFWRK1(40)
      IF (IPLCT.EQ.1) GO TO 13
      IF (NBODE) 3,1,2
    1 WRITE (6,34)
      WRITE (6,35)
      WRITE (6,36)
      WRITE (6,37) NTITLE
      WRITE (6,35)
      CALL PLCTP (FREQ,AMAG,NOMEG,1)
      CALL PLCTP (WFREQS,SAMAG,500,2)
      CALL PLCTP (FREQ,ACMAG,NOMEG,3)
      WRITE (6,34)
      WRITE (6,35)
      WRITE (6,38)
      WRITE (6,37) NTITLE
      WRITE (6,35)
      CALL PLCTP (FREQ,CPHSD,NOMEG,1)
      CALL PLCTP (WFREQS,SAPHSD,500,2)
      CALL PLCTP (FREQ,DCPHSD,NOMEG,3)
      GO TO 4
    2 WRITE (6,35)
      WRITE (6,39)
```

```
      WRITE (6,35)
      GO TO 4
    3 WRITE (6,35)
      WRITE (6,40)
      WRITE (6,35)
    4 CONTINUE
      IF (NICHOL) 7,5,6
    5 WRITE (6,34)
      WRITE (6,35)
      WRITE (6,41)
      WRITE (6,37) NTITLE
      WRITE (6,35)
      CALL FLCTP (QPHSD,AMAG,NOMEG,1)
      CALL FLCTP (SAPHSD,SAMAG,500,2)
      CALL FLCTP (DQPHSD,ADMAG,NOMEG,3)
      GO TO 8
    6 WRITE (6,35)
      WRITE (6,42)
      WRITE (6,35)
      GO TO 8
    7 WRITE (6,35)
      WRITE (6,43)
    8 CONTINUE
      IF (NYQST) 11,9,10
    9 WRITE (6,34)
      WRITE (6,35)
      WRITE (6,44)
      WRITE (6,37) NTITLE
      WRITE (6,35)
      CALL FLCTP (QR,QI,500,0)
      GO TO 12
   10 WRITE (6,35)
      WRITE (6,45)
      WRITE (6,35)
      GO TO 12
   11 WRITE (6,35)
      WRITE (6,46)
      WRITE (6,35)
   12 CONTINUE
      WRITE (6,34)
      WRITE (6,35)
      WRITE (6,47)
      WRITE (6,37) NTITLE
      WRITE (6,35)
      CALL FLCTP (FREQ,DIFF1,NOMEG,0)
      WRITE (6,34)
      WRITE (6,35)
      WRITE (6,48)
```

```
      WRITE (6,37) NTITLE
      WRITE (6,35)
      CALL PLOTP (FREQ,DIFF2,NOMEG,0)
      RETURN
C
   13 DO 14 I=1,NOMEG
      X1 = SCALE(1)
      X2 = SCALE(2)
      XX1 = SCALE1(1)
      XX2 = SCALE1(2)
      XXX1 = SCALE2(1)
      XXX2 = SCALE2(2)
      Z1 = AMAG(I)
      Z2 = ADMAG(I)
      Z3 = QPHSD(I)
      Z4 = DQPHSD(I)
      Z5 = FREQ(I)
      SCALE(1) = AMAX1(X1,Z1,Z2)
      SCALE(2) = AMIN1(X2,Z1,Z2)
      SCALE1(1) = AMAX1(XX1,Z3,Z4)
      SCALE1(2) = AMIN1(XX2,Z3,Z4)
      SCALE2(1) = AMAX1(XXX1,Z5)
      SCALE2(2) = AMIN1(XXX2,Z5)
   14 CONTINUE
C
C
      DO 15 J=1,500
      SCALE(1) = AMAX1(SCALE(1),SAMAG(J))
      SCALE(2) = AMIN1(SCALE(2),SAMAG(J))
      SCALE1(1) = AMAX1(SCALE1(1),SAPHSE(J))
      SCALE1(2) = AMIN1(SCALE1(2),SAPHSE(J))
   15 CONTINUE
C
      IF (NBCDE) 20,16,19
C
   16 DO 17 I=1,12
      RTE(I+4) = NTITLE(I)
      RTE(I+16) = TITLE1(I)
   17 CONTINUE
C
C
C     SET UP AND PLOT GAIN(DB) VS. FREQUENCY
C
      ITE(1) = 1
      ITE(2) = 2
      ITE(3) = 9
      ITE(4) = 6
      CALL DRAWP (2,SCALE2,SCALE,ITB,RTE)
```

```
      ITB(1) = 2                                                        GRAP1450
      ITB(2) = 4                                                        GRAP1460
      CALL DRAWP (NOMEG,FREQ,ADMAG,ITB,RTB)                             GRAP1470
      ITB(1) = 2                                                        GRAP1480
      ITB(2) = 1                                                        GRAP1490
      CALL DRAWP (NOMEG,FREQ,AMAG,ITB,RTB)                              GRAP1500
      ITB(1) = 3                                                        GRAP1510
      ITB(2) = 0                                                        GRAP1520
      CALL DRAWP (500,WFREQS,SAMAG,ITB,RTB)                             GRAP1530
C                                                                       GRAP1540
      DO 18 I=1,12                                                      GRAP1550
      RTB(I+16) = TITLE2(I)                                             GRAP1560
   18 CONTINUE                                                          GRAP1570
C                                                                       GRAP1580
      ITB(1) = 1                                                        GRAP1590
      ITB(2) = 2                                                        GRAP1600
      ITB(3) = 9                                                        GRAP1610
      ITB(4) = 6                                                        GRAP1620
      CALL DRAWP (2,SCALE2,SCALE1,ITB,RTB)                              GRAP1630
      ITB(1) = 2                                                        GRAP1640
      ITB(2) = 4                                                        GRAP1650
      CALL DRAWP (NOMEG,FREQ,DQPHSD,ITB,RTB)                            GRAP1660
      ITB(1) = 2                                                        GRAP1670
      ITB(2) = 1                                                        GRAP1680
      CALL DRAWP (NOMEG,FREQ,QPHSD,ITB,RTB)                             GRAP1690
      ITB(1) = 3                                                        GRAP1700
      ITB(2) = 0                                                        GRAP1710
      CALL DRAWP (500,WFREQS,SAPHSD,ITB,RTB)                            GRAP1720
      WRITE (6,35)                                                      GRAP1730
      WRITE (6,49)                                                      GRAP1740
      WRITE (6,35)                                                      GRAP1750
      GO TO 21                                                          GRAP1760
   19 WRITE (6,35)                                                      GRAP1770
      WRITE (6,39)                                                      GRAP1780
      WRITE (6,35)                                                      GRAP1790
      GO TO 21                                                          GRAP1800
   20 WRITE (6,35)                                                      GRAP1810
      WRITE (6,40)                                                      GRAP1820
      WRITE (6,35)                                                      GRAP1830
   21 IF (NICHOL) 25,22,24                                              GRAP1840
C                                                                       GRAP1850
   22 DO 23 I=1,12                                                      GRAP1860
      RTB(I+16) = TITLE3(I)                                             GRAP1870
   23 CONTINUE                                                          GRAP1880
C                                                                       GRAP1890
      ITB(1) = 1                                                        GRAP1900
      ITB(2) = 2                                                        GRAP1910
      ITB(3) = 6                                                        GRAP1920
```

```
      ITB(4) = 9                                                        GRAP1930
      CALL DRAWP (2,SCALE1,SCALE,ITB,RTB)                               GRAP1940
      ITB(1) = 2                                                        GRAP1950
      ITB(2) = 4                                                        GRAP1960
      CALL DRAWP (NOMEG,DQPHSD,ADMAG,ITB,RTB)                           GRAP1970
      ITB(1) = 2                                                        GRAP1980
      ITB(2) = 1                                                        GRAP1990
      CALL DRAWP (NOMEG,QPHSD,AMAG,ITB,RTB)                             GRAP2000
      ITB(1) = 3                                                        GRAP2010
      ITB(2) = 0                                                        GRAP2020
      CALL DRAWP (500,SAPHSD,SAMAG,ITB,RTB)                             GRAP2030
      WRITE (6,35)                                                      GRAP2040
      WRITE (6,50)                                                      GRAP2050
      WRITE (6,35)                                                      GRAP2060
      GO TO 26                                                          GRAP2070
   24 WRITE (6,35)                                                      GRAP2080
      WRITE (6,42)                                                      GRAP2090
      WRITE (6,35)                                                      GRAP2100
      GO TO 26                                                          GRAP2110
   25 WRITE (6,35)                                                      GRAP2120
      WRITE (6,43)                                                      GRAP2130
      WRITE (6,35)                                                      GRAP2140
   26 IF (NYQST) 30,27,29                                               GRAP2150
C                                                                       GRAP2160
   27 DO 28 I=1,12                                                      GRAP2170
      RTB(I+16) = TITLE4(I)                                             GRAP2180
   28 CONTINUE                                                          GRAP2190
C                                                                       GRAP2200
      ITB(1) = 0                                                        GRAP2210
      ITB(2) = 0                                                        GRAP2220
      ITB(3) = 8                                                        GRAP2230
      ITB(4) = 8                                                        GRAP2240
      CALL DRAWP (500,QR,QI,ITB,RTB)                                    GRAP2250
      WRITE (6,35)                                                      GRAP2260
      WRITE (6,51)                                                      GRAP2270
      WRITE (6,35)                                                      GRAP2280
      GO TO 31                                                          GRAP2290
   29 WRITE (6,35)                                                      GRAP2300
      WRITE (6,45)                                                      GRAP2310
      WRITE (6,35)                                                      GRAP2320
      GO TO 31                                                          GRAP2330
   30 WRITE (6,35)                                                      GRAP2340
      WRITE (6,46)                                                      GRAP2350
      WRITE (6,35)                                                      GRAP2360
   31 CONTINUE                                                          GRAP2370
C                                                                       GRAP2380
      DO 32 I=1,12                                                      GRAP2390
      RTB(I+16) = TITLE5(I)                                             GRAP2400
```

```
32 CONTINUE
C
      ITE(1) = 0
      ITE(2) = 5
      ITE(3) = 9
      ITE(4) = 6
      CALL DRAWP (NOMEG,FREQ,DIFF1,ITB,RTB)
C
      DO 33 I=1,12
      RTE(I+16) = TITLE6(I)
33 CONTINUE
C
      CALL DRAWP (NOMEG,FREQ,DIFF2,ITB,RTB)
      RETURN
C
34 FORMAT ('1')
35 FORMAT (' ',80('='))
36 FORMAT ('0',20X,'BODE PLOT --- GAIN(DB)  VS.  FREQUENCY (RADIANS)'
   1)
37 FORMAT ('0',10X,'TITLE --- ',12A4,/)
38 FORMAT ('0',20X,'BODE PLOT --- PHASE(DEGREES)  VS.  FREQUENCY (RAD
   1IANS)')
39 FORMAT ('0',10X,'BODE PLOT NOT DESIRED',/)
40 FORMAT ('0',80('*'),/,5('*'),3X,'ERROR EXISTS IN GRAPH SELECTION
   1 NUMBER',/,5X,'NBODE.  THIS NUMBER SHOULD NOT BE NEGATIVE; CHECK',
   2/,5X,'DATA DECK FOR POSSIBLE ERRORS.',/,'0',80('*'))
41 FORMAT ('0',20X,'NICHOLS PLOT')
42 FORMAT ('0',10X,'NICHOLS PLOT NOT DESIRED',/)
43 FORMAT ('0',80('*'),/,5('*'),3X,'ERROR EXISTS IN GRAPH SELECTION
   1 NUMBER',/,5X,'NICHOL.  THIS NUMBER SHOULD NOT BE NEGATIVE; CHECK'
   2,/,5X,'DATA DECK FOR POSSIBLE ERRORS.',/,'0',80('*'))
44 FORMAT ('0',20X,'NYQUIST PLOT')
45 FORMAT ('0',10X,'NYQUIST PLOT NOT DESIRED',/)
46 FORMAT ('0',80('*'),/,5('*'),3X,'ERROR EXISTS IN GRAPH SELECTION
   1 NUMBER',/,5X,'NYQST.  THIS NUMBER SHOULD NOT BE NEGATIVE; CHECK',
   2/,5X,'DATA DECK FOR POSSIBLE ERRORS.',/,'0',80('*'))
47 FORMAT ('0',10X,'MAGNITUDE DIFFERENCE IN DB VS. FREQUENCY(RADS)')
48 FORMAT ('0',10X,'PHASE DIFFERENCE(DEGREES) VS. FREQUENCY(RADS)')
49 FORMAT ('0',10X,'BODE PLOT HAS BEEN DRAWN',/)
50 FORMAT ('0',10X,'NICHOLS PLOT HAS BEEN DRAWN',/)
51 FORMAT ('0',10X,'NYQUIST PLOT HAS BEEN DRAWN',/)
      END
```

```
      FUNCTION KE (X)
      DIMENSION X(24),  Y(12),  Z(12)
      COMMON NCN,NCD,NTFN,NTFD,NOMEG,A(20),B(20),CMAG(500),DQPHSD(500),W
     1FREQ(500),NSN,NSD,COFASN(40),COFASD(40),CMAG(500),CR(500),CI(500),
     2CPHSR(500),CPHSD(500),N1,N2,N3,N4,PI,T,WS,IZOH,ADMAG(500),NTITLE(1
     32),FREQ(500),AMAG(500),NMINFS,COFWRK(40),IPLOT,WFREQS(500),SAPHSD(
     4500),SAMAG(500),DIFF1(500),DIFF2(500),ICOST,CFWRK1(40)
      KE = 0
C
      DO 1 I=1,N1
      Y(I) = X(I)
    1 CONTINUE
C
C
      DO 2 J=1,N2
      Z(J) = X(N1+J)
    2 CONTINUE
C
C COMPUTE THE SYSTEM OPEN LOOP NUMERATOR AND DENOMINATOR POLYNOMIALS
C
      CALL PMPY (COFASN,N7,Y,N1,A,N3)
      CALL PMPY (COFASD,N8,Z,N2,B,N4)
      NSN = N7-1
      NSD = N8-1
      CALL PVAL (COFASN,NSN,0.0,WFREQ(1),QNUMR,QNUMI)
      CALL PVAL (COFASD,NSD,0.0,WFREQ(1),QDENR,QDENI)
      IF (ABS(QNUMR).GT.2.9E+37) K=1
      IF (ABS(QNUMI).GT.2.9E+37) K=1
      IF (ABS(QDENR).GT.2.9E+37) K=1
      IF (ABS(QDENI).GT.2.9E+37) K=1
      IF (KE.EQ.1) WRITE (6,5)
      IF (KE.EQ.1) GO TO 4
      IF (NMINFS.EQ.1) GO TO 3
C
C CHECK FOR RIGHT HALF PLANE ROOTS OF COMPENSATOR NUMERATOR
C
      CALL PINVRT (Y,N1)
      CALL ROUTH (Y,N1,ISTN)
      CALL PINVRT (Y,N1)
      KE = ISTN
      IF (KE.EQ.1) GO TO 4
C
C CHECK FOR RIGHT HALF PLANE ROOTS OF COMPENSATOR DENOMINATOR
C
    3 CALL PINVRT (Z,N2)
      CALL ROUTH (Z,N2,ISTD)
      CALL PINVRT (Z,N2)
      KE = ISTD
```

```
      IF (KE.EQ.1) GO TO 4
    4 FETURN
C
    5 FCRMAT ('0',80('-'),/,' ',4(' * * * WARNING * * *'),/,'0','THE SIZ
     1E CF THE REAL AND IMAGINARY PARTS OF THE NUMERATOR AND DENOMINATOR
     2',/,' ','ARE CLOSE TO THE MAXIMUM VALUE THAT THE COMPUTER CAN HANC
     3LE.  FREQUENCY',/,' ','SCALING CF THE PRCBLEM IS RECOMMENDED TC AL
     4LEVIATE THE PROBLEM',/,'0',4(' * * * WARNING * * *'),/,' ',80('-')
     5)
      END
```

```
      FUNCTION FE (X)
      DIMENSION X(24), Y(12), Z(12)
      COMMON NCN,NCD,NTFN,NTFD,NOMEG,A(20),B(20),CMAG(500),DQPHSD(500),W
     1FREQ(500),NSN,NSD,COFASN(40),COFASD(40),QMAG(500),CR(500),CI(500),
     2CPHSR(500),CPHSD(500),N1,N2,N3,N4,PI,T,WS,IZOH,ADMAG(500),NTITLE(1
     32),FREQ(500),AMAG(500),NMINFS,COFWRK(40),IPLOT,WFRECS(500),SAPHSD(
     4500),SAMAG(500),DIFF1(500),DIFF2(500),ICOST,CFWRK1(40)
      PHICHK = 0.0
      IDELNU = 0
      IDELDE = 0
      PHI = 0.0
      IPHI = 0
      PSICHK = 0.0
      PSI = 0.0
      IPSI = 0
C
      DO 1 I=1,N1
      Y(I) = X(I)
    1 CONTINUE
C
C
      DO 2 I=1,N2
      Z(I) = X(N1+I)
    2 CONTINUE
C
C COMPUTE SYSTEM OPEN LOOP NUMERATOR COEFFICENTS AS BOXPLX CHANGES
C THE COMPENSATOR PARAMETERS
C
      CALL PMPY (COFASN,N7,Y,N1,A,N3)
      CALL PMPY (COFASD,N8,Z,N2,B,N4)
      NSN = N7-1
      NSD = N8-1
      COST = 0.0
C
C COMPUTE OPEN LOOP FREQUENCY RESPONSE OF SYSTEM AT THE DISCRETE
C FREQUENCIES SPECIFIED
C
      DO 20 K=1,NOMEG
      CALL PVAL (COFASN,NSN,0.0,WFREQ(K),QNUMR,QNUMI)
      CALL PVAL (COFASD,NSD,0.0,WFREQ(K),QDENR,QDENI)
      IF (ABS(QNUMR).GT.3.E+37) QNUMR=SIGN(3.E+37,QNUMR)
      IF (ABS(QNUMI).GT.3.E+37) QNUMI=SIGN(3.E+37,QNUMI)
      IF (ABS(QDENR).GT.3.E+37) QDENR=SIGN(3.E+37,QDENR)
      IF (ABS(QDENI).GT.3.E+37) QDENI=SIGN(3.E+37,QDENI)
      IF (SQRT(QDENI**2+QDENR**2)) 4,3,4
    3 CMAG(K) = 1.E+20
      CR(K) = 1.E+20
      CI(K) = 1.E+20
```

```
      GO TO 5
    4 QMAG(K) = SQRT(QNUMI**2+QNUMR**2)/SQRT(QDENI**2+QDENR**2)
C
C ADJUST MAGNITUDE RESPONSE IF ZERO ORDER HOLD IS PRESENT IN THE SYSTEM
C
      IF (IZOH.EQ.1) QMAG(K)=QMAG(K)*ABS((SIN(PI*WFREQ(K)/WS))/(PI*WFREQ
     1(K)/WS))
      QR(K) = (QNUMR*QDENR+QNUMI*QDENI)/(QDENR**2+QDENI**2)
      QI(K) = (QNUMI*QDENR-QNUMR*QDENI)/(QDENR**2+QDENI**2)
    5 IF ((QNUMI.EQ.0.0).AND.(QNUMR.EQ.0.0)) QNUMR=1.0
      ANGNUR = ATAN2(QNUMI,QNUMR)
      ANGNUD = ANGNUR*57.29578
      IF (ANGNUD.LT.0.0) ANGNUD=360.0+ANGNUD
      DELPHI = ANGNUD-PHI
      IF (IDELNU.EQ.0) GO TO 8
      IF (DELPHI.GE.0.0.AND.IDELNU.EQ.1) GO TO 8
      IF (DELPHI.LT.0.0.AND.IDELNU.EQ.1) GO TO 6
      IF (DELPHI.GE.0.0.AND.IDELNU.EQ.-1) GO TO 7
      IF (DELPHI.LT.0.0.AND.IDELNU.EQ.-1) GO TO 8
    6 IF (ABS(DELPHI).LT.180.0) GO TO 8
      DELPHI = ANGNUD+360.0-PHI
      GO TO 8
    7 IF (ABS(DELPHI).LT.180.0) GO TO 8
      DELPHI = ANGNUD-360.0-PHI
    8 PHICHK = PHICHK+DELPHI
      IPHI = PHICHK/360.0
      PHI = ANGNUD
      ANGNUD = ANGNUD+IPHI*360.0
      ANGDER = ATAN2(QDENI,QDENR)
      ANGDED = ANGDER*57.29578
      IF (ANGDED.LT.0.0) ANGDED=360.0+ANGDED
      DELPSI = ANGDED-PSI
      IF (IDELDE.EQ.0) GO TO 11
      IF (DELPSI.GE.0.0.AND.IDELDE.EQ.1) GO TO 11
      IF (DELPSI.LT.0.0.AND.IDELDE.EQ.1) GO TO 9
      IF (DELPSI.GE.0.0.AND.IDELDE.EQ.-1) GO TO 10
      IF (DELPSI.LT.0.0.AND.IDELDE.EQ.-1) GO TO 11
    9 IF (ABS(DELPSI).LT.180.0) GO TO 11
      DELPSI = ANGDED+360.0-PSI
      GO TO 11
   10 IF (ABS(DELPSI).LT.180.0) GO TO 11
      DELPSI = ANGDED-360.0-PSI
   11 PSICHK = PSICHK+DELPSI
      IPSI = PSICHK/360.0
      PSI = ANGDED
      ANGDED = ANGDED+IPSI*360.0
      IF (DELPHI.GE.0.0) IDELNU=1
      IF (DELPHI.LT.0.0) IDELNU=-1
```

```
      IF (DELPSI.GE.0.0) IDELDE=1
      IF (DELPSI.LT.0.0) IDELDE=-1
      QPHSD(K) = ANGNUD-ANGDED
      QPHSR(K) = ANGNUR-ANGDER
      IF (IZOH.NE.1) GO TO 12
      IWS = WFREQ(K)/WS
C
C ADJUST PHASE RESPONSE IF ZERO ORDER HOLD IS PRESENT IN THE SYSTEM
C
      QPHSD(K) = QPHSD(K)-(PI*WFREQ(K)/WS)*57.29578-IWS*180.0
   12 IF (ICOST) 14,14,13
   13 GO TO (14,15,16,17,18,19), ICOST
   14 COST = COST+(1.0-QMAG(K)/DMAG(K))**2+(1.0-QPHSD(K)/DQPHSD(K))**2
      GO TO 20
   15 COST = COST+(1.0-QMAG(K)/DMAG(K))**2
      GO TO 20
   16 COST = COST+(1.0-QPHSD(K)/DQPHSD(K))**2
      GO TO 20
   17 COST = COST+ABS(1.0-QMAG(K)/DMAG(K))+ABS(1.0-QPHSD(K)/DQPHSD(K))
      GO TO 20
   18 COST = COST+ABS(1.0-QMAG(K)/DMAG(K))
      GO TO 20
   19 COST = COST+ABS(1.0-QPHSD(K)/DQPHSD(K))
   20 CONTINUE
C
      FE = COST
      RETURN
C
      END
```

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++C
C++++++++++++++++++++       SUBROUTINE PMPY      ++++++++++++++++++++C
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++C
C++++                                                            ++++C
C++++ PURPOSE: TO MULTIPLY THE COEFFICIENTS OF TWO POLYNOMIALS AND ++++C
C++++          RETURN THE COEFFICIENTS OF THE RESULTING POLYNOMIAL. ++++C
C++++          POLYNOMIALS TO BE MULTIPLIED MUST HAVE REAL        ++++C
C++++          COEFFICIENTS, BUT NEED NOT BE OF THE SAME ORDER.   ++++C
C++++                                                            ++++C
C++++ CALLING SEQUENCE:                                          ++++C
C++++ CALL PMPY(Z,IDIMZ,X,IDIMX,Y,IDIMY)                         ++++C
C++++                                                            ++++C
C++++ DESCRIPTION OF PARAMETERS:                                 ++++C
C++++ Z        = VECTOR OF RESULTANT COEFFICIENTS, ORDERED IN    ++++C
C++++            ASCENDING POWERS OF THE INDEPENDENT VARIABLE.   ++++C
C++++ IDIMZ    = CALCULATED DIMENSION OF THE RESULTING POLYNOMIAL ++++C
C++++            VECTOR Z.                                       ++++C
C++++ X        = VECTOR OF COEFFICIENTS OF FIRST POLYNOMIAL TO BE ++++C
C++++            MULTIPLIED IN ASCENDING POWERS OF THE INDEPENDENT ++++C
C++++            VARIABLE.                                       ++++C
C++++ IDIMX    = DIMENSION OF THE VECTOR X.                      ++++C
C++++ Y        = VECTOR OF COEFFICIENTS OF THE SECOND POLYNOMIAL TO ++++C
C++++            BE MULTIPLIED ORDERED IN ASCENDING POWERS OF THE ++++C
C++++            INDEPENDENT VARIABLE.                           ++++C
C++++ IDIMY    = DIMENSION OF THE VECTOR Y.                      ++++C
C++++                                                            ++++C
C++++ SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED:             ++++C
C++++     NONE                                                   ++++C
C++++                                                            ++++C
C++++ METHOD:                                                    ++++C
C++++ THE COEFFICIENTS OF THE RESULTANT POLYNOMIAL ARE CALCULATED ++++C
C++++ AS THE SUM OF PRODUCTS OF THE COEFFICIENTS OF THE INPUT    ++++C
C++++ POLYNOMIALS WHOSE EXPONENTS ADD UP TO FORM THE CORRESPONDING ++++C
C++++ EXPONENT OF THE RESULTING COEFFICIENTS.                    ++++C
C++++                                                            ++++C
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++C
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++C
C
      SUBROUTINE PMPY (Z,IDIMZ,X,IDIMX,Y,IDIMY)
      DIMENSION Z(1), X(1), Y(1)
C
      IF (IDIMX*IDIMY) 1,1,2
    1 IDIMZ = 0
      GO TO 5
    2 IDIMZ = IDIMX+IDIMY-1
C
      DO 3 I=1,IDIMZ
    3 Z(I) = 0.
```

```
C
C
      DO 4 I=1,IDIMX
C
      DO 4 J=1,IDIMY
      K = I+J-1
    4 Z(K) = X(I)*Y(J)+Z(K)
C
    5 RETURN
      END
```

```
C     ..................................................................
C
C        SUBROUTINE BOXPLX                 (CATEGORY HO)
C
C        PURPOSE
C
C           BOXPLX IS A SUBROUTINE USED TO SOLVE THE PROBLEM OF LOCATING
C           A MINIMUM (OR MAXIMUM) OF AN ARBITRARY OBJECTIVE FUNCTION
C           SUBJECT TO ARBITRARY EXPLICIT AND/OR IMPLICIT CONSTRAINTS BY
C           THE COMPLEX METHOD OF M.J. BOX.  EXPLICIT CONSTRAINTS ARE
C           DEFINED AS UPPER AND LOWER BOUNDS ON THE INDEPENDENT VARIABLES.
C           IMPLICIT CONSTRAINTS MAY BE ARBITRARY FUNCTIONS OF THE VAR-
C           IABLES.  TWO FUNCTION SUBPROGRAMS TO EVALUATE THE OBJECTIVE
C           FUNCTION AND IMPLICIT CONSTRAINTS, RESPECTIVELY, MUST BE
C           SUPPLIED BY THE USER (SEE EXAMPLE BELOW).  BOXPLX ALSO HAS
C           THE OPTION TO PERFORM INTEGER PROGRAMMING, WHERE THE VALUES
C           OF THE INDEPENDENT VARIABLES ARE RESTRICTED TO INTEGERS.
C
C        USAGE
C
C           CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)
C
C
C        DESCRIPTION OF PARAMETERS
C
C           NV    AN INTEGER INPUT DEFINING THE NUMBER OF INDEPENDENT
C                 VARIABLES OF THE OBJECTIVE FUNCTION TO BE MINIMIZED.
C                 NOTE:  MAXIMUM NV + NAV IS PRESENTLY 50.  MAXIMUM NV IS
C                 25.  IF THESE LIMITS MUST BE EXCEEDED, PUNCH A SOURCE
C                 DECK IN THE USUAL MANNER, AND CHANGE THE DIMENSION
C                 STATEMENTS.
C
C           NAV   AN INTEGER INPUT DEFINING THE NUMBER OF AUXILIARY VAR-
C                 IABLES THE USER WISHES TO DEFINE FOR HIS OWN CONVENIENCE.
C                 TYPICALLY HE MAY WISH TO DEFINE THE VALUE OF EACH IMPLICIT
C                 CONSTRAINT FUNCTION AS AN AUXILIARY VARIABLE.  IF THIS
C                 IS DONE, THE OPTIONAL OUTPUT FEATURE OF BOXPLX CAN BE
C                 USED TO OBSERVE THE VALUES OF THOSE CONSTRAINTS AS THE
C                 SOLUTION PROGRESSES.  AUXILIARY VARIABLES, IF USED,
C                 SHOULD BE EVALUATED IN FUNCTION KE (DEFINED BELOW).
C                 NAV MAY BE ZERO.
C
C           NPR   INPUT INTEGER CONTROLLING THE FREQUENCY OF OUTPUT DESIRED
C                 FOR DIAGNOSTIC PURPOSES.  IF NPR .LE. 0, NO OUTPUT WILL BE
C                 PRODUCED BY BOXPLX.  OTHERWISE, THE CURRENT COMPLEX OF
C                 K= 2*NV VERTICES AND THEIR CENTROID WILL BE OUTPUT AFTER
C                 EACH NPR PERMISSIBLE TRIALS.  THE NUMBER OF TOTAL TRIALS,
C                 NUMBER OF FEASIBLE TRIALS, NUMBER OF FUNCTION EVALUATIONS
```

```
C           AND NUMBER OF IMPLICIT CONSTRAINT EVALUATIONS ARE IN-
C           CLUDED IN THE OUTPUT.
C           ADDITIONALLY, (WHEN NPR .GT. 0) THE SAME INFORMATION
C           WILL BE OUTPUT:
C
C             1) IF THE INITIAL POINT IS NOT FEASIBLE,
C             2) AFTER THE FIRST COMPLETE COMPLEX IS GENERATED,
C             3) IF A FEASIBLE VERTEX CANNOT BE FOUND AT SOME TRIAL,
C             4) IF THE OBJECTIVE VALUE OF A VERTEX CANNOT BE MADE
C                NO-LONGER-WORST.
C             5) IF THE LIMIT ON TRIALS (NTA) IS REACHED AND,
C             6) WHEN THE OBJECTIVE FUNCTION HAS BEEN UNCHANGED FOR
C                2*NV TRIALS, INDICATING A LOCAL MINIMUM HAS BEEN
C                FOUND.
C
C           IF THE USER WISHES TO TRACE THE PROGRESS OF A SOLUTION,
C           A CHOICE OF NPR = 25, 50 OR 100 IS RECOMMENDED.
C
C     NTA   INTEGER INPUT OF LIMIT ON THE NUMBER OF TRIALS ALLOWED
C           IN THE CALCULATION.  IF THE USER INPUTS NTA .LE. 0, A
C           DEFAULT VALUE OF 2000 IS USED.  WHEN THIS LIMIT IS REACHED
C           CONTROL RETURNS TO THE CALLING PROGRAM WITH THE BEST
C           ATTAINED OBJECTIVE FUNCTION VALUE IN YMN, AND THE BEST
C           ATTAINED SOLUTION POINT IN XS.
C
C     R     A REAL NUMBER INPUT TO DEFINE THE FIRST RANDOM NUMBER
C           USED IN DEVELOPING THE INITIAL COMPLEX OF 2*NV VERTICIES.
C           (0. .GT. R .LT. 1.) IF R IS NOT WITHIN THESE BOUNDS,
C           IT WILL BE REPLACED BY 1./3. .
C
C     XS    INPUT REAL ARRAY DIMENSIONED AT LEAST NV+NAV. THE FIRST
C           NV MUST CONTAIN A FEASIBLE ORIGIN FOR STARTING THE CAL-
C           CULATION.  THE LAST NAV NEED NOT BE INITIALIZED.  UPON
C           RETURN FROM BOXPLX, THE FIRST NV ELEMENTS OF THE ARRAY
C           CONTAIN THE COORDINATES OF THE MINIMUM OBJECTIVE FUNCTION,
C           AND THE REMAINING NAV (NAV .GE. 0) CONTAIN THE VALUES OF
C           THE CORRESPONDING AUXILIARY VARIABLES.
C
C     IP    INTEGER INPUT FOR OPTIONAL INTEGER PROGRAMMING.  IF IP=1,
C           THE VALUES OF THE INDEPENDENT VARIABLES WILL BE REPLACED
C           WITH INTEGER VALUES (STILL STORED AS REAL*4).
C
C     XU    A REAL ARRAY DIMENSIONED AT LEAST NV INPUTTING THE UPPER
C           BOUND ON EACH INDEPENDENT VARIABLE, (EACH EXPLICIT CON-
C           STRAINT).  INPUT VALUES ARE SLIGHTLY ALTERED BY BOXPLX.
C
C     XL    A REAL ARRAY DIMENSIONED AT LEAST NV INPUTTING THE LOWER
C           BOUND ON EACH INDEPENDENT VARIABLE, (EACH EXPLICIT CON-
```

```
C                  STRAINT).  NOTE:  FOR BOTH XU AND XL CHOOSE REASONABLE
C                  VALUES IF NONE ARE GIVEN, NOT VALUES WHICH ARE MAGNITUDES
C                  ABOVE OR BELOW THE EXPECTED SOLUTION.  INPUT VALUES ARE
C                  SLIGHTLY ALTERED BY BOXPLX.
C
C              YMN THIS OUTPUT IS THE VALUE (REAL*4) OF THE OBJECTIVE FUNC-
C                  TION, CORRESPONDING TO THE SOLUTION POINT OUTPUT IN XS.
C
C              IER INTEGER ERROR RETURN.  TO BE INTERROGATED UPON RETURN
C                  FROM BOXPLX.  IER WILL BE ONE OF THE FOLLOWING:
C
C                    =-1   CANNOT FIND FEASIBLE VERTEX OR FEASIBLE CENTROID
C                          AT THE START OR A RESTART (SEE 'METHOD' BELOW).
C                    =0    FUNCTION VALUE UNCHANGED FOR 'N' TRIALS.  (WHERE
C                          N=6*NV+10)  THIS IS THE NORMAL RETURN PARAMETER.
C                    =1    CANNOT DEVELOP FEASIBLE VERTEX.
C                    =2    CANNOT DEVELOP A NO-LONGER-WORST VERTEX.
C                    =3    LIMIT ON TRIALS REACHED.  (NTA EXCEEDED)
C                    NOTE:  VALID RESULTS MAY BE RETURNED IN ANY OF THE
C                          ABOVE CASES.
C
C          EXAMPLE OF USAGE
C
C              THIS EXAMPLE MINIMIZES THE OBJECTIVE FUNCTION SHOWN IN THE
C              EXTERNAL FUNCTION FE(X).  THERE ARE TWO INDEPENDENT VAR-
C              IABLES X(1) & X(2), AND TWO IMPLICIT CONSTRAINT FUNCTIONS
C              X(3) & X(4) WHICH ARE EVALUATED AS AUXILIARY VARIABLES (SEE
C              EXTERNAL FUNCTION KE(X) ).
C
C          DIMENSION  XS(4),XU(2),XL(2)
C
CC      STARTING GUESS
C          XS(1) = 1.0
C          XS(2) = 0.5
CC      UPPER LIMITS
C          XL(1) = 6.0
C          XL(2) = 6.0
CC      LOWER LIMITS
C          XL(1) = 0.0
C          XL(2) = 0.0
C
C          R = 9./13.
C          NTA = 5000
C          NFR = 50
C          NAV = 2
C          NV = 2
C          IP = 0
C
```

```
C       CALL BCXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)
C       WRITE(6,1) ((XS(I),I=1,4),YMN,IER)
C     1 FORMAT (////,'    THE POINT IS LOCATED AT (XS(I)=) ',4(E13.7,5X),
C      1//,'    AND THE FUNCTION VALUE IS ',E13.7,' IER = ',I5)
C
C       STOP
C       END
C
C
C       FUNCTION KE(X)
CC    EVALUATE CONSTRAINTS.  SET KE=0 IF NO IMPLICIT CONSTRAINT IS
CC    VIOLATED, OR SET KE=1 IF ANY IMPLICIT CONSTRAINT IS VIOLATED.
C       DIMENSION X(4)
C       X1 = X(1)
C       X2 = X(2)
C       KE = 0
C       X(3) = X1 + 1.732051*X2
C       IF (X(3) .LT. 0. .OR. X(3) .GT. 6.) GO TO 1
C       X(4) = X1/1.732051 -X2
C       IF (X(4) .GE. 0.)  RETURN
C
C     1 KE = 1
C       RETURN
C       END
C
C
C       FUNCTION FE(X)
C       DIMENSION X(4)
C
CC    THIS IS THE OBJECTIVE FUNCTION.
C       FE= -(X(2)**3 *(9.-(X(1)-3.)**2)/(46.76538))
C       RETURN
C       END
C
C       METHOD
C
C          THE COMPLEX METHOD IS AN EXTENSION AND ADAPTION OF THE SIM-
C          PLEX METHOD OF LINEAR PROGRAMMING.  STARTING WITH ANY ONE
C          FEASIBLE POINT IN N-DIMENSION SPACE A "COMPLEX" OF 2*N
C          VERTICES IS CONSTRUCTED BY SELECTING RANDOM POINTS WITHIN THE
C          FEASIBLE REGION.  FOR THIS PURPOSE N COORDINATES ARE FIRST
C          RANDOMLY CHOSEN WITHIN THE SPACE BOUNDED BY EXPLICIT CON-
C          STRAINTS.  THIS DEFINES A TRIAL INITIAL VERTEX.  IT IS THEN
C          CHECKED FOR POSSIBLE VIOLATION OF IMPLICIT CONSTRAINTS.  IF
C          ONE OR MORE ARE VIOLATED, THE TRIAL INITIAL VERTEX IS DISPLACED
C          HALF OF ITS DISTANCE FROM THE CENTROID OF PREVIOUSLY SELECTED
C          INITIAL VERTICES.  IF NECESSARY THIS DISPLACEMENT PROCESS IS
C          REPEATED UNTIL THE VERTEX HAS BECOME FEASIBLE.  IF THIS FAILS
```

```
C        TO HAPPEN AFTER 5*N+10 DISPLACEMENTS, THE SOLUTION IS ABAND-   BXPL1930
C        ONED.  AFTER EACH VERTEX IS ADDED TO THE COMPLEX, THE CURRENT  BXPL1940
C        CENTROID IS CHECKED FOR FEASIBILITY.  IF IT IS INFEASIBLE,     BXPL1950
C        THE LAST TRIAL VERTEX IS ABANDONED AND AN EFFORT TO GENERATE   BXPL1960
C        AN ALTERNATIVE TRIAL VERTEX IS MADE.  IF 5*N+10 VERTICES ARE   BXPL1970
C        ABANDONED CONSECUTIVELY, THE SOLUTION IS TERMINATED.           BXPL1980
C                                                                       BXPL1990
C        IF AN INITIAL COMPLEX IS ESTABLISHED, THE BASIC COMPUTATION    BXPL2000
C        LOOP IS INITIATED.  THESE INSTRUCTIONS FIND THE CURRENT WORST  BXPL2010
C        VERTEX, THAT IS, THE VERTEX WITH THE LARGEST CORRESPONDING     BXPL2020
C        VALUE FOR THE OBJECTIVE FUNCTION, AND REPLACE THAT VERTEX BY   BXPL2030
C        ITS OVER-REFLECTION THROUGH THE CENTROID OF ALL OTHER VERTICES.BXPL2040
C        (IF THE VERTEX TO BE REPLACED IS CONSIDERED AS A VECTOR IN     BXPL2050
C        N-SPACE, ITS OVER-REFLECTION IS OPPOSITE IN DIRECTION, IN-     BXPL2060
C        CREASED IN LENGTH BY THE FACTOR 1.3, AND COLLINEAR WITH        BXPL2070
C        THE REPLACED VERTEX AND CENTROID OF ALL OTHER VERTICES.)       BXPL2080
C                                                                       BXPL2090
C        WHEN AN OVER-REFLECTION IS NOT FEASIBLE OR REMAINS WORST, IT   BXPL2100
C        IS CONSIDERED NOT-PERMISSIBLE AND IS DISPLACED HALFWAY TOWARD  BXPL2110
C        THE CENTROID.  AFTER FOUR SUCH ATTEMPTS ARE MADE UNSUCCESSFULLYBXPL2120
C        EVERY FIFTH ATTEMPT IS MADE BY REFLECTING THE OFFENDING VERTEX BXPL2130
C        THROUGH THE PRESENT BEST VERTEX, INSTEAD OF THROUGH THE CEN-   BXPL2140
C        TROID.  IF 5*N+10 DISPLACEMENTS AND OVER-REFLECTIONS OCCUR     BXPL2150
C        WITHOUT A SUCCESSFUL (PERMISSIBLE) RESULT, THE CURRENT BEST    BXPL2160
C        VERTEX IS TAKEN AS AN INITIAL FEASIBLE POINT FOR A RESTART     BXPL2170
C        RUN OF THE COMPLETE PROCESS.  RESTARTING IS ALSO UNDERTAKEN    BXPL2180
C        WHEN 6*NV+10 CONSECUTIVE TRIALS HAVE BEEN MADE WITH NO SIGNIF- BXPL2190
C        ICANT CHANGE IN THE VALUE OF THE OBJECTIVE FUNCTION.  IN ALL   BXPL2200
C        CASES, RESTARTING IS INHIBITED IF THE LAST RESTART DID NOT     BXPL2210
C        PRODUCE A SIGNIFICANT IMPROVEMENT IN THE MINIMUM ATTAINED.     BXPL2220
C                                                                       BXPL2230
C        IT IS RECOMMENDED THAT THE USER READ THE REFERENCE FOR         BXPL2240
C        FURTHER USEFUL INFORMATION.  IT SHOULD BE NOTED THAT THE       BXPL2250
C        ALGORITHM DEFINED THERE HAS BEEN ALTERED TO FIND THE           BXPL2260
C        CONSTRAINED MINIMUM, RATHER THAN THE MAXIMUM.                  BXPL2270
C                                                                       BXPL2280
C                                                                       BXPL2290
C     REMARKS                                                           BXPL2300
C                                                                       BXPL2310
C        THE INTEGER PROGRAMMING OPTION WAS ADDED TO THIS PROGRAM       BXPL2320
C        AS SUGGESTED IN REFERENCE (2).  A MIXED INTEGER/CONTINUOUS     BXPL2330
C        VARIABLE VERSION OF BOXPLX WOULD BE EASY TO CREATE BY DE-      BXPL2340
C        CLARING "IP" TO BE AN ARRAY OF NV CONTROL VARIABLES WHERE IP   BXPL2350
C        (I)=1 WOULD INDICATE THAT THE I-TH VARIABLE IS TO BE CONFINED  BXPL2360
C        TO INTEGER VALUES.  EACH STATEMENT OF THE FORM 'IF (IP .EQ.    BXPL2370
C        1)' ETC. WOULD THEN NEED TO BE ALTERED TO 'IF (IP(I) .EQ. 1)'  BXPL2380
C        ETC., WHERE THE SUBSCRIPT IS APPROPRIATELY CHOSEN.  NORMALLY,  BXPL2390
C        XU AND XL VALUES ARE ALTERED TO BE AN EPSILON 'WITHIN' ACTUAL  BXPL2400
```

```
C         VALUES DECLARED BY THE USER.  THIS ADJUSTMENT IS NOT MADE
C         WHEN IP=1.
C
C         NOTE:  NO NON-LINEAR PROGRAMMING ALGORITHM CAN GUARANTEE THAT
C         THE ANSWER FOUND IS THE GLOBAL MINIMUM, RATHER THAN JUST A
C         LOCAL MINIMUM.  HOWEVER, ACCORDING TO REF.2, THE COMPLEX
C         METHOD HAS AN ADVANTAGE IN THAT IT TENDS TO FIND THE GLOBAL
C         MINIMUM MORE FREQUENTLY THAN MANY OTHER NON-LINEAR PROGRAM-
C         MING ALGORITHMS.
C
C         IT SHOULD BE NOTED THAT THE AUXILIARY VARIABLE FEATURE CAN
C         ALSO BE USED TO DEAL WITH PROBLEMS CONTAINING EQUALITY CON-
C         STRAINTS.  ANY EQUALITY CONSTRAINT IMPLIES THAT A GIVEN VAR-
C         IABLE IS NOT TRULY INDEPENDENT.  THEREFORE, IN GENERAL, ONE
C         VARIABLE INVOLVED IN AN EQUALITY CONSTRAINT CAN BE RENUMBERED
C         FROM THE SET OF NV INDEPENDENT VARIABLES AND ADDED TO THE SET
C         OF NAV AUXILIARY VARIABLES.  THIS USUALLY INVOLVES RENUMBERING
C         THE INDEPENDENT VARIABLES OF THE GIVEN PROBLEM.
C
C      SUBROUTINES AND FUNCTIONS REQUIRED
C
C         SUBROUTINE 'BOUT' AND FUNCTION 'FBV' ARE INTEGRAL PARTS OF
C         THE BOXPLX PACKAGE.
C
C         TWO FUNCTIONS MUST BE SUPPLIED BY THE USER.  THE FIRST, KE(X),
C         IS USED TO EVALUATE THE IMPLICIT CONSTRAINTS.  SET KE=0 AT THE
C         BEGINING OF THE FUNCTION, THEN EVALUATE THE IMPLICIT CON-
C         STRAINTS.  IN THE EXAMPLE ABOVE, THE FIRST CONSTRAINT, X(3),
C         MUST BE WITHIN THE RANGE (0. .LE. X(3) .LE. 6.).  THE SECOND
C         CONSTRAINT X(4), MUST BE .GE. 0. .  IF EITHER CONSTRAINT IS
C         NOT WITHIN THESE BOUNDS, CONTROL IS TRANSFERRED TO STATEMENT 1,
C         AND KE IS SET TO "1" AND CONTROL IS RETURNED TO BOXPLX.
C
C         THE SECOND FUNCTION THE USER MUST PROVIDE EVALUATES THE OB-
C         JECTIVE FUNCTION.  IT IS CALLED FE(X) AS SHOWN IN THE EXAM-
C         PLE ABOVE, AND FE MUST BE SET TO THE VALUE OF THE OBJECTIVE
C         FUNCTION CORRESPONDING TO CURRENT VALUES OF THE NV INDEPENDENT
C         VARIABLES IN ARRAY 'X'.
C
C      REFERENCES
C
C         BOX, M. J., "A NEW METHOD OF CONSTRAINED OPTIMIZATION AND A
C         COMPARISON WITH OTHER METHODS", COMPUTER JOURNAL, 8 APR. '65,
C         PP. 45-52.
C
C         BEVERIDGE G., AND SCHECHTER R., "OPTIMIZATION: THEORY AND
C         PRACTICE", MCGRAW-HILL, 1970.
C
```

```
C     PROGRAMMER                                                        BXPL2890
C                                                                       BXPL2900
C        R.R. HILLEARY 1/1966.                                          BXPL2910
C        REVISED FOR SYSTEM 360 4/1967                                  BXPL2920
C        CORRECTED  1/1969                                              BXPL2930
C        REVISED/EXTENDED BY L.NOLAN/R.HILLEARY  2/1975                 BXPL2940
C        MODIFIED FOR USE IN CALICO PROGRAM 1/76                        BX7L2950
C                                                                       BXPL2960
C     ..................................................................BXPL2970
C                                                                       BXPL2980
C                                                                       BXPL2990
      SUBROUTINE BOXPLX (NV,NAV,NPR,NTZ,RZ,XS,IP,BU,BL,YMN,IER)         BXPL3000
C                                                                       BXPL3010
      DIMENSION V(50,50), FUN(50), SUM(25), CEN(25), XS(NV), BL(NV), BL(BXPL3020
     1NV)                                                               BXPL3030
      COMMON NCN,NCD,NTFN,NTFD,NOMEG,A(20),B(20),CMAG(500),DQPHSD(500),WBXPL3040
     1FREQ(500),NSN,NSD,COFASN(40),COFASD(40),QMAG(500),CR(500),CI(500),BXPL3050
     2QPHSR(500),QPHSD(500),N1,N2,N3,N4,PI,T,WS,IZCH,ADMAG(500),NTITLE(1BXPL3060
     32),FREQ(500),AMAG(500),NMINFS,COFWRK(40),IPLOT,WFREQS(500),SAPHSD(BXPL3070
     4500),SAMAG(500),DIFF1(500),DIFF2(500),ICOST,CFWRK1(40)            BXPL3080
C                                                                       BXPL3090
      KV = 5                                                            BXPL3100
      EF = 1.E-6                                                        BXPL3110
      NTA = 2000                                                        BXPL3120
      IF (NTZ.GT.0) NTA = NTZ                                           BXPL3130
      R = RZ                                                            BXPL3140
      IF (R.LE.0..OR.R.GE.1.) R=1./3.                                   BXPL3150
      NVT = NV+NAV                                                      BXPL3160
C                                                                       BXPL3170
C           TOTAL VARS, EXPLICIT PLUS IMPLICIT                          BXPL3180
      NT = 0                                                            BXPL3190
C           CURRENT TRIAL NO.                                           BXPL3200
      NPT = 0                                                           BXPL3210
C           CURRENT NO. OF PERMISSIBLE TRIALS                           BXPL3220
      NTFS = 0                                                          BXPL3230
C           CURRENT NO. OF TIMES F HAS BEEN ALMOST UNCHANGED            BXPL3240
C                                                                       BXPL3250
C              CHECK FEASIBILITY OF START POINT                         BXPL3260
C                                                                       BXPL3270
C                                                                       BXPL3280
      DO 4 I=1,NV                                                       BXPL3290
      VT = XS(I)                                                        BXPL3300
      IF (BL(I).LE.VT) GO TO 1                                          BXPL3310
      II = -I                                                           BXPL3320
      VT = BL(I)                                                        BXPL3330
      GO TO 2                                                           BXPL3340
    1 IF (BU(I).GE.VT) GO TO 3                                          BXPL3350
      II = I                                                            BXPL3360
```

```
      VT = BU(I)                                                        BXPL3370
    2 IF (NPR.GT.0) WRITE (6,51) II                                     BXPL3380
    3 V(I,1) = VT                                                       BXPL3390
      CEN(I) = VT                                                       BXPL3400
      IF (IP.EQ.1) GO TO 4                                              BXPL3410
      BL(I) = BL(I)+AMAX1(EP,EP*ABS(BL(I)))                             BXPL3420
      BL(I) = BU(I)-AMAX1(EP,EP*ABS(BU(I)))                             BXPL3430
    4 SUM(I) = VT                                                       BXPL3440
C                                                                       BXPL3450
C                                                                       BXPL3460
C                                                                       BXPL3470
      NCE = 1                                                           BXPL3480
C              NUMBER OF CONSTRAINT EVALUATIONS                         BXPL3490
      I = 1                                                             BXPL3500
      IF (KE(V(1,1)).EQ.0) GO TO 5                                      BXPL3510
      IF (NPR.LE.0) GO TO 12                                            BXPL3520
      WRITE (6,52)                                                      BXPL3530
      GO TO 12                                                          BXPL3540
    5 NFE = 1                                                           BXPL3550
C                                                                       BXPL3560
C    NUMBER OF VERTICES (K) = 2 TIMES NO. OF VARIABLES.                 BXPL3570
      K = 2*NV                                                          BXPL3580
C                                                                       BXPL3590
C    NUMBER OF DISPLACEMENTS ALLOWED.                                   BXPL3600
      NLIM = 5*NV+10                                                    BXPL3610
C                                                                       BXPL3620
C    NUMBER OF CONSECUTIVE TRIALS WITH UNCHANGED FE TO TERMINATE.       BXPL3630
      NCT = NLIM+NV                                                     BXPL3640
      ALPHA = 1.3                                                       BXPL3650
      FK = K                                                            BXPL3660
      FKM = FK-1.                                                       BXPL3670
      BETA = ALPHA+1.                                                   BXPL3680
C                                                                       BXPL3690
C    INSURE SEED OF RANDOM NUMBER GENERATOR IS ODD.                     BXPL3700
      IQR = R*1.E7                                                      BXPL3710
      IF (MOD(IQR,2).EQ.0) IQR=IQR+101                                  BXPL3720
C                                                                       BXPL3730
C                        SET UP INITIAL VERTICES                        BXPL3740
      FUN(1) = FE(V(1,1))                                               BXPL3750
      YMN = FUN(1)                                                      BXPL3760
    6 FI = 1.                                                           BXPL3770
      FUNOLD = FUN(1)                                                   BXPL3780
C                                                                       BXPL3790
C                                                                       BXPL3800
      DO 17 I=2,K                                                       BXPL3810
      FI = FI+1.                                                        BXPL3820
      LIMT = 0                                                          BXPL3830
    7 LIMT = LIMT+1                                                     BXPL3840
```

```
C
C     END CALCULATION IF FEASIBLE CENTROID CANNOT BE FOUND.
        IF (LIMT.GE.NLIM) GO TO 11
C
C
        DO 8 J=1,NV
C
C     RANDOM NUMBER GENERATOR  (RANDU)
        IQR = IQR*65539
        IF (IQR.LT.0) IQR = IQR+2147483647+1
        RQX = IQR
        RQX = RQX*.4656613E-9
        V(J,I) = BL(J)+RQX*(BL(J)-BL(J))
        IF (IP.EQ.1) V(J,I)=AINT(V(J,I)+.5)
      8 CONTINUE
C
C
C
        DO 10 L=1,NLIM
        NCE = NCE+1
        IF (KE(V(1,I)).EQ.0) GO TO 13
C
C
        DO 9 J=1,NV
        VT = .5*(V(J,I)+CEN(J))
        IF (IP.EQ.1) VT = AINT(VT+.5)
        V(J,I) = VT
      9 CONTINUE
C
C
     10 CONTINUE
C
C
     11 IF (NPR.LE.0) GO TO 12
        WRITE (6,53) I
        CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,I,FUN,CEN,I)
     12 IER = -1
        GO TO 50
C
C
     13 DO 14 J=1,NV
        SUM(J) = SUM(J)+V(J,I)
     14 CEN(J) = SUM(J)/FI
C
C
C     TRY TO ASSURE FEASIBLE CENTROID FOR STARTING.
        NCE = NCE+1
        IF (KE(CEN).EQ.0) GO TO 16
```

```
C
      DO 15 J=1,NV
   15 SUM(J) = SUM(J)-V(J,I)
C
      GO TO 7
   16 NFE = NFE+1
      FUN(I) = FE(V(1,I))
   17 CONTINUE
C
C
C     END OF LOOP SETTING OF INITIAL COMPLEX.
      IF (NPR.LE.0) GO TO 19
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CENT,0)
C
C     FIND THE WORST VERTEX, THE 'J'TH.
      J = 1
C
C
      DO 18 I=2,K
      IF (FUN(J).GE.FUN(I)) GO TO 18
      J = I
   18 CONTINUE
C
C
C     BASIC LOOP.  ELIMINATE EACH WORST VERTEX IN TURN.  IT MUST BECOME
C     NO LONGER WORST, NOT MERELY IMPROVED.  FIND NEXT-TO-WORST VERTEX,
C     THE 'JN'TH ONE.
   19 JN = 1
      IF (J.EQ.1) JN = 2
C
C
      DO 20 I=1,K
      IF (I.EQ.J) GO TO 20
      IF (FUN(JN).GE.FUN(I)) GO TO 20
      JN = I
   20 CONTINUE
C
C
C     LIMT = NUMBER OF MOVES DURING THIS TRIAL TOWARD THE CENTROID
C      DUE TO FUNCTION VALUE.
      LIMT = 1
C
C     COMPUTE CENTROID AND OVER REFLECT WORST VERTEX.
C
C
      DO 21 I=1,NV
      VT = V(I,J)
      SUM(I) = SUM(I)-VT
```

```
      CEN(I) = SUM(I)/FKM
      VT = BETA*CEN(I)-ALPHA*VT
      IF (IP.EQ.1) VT = AINT(VT+.5)
C
C    INSURE THE EXPLICIT CONSTRAINTS ARE OBSERVED.
   21 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))
C
C
      NT = NT+1
C
C    CHECK FOR IMPLICIT CONSTRAINT VIOLATION.
C
C
   22 DO 27 N=1,NLIM
      NCE = NCE+1
      IF (KE(V(1,J)).EQ.0) GO TO 28
C
C    EVERY 'KV'TH TIME, OVER-REFLECT THE OFFENDING VERTEX THROUGH THE
C    BEST VERTEX.
      IF (MOD(N,KV).NE.0) GO TO 24
      CALL FBV (K,FUN,M)
C
C
      DO 23 I=1,NV
      VT = BETA*V(I,M)-ALPHA*V(I,J)
      IF (IP.EQ.1) VT = AINT(VT+.5)
   23 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))
C
C
      GO TO 26
C
C    CONSTRAINT VIOLATION:  MOVE NEW POINT TOWARD CENTROID.
C
C
   24 DO 25 I=1,NV
      VT = .5*(CEN(I)+V(I,J))
      IF (IP.EQ.1) VT = AINT(VT+.5)
      V(I,J) = VT
   25 CONTINUE
C
C
   26 NT = NT+1
   27 CONTINUE
C
C
      IER = 1
C
C    CANNOT GET FEASIBLE VERTEX BY MOVING TOWARD CENTROID,
```

```
C     OR BY OVER-REFLECTING THRU THE BEST VERTEX.                       BXPL5290
         IF (NPR.LE.0) GO TO 44                                          BXPL5300
         WRITE (6,54) NT,J                                               BXPL5310
         CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,J)                 BXPL5320
         GO TO 44                                                        BXPL5330
C                                                                        BXPL5340
C     FEASIBLE VERTEX FOUND, EVALUATE THE OBJECTIVE FUNCTION.            BXPL5350
   28 NFE = NFE+1                                                        BXPL5360
         FUNTRY = FE(V(1,J))                                             BXPL5370
C                                                                        BXPL5380
C     TEST TO SEE IF FUNCTION VALUE HAS NOT CHANGED.                     BXPL5390
         AFC = ABS(FUNTRY-FUNOLD)                                        BXPL5400
         AMX = AMAX1(ABS(EP*FUNOLD),EP)                                  BXPL5410
C                                                                        BXPL5420
C     ACTIVATE THE FOLLOWING TWO STATEMENTS FOR DIAGNOSTIC PURPOSES ONLY. BXPL5430
C        WRITE (6,99) J,AFO,AMX,FUNTRY,FUNOLD,FUN(J),FUN(JN),NTFS,N      BXPL5440
C   99 FORMAT (1X,I3,6E15.7,2I5)                                         BXPL5450
         IF (AFO.GT.AMX) GO TO 29                                        BXPL5460
         NTFS = NTFS+1                                                   BXPL5470
         IF (NTFS.LT.NCT) GO TO 30                                       BXPL5480
         IER = 0                                                         BXPL5490
         IF (NPR.LE.0) GO TO 44                                          BXPL5500
         WRITE (6,55) K                                                  BXPL5510
         GO TO 44                                                        BXPL5520
   29 NTFS = 0                                                           BXPL5530
C                                                                        BXPL5540
C     IS THE NEW VERTEX NO LONGER WORST?                                 BXPL5550
   30 IF (FUNTRY.LT.FUN(JN)) GO TO 36                                    BXPL5560
C                                                                        BXPL5570
C     TRIAL VERTEX IS STILL WORST; ADJUST TOWARD CENTROID.               BXPL5580
C     EVERY 'KV'TH TIME, OVER-REFLECT THE OFFENDING VERTEX THROUGH THE   BXPL5590
C     BEST VERTEX.                                                       BXPL5600
         LIMT = LIMT+1                                                   BXPL5610
         IF (MOD(LIMT,KV).NE.0) GO TO 32                                 BXPL5620
         CALL FBV (K,FUN,M)                                              BXPL5630
C                                                                        BXPL5640
C                                                                        BXPL5650
         DO 31 I=1,NV                                                    BXPL5660
         VT = BETA*V(I,M)-ALPHA*V(I,J)                                   BXPL5670
         IF (IP.EQ.1) VT = AINT(VT+.5)                                   BXPL5680
   31 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))                              BXPL5690
C                                                                        BXPL5700
C                                                                        BXPL5710
         GO TO 34                                                        BXPL5720
C                                                                        BXPL5730
C                                                                        BXPL5740
   32 DO 33 I=1,NV                                                       BXPL5750
         VT = .5*(CEN(I)+V(I,J))                                         BXPL5760
```

```
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     BXPL5770
      V(I,J) = VT                                                       BXPL5780
   33 CONTINUE                                                          BXPL5790
C                                                                       BXPL5800
C                                                                       BXPL5810
   34 IF (LIMT.LT.NLIM) GO TO 35                                        BXPL5820
C                                                                       BXPL5830
C    CANNOT MAKE THE 'J'TH VERTEX NO LONGER WORST BY DISPLACING TOWARD  BXPL5840
C    THE CENTROID OR BY OVER-REFLECTING THRU THE BEST VERTEX.           BXPL5850
      IER = 2                                                           BXPL5860
      IF (NPR.GT.0) WRITE (6,54) NT,J                                   BXPL5870
      GO TO 44                                                          BXPL5880
   35 NT = NT+1                                                         BXPL5890
      GO TO 22                                                          BXPL5900
C                                                                       BXPL5910
C    SUCCESS:  WE HAVE A REPLACEMENT FOR VERTEX J.                      BXPL5920
   36 FUN(J) = FUNTRY                                                   BXPL5930
      FUNOLD = FUNTRY                                                   BXPL5940
      NFT = NFT+1                                                       BXPL5950
C                                                                       BXPL5960
C    EVERY 100'TH PERMISSIBLE TRIAL, RECOMPUTE CENTROID SUMMATION TO    BXPL5970
C    AVOID CREEPING ERROR.                                              BXPL5980
      IF (MOD(NPT,100).NE.0) GO TO 39                                   BXPL5990
C                                                                       BXPL6000
C                                                                       BXPL6010
      DO 38 I=1,NV                                                      BXPL6020
      SUM(I) = 0.                                                       BXPL6030
C                                                                       BXPL6040
C                                                                       BXPL6050
      DO 37 N=1,K                                                       BXPL6060
   37 SUM(I) = SUM(I)+V(I,N)                                            BXPL6070
C                                                                       BXPL6080
C                                                                       BXPL6090
      CEN(I) = SUM(I)/FK                                                BXPL6100
   38 CONTINUE                                                          BXPL6110
C                                                                       BXPL6120
C                                                                       BXPL6130
      LC = 0                                                            BXPL6140
      GO TO 41                                                          BXPL6150
C                                                                       BXPL6160
C                                                                       BXPL6170
   39 DO 40 I=1,NV                                                      BXPL6180
   40 SUM(I) = SUM(I)+V(I,J)                                            BXPL6190
C                                                                       BXPL6200
C                                                                       BXPL6210
      LC = J                                                            BXPL6220
C                                                                       BXPL6230
   41 IF (NPR.LE.0) GO TO 42                                            BXPL6240
```

```
      IF (MOD(NFT,NPR).NE.0) GO TO 42                                   BXPL6250
C                                                                       BXPL6260
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,LC)                  BXPL6270
C                                                                       BXPL6280
C   HAS THE MAX. NUMBER OF TRIALS BEEN REACHED WITHOUT CONVERGENCE?     BXPL6290
C   IF NOT, GO TO NEW TRIAL.                                            BXPL6300
   42 IF (NT.GE.NTA) GO TO 43                                           BXPL6310
C                                                                       BXPL6320
C   NEXT-TO-WORST VERTEX NOW BECOMES WORST.                             BXPL6330
      J = JN                                                            BXPL6340
      GO TO 19                                                          BXPL6350
   43 IER = 3                                                           BXPL6360
      IF (NPR.GT.0) WRITE (6,56)                                        BXPL6370
C                                                                       BXPL6380
C   COLLECTOR POINT FOR ALL ENDINGS.                                    BXPL6390
C    1)  CANNOT DEVELOP FEASIBLE VERTEX.              IER =  1          BXPL6400
C    2)  CANNOT DEVELOP A NO-LONGER-WORST VERTEX.     IER =  2          BXPL6410
C    3)  FUNCTION VALUE UNCHANGED FOR K TRIALS.       IER =  0          BXPL6420
C    4)  LIMIT ON TRIALS REACHED.                     IER =  3          BXPL6430
C    5)  CANNOT FIND FEASIBLE VERTEX AT START.        IER = -1          BXPL6440
   44 CONTINUE                                                          BXPL6450
C                                                                       BXPL6460
C   FIND BEST VERTEX.                                                   BXPL6470
      CALL FBV (K,FUN,M)                                                BXPL6480
      IF (IER.GE.3) GO TO 46                                            BXPL6490
C                                                                       BXPL6500
C   RESTART IF THIS SOLUTION IS SIGNIFICANTLY BETTER THAN THE PREVIOUS, BXPL6510
C   OR IF THIS IS THE FIRST TRY.                                        BXPL6520
      IF (NPR.LE.0) GO TO 45                                            BXPL6530
      WRITE (6,57) (M,YMN,FUN(M))                                       BXPL6540
   45 IF (FUN(M).GE.YMN) GO TO 49                                       BXPL6550
      IF (ABS(FUN(M)-YMN).LE.AMAX1(EP,EP*YMN)) GO TO 49                 BXPL6560
C                                                                       BXPL6570
C   GIVE IT ANOTHER TRY UNLESS LIMIT ON TRIALS REACHED.                 BXPL6580
   46 YMN = FUN(M)                                                      BXPL6590
      FUN(1) = FUN(M)                                                   BXPL6600
C                                                                       BXPL6610
C                                                                       BXPL6620
      DO 47 I=1,NV                                                      BXPL6630
      CEN(I) = V(I,M)                                                   BXPL6640
      SUM(I) = V(I,M)                                                   BXPL6650
   47 V(I,1) = V(I,M)                                                   BXPL6660
C                                                                       BXPL6670
C                                                                       BXPL6680
C                                                                       BXPL6690
      DO 48 I=1,NVT                                                     BXPL6700
   48 XS(I) = V(I,M)                                                    BXPL6710
C                                                                       BXPL6720
```

```
C                                                                     BXPL6730
      IF (IER.LT.3) GO TO 6                                           BXPL6740
   49 IF (NPR.LE.0) GO TO 50                                          BXPL6750
      CALL BCUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,V(1,M),-1)             BXPL6760
      WRITE (6,58) FUN(M)                                             BXPL6770
   50 RETURN                                                          BXPL6780
C                                                                     BXPL6790
C                                                                     BXPL6800
   51 FORMAT (50HOINDEX AND DIRECTION OF OUTLYING VARIABLE AT STARTI5) BXPL6810
   52 FORMAT (50HOIMPLICIT CONSTRAINT VIOLATED AT START. DEAD END. )  BXPL6820
   53 FORMAT ('OCANNOT FIND FEASIBLE',I4,'TH VERTEX OR CENTROID AT STARTBXPL6830
     1.')                                                             BXPL6840
   54 FORMAT (10HOAT TRIAL I4,54H CANNOT FIND FEASIBLE VERTEX WHICH IS NBXPL6850
     1O LONGER WORST,I4,15X,'RESTART FROM BEST VERTEX.')              BXPL6860
   55 FORMAT (40HOFUNCTION HAS BEEN ALMOST UNCHANGED FOR I5,7H TRIALS) BXPL6870
   56 FORMAT (27HOLIMIT ON TRIALS EXCEEDED. )                         BXPL6880
   57 FORMAT ('OBEST VERTEX IS NO.',I3,'  OLD MIN WAS ',E15.7,        BXPL6890
     1 '  NEW MIN IS ',E15.7)                                         BXPL6900
   58 FORMAT ('OMIN OBJECTIVE FUNCTION IS ',E15.7)                    BXPL6910
      END                                                             BXPL6920
```

```
      SUBROUTINE FBV (K,FUN,M)
      DIMENSION FUN(50)
      M = 1
C
C
      DO 1 I=2,K
      IF (FUN(M).LE.FUN(I)) GO TO 1
      M = I
    1 CONTINUE
C
C
      RETURN
      END
```

```
      SUBROUTINE BOUT (NT,NFT,NFE,NCE,NV,NVT,V,K,FN,C,IK)
      DIMENSION V(50,50), FN(50), C(25)
      WRITE (6,4) NT,NFT,NFE,NCE
C
C
      DO 1 I=1,K
      WRITE (6,5) FN(I),(V(J,I),J=1,NV)
      IF (NVT.LE.NV) GO TO 1
      NVP = NV+1
      WRITE (6,6) (V(J,I),J=NVP,NVT)
    1 CONTINUE
C
C
      IF (IK.NE.0) GO TO 2
C
      WRITE (6,7) (C(I),I=1,NV)
      RETURN
    2 IF (IK.GE.0) GO TO 3
      WRITE (6,8) (C(I),I=1,NV)
      RETURN
    3 WRITE (6,9) IK,(C(I),I=1,NV)
      RETURN
C
C
    4 FORMAT ('ONO. TOTAL TRIALS = ',I5,4X,'NO. FEASIBLE TRIALS = ',
     1I5,4X,'NO. FUNCTION EVALUATIONS = ',I5,4X,'NO. CONSTRAINT EVALUATI
     2ONS = ',I5/'O     FUNCTION VALUE',6X,'INDEPENDENT VARIABLES/DEPEND
     3ENT OR IMPLICIT CONSTRAINTS')
    5 FORMAT (1H ,E18.7,2X,7E14.7/(21X,7E14.7))
    6 FORMAT (21X,7E14.7)
    7 FORMAT (10HOCENTROID 11X,7E14.7/(21X,7E14.7))
    8 FORMAT ('O  BEST VERTEX',7X,7E14.7/(21X,7E14.7))
    9 FORMAT ('OCENTROID LESS VX',I2,2X,7E14.7/(21X,7E14.7))
      END
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPINV
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPINV
C++++++++++++++++++++       SUBROUTINE PINVRT       ++++++++++++++++++++CPINV
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPINV
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPINV
C++++                                                               ++++CPINV
C++++ PURPOSE: TO REARRANGE A VECTOR OF REAL POLYNOMIAL             ++++CPINV
C++++          COEFFICIENTS IN ASCENDING POWERS OF THE INDEPENDENT  ++++CPINV
C++++          VARIABLE TO A VECTOR OF COEFFICIENTS IN DESCENDING   ++++CPINV
C++++          POWERS OF THE INDEPENDENT VARIABLE, OR VICE VERSA.   ++++CPINV
C++++                                                               ++++CPINV
C++++ CALLING SEQUENCE                                              ++++CPINV
C++++ CALL PINVRT(X,N)                                              ++++CPINV
C++++                                                               ++++CPINV
C++++ DESCRIPTION OF PARAMETERS:                                    ++++CPINV
C++++ X       = A VECTOR OF REAL COEFFICIENTS, UPON RETURN FROM     ++++CPINV
C++++           THE SUBROUTINE THIS VECTOR WILL CONTAIN THE         ++++CPINV
C++++           COEFFICIENTS IN REVERSE SEQUENCE FROM THAT INPUT.   ++++CPINV
C++++                                                               ++++CPINV
C++++ N       = THE DIMENSION OF THE VECTOR X.                      ++++CPINV
C++++                                                               ++++CPINV
C++++ SUBROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:               ++++CPINV
C++++     NONE                                                      ++++CPINV
C++++                                                               ++++CPINV
C++++ METHOD:                                                       ++++CPINV
C++++ THE SEQUENCE OF THE INPUT VECTOR IS REVERSED BY USE OF TWO    ++++CPINV
C++++ NESTED DO-LOOPS.  THE SIZE OF THE INPUT VECTOR IS LIMITED BY  ++++CPINV
C++++ ITS DIMENSION IN THE CALLING PROGRAM.                         ++++CPINV
C++++                                                               ++++CPINV
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPINV
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPINV
      SUBROUTINE PINVRT (X,N)                                            PINV
      DIMENSION X(1)                                                     PINV
      N2 = N-1                                                           PINV
C                                                                        PINV
      DO 2 J=1,N2                                                        PINV
      TEMP = X(1)                                                        PINV
      N1 = N+1-J                                                         PINV
C                                                                        PINV
      DO 1 I=2,N1                                                        PINV
      X(I-1) = X(I)                                                      PINV
    1 CONTINUE                                                           PINV
C                                                                        PINV
      X(N1) = TEMP                                                       PINV
    2 CONTINUE                                                           PINV
C                                                                        PINV
      RETURN                                                             PINV
      END                                                                PINV
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CROUT
C++++++++++++++++++++                              ++++++++++++++++++++CROUT
C++++++++++++++++++++      SUBROUTINE ROUTH        ++++++++++++++++++++CROUT
C++++++++++++++++++++                              ++++++++++++++++++++CROUT
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CROUT
C++++                                                              ++++CROUT
C++++ PURPOSE: TO DETERMINE IF A POLYNOMIAL IN S HAS RIGHT HALF    ++++CROUT
C++++          PLANE ROOTS.                                        ++++CROUT
C++++                                                              ++++CROUT
C++++ CALLING SEQUENCE:                                            ++++CROUT
C++++ CALL ROUTH(Y,N,ISTABL)                                       ++++CROUT
C++++                                                              ++++CROUT
C++++ DESCRIPTION OF PARAMETERS:                                   ++++CROUT
C++++ Y      = A ONE DIMENSIONAL ARRAY CONTAINING THE COEFFICIENTS ++++CROUT
C++++          OF THE POLYNOMIAL IN DESCENDING POWERS OF THE       ++++CROUT
C++++          INDEPENDENT VARIABLE S.                             ++++CROUT
C++++ N      = DIMENSION OF THE ARRAY Y.                           ++++CROUT
C++++ ISTABL = INTEGER OUTPUT;                                     ++++CROUT
C++++                0---INDICATES THE POLYNOMIAL HAS NO RIGHT HALF ++++CROUT
C++++                    PLANE ROOTS.                              ++++CROUT
C++++                1---INDICATES THE POLYNOMIAL HAS RIGHT HALF   ++++CROUT
C++++                    PLANE ROOTS.                              ++++CROUT
C++++                                                              ++++CROUT
C++++ SUBROUTINES AND FUNCTION SUB-PROGRAMS REQUIRED:              ++++CROUT
C++++     NONE                                                     ++++CROUT
C++++                                                              ++++CROUT
C++++ METHOD:                                                      ++++CROUT
C++++ THE POLYNOMIAL IS CHECKED FOR RIGHT HALF PLANE ROOTS BY      ++++CROUT
C++++ FORMING THE ROUTH ARRAY FROM THE POLYNOMIAL COEFFICIENTS     ++++CROUT
C++++ INPUT TO THE SUBROUTINE.                                     ++++CROUT
C++++                                                              ++++CROUT
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CROUT
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CROUT
      SUBROUTINE ROUTH (Y,N,ISTABL)                                     ROUT
      DIMENSION A(40,21), Y(1)                                          ROUT
C                                                                       ROUT
      DO 2 I=1,40                                                       ROUT
C                                                                       ROUT
      DO 1 J=1,21                                                       ROUT
      A(I,J) = 0.0                                                      ROUT
    1 CONTINUE                                                          ROUT
C                                                                       ROUT
    2 CONTINUE                                                          ROUT
C                                                                       ROUT
      ISTABL = 0                                                        ROUT
      L = 0                                                             ROUT
      LL = 0                                                            ROUT
      KK = N/2.0                                                        ROUT
```

```
      KK1 = KK+1
      FF = KK1
      K = KK*2.0
      IF (K.EQ.N) GO TO 5
      N2 = N-1
C
      DO 3 J=1,N,2
      L = L+1
      A(1,L) = Y(J)
    3 CONTINUE
C
C
      DO 4 J1=2,N2,2
      LL = LL+1
      A(2,LL) = Y(J1)
      IF (A(2,1).EQ.0.0) A(2,1)=0.0001
    4 CONTINUE
C
      GO TO 8
    5 KK1 = KK
C
      DO 6 J=1,N,2
      L = L+1
      A(1,L) = Y(J)
    6 CONTINUE
C
C
      DO 7 J1=2,N,2
      LL = LL+1
      A(2,LL) = Y(J1)
    7 CONTINUE
C
      IF (A(2,1).EQ.0.0) A(2,1)=0.0001
C
    8 DO 10 I=3,N
C
      DO 9 M=1,KK
      A(I,M) = (A(I-1,1)*A(I-2,M+1)-A(I-2,1)*A(I-1,M+1))/A(I-1,1)
    9 CONTINUE
C
      IF (A(I,1).EQ.0.0) A(I,1)=0.0001
      FF = FF-0.5
      KK = FF
   10 CONTINUE
C
C
      DO 11 I=1,N
      IF (A(I,1).LT.0.0) ISTABL=1
```

```
   11 CONTINUE
C
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE PADD
C
C        PURPOSE
C           ADD TWO POLYNOMIALS
C
C        USAGE
C           CALL PADD(Z,IDIMZ,X,IDIMX,Y,IDIMY)
C
C        DESCRIPTION OF PARAMETERS
C           Z     - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM
C                   SMALLEST TO LARGEST POWER
C           IDIMZ - DIMENSION OF Z (CALCULATED)
C           X     - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL, ORDERED
C                   FROM SMALLEST TO LARGEST POWER
C           IDIMX - DIMENSION OF X (DEGREE IS IDIMX-1)
C           Y     - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,
C                   ORDERED FROM SMALLEST TO LARGEST POWER      ,
C           IDIMY - DIMENSION OF Y (DEGREE IS IDIMY-1)
C
C        REMARKS
C           VECTOR Z MAY BE IN SAME LOCATION AS EITHER VECTOR X OR
C           VECTOR Y ONLY IF THE DIMENSION OF THAT VECTOR IS NOT LESS
C           THAN THE OTHER INPUT VECTOR
C           THE RESULTANT POLYNOMIAL MAY HAVE TRAILING ZERO COEFFICIENTS
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DIMENSION OF RESULTANT VECTOR IDIMZ IS CALCULATED AS THE
C           LARGER OF THE TWO INPUT VECTOR DIMENSIONS. CORRESPONDING
C           COEFFICIENTS ARE THEN ADDED TO FORM Z.
C
C     ..................................................................
C
      SUBROUTINE PADD (Z,IDIMZ,X,IDIMX,Y,IDIMY)
      DIMENSION Z(1), X(1), Y(1)
C
C     TEST DIMENSIONS OF SUMMANDS
C
      NDIM = IDIMX
      IF (IDIMX-IDIMY) 1,2,2
    1 NDIM = IDIMY
    2 IF (NDIM) 9,9,3
C
    3 DO 8 I=1,NDIM
```

```
      IF (I-ICIMX) 4,4,6                                          PACC 490
    4 IF (I-ICIMY) 5,5,7                                          PADD 500
    5 Z(I) = X(I)+Y(I)                                            PADD 510
      GC TO 8                                                     PADD 520
    6 Z(I) = Y(I)                                                 PADD 530
      GC TO 8                                                     PACD 540
    7 Z(I) = X(I)                                                 PACC 550
    8 CCNTINLE                                                    PADD 560
C                                                                 PADD 570
    9 ICIMZ = NDIM                                                PACD 580
      RETLRN                                                      PADD 590
      ENC                                                         PACD 600
```

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPHAS
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPHAS
C++++++++++++++++++++                              ++++++++++++++++++++CPHAS
C++++++++++++++++++++       SUBROUTINE PHASE       ++++++++++++++++++++CPHAS
C++++++++++++++++++++                              ++++++++++++++++++++CPHAS
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPHAS
C++++                                                              ++++CPHAS
C++++ PURPOSE: TO DETERMINE THE PHASE OF A POLYNOMIAL IN THE       ++++CPHAS
C++++          COMPLEX VARIABLE S, EVALUTATED AT A PARTICULAR      ++++CPHAS
C++++          FREQUENCY S = 0.0 + JW, GIVEN THE ROOTS OF THE POLY- ++++CPHAS
C++++          NOMIAL IN S.                                        ++++CPHAS
C++++                                                              ++++CPHAS
C++++ CALLING SEQUENCE:                                            ++++CPHAS
C++++ CALL PHASE(RR,RI,N,OMEGA,TANGD)                              ++++CPHAS
C++++                                                              ++++CPHAS
C++++ DESCRIPTION OF PARAMETERS:                                   ++++CPHAS
C++++ RR      = A ONE DIMENSIONAL INPUT ARRAY CONTAINING THE REAL  ++++CPHAS
C++++           PARTS OF THE ROOTS OF THE POLYNOMIAL.              ++++CPHAS
C++++ RI      = A ONE DIMENSIONAL INPUT ARRAY CONTAINING THE IMAGIN-++++CPHAS
C++++           ARY PARTS OF THE ROOTS OF THE POLYNOMIAL.          ++++CPHAS
C++++ N       = INPUT INTEGER SPECIFYING THE NUMBER OF ROOTS OF    ++++CPHAS
C++++           THE POLYNOMIAL.                                    ++++CPHAS
C++++ OMEGA   = INPUT VARIABLE GIVING THE FREQUENCY IN RADIANS AT  ++++CPHAS
C++++           WHICH THE PHASE IS TO BE COMPUTED.  THIS VALUE RE- ++++CPHAS
C++++           PRESENTS THE IMAGINARY PART OF THE COMPLEX VARIABLE ++++CPHAS
C++++           S OF THE POLYNOMIAL, THE REAL PART IS TAKEN TO BE  ++++CPHAS
C++++           ZERO.                                              ++++CPHAS
C++++ TANGD   = OUTPUT VARIABLE GIVING THE PHASE OF THE POLYNOMIAL ++++CPHAS
C++++           IN DEGREES.                                        ++++CPHAS
C++++                                                              ++++CPHAS
C++++ SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED:               ++++CPHAS
C++++     NONE                                                     ++++CPHAS
C++++                                                              ++++CPHAS
C++++ METHOD:                                                      ++++CPHAS
C++++     EACH INDIVIDUAL ROOT OF THE POLYNOMIAL IS TAKEN AS A     ++++CPHAS
C++++ FIRST ORDER FACTOR OF THE POLYNOMIAL. THE PHASE CONTRIBUTION ++++CPHAS
C++++ OF EACH FIRST ORDER TERM IS THEN CALCULATED AND THE OVERALL  ++++CPHAS
C++++ PHASE ANGLE, AT A PARTICULAR FREQUENCY, IS FORMED AS THE SUM ++++CPHAS
C++++ OF THESE INDIVIDUAL PHASE CONTRIBUTIONS.                     ++++CPHAS
C++++                                                              ++++CPHAS
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPHAS
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CPHAS
      SUBROUTINE PHASE (RR,RI,N,OMEG,TANGD)                             PHAS
      DIMENSION RR(1), RI(1)                                            PHAS
      TANGD = 0.0                                                       PHAS
      IF (N.EQ.0) GO TO 4                                               PHAS
C                                                                      CPHAS
C COMPUTE THE PHASE CONTRIBUTIONS OF THE INDIVIDUAL FIRST ORDER TERMS   CPHAS
```

```
C AND FORM THE SUM OF THESE TO FIND THE TOTAL PHASE SHIFT INVOLVED.
C
      DO 3 I=1,N
      VI = OMEG-RI(I)
      VR = -RR(I)
      IF ((VR.EQ.0.0).AND.(VI.EQ.0.0)) GO TO 1
      ANGR = ATAN2(VI,VR)
      ANGD = ANGR*57.29578
      GO TO 2
    1 ANGD = 0.0
    2 TANGD = TANGD+ANGD
    3 CONTINUE
C
    4 RETURN
      END
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSIMU
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSIMU
C++++++++++++++++++++                              ++++++++++++++++++++CSIMU
C++++++++++++++++++++      SUBROUTINE SIMULA       ++++++++++++++++++++CSIMU
C++++++++++++++++++++                              ++++++++++++++++++++CSIMU
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSIMU
C++++                                                               ++++CSIMU
C++++ PURPOSE: TO COMPUTE THE FREQUENCY RESPONSE OF THE OPEN LOOP   ++++CSIMU
C++++          SYSTEM AFTER THE OPTIMIZED COMPENSATOR COEFFICIENTS  ++++CSIMU
C++++          HAVE BEEN DETERMINED.  THE FREQUENCY RESPONSE IS     ++++CSIMU
C++++          COMPUTED OVER THE FREQUENCY RANGE SPECIFIED BY THE   ++++CSIMU
C++++          USER, BUT AT A DENSER INTERVAL THAN IS NORMALLY USED ++++CSIMU
C++++          IN THE OPTIMIZATION PART OF THE PROGRAM.  THIS SERVES++++CSIMU
C++++          TO SHOW ANY RESONANT PEAKS WHICH MIGHT NOT BE        ++++CSIMU
C++++          OBVIOUS IN THE GRAPHS IF THE DISCRETE FREQUENCIES    ++++CSIMU
C++++          CHOSEN BY THE USER ARE WIDELY SPACED.                ++++CSIMU
C++++                                                               ++++CSIMU
C++++ CALLING SEQUENCE:                                             ++++CSIMU
C++++ CALL SIMULA(WMIN,WMAX)                                        ++++CSIMU
C++++                                                               ++++CSIMU
C++++ DESCRIPTION OF PARAMETERS:                                    ++++CSIMU
C++++ WMIN    = MINIMUM VALUE OF FREQUENCY RANGE BEING COMPUTED     ++++CSIMU
C++++ WMAX    = MAXIMUM VALUE OF FREQUENCY RANGE BEING COMPUTED     ++++CSIMU
C++++                                                               ++++CSIMU
C++++ SUBROUTINED AND FUNCTION SUBPROGRAMS REQUIRED                 ++++CSIMU
C++++      PVAL, CCNORM, PROD, PRBM, PQFB, PHASE                    ++++CSIMU
C++++                                                               ++++CSIMU
C++++ NOTE:                                                         ++++CSIMU
C++++      THE MAGNITUDE AND PHASE VALUES COMPUTED ALONG WITH THE   ++++CSIMU
C++++ FREQUENCIES AT WHICH THEY ARE COMPUTED ARE STORED IN THE      ++++CSIMU
C++++ ARRAYS SAMAG, SAPHSD, AND WFREQS.  THESE ARRAYS ARE           ++++CSIMU
C++++ OUTPUT VIA THE COMMON STATEMENT.                              ++++CSIMU
C++++                                                               ++++CSIMU
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSIMU
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++CSIMU
      SUBROUTINE SIMULA (WMIN,WMAX)                                      SIMU
      COMMON NCN,NCD,NTFN,NTFD,NOMEG,A(20),B(20),DMAG(500),DQPHSD(500),WSIMU
     1FREQ(500),NSN,NSD,COFASN(40),COFASD(40),CMAG(500),CR(500),CI(500),SIMU
     2QPHSR(500),QPHSD(500),N1,N2,N3,N4,PI,T,WS,IZCH,ADMAG(500),NTITLE(1SIMU
     32),FREQ(500),AMAG(500),NMINFS,COFWRK(40),IPLOT,WFREQS(500),SAPHSD(SIMU
     4500),SAMAG(500),DIFF1(500),DIFF2(500),ICOST,CFWRK1(40)              SIMU
      DIMENSION RNR(40), RNI(40), RDR(40), RDI(40)                       SIMU
C                                                                        SIMU
      DO 1 I=1,500                                                       SIMU
      WFREQS(I) = 0.0                                                    SIMU
      SAPHSD(I) = 0.0                                                    SIMU
      SAMAG(I) = 0.0                                                     SIMU
    1 CONTINUE                                                           SIMU
```

```
C
      N7 = NSN+1
      N8 = NSD+1
      XCMEG = 500.0
      WFREQS(1) = WMIN
      Y = (ALOG10(WMAX)-ALOG10(WMIN))/(XCMEG-1.0)
      Z = 10.0**Y
C
      DO 2 I=2,500
      J = I-1
    2 WFREQS(I) = WFREQS(J)*Z
C
C
      DO 5 K=1,500
      CALL PVAL (COFASN,NSN,0.0,WFREQS(K),QNUMR,QNUMI)
      CALL PVAL (COFASD,NSD,0.0,WFREQS(K),QDENR,QDENI)
      IF (SQRT(QDENI**2+QDENR**2)) 4,3,4
    3 SAMAG(K) = 1.E+30
      GR(K) = 1.E+30
      GI(K) = 1.E+30
      GO TO 5
    4 SAMAG(K) = SQRT(QNUMI**2+QNUMR**2)/SQRT(QDENI**2+QDENR**2)
      IF (IZOH.EQ.1) SAMAG(K)=SAMAG(K)*ABS((SIN(PI*WFREQS(K)/WS))/(PI*WF
     1REQS(K)/WS))
      GR(K) = (QNUMR*QDENR+QNUMI*QDENI)/(QDENR**2+QDENI**2)
      GI(K) = (QNUMI*QDENR-QNUMR*QDENI)/(QDENR**2+QDENI**2)
    5 CONTINUE
C
      IF (N7.EQ.1) GO TO 7
C
      DO 6 I=1,N7
      COFWRK(I) = COFASN(I)
    6 CONTINUE
C
      CALL CONORM (COFWRK,N7,XNORM1)
      CALL PRQD (COFWRK,N7,RNR,RNI,CFWRK1,NRTSN1,IERN1)
      IF (IERN1.EQ.0) GO TO 8
      CALL PRBM (COFWRK,N7,RNR,RNI,CFWRK1,NRTSN2,IERN2)
      IF (IERN2.EQ.0) GO TO 8
      WRITE (6,14)
      WRITE (6,15)
      WRITE (6,16)
      WRITE (6,15)
      WRITE (6,14)
      GO TO 12
    7 IF (N8.EQ.1) GO TO 10
C
    8 DO 9 J=1,N8
```

```
      COFWRK(J) = COFASC(J)
    9 CONTINUE
C
      CALL CCNORM (COFWRK,N8,XNORM2)
      GAINK = XNORM1/XNORM2
      CALL PROD (COFWRK,N8,RDR,RDI,CFWRK1,NRTSC1,IERD1)
      IF (IERD1.EQ.0) GO TO 10
      CALL PRBM (COFWRK,N8,RDR,RDI,CFWRK1,NRTSC2,IERD2)
      IF (IERD2.EQ.0) GO TO 10
      WRITE (6,14)
      WRITE (6,15)
      WRITE (6,17)
      WRITE (6,15)
      WRITE (6,14)
      GO TO 12
C
   10 DO 11 K=1,500
      CALL PHASE (RNR,RNI,NSN,WFREQS(K),ANGNUD)
      CALL PHASE (RDR,RDI,NSD,WFREQS(K),ANGDED)
      SAPHSD(K) = ANGNUD-ANGDED
      IF (GAINK.LT.0.0) SAPHSD(K)=SAPHSD(K)-180.0
      IF (IZOH.NE.1) GO TO 11
      IWS = WFREQS(K)/WS
      SAPHSD(K) = SAPHSD(K)-(PI*WFREQS(K)/WS)*57.29578-IWS*180.0
   11 CONTINUE
C
C
   12 DO 13 J=1,500
      WFREQS(J) = ALOG10(WFREQS(J))
      SAMAG(J) = 20.0*ALOG10(SAMAG(J))
   13 CONTINUE
C
      RETURN
C
   14 FORMAT (' ',72('-'))
   15 FORMAT (' ',8('*WARNING*'))
   16 FORMAT ('0','THE PHASE SIMULATION OF THE SYSTEM WAS ABANDONED',/,
     1 ' SINCE THE CALLS TO TWO SEPARATE ROOT FINDING SUBROUTINES',/,
     2 ' FAILED TO FIND THE SYSTEM NUMERATOR ROOTS',/)
   17 FORMAT ('0','THE PHASE SIMULATION OF THE SYSTEM WAS ABANDONED',/,
     1 ' SINCE THE CALLS TO TWO SEPARATE ROOT FINDING SUBROUTINES',/,
     2 ' FAILED TO FIND THE SYSTEM DENOMINATOR ROOTS',/)
      END
```

APPENDIX C

DESCRIPTION OF FORTRAN VARIABLE NAMES

FORTRAN VARIABLE	UNIT	DESCRIPTION
A(I)	None	Array containing the coefficients of the plant transfer function numerator.
ADAMG(I)	db	Array containing the desired open loop magnitude response in db at the discrete frequencies specified by the designer.
AMAG(I)	db	Output array containing the magnitude of the open loop frequency response in db at the discrete frequencies specified.
ANGDED	Degrees	Phase angle contribution of the open loop system transfer function denominator at a specific frequency.
ANGDER	Radians	Phase angle contribution of the open loop system transfer function denominator at a specific frequency.
ANGNUD	Degrees	Phase angle contribution of the open loop system transfer function numerator at a specific frequency.
ANGNUR	Radians	Phase angle contribution of the open loop system transfer function numerator at a specific frequency.
B(I)	None	Array containing the coefficients of the plant transfer function denominator.
CO(I)	None	One dimensional working array used primarily in reading polynomial coefficients and inputing data to root finding subroutines.
COF(I)	None	One dimensional working array used primarily to store resulting coefficients used in checking accuracy of root finding subroutines.
COFASN(I)	None	Array containing the coefficients of the open loop transfer function numerator.

FORTRAN VARIABLE	UNIT	DESCRIPTION
COFASL(I)	None	Array containing the coefficients of the open loop system transfer function denominator.
COFWRK(I)	None	One dimensional working array used primarily to temporarily store polynomial coefficients in performing polynomial manipulations.
DELPHI	Degrees	Difference between two numerator phase angle contributions at successive discrete frequencies.
DELPSI	Degrees	Difference between two denominator phase angle contributions at successive discrete frequencies.
DIFF1(I)	db	Difference between the desired and actual magnitude of the open loop frequency response at the discrete frequency values specified in the input.
DIFF2(I)	Degrees	Difference between the desired and actual phase of the open loop frequency response at the discrete frequency values specified in the input.
DMAG(I)	None	Array containing the desired magnitude response at the discrete frequencies specified by the designer.
DQPHSL(I)	Degrees	Input array containing the desired phase profile of the open loop system at the discrete frequencies chosen.
DWFQ	Radians	Interval to be used if discrete frequency values are to be incremented linearly.
FREQ(I)	None	Logarithm of the discrete frequencies used in constructing abscissa for Bode plots.
GAIN	None	Gain of plant transfer function.
GAINCC	None	Gain of compensator transfer function.

FORTRAN VARIABLE	UNIT	DESCRIPTION
IDB	None	Input integer to be set equal to unity if the desired open loop magnitude response is input in db.
IERBXP	None	Output integer indicating the completion code of the minimization routine.
INFORM	None	Alphabetic input indicating if coefficients are to be read in factored or polynomial form.
IPLOT	None	Integer input, to be set to unity if CALCOMP plots of results are desired.
IZOH	None	Integer input set equal to unity if there is a zero order hold in the error signal path prior to the plant and compensator.
KNOW	None	Integer input indicating if discrete frequency values are to be read in or incremented linearly from the minimum frequency.
NBODE	None	Integer input to be set equal to unity if Bode plot is not desired.
NCD	None	Order of compensator transfer function denominator.
NCN	None	Order of compensator transfer function numerator.
NICHCL	None	Integer input to be set equal to unity if Nichol's plot is not desired.
NMINFS	None	Integer input to be set equal to unity if a nonminimum phase solution is to be allowed.
NOMEG	None	Integer specifying the total number of discrete frequency points being considered in the input frequency response profile. This number must be { 500.
NSD	None	Order fo the system open loop transfer function denominator.

FORTRAN VARIABLE	UNIT	DESCRIPTION
NSN	None	Order of the system open loop transfer function numerator.
NTFD	None	Order of plant transfer function denominator.
NTFN	None	Order of plant transfer function numerator.
NYQST	None	Integer input to be set equal to unity if Nyquist plot is not desired.
N3	None	Dimension of array containing plant transfer function numerator coefficients.
N4	None	Dimension of array containging plant transfer function denominator coefficients.
QDENI	None	Value of the immaginary part of the system open loop transfer function denominator evaluated at a particular frequency.
QDENR	None	Value of the real part of the system open loop transfer function denominator evaluated at a particular frequency.
QMAG(I)	None	Output array containing the magnitude of the frequency response computed for the open loop system transfer function at the discrete frequencies specified.
QNUMI	None	Value of the immaginary part of the system open loop transfer function numerator evaluated at a particular frequency.
QNUMR	None	Value of the real part of the system open loop transfer function numerator evaluated at a particular frequency.
QPHSD(I)	Degrees	Output array containing the phase of the frequency response of the open loop system at the discrete frequencies specified.

FORTRAN VARIABLE	UNIT	DESCRIPTION
CPHSR(I)	Radians	Output array containing the phase of the frequency response of the open loop system transfer function at the discrete frequencies specified.
SAMAG(I)	db	Array containing the magnitude of the simulation of the resulting optimized system transfer function.
SAPHSD(I)	Degrees	Array containing the phase of the simulation of the resulting optimized system transfer function.
T	Secs	Sampling period if a zero order hold is present in the system.
WFREQ(I)	Radians	Array containing the discrete frequencies specified by the designer.
WMAX	Radians	Input specifying the maximum frequency value of the range of frequencies being considered.
WMIN	Radians	Input specifying the minimum frequency value of the range of frequencies being considered.
WS	Radians	Output giving the sampling frequency if a zero order hold is present in the system.
XS(I)	None	Array in which values to be varied by the minimization routine are stored. At the start this array contains the initial guess of the coefficient values and at completion the array contains the coefficient values that yield the minimum cost function.

LIST OF REFERENCES

1. Massachusetts Institute of Technology, Electronic Systems Laboratory Report ESL-R-409, On the Design of a Digital Computer Program for the Design of Feedback Compensators in Transfer Function Form, by M. Athans, December 1969.

2. Beveridge, G. S. G. and Schechter, R. S., Optimization: Theory and Practice, p.453-456, McGraw-Hill, 1970.

3. Bode, H. W., Analysis and Feedback Amplifier Design, D. Van Nostrand Company, 1945.

4. Box, M. J., "A New Method of Constrained Optimization and a Comparison With Other Methods", Computer Journal, p.42-52, 8 April 1965.

5. Aerospace Corporation Report TR-1001(2307)-20, The Application of Modern Computing Technology to Control System Analysis and Design Problems, by T. C. Coffey, June 1967.

6. Jacoby, S. L. S., Kowalik, J. S., and Pizzo, J. T., Iterative Methods for Nonlinear Optimization Problems, p.195-197, Prentice-Hall, 1972.

7. Kuo, F. F., and Kaiser, J. F., Editors, Systems Analysis by Digital Computer, Wiley, 1966.

8. Leondes, C. T., Editor, Control and Dynamic Systems Advances in Theory and Applications, v. 11, p.25-143, Academic Press, 1974.

9. Melsa, J. L. and Jones, S. K., Computer Programs for Computational Assistance in the Study of Linear Control Theory, 2d. ed., McGraw Hill, 1973.

10. Melsa, J. L. and Schultz, D. G., Linear Control Systems, p. 513-591, McGraw Hill, 1969.

11. NASA Contractor Report CR-2248, An Innovative Approach to Compensator Design, by Jerrell R. Mitchell and Willie L. McDaniel, Jr., May 1973.

12. Oldenburger, R., Editor, Frequency Response, The MacMillian Co., 1956.

13. Page, J. A., Automated Design of Feedback Control Systems, Ph.D. Thesis, University of California, Los Angeles, 1970.

14. Rosenbrock, H. H., Computer Aided Control System Design, Academic Press, 1974.

15. Sanathanan, C. K. and Tsukui, H., "Synthesis of Transfer Function from Frequency Response Data", International Journal of Systems Science, v. 5, no. 4, p.41-54, May 1974.

16. NASA Technical Memorandum X-1443, FORMAC Program to Assist in the Analysis of Linear Control Systems Using State Varialbe Feedback Design Technique, by C. R. Slivinsky, L. T. Dellener, and D. J. Arpasi, October 1967.

17. Thaler, G. J., Design of Feedback Systems, Dowden, Hutchinson, and Ross, 1973.

18. Thaler, G. J. and Brown, R. G., Analysis and Design of Feedback Control Systems, 2d. ed., McGraw Hill, 1960.

19. Truxal, J. G., Introductory System Engineering, McGraw Hill, 1972.

20. Office of Naval Research London Conference Report C-20-73, Conference on "Computer Aided Control System Design", by A. H. Waynick, 10 September 1973.

21. Computer Aided Control System Design Conference Publication Number 96, A Method for Linear Systems Order Reduction, by S. C. Chuang, p.213-218, 4 April 1973.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 62 Naval Postgraduate School Monterey, California 93940	2
4.	Professor George J. Thaler, Code 62TR Naval Postgraduate School Monterey, California 93940	5
5.	Lt. Anthony J. Mancini, USN 1514 Electric St. Scranton, Pa. 18509	3
6.	Dr. Jerrel R. Mitchell Department of Electrical Engineering Mississippi State University Mississippi State, Mississippi 39762	1
7.	Mr. Jay Cameron IBM Monterey and Cottle Roads San Jose, California 95114	1

8. Mr. Peter Gramata 1
IBM
Monterey and Cottle Roads
San Jose, California 95114

9. Mr. Robert McIntosh 1
IBM
Monterey and Cottle Roads
San Jose, California 95114

10. Mr. Martin Dost 1
IBM
Monterey and Cottle Roads
San Jose, California 95114

11. Mr. Ron Palmer 1
IBM
Monterey and Cottle Roads
San Jose, California 95114

12. Mr. W. M. Syn 1
IBM
Monterey and Cottle Roads
San Jose, California 95114

13. Professor A. G. J. MacFarlane 1
Dept. of Electrical Engineering
University of Manchester
Inst. of Science and Technology
Manchester England 067609

14. Dr. Rui-Tien Rung 1
Dept. of Applied Sciences
Universite du Quebec a Chicoutimi
930 est, rue Jacques-Cartier
Chicoutimi, Quebec
G7H 2B1

15. Mr. Dave Noble 1
IBM
Monterey and Cottle Roads
San Jose, California 95114