United States
Naval Postgraduate School

THESIS

COMPUTER-AIDED NETWORK DESIGN BY OPTIMIZATION
IN THE FREQUENCY DOMAIN

by

James Lau

December 1969

Computer-Aided Network Design by Optimization
in the Frequency Domain

by

James Lau
Captain, United States Marine Corps
B.S., United States Military Academy, 1962

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1969

Author: James Lau

Approved by: Donald E. Kirk
Thesis Advisor

Charles H Rothauge
Chairman, Department of Electrical Engineering

R. F. Rinehart
Academic Dean

ABSTRACT

The filter design problem is considered as an optimization problem. An iterative search technique is employed to adjust the variable network element values to approximate some desired network response, with a minimum of error. Explicit constraints are employed to ensure physical realizability. The design process uses a combination of a modified version of Calahan's network analysis program with a direct search method of minimization developed by Hooke and Jeeves. The result is a procedure which utilizes the circuit designer's experience and knowledge to set up the problem but relieves him of the tedious labor now performed by the high-speed digital computer.

TABLE OF CONTENTS

ACKNOWLEDGEMENT

The author wishes to express his appreciation to Professor D. E. Kirk for providing invaluable assistance, guidance, and helpful suggestions throughout the course of the investigation. The author is also grateful to his wife, Eloise, who patiently typed the preliminary efforts, and the final draft.

I. INTRODUCTION

A. COMPUTER-AIDED NETWORK DESIGN

Mathematical programming techniques have found wide use in operations research, economics, and other related fields. However, it has only been in recent years that such techniques have gained acceptance as tools for the design and evaluation of electronic circuits. The development of several general network-analysis programs has made computer-aided network design quite attractive. What is computer-aided network design? The circuit operation is first analyzed by means of a computer. It is then modified and analyzed again until the desired result is achieved--a trial-and-error procedure. Naturally the more experienced the engineer, the fewer the trials before a satisfactory design is realized.

The engineer today has a variety of analysis programs which may suit his needs in the design of networks. Some of the more well-known ones are: NET-1, ECAP, SCEPTRE, NASAP, CIRCUS, LISA, PANE, CALAHAN, and CORNAP. Programs such as these have offered great assistance to the engineer in the analysis and design of networks. Although the obvious advantages in saving of time and tedious labor are quite apparent, there are certain features that would be desirable and perhaps possible in future programs of the type mentioned. Some of these features may include:

1. A graphical output on remote terminals which will allow the engineer to check his results and make on the spot changes as necessary.

2. Automatic means for improving the circuit design; i.e., some optimization technique to obtain "best" element values.

As valuable an aid as the computer is, a significant part of the design procedure will still require the engineering judgement of the designer. The cost for relieving the engineer of all the tedious calculations required for analysis is not an inexpensive one. The engineer must use his knowledge to specify the network topology, the response desired, constraints on element values, error criteria, reasonable initial element values, and other information which only he can provide.

B. USE OF OPTIMIZATION TECHNIQUES IN COMPUTER-AIDED NETWORK DESIGN

The network designer is basically confronted with the problem of designing a circuit to meet some prescribed performance requirements. The design may be accomplished in one of many ways. If the requirements are such that an existing synthesis technique will provide the answer, the problem is essentially solved, and a satisfactory solution is obtained. In some cases a perfectly good design may be achieved in the laboratory by physically wiring the circuit on the "bread-board" and experimentally determining the "best" element values for the design.

There are classical synthesis techniques which provide a step-by-step design procedure, resulting in the circuit configuration and element values [1]. However, there are some design problems which may not be amenable to solution by any of the known synthesis techniques. The designer may be given a requirement in the form of a table of values or a graph of the response desired. Such a requirement cannot be satisfied by the classical synthesis techniques. If the circuit contains a large number of variable elements, design by a trial-and-error process in the laboratory is also highly unfeasible. Again, as with the analysis, the high-speed digital computer has offered an alternative approach to the problem. We can now use some optimization technique to find element values in a given design configuration which yield a solution nearest to the prescribed performance requirement. The optimization technique iteratively adjusts the element values until the requirement is approximated as closely as possible.

Although synthesis techniques are available for the design of standard high- and low-pass filters, they do not take into account any constraints on the network configuration or element values. Problems of this nature would certainly be amenable to solution by an optimization technique. Networks whose transfer functions are extremely complex comprise another class of problems which could be solved by optimization. Optimization may also be used to obtain models for active devices. An optimization scheme could well be used in the design of matched filters. There are countless other examples, but suffice it to say that a combination of a good network analysis program and an efficient optimization program is certainly an excellent application of computer-aided network design.

C. OPTIMIZATION TECHNIQUES

In any optimization procedure two requirements must be satisfied. First, there must be some means to determine the behavior or performance of the system for any set of parameter values. Second, a performance measure must be selected which is a numerical measure of the behavior of the system. The optimization is basically a matter of minimizing the performance measure, which is a function of the parameters. If we think of this in geometric terms, the points in the parameter space represent different circuit element values and any change in the element value will result in movement to another point in the space. The performance measure is defined on this parameter space and requires an additional dimension if it is to be represented geometrically.

Minimization techniques generally do not yield the global minimum. What is found is a local minimum, but by changing the starting values of the parameters, it can be determined whether the local minimum is also the global minimum. If the minimization procedure converges to different values of the performance measure, the smallest value of the performance measure is then selected as the global minimum. The different minimization techniques may be classified by the method which is utilized to find a local minimum. They may be generally classified in the three following categories:

1. Direct search methods: those which do not compute the partial derivatives of the performance measure with respect to the parameters, but use only the value of the performance measure.

2. Gradient methods: those which require the calculation of the first partial derivatives of the performance measure with respect to the parameters.

3. Second-Order methods: those requiring higher-order partial derivatives.

No attempt will be made here to discuss the various methods under each category. An excellent discussion of the methods can be found in Ref. [2].

D. THE GENERAL NATURE OF THE PROBLEM

Earlier it was stated that for problems which cannot be solved by existing synthesis techniques or by experimenting with the circuit, an optimization technique may be feasible as an alternative solution.

Problems which are amenable to solution by optimization will generally be stated as follows: "Given a particular network with a fixed number of variable elements, adjust these elements until the response of the network minimizes some preassigned criterion". The key words in this general statement are "particular network with a fixed number of variable elements". For a particular desired response there may be several network configurations which will yield comparable results. The job of the designer then is to choose among the configurations he tries, and to select the best design which satisfies the requirements. One of the desirable features the engineer would like in future programs for computer-aided design is the ability of the program to also produce the network configuration as well as the element values for the optimal network for a given response. With the existing programs the engineer starts with a particular network configuration and a fixed number of variable elements are adjusted until the performance meets some preassigned criterion.

Now that the type of design problem is defined, the important features of the optimization process may be studied. Figure I-1 gives the essential elements which should be a part of the optimization technique for the circuit design problems.

The choice of the network configuration and the initial element values is the task of the engineer before the actual optimization process begins. The features to be discussed now are the evaluation of the response, the performance measure and the decision-making process.

The first thing the optimization program must be capable of doing is to evaluate the response from a given set of element values. This

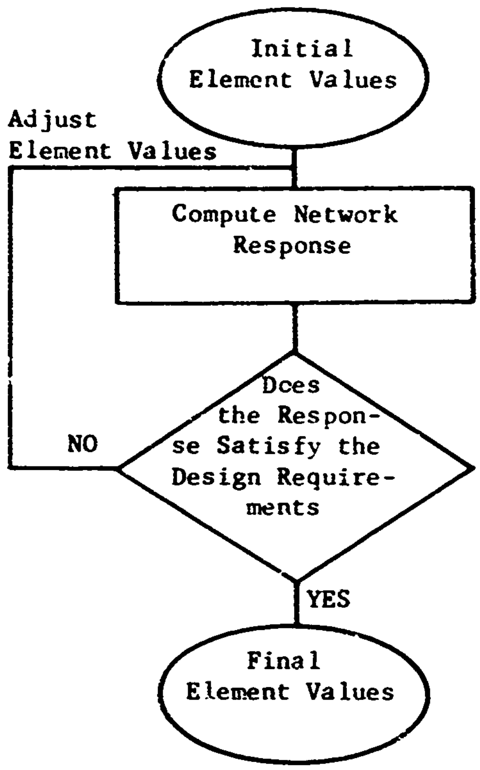


Fig. I-1 The Optimization Process

must be done each time the element values are adjusted. The analysis programs mentioned above all have this capability.

Once the response has been evaluated, it must be compared with the desired response. The measure of behavior is the performance measure. It is impossible to choose a single criterion and call it the universal performance measure. The nature of the problem may determine what performance measure is to be used. The experienced circuit designer will generally have in mind what performance measure is best for a given situation. The performance measure chosen must remain the same throughout the design procedure. Some problems will dictate what performance measure is to be used, but in many cases the choice is purely subjective [3].

There are several typical forms of the performance measure the designer may use in the optimization process. The simplest form would be

$$J(\underset{\sim}{p}) = \sum_{i=1}^{N} \left| R_A(f_i) - R_D(f_i) \right|$$

where $J(\underset{\sim}{p})$ is the performance measure, a function of the parameter values $\underset{\sim}{p}$, the terms $R_A(f_i)$ and $R_D(f_i)$ represent the actual and desired frequency responses, respectively, and N is the total number of points. This form of the performance measure indicates that only the magnitude of the difference is of interest, positive and negative deviations having equal weight. Another performance measure is

$$J(\underset{\sim}{p}) = \sum_{i=1}^{N} \left[R_A(f_i) - R_D(f_i) \right]^n$$

where n is some even integer. When the difference between the actual and desired responses are small, say less than 1.0, then the value of this performance measure will decrease with increasing values of the exponent, n. These are just two examples of what may be used for performance measures. By defining the performance measure in some way such as mentioned above, the optimization procedure is one in which a search is conducted to find the minimum of the performance measure.

The search for this minimum generally results in finding a local minimum as mentioned above. Only if it is known that the function is unimodal will the local minimum be the global minimum. Otherwise, it is necessary to conduct a systematic search of the entire parameter space in order to locate the global minimum. For problems that have more than three or four variable parameters this may not be feasible. However, if the search is started with new initial values of the elements, and the function value converges to the same value, then one can be relatively certain that a global minimum has been located. By finding a minimum, whether it be local or global, a perfectly acceptable solution may be obtained. In the final analysis, it is the engineer's decision whether the final network configuration behaves in a satisfactory manner or whether further investigation is necessary to locate a "better" minimum.

This final aspect, the decision of the engineer, is perhaps the most critical item in the optimization. He must weigh the cost of further exploration to find a smaller minimum against the solution he already has. A great deal depends on what the circuit requirements are. The tolerances on the element values may be such that the procedure may have to be repeated again and again. On the other hand, there may be very weak restrictions on the element values so long as the response matches the desired response within say 0.1%. The computer relieves the engineer of the tedious work involved in any optimization procedure, but he is still responsible for making the knowledgable decisions to use the computer most advantageously.

II. THE OPTIMIZATION PROGRAM

A. ANALYSIS PROGRAM

As mentioned in Chapter I, the optimization procedures will include an analysis program and a minimization program. Among the various programs available, the CALAHAN and ECAP programs were considered as likely choices. Both programs were subjects of study in a course in computer-aided design, taught by Professor S. G. Chan, offered at the Naval Postgraduate School. The CALAHAN program was the final choice since it provides for a graphical output of the frequency response, an output not available with ECAP. Presumably the engineer who is designing by optimization will either choose a program which he has used successfully, or he will write a program to suit his needs.

1. A Linear Network Analysis Program

The CALAHAN program is a general-purpose program designed for the analysis of linear electrical networks [4]. Input data to the program consists of the number of nodes in the circuit, the number of passive elements, the number of active elements, the input and output node numbers, and the type of output desired. A list of element values must be provided as well as a range of values of frequency (time) over which the frequency (transient) response is to be calculated. Outputs from the program are the coefficients of the specified network function, the poles and zeros, frequency and/or transient responses. The program is designed so that the user need only provide the required data cards, to obtain the desired output.

2. Modification of CALAHAN for Use in the Optimization Program

In order to incorporate CALAHAN into the optimization program, it was necessary to make some modifications to the original CALAHAN program. Before this was done, the original version was run a considerable number of times to yield frequency responses of circuits for which the actual responses were known. From the closed-form expression of the voltage transfer function of the Butterworth filter [5], the theoretical frequency responses were obtained for various orders of this type of filter. Using values of the normalized Butterworth filters [6], frequency responses were calculated by CALAHAN for different orders of the filter. The responses calculated by CALAHAN were almost exactly the same as the theoretical responses. The procedure was also repeated with the Chebyshev filter and similarly good results were obtained.

Since the goal of the optimization is to determine a set of element values for a particular network configuration whose frequency response is to match a given response as closely as possible, only the portion of CALAHAN that calculates the frequency response is needed. The main program from the original CALAHAN was reduced until only the portions involving the frequency response remained. Several subroutines that are not essential to the calculation of the frequency response were removed.

B. THE MINIMIZATION PROGRAM

The minimization program used in conjunction with CALAHAN to form the optimization program is a direct search technique [7]. This category of minimization techniques requires only the calculation of the value of the performance measure, the calculation of derivatives not being a requirement. A gradient technique used in the design of filters by optimization is the subject of the Naval Postgraduate School thesis written by Major Charles A. Henry, USMC. Results obtained using the two methods are discussed and compared in Chapter IV.

1. Direct Search

Direct search may be basically described as a sequential examination of trial solutions which involves the comparison of the trial solution with the "best" solution obtained up to that time and a method for determining what the next trial solution will be [7]. Among the various types of problems which can be solved by direct search are solution of system of equations, curve-fitting problems, solution of integral equations, and minimizing (or maximizing) functions with or without constraints on the variables. The application of direct search methods to the solution of problems of the types mentioned above is basically the same regardless of the type of problem.

A space of P points, representing the solution space, must be defined. There must be some means to determine that one point is "better" than another. Presumably there is a "best" solution P* in the solution space. Direct search is then accomplished in the following manner: A point B_1, designated the first base point, is arbitrarily selected in the space. A second point, P_2, is then selected and compared with B_1. If P_2 is "better" than B_1 then P_2 becomes B_2, the second base point. However if P_2 is not "better" than B_1, then B_1 remains the base point. The process continues with each new point selected and compared with the current base point. The technique for selecting new trial points is determined by various conditions which arise as a function of results of trials made. The technique to be used in the minimization program is pattern search.

2. The Specific Technique--Pattern Search

Pattern search is a direct search technique for finding the minimum of a function $F(\underset{\sim}{p})$ of the variables $\underset{\sim}{p} = (p_1, p_2, p_3, \ldots, p_N)^T$. The argument $\underset{\sim}{p}$ is varied until a minimum value of $F(\underset{\sim}{p})$ is obtained. The successive values of $\underset{\sim}{p}$ represent points in an N-dimensional space.

The operation of the pattern-search routine will now be described. First, a few definitions will be of aid in the ensuing discussion. The procedure of going from one point to another point is termed a move. If the value of $F(\underset{\sim}{p})$ decreases, then the move is a success; if the function $F(\underset{\sim}{p})$ does not decrease, then the move is a failure. Pattern search makes two types of moves. The explore move is to acquire knowledge about the behavior of the function $F(\underset{\sim}{p})$. The second type of move is the pattern move which utilizes the information gained from the explore moves to accomplish the actual minimization of the function by moving in the direction of an established pattern. The point from which a pattern move is made is known as a base point. Basically the pattern-search procedure is movement from base point to base point.

The explore move provides the information which indicates a probable direction for a successful move. A pattern is thus established. The pattern move from a given base point duplicates the combined moves from a previous base point if the direction of the pattern is unchanged. This process continues as long as the moves are successful, the step lengths increasing in magnitude. The result of each pattern move then is either a success or a failure. If the pattern move is a success, then a series of explore moves is carried out to see if the result can be further improved.

Each explore move is carried out in the following manner: a single coordinate of the point is varied by either increasing or decreasing the coordinate by some fixed amount and seeing if the move is a success. If a success occurs, the new coordinate value is used; otherwise the old coordinate is retained. For each coordinate, these explore moves are made until the final point, as a result of all the explore moves, becomes the new base point.

If, on the other hand, the pattern move is a failure, the search continues by retreating to the base point and starting over again with new explore moves until a new pattern is established.

The pattern-search technique can be better understood with the aid of a simple example. Figure II-1 serves as an illustration of what has been discussed in the previous paragraphs. A two-dimensional parameter space is shown with equal-cost contours represented by F_1, F_2, ..., F_8; where $F_k > F_{k+1}$. The argument of the function F is $\underset{\sim}{p} = (p_1,p_2)^T$. The point B_1 is selected as the first base point and explore moves are conducted from this point. First the p_1 coordinate is stepped in its positive and negative directions; a success is achieved when the step is negative. Next the p_2 coordinate is tested and it is determined that a positive step yields a success. The explore move produces a new base point B_2. The most probable direction of a success is in the direction of the line segment $\overline{B_1B_2}$; therefore, the pattern is established and the pattern move results in TP_1, a temporary point. Each pattern move is followed by a series of explore moves. If the result of the explore moves is a success, then the pattern move is termed a

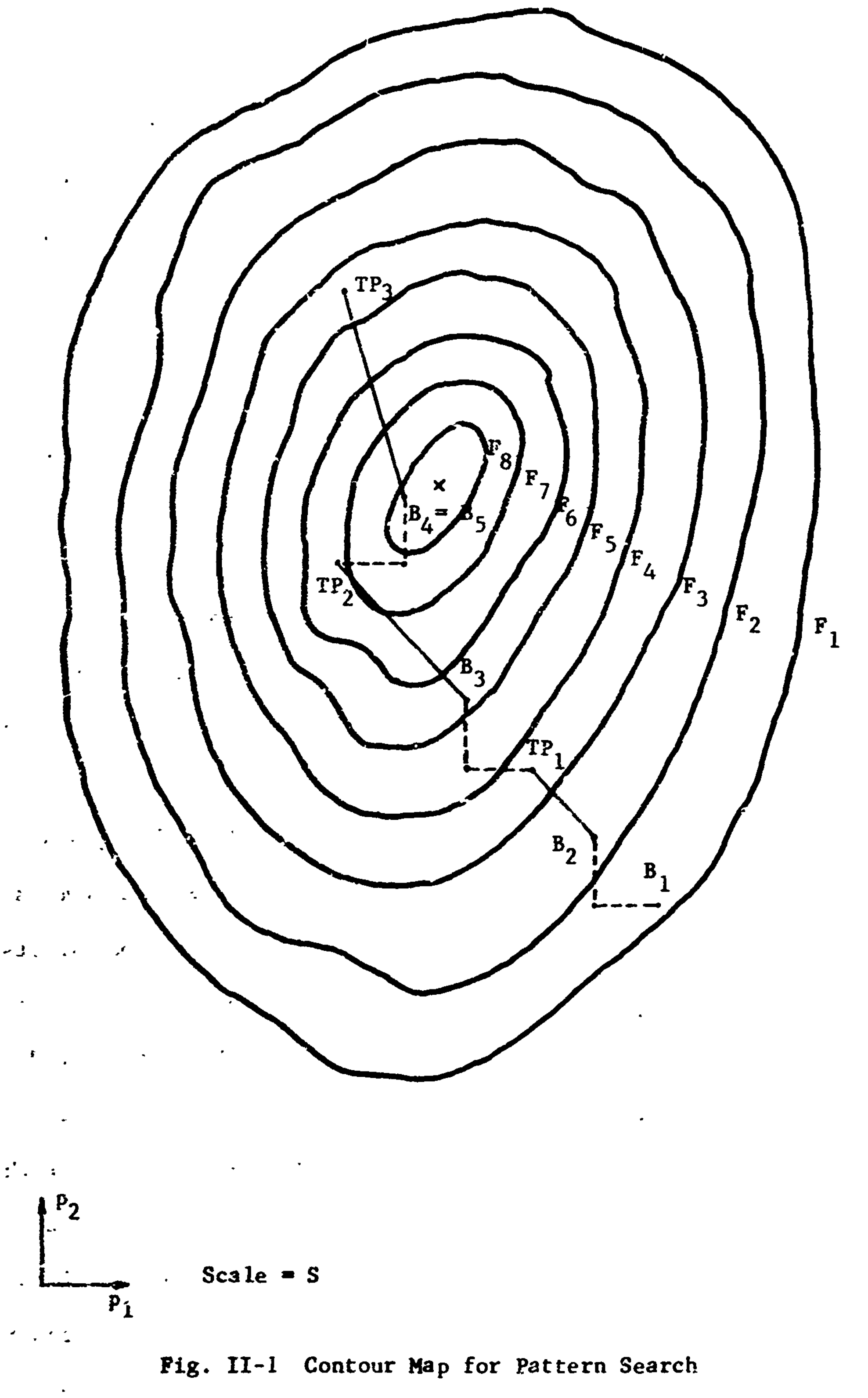
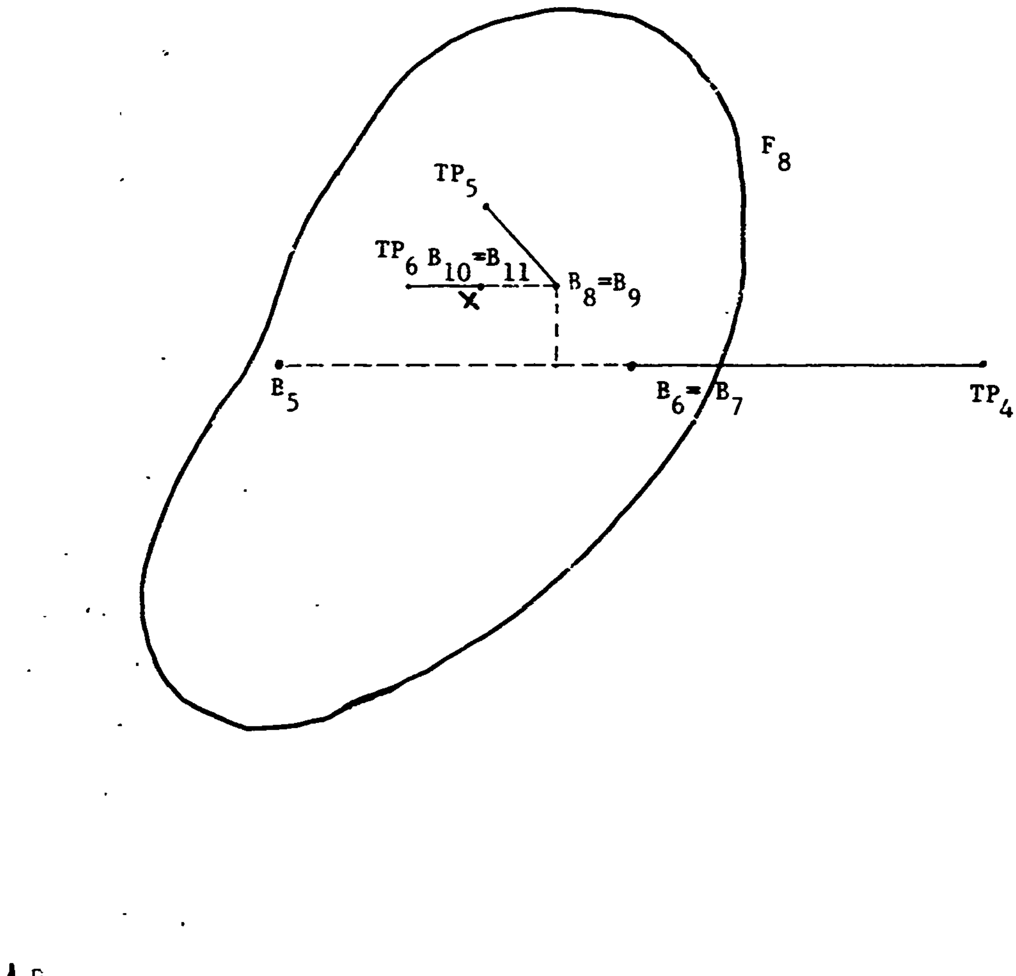


Fig. II-1 Contour Map for Pattern Search success. The point resulting from the successful explore moves becomes the new base point. If, however, the explore moves are failures, the function value at the temporary point is calculated and if this value is greater than the function value at the previous base point, the old base point becomes the new base point and explore moves are tried again. At TP_1 explore trials are made and a successful move is made to B_3, which becomes the new base point. This indicates that the pattern move to TP_1 was a success. By the same line of reasoning, the pattern move to TP_2 is a success. The pattern move to TP_3 is a failure since all explore moves from this point are failures and the value of the function at TP_3 is greater than the value at B_4; therefore B_4 becomes B_5, the new base point, and the explore moves are tried again. The region within the F_8 contour is enlarged by a factor of five and shown in Fig. II-2. The point B_6 is a successful explore but it should be noted that there is no change in the p_2 coordinate from B_5. A change in the p_2 coordinate would yield a failure. The pattern move to TP_4 is a failure, so B_6 now becomes B_7 and explore moves are made. Perturbations of both p_1 and p_2 in the original step size do not produce any successful explore moves. It is, therefore, necessary to reduce the step size by some fixed amount. The step reduction factor in this case is 0.2, which means that the original step size has now been reduced by a factor of five. Once again explore trials are made, now with the new step size, and a success is achieved at B_8. The pattern move to TP_5 is a failure, so B_8 becomes B_9. A successful explore move is achieved at B_{10} but the pattern move to TP_6 is a failure and B_{10} becomes B_{11}. This process of reducing the step size and then making the pattern moves continues until the difference between two consecutive steps is less than



Scale = 5S

Fig. II-2 Enlargement of F_8 Contour

some prescribed amount. If this criterion is a very small number then the step size will be sufficiently small to ensure that the minimum has been closely approximated. Care must be taken in the choice of both the step size and the step reduction factor. Too large a reduction factor will result in a slowdown of the search procedure. If an initial step size is too large, the minimum may be missed altogether.

The direct search procedure described above is termed pattern search since the minimization is basically performed by the pattern moves. Although the explore moves provide some reduction, their main purpose is to provide information for the improvement of the pattern move. The pattern-search program used in the optimization program is a program written by R. Hilleary[1], with some modifications to include constraints on the independent variables.

C. THE OPTIMIZATION PROGRAM--A COMBINATION

In sections A and B the individual programs in the optimization program were discussed in some detail. A brief description was given of the modifications to CALAHAN to accommodate the particular problems to be considered. How are CALAHAN and DIRECT together to be implemented into one program to be used in the design of networks by optimization? The answer to this question is the subject of this section.

The basic type of filter design problem which will be solved by the optimization program is one in which a particular frequency response is given and the objective is to design a filter which approximates the response as closely as possible. In the next chapter the exact problem

[1]Subroutine DIRECT, Naval Postgraduate School Computer Facility.

problem formulation and specific example problems will be discussed in detail, but the above problem description is adequate for a general discussion of the optimization program. After the particular circuit has been selected, a choice of initial element values must be made, the engineer's knowledge and experience playing a vital role in the choice. Other information which must be supplied to the program includes the following: the number of frequency points to be matched, the values of the desired frequency response at the points to be matched, the explicit constraints on the element values, the step size, the step reduction factor, and the termination criterion for the minimization.

The initial element values serve as coordinates of the first base point for the pattern search routine, called DIRECT. An external function subprogram then calculates the value of the function to be minimized by DIRECT. The exact form of this function may differ for different problems, but in all cases it is a comparison between the actual frequency response and the desired response. This calculation is performed as part of the function subprogram utilizing the modified version of CALAHAN. Once the minimum has been determined, the element values producing the minimum are supplied to the analysis program and the actual frequency response is calculated. This process may be repeated until the overall design satisfies all of the requirements.

III. IMPLEMENTATION OF THE OPTIMIZATION PROGRAM

The use of the optimization program is dictated by the requirements for the design. An optimization technique should be used only when classical synthesis methods and experimental methods are either impossible or unfeasible. The purpose of this chapter is to discuss the use of the optimization program in circuit design. The first section of the chapter is a general discussion of the features of the program. In the concluding section several examples are given to illustrate the use of the program.

A. PROGRAM FEATURES

1. The Input Data

For input data, the optimization program requires a topological description of the network, a list of element values, a range of frequencies over which the desired and actual responses are to be matched, a description of the desired response, the number of varying and non-varying elements, and a list of the constraints on the varying elements. The following information required by DIRECT must also be included in the input data: the step size, the step reduction factor, the termination criterion, and the maximum allowable number of evaluations. Figure III-1 is a flow chart showing the sequence and coding of the input data cards.

2. The Output

The output from the optimization program consists of two parts. The first part is a result of the minimization and includes the value of the function at convergence, and the optimum values of the variable

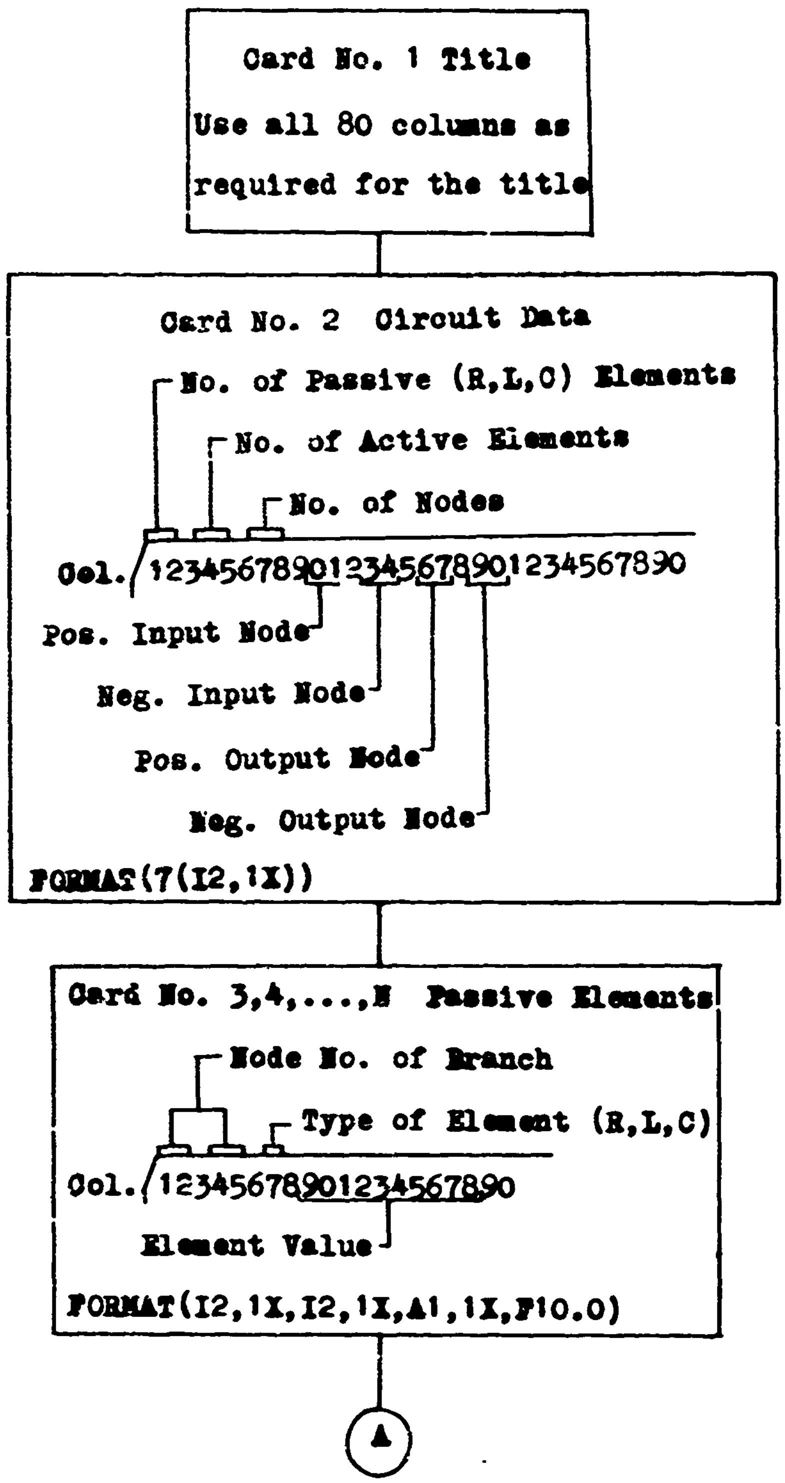


Fig. III-1 Coding Flow Chart for Optimization Program

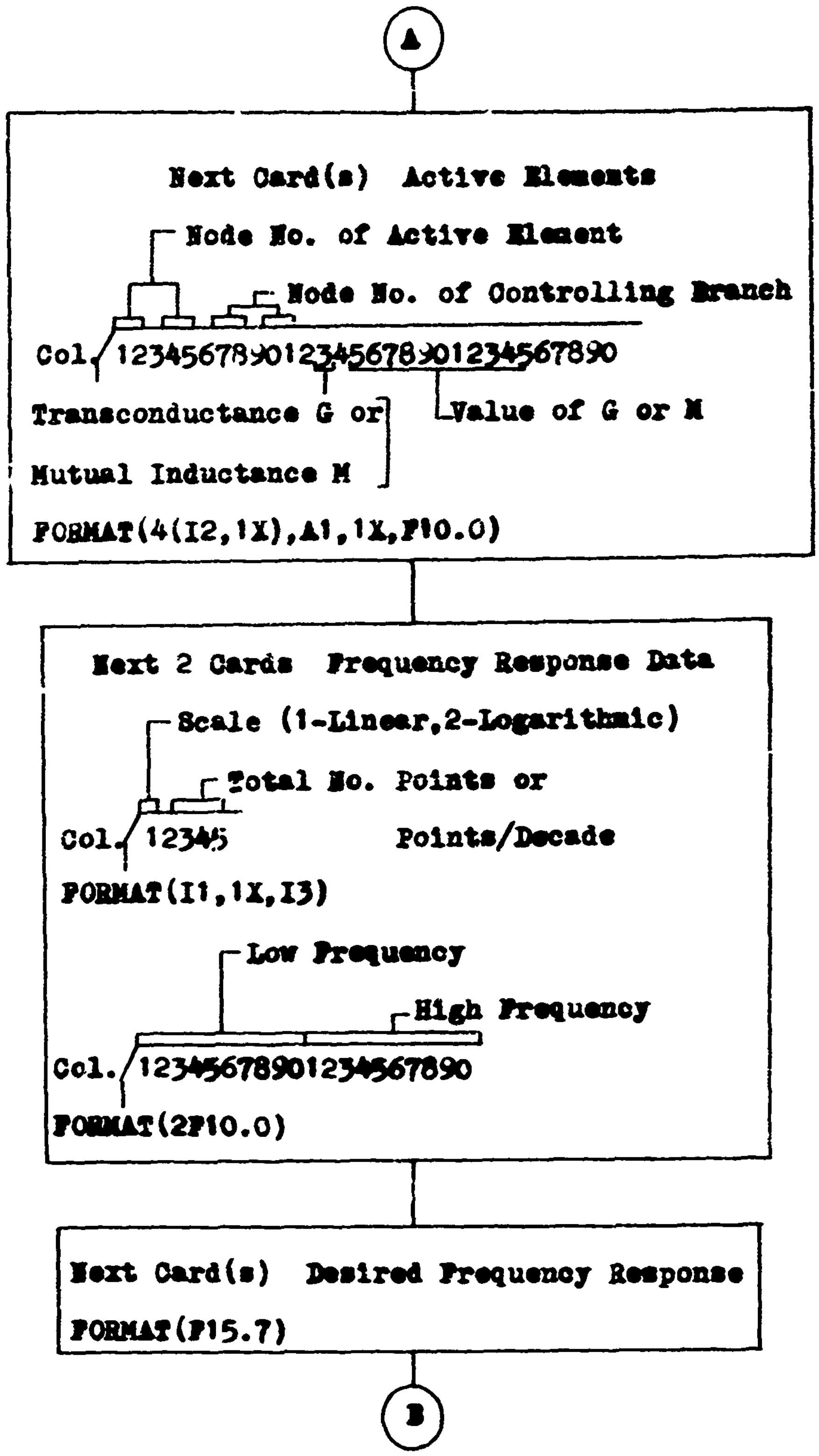


Fig. III-1 (continued)

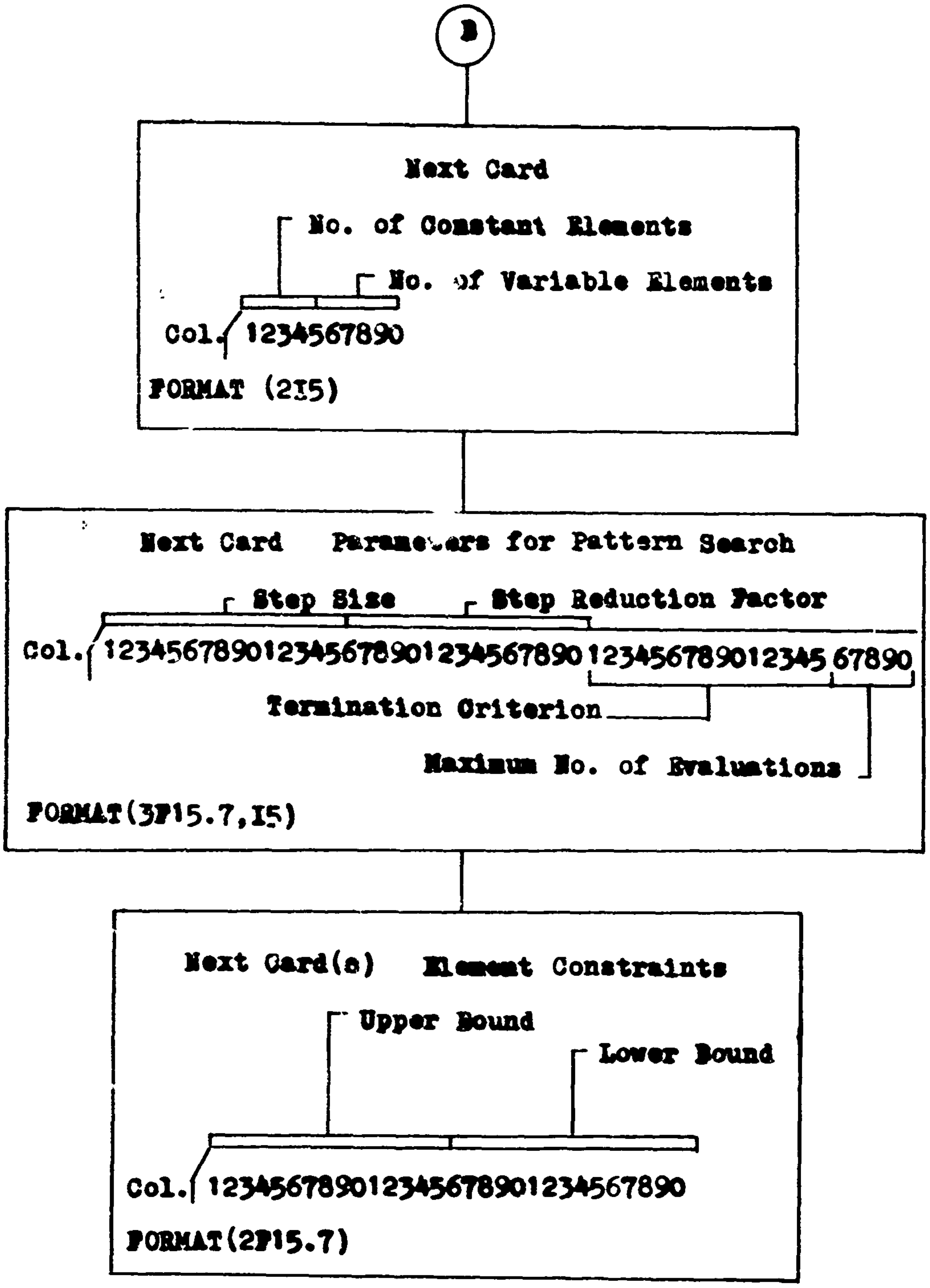


Fig. III-1 (continued)

elements. The second part of the output is a result of the analysis program. With the optimum element values calculated by DIRECT, the frequency response is calculated. The output is in tabular form as well as in a graphical form. The circuit designer merely compares the values of the calculated response with those of the desired response. If the design requirements are satisfied, the element values calculated in the first part of the output are the element values of the design.

3. Accuracy of the Optimization Program

In all examples used in testing the program, the performance measure to be minimized was of the form,

$$J(\underset{\sim}{p}) = \sum_{i=1}^{N} \left[R_A (f_i) - R_D (f_i) \right]^n$$

where $J(\underset{\sim}{p})$ is a function of the network elements $\underset{\sim}{p}$. $R_A (f_i)$ is the actual frequency response at the i^{th} comparison point and $R_D (f_i)$ is the desired response at the same point of comparison. N is the total number of frequency points to be compared and n is some even integer. Theoretically, the minimum of this performance measure is zero if a perfect match of frequency responses is achieved. In practice, however, a zero output is rarely, if ever, achieved. The measure of accuracy is determined by the function value at exit from DIRECT; the smaller the function value, the closer the actual response approaches the desired response.

The accuracy of the output is basically dependent upon the choice of the termination criterion for DIRECT. The pattern search ends when the difference between consecutive step sizes falls below this pre-selected termination criterion. A small criterion will result in a small function value, consequently a closer approximation to the desired response. The program allows the user to specify the termination criterion as part of the input. Table III-1 shows a comparison of execution times, and function values, for a normalized fourth-order Butterworth filter, as a function of the step size, the step reduction factor, and the termination criterion. The performance measure used was

$$J(\underset{\sim}{p}) = \sum_{i=1}^{21} \left[R_A (f_i) - R_D (f_i)\right]^2 .$$

The function values for a termination criterion of 10^{-4} differ by a factor of 100 from those for a termination criterion of 10^{-6}; whereas the difference between function values for termination criteria 10^{-6} and 10^{-9} is insignificant. In this case there is no particular advantage in the choice of a termination criterion less than 10^{-6}, since the function values only change slightly but the execution times are longer. A comparison between the desired response, for the frequency range specified, and the largest and smallest function values is given in Table III-2.

4. Execution Time

The execution time for the program is dependent on several factors which will be discussed in this section. The initial choice of element values will certainly affect the execution time; if the initial guess is a poor choice the program may take an inordinate amount of time if it converges at all to a minimum. Convergence to a minimum also may be quite slow for circuits with a large number of elements. The only solution to this problem is to choose a simpler circuit configuration which may yield a response within acceptable tolerances. In the pattern search, the execution time is a function of the termination criterion. The choice of the termination criterion is a compromise between speed and accuracy; one is sacrificed for the other. If more

TABLE III-1

Trial	Step Size	Step Red. Factor	Termination Criterion	Execution Time (Sec)	Function Value x 10^8
A	0.05	0.25	10^{-4}	50.38	203.67
B	0.05	0.25	10^{-6}	68.50	0.31191
C	0.05	0.25	10^{-9}	78.01	0.23612
D	0.05	0.125	10^{-4}	54.93	105.01
E	0.05	0.125	10^{-6}	70.18	0.11424
F	0.05	0.125	10^{-9}	71.77	0.11424
G	0.1	0.25	10^{-4}	46.37	386.91
H	0.1	0.25	10^{-6}	70.90	0.55285
I	0.1	0.25	10^{-9}	78.07	0.54994
J	0.1	0.125	10^{-4}	49.09	60.625
K	0.1	0.125	10^{-6}	57.52	7.753
L	0.1	0.125	10^{-9}	63.03	7.723
M	0.5	0.25	10^{-4}	68.27	16.730
N	0.5	0.25	10^{-6}	87.31	0.14581
O	0.5	0.25	10^{-9}	90.89	0.14581
P	0.5	0.125	10^{-4}	85.22	284.77
Q	0.5	0.125	10^{-6}	104.78	0.12187
R	0.5	0.125	10^{-9}	110.14	0.12187

TABLE III-2

Desired Response	Trial G Response	Trial F Response
- 0.1042320	- 0.1032002	- 0.1042283
- 0.2204427	- 0.2196893	- 0.2204416
- 0.4314420	- 0.4310329	- 0.4314427
- 0.7850979	- 0.7850114	- 0.7851012
- 1.3305276	- 1.3306713	- 1.3305340
- 2.1019602	- 2.1021585	- 2.1019697
- 3.1032674	- 3.1033316	- 3.1032581
- 4.3047100	- 4.3045549	- 4.3047056
- 5.6547211	- 5.6543064	- 5.6547127
- 7.0972899	- 7.0966568	- 7.0972862
- 8.5844102	- 8.5836172	- 8.5844040
-10.0806382	-10.0797758	-10.0806456
-11.5623696	-11.5614929	-11.5623751
-13.0151248	-13.0142879	-13.0151329
-14.4307494	-14.4300060	-14.4307604
-15.8052081	-15.8045635	-15.8052015
-17.1370394	-17.1365509	-17.1370392
-18.4263366	-18.4259949	-18.4263458
-19.6740983	-19.6738892	-19.6740875
-20.8818219	-20.8817902	-20.8818054
-22.0512536	-22.0513916	-22.0512390

accurate results are required then the execution time is necessarily longer. Table III-1 shows the effects of different step reduction factors and termination criteria. A further comparison of execution time as a function of the number of points compared is made for the normalized fourth-order Butterworth filter. The results of this comparison are shown in Table III-3.

TABLE III-3

No. Points	Execution Time (Sec)	Function Value
50	147.06	0.8699953×10^{-8}
40	112.77	0.4037205×10^{-7}
30	83.93	0.3040103×10^{-8}
20	71.77	0.1142364×10^{-8}

B. DESIGN EXAMPLES

To illustrate the use of the optimization program, several examples of filter design will be discussed in this section. In all of the examples, the desired frequency response is in the form of a table of values. These values are to be matched as closely as possible by the circuit configuration selected. In general, design specifications are not quite as stringent as this. A more likely specification would be to design a maximally flat filter in a pass band whose cut-off frequencies are at f_1 and f_2 and with a dropoff of a specified number of db per octave; however, to illustrate the capability of the program, point-by-point comparisons will be made.

Example 1

Problem: Find the optimum element values for the filter configuration shown in Fig. III-2, whose frequency response from 0.15Hz to 0.24Hz most closely approximates the 5^{th}-order Butterworth response over the same range of frequencies.

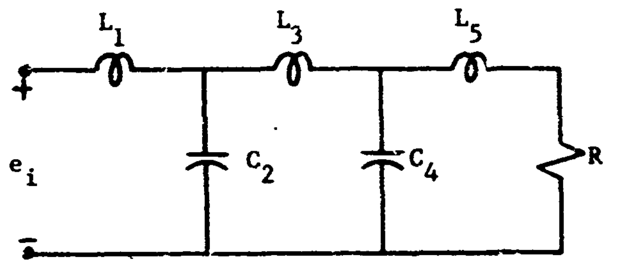


Fig. III-2 Circuit for Example 1

Constraints on element values

$0.5 \leq L_1 \leq 1.75$

$1.0 \leq C_2 \leq 2.5$

$1.0 \leq L_3 \leq 1.6$ $R = 1.0$

$0.4 \leq C_4 \leq 1.0$

$0.1 \leq L_5 \leq 0.75$

Solution:

Trial 1 Initial guess: $L_1=1.0$, $C_2=2.0$, $L_3=1.5$, $C_4=.8$, $L_5=.5$

At exit from program: $L_1=1.29$, $C_2=2.12$, $L_3=1.19$, $C_4=.888$, $L_5=.161$

Function value = $.2435 \times 10^{-6}$

Trial 2 Initial guess: $L_1=1.6$, $C_2=2.0$, $L_3=1.5$, $C_4=.8$, $L_5=.5$

At exit from program: $L_1=1.7$, $C_2=1.5$, $L_3=1.51$, $C_4=.875$, $L_5=.372$

Function value = $.6473 \times 10^{-7}$

Trial 3 Initial guess: $L_1=1.6$, $C_2=2.0$, $L_3=1.5$, $C_4=.9$, $L_5=.5$

At exit from program: $L_1=1.54$, $C_2=1.7$, $L_3=1.38$, $C_4=.895$, $L_5=.307$

Function value = $.5150 \times 10^{-13}$

Trial 4 Initial guess: $L_1=1.6$, $C_2=2.0$, $L_3=1.5$, $C_4=.9$, $L_5=.3$

At exit from program: $L_1=1.43$, $C_2=1.87$, $L_3=1.29$, $C_4=.903$, $L_5=.253$

Function value = $.1384 \times 10^{-7}$

Trial 5 Initial guess: L_1=1.6, C_2=1.8, L_3=1.5, C_4=.9, L_5=.4

At exit from program: L_1=1.54, C_2=1.7, L_3=1.38, C_4=.895, L_5=.307

Function value = $.429 \times 10^{-13}$

The optimum element values are those values calculated in trials 3 and 5. Comparison of the trial frequency responses with the Butterworth response is shown in Table III-4.

Discussion--In this problem only ten points were compared, if more accurate results are desired more points should be compared. The element values for trials 3 and 5 are very close to the values for the fifth order normalized Butterworth filter.

TABLE III-4

Freq.	Butterworth	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
0.15	- 1.9116645	- 1.8978281	- 1.9217176	- 1.9116898	- 1.9047451	- 1.9116688
0.16	- 3.1268136	- 3.1432962	- 3.1149416	- 3.1270609	- 3.1350603	- 3.1270981
0.17	- 4.6734886	- 4.6764345	- 4.6712990	- 4.6733904	- 4.6751757	- 4.6734352
0.18	- 6.4580720	- 6.4457130	- 6.4668531	- 6.4577456	- 6.4521284	- 6.4577742
0.19	- 8.3755182	- 8.3625135	- 8.3847713	- 8.3752613	- 8.3690329	- 8.3752661
0.20	-10.3421541	-10.3396997	-10.3439083	-10.3421764	-10.3406610	-10.3421516
0.21	-12.3032792	-12.3126831	-12.2965260	-12.3035507	-12.3075638	-12.3035259
0.22	-14.2275041	-14.2418127	-14.2172155	-14.2278433	-14.2342997	-14.2278185
0.23	-16.0987370	-16.1062012	-16.0933380	-16.0988770	-16.1024323	-16.0988617
0.24	-17.9099567	-17.8969879	-17.9191589	-17.9096069	-17.9040222	-17.9096222

Example 2

Problem: Design a filter which approximates the straight-line characteristic shown in Fig. III-4.

Solution: Select the circuit configuration by determining the slope of the straight line after cutoff. The slope is approximately 24db per octave. Each 6db/octave represents one order of a low-pass filter; therefore the circuit to be used for the design is a fourth-order low-pass filter as shown in Fig. III-3.

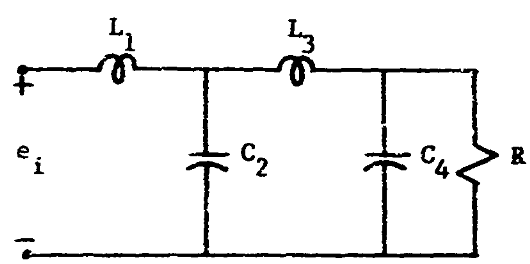


Fig. III-3 Circuit for Example 2

Results:

The first initial guess: L_1=2.0, C_2=2.0, L_3=2.0, C_4=2.0

At exit from program: L_1=1.44, C_2=1.62, L_3=1.10, C_4=0.379

The second initial guess: L_1=1.5, C_2=1.5, L_3=1.0, C_4=0.5

At exit from program: L_1=1.44, C_2=1.63, L_3=1.10, C_4=0.381

A plot of the actual and desired responses is shown in Fig. III-4.

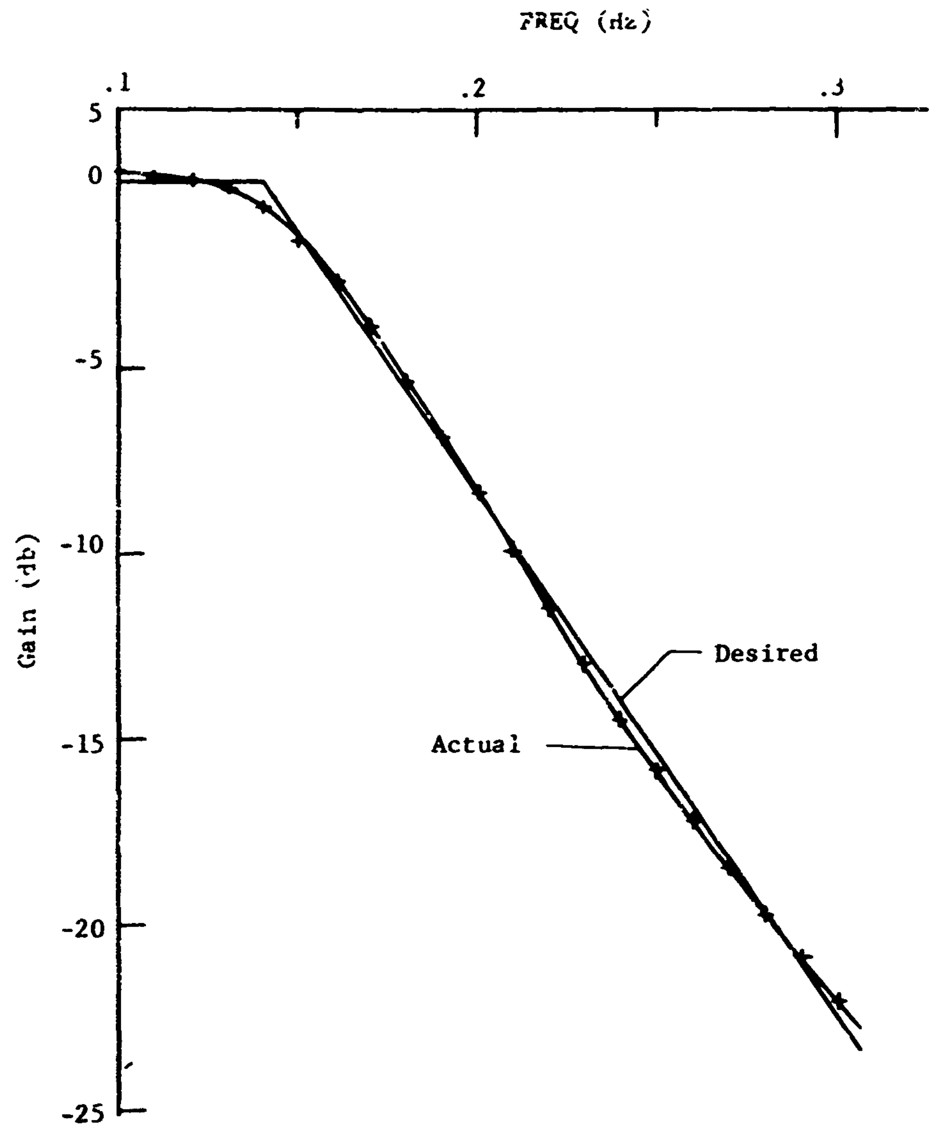


Fig. III-4 Desired and Actual Responses for Example 2

Example 3

Problem: Design a filter that has a Gaussian distribution response in the frequency range from 0Hz to 4Hz.

Solution: The first step in the solution is to change the Gaussian response from a voltage ratio to db for use in the optimization program. Table III-5 contains the values for a 21-point comparison. The response is plotted in Fig. III-5.

The first trial design was a ninth-order low-pass filter. The resulting response is plotted in Fig. III-5 showing rather marked deviations from the desired response. At the lower frequencies the deviations are much greater.

The second trial design was a modified fifth-order low-pass filter. The response is plotted in Fig. III-5. There is a slight improvement in the approximation; however the deviations at some points are quite large.

Discussion: In this problem, only low-pass filter configurations were considered. Both design responses deviated considerably from the desired responses. This points out a limitation of the optimization program. The success of the optimization technique is dependent upon the circuit configuration selected. In this example, presumably there is a better circuit configuration which would approximate the desired response with less deviation.

TABLE III-5

Freq.	Desired Gain	Design 1 Gain	Design 2 Gain
0.	0.	0.	0.
0.2	- 0.174	0.389	- 1.296
0.4	- 0.693	1.827	- 2.886
0.6	- 1.563	1.172	- 3.000
0.8	- 2.779	- 3.918	- 1.969
1.0	- 4.341	- 7.501	- 3.687
1.2	- 6.253	- 8.378	- 8.179
1.4	- 8.512	- 6.581	-11.557
1.6	-11.119	-10.663	-13.150
1.8	-14.067	-17.770	-12.961
2.0	-17.368	-22.048	-14.552
2.2	-21.012	-23.421	-21.813
2.4	-25.005	-18.813	-28.598
2.6	-29.345	-22.974	-34.113
2.8	-34.067	-36.294	-38.755
3.0	-39.172	-44.084	-42.794
3.2	-44.437	-50.013	-46.388
3.4	-50.458	-54.899	-49.640
3.6	-56.478	-59.072	-52.618
3.8	-61.938	-62.697	-55.371
4.0	-70.458	-65.865	-57.933

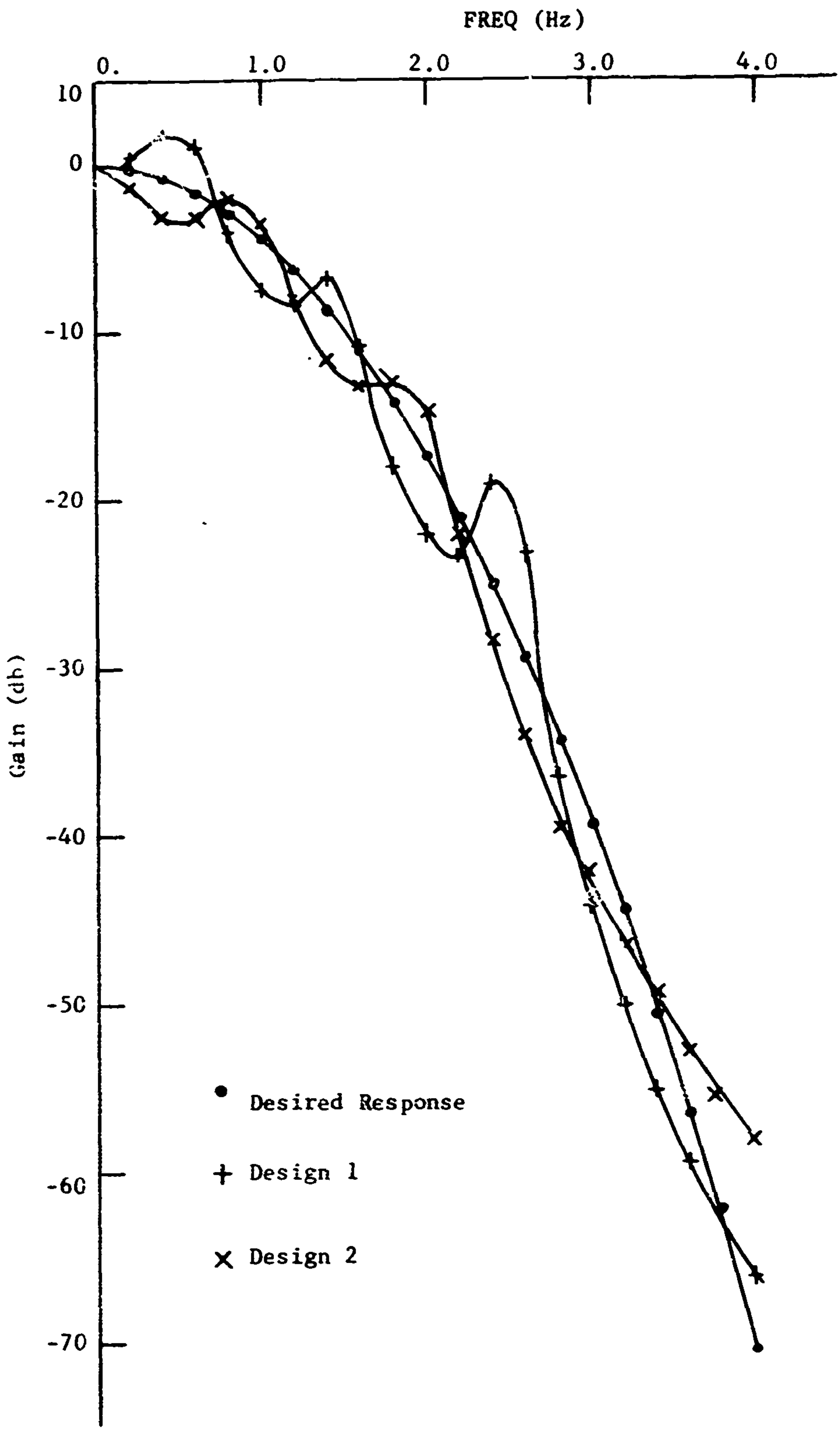


Fig. III-5 Comparison of Two Designs with the Gaussian Response

Example 4

Problem: Design a simple bandpass filter to match the desired frequency response of Table III-6.

Solution: A simple third-order low-pass filter is transformed into a bandpass filter by frequency transformation. The resultant circuit is shown in Fig. III-6.

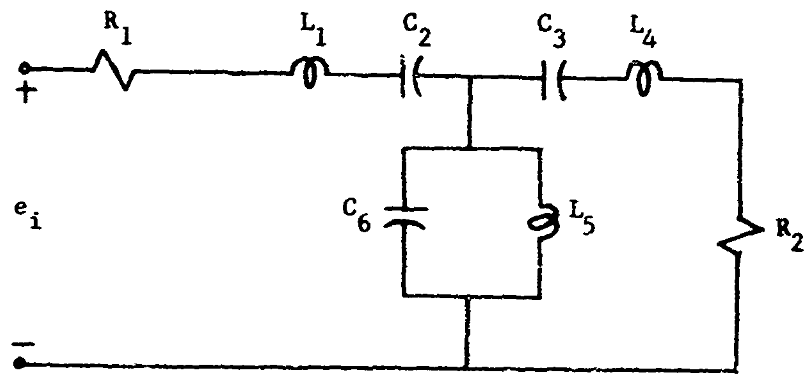


Fig. III-6 Bandpass Filter Design

Results:

The initial guess: L_1=0.5, C_2=1.2, C_3=1.2, L_4=0.5, L_5=0.75, C_6=0.75

Exit from program: L_1=0.278, C_2=0.968, C_3=1.07, L_4=0.231, L_5=0.512, C_6=0.5

Results are tabulated in Table III-6.

TABLE III-6

Freq.	Desired Gain	Actual Gain
0.12	-10.6043911	-10.6046772
0.14	- 7.8007050	- 7.8000298
0.16	- 6.5953960	- 6.5954628
0.18	- 6.1859341	- 6.1866503
0.20	- 6.0634375	- 6.0643120
0.22	- 6.0301752	- 6.0309124
0.24	- 6.0222807	- 6.0227900
0.26	- 6.0207853	- 6.0211134
0.28	- 6.0205956	- 6.0207987
0.30	- 6.0205870	- 6.0207443
0.32	- 6.0205832	- 6.0207691
0.34	- 6.0205908	- 6.0208607
0.36	- 6.0206118	- 6.0210257
0.38	- 6.0207939	- 6.0213528
0.40	- 6.0214853	- 6.0221939
0.42	- 6.0233574	- 6.0242023
0.44	- 6.0275412	- 6.0284853
0.46	- 6.0356464	- 6.0366526
0.48	- 6.0498857	- 6.0509090
0.50	- 6.0731249	- 6.0740948
0.52	- 6.1088524	- 6.1096964
0.54	- 6.1611710	- 6.1618414
0.65	- 6.2347078	- 6.2351713

TABLE III-6 (continued)

Freq.	Desired Gain	Actual Gain
0.58	- 6.3344698	- 6.3346691
0.60	- 6.4655190	- 6.4654360
0.62	- 6.6326761	- 6.6323280
0.64	- 6.8401451	- 6.8395615
0.66	- 7.0910606	- 7.0902948
0.68	- 7.3872252	- 7.3863807
0.70	- 7.7288332	- 7.7280092
0.72	- 8.1145258	- 8.1138344
0.74	- 8.5414162	- 8.5409737
0.76	- 9.0054874	- 9.0054045
0.78	- 9.5018768	- 9.5022497
0.80	-10.0253372	-10.0262442
0.82	-10.5705452	-10.5720444

Example 5

Problem: Determine the element values of a fourth-order low-pass filter whose frequency response approximates the response of a fourth-order Butterworth filter using:

(a) ideal elements

(b) inductances with nominal resistance of 0.01 ohms

(c) inductances with nominal resistance of 0.5 ohms

Solution: The circuit selected is the same as in Fig. III-3 but there are series resistors with the inductances when non-ideal elements are considered. The initial guess for the elements is the same for all three situations. All parameters for DIRECT remain the same. The frequency range is from 0.1 Hz to 0.3 Hz, comparing 21 points. The results are shown in Table III-7. Figure III-7 is a comparison of the frequency responses for circuits with ideal and non-ideal elements.

Discussion: For nominal resistances of 0.01 ohms the element values did not change much from the values of the ideal elements since the resistances are so small. When the resistance is of the same order of magnitude as the inductance then the final element values differ considerably from the ideal element values. Also, with non-ideal elements the frequency response as shown in Fig. III-7 is attenuated at the low-frequency end.

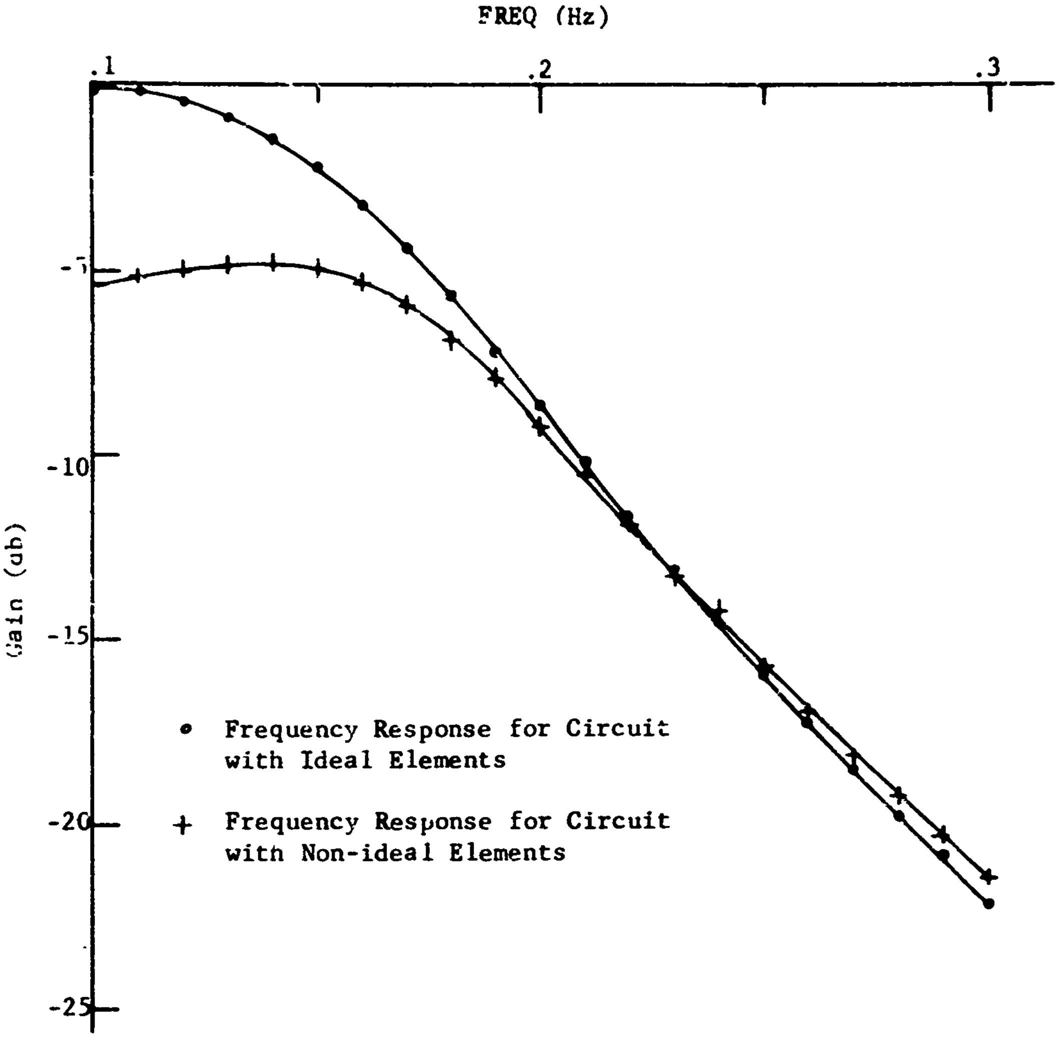


Fig. III-7 Response of Ideal and Non-ideal Circuits

TABLE III-7

Ideal Element Values	Non-ideal Element Values	
$R_s = 0$	$R_s = 0.01$	$R_s = 0.5$
$L_1 = 1.53$	$L_1 = 1.51$	$L_1 = 1.11$
$C_2 = 1.58$	$C_2 = 1.59$	$C_2 = 1.31$
$L_3 = 1.08$	$L_3 = 1.09$	$L_3 = 1.53$
$C_4 = 0.383$	$C_4 = 0.386$	$C_4 = 0.215$

IV. SUMMARY AND CONCLUSIONS

The subject of computer-aided design by optimization techniques, although only one facet of computer-aided design, is in itself quite a diverse field. There is a large variety of optimization methods which can be effectively employed in network design. The main reason for using an optimization technique instead of a classical synthesis technique in circuit design is that classical techniques cannot satisfy all possible design specifications. A specification such as matching the response of a circuit to some desired response given by a table of values or a graph cannot be realized by classical techniques. Constraints on circuit element values generally cannot be accommodated by classical methods. Such design specifications which cannot be realized by classical methods can often be satisfied by optimization techniques.

A. SUMMARY

Chapter I is an introductory chapter presenting a general discussion of computer-aided design and application of optimization techniques in computer-aided design. The three basic categories of optimization techniques are described and the general nature of the problem is presented.

In Chapter II the optimization program is described. The optimization program used is a combination of the linear network analysis program by Calahan and the pattern-search technique for minimization of a function of several variables. Modifications to the original analysis program were made in order to incorporate it into the optimization program. Basically this was a matter of reducing the size of CALAHAN, since only the portion perraining to frequency response was required. The specific method of pattern search, DIRECT, in conjunction with the modified version of CALAHAN constitute the optimization program.

The specific details regarding the implementation of the optimization program are included in Chapter III. Instructions for coding of input data cards are shown as a coding flowchart in Fig. III-1. The factors affecting the accuracy of the program are shown by the data of Table III-1. A comparison of the accuracy of the program for the worst and best approximations over a series of trial runs is shown in Table III-1 and Table III-3. Five design examples are provided at the end of the chapter.

B. CAPABILITIES AND LIMITATIONS OF THE PROGRAM

For the designs attempted, results indicated that the optimization program is highly accurate and relatively fast. A comparative study between the gradient-projection method described in the Naval Postgraduate School thesis by Major C. A. Henry, and the pattern-search method was conducted to determine the relative accuracy and speed of the two methods. Examples 2, 4, and 5(a) in Chapter III were selected for the comparisons. The results are shown in Table IV-1. Very accurate results can be achieved, as shown by Example 4 in Chapter III. On the other hand, results may deviate considerably from what is desired, as illustrated in Example 3 in Chapter III. A high degree of accuracy can be achieved if the circuit configuration chosen is the proper one for the desired response. At present there is no known optimization program that automatically alters the configuration of the network to yield an optimum solution.

TABLE IV-1

Design Problem	Method	Function Value	Execution Time(sec)
Straight-Line Approx.	Gradient Projection	1.655	132
	Pattern Search	1.651	50
Fifth-Order Butterworth	Gradient Projection	0.458×10^{-2}	133
	Pattern Search	0.204×10^{-5}	49
Band Pass	Gradient Projection	0.564×10^{-4}	328
	Pattern Search	0.220×10^{-5}	184

One of the main limitations of the optimization program is that an excessive execution time is required for circuits with more than 12 or 13 elements. The reason for this is that CALAHAN finds the tree for the network each time the elements are perturbed. This is not necessary since the circuit configuration remains the same throughout the optimization process; however no attempt was made to alter this.

The total memory requirements for the program are approximately 110 K bytes. This may or may not present a problem depending upon the computer system available to the circuit designer.

C. FUTURE REFINEMENTS

Possible areas in which the program may be improved or implemented are:

(1) Modification of the tree-finding process so that the tree is found only once for each circuit configuration.

(2) Use of the program to optimize active networks.

(3) Development of a means to "grow" elements; i.e. development of a technique that will change the circuit configuration. In this manner the circuit configuration as well as optimum element values would be calculated.

```
C                  A NETWORK OPTIMIZATION PROGRAM
C
C
C
C
C      A COMBINATION OF THE CALAHAN LINEAR NETWORK ANALYSIS
C      PROGRAM AND THE DIRECT SEARCH MINIMIZATION PROGRAM
C      FOR THE SOLUTION OF CIRCUIT DESIGN PROBLEMS
C
C
C                        MAIN PROGRAM
C
C
      EXTERNAL FE
      DIMENSION MP(100,3),ML(50,5),ELT(100),MAP(20,5),ELTA
     1(20),VAL(100),VALA(20),C(50),G(50),H(50),Y11(60),Y12
     2(60),Y(60),Z(60),Y21(60),Y22(60),VALL(50),ZZ(60,2),D
     3(100),R2(100),PP(60,2),BU(15),BL(15)
      DIMENSION LABEL(20)
      COMMON VAL,OMGMIN,OMGMAX,Y,R2,D,ELT,ELTA,VALA,Y11,Y12,
     1VALL,Y21,Y22,Z,ZZ,PP,LIN,NOM,JP,JZ,KEY1,ND,NPL,NN,JI,
     2KI,JO,KO,NAL,KEY2,MP,MAP,JW,NVAR,KEY3,NRES
      REAL IHC / 4HC    /
   10 CLOCK=ITIME(0)*.01
      READ(5,11,END=26) LABEL
   11 FORMAT(20A4)
      WRITE (6,12) LABEL
   12 FORMAT (1H1,10X,20A4)
C       READ,PRINT RLC ELEMENTS
      READ(5,13)NPL,NAL,NN,JI,KI,JO,KO,(MP(J,1),MP(J,2),ELT
     1(J),VAL(J),J=1,NPL)
   13 FORMAT(7(I2,1X)/(I2,1X,I2,1X,A1,1X,F10.0))
      WRITE(6,14)
   14 FORMAT(//,10X,19HCIRCUIT INPUT DATA       ,//)
      WRITE(6,15) NPL,NAL,NN,JI,KI,JO,KO,(MP(J,1),MP(J,2),
     1ELT(J),VAL(J),J=1,NPL)
   15 FORMAT(9X,7(I2,1X)/(9X,I2,1X,I2,1X,A1,1X,F18.9))
C     IF ACTIVE ELEMENTS,READ AND PRINT
      IF(NAL)16,19,16
   16 READ (5,17) ((MAP(J,I),I=1,4),ELTA(J),VALA(J),J=1,NAL)
   17 FORMAT(4(I2,1X),A1,1X,F10.0)
      WRITE(6,18)((MAP(J,I),I=1,4),ELTA(J),VALA(J),J=1,NAL)
   18 FORMAT(9X,4(I2,1X),A1,1X,F18.9)
   19 KEY1=1
      KEY2=2
      KEY3=1
      NVAR=0
      NVAL=0
      READ(5,20)LIN,NOM,OMGMIN,OMGMAX
   20 FORMAT(I1,1X,I3/2F10.0)
C     ND INCLUDES THE STARTING POINT
      ND=NOM+1
C     D(I) IS THE DESIRED FREQUENCY RESPONSE
      READ(5,21)(D(I),I=1,ND)
   21 FORMAT(F15.7)
      JW=1
C     NRES IS THE NUMBER OF CONSTANT ELEMENTS
C     NEX IS THE NUMBER OF VARIABLE ELEMENTS
      READ(5,22) NRES,NEX
   22 FORMAT(2I5)
      READ(5,23)DEL,RHO,DEC,MAXEV
   23 FORMAT(3F15.7,I5)
C     BU(I) IS THE UPPER BOUND
C     BL(I) IS THE LOWER BOUND
      READ(5,24)(BU(I),BL(I),I=1,NEX)
   24 FORMAT(2F15.7)
      NPM=NPL-NRES
      CALL DIRECT(VAL,NPM,SPSI,DEL,RHO,DEC,FE,KON,MAXEV,-1,
     1BU,BL)
      KEY1=2
      CALL FREQQ(LIN,NOM,OMGMIN,OMGMAX,JP,JZ,Y,Z,KEY1,R2)
      CLOCK=ITIME(0)*.01-CLOCK
```

```
      WRITE(6,25) CLOCK
   25 FORMAT(1X,'EXECUTION TIME=',F7.2,'SEC.',/)
      GO TO 10
   26 STOP
      END
C
C
C                     SUBROUTINE DIRECT
C
C
C
C
C
C     TO LOCATE A MINIMUM OF A FUNCTION,S, OF K VARIABLES
C        BY THE METHOD OF DIRECT SEARCH (HOOKE AND JEEVES)
C
C
C
C      DESCRIPTION OF PARAMETERS
C
C
C     PSI IS THE VECTOR OF K INDEPENDENT VARIABLES. IT IS
C     INITIALLY FILLED BY USER WITH FIRST GUESS OF SOLUTION
C     AT EXIT FROM DIRECT IT CONTAINS BEST VALUES ATTAINED.
C
C     K IS THE NO. OF INDEPENDENT VARIABLES OF THE FUNCTION,
C     S,TO BE MINIMIZED
C
C     SPSI AT EXIT FROM DIRECT CONTAINS SMALLEST S(PSI)
C     ATTAINED
C
C     DELCAP IS THE INITIAL STEP LENGTH
C     DELCAP IS ALTERED BY DIRECT. DO NOT USE A NUMERICAL
C     VALUE IN THE CALLING LIST
C
C     RHO IS THE STEP REDUCTION FACTOR   SUGGESTED VALUES
C     ARE .125 OR .25
C
C     DELLC IS THE TERMINATION CRITERION  WHEN THE CURRENT
C     STEP SIZE IS LESS THAN DELLC THE SEARCH IS ENDED.
C
C     S IS THE NAME OF THE EXTERNAL FUNCTION,S(PHI),TO BE
C     MINIMIZED.  A FUNCTION SUBPROGRAM OF THE SAME NAME
C     MUST BE SUPPLIED BY THE USER
C
C     KONVRG IS AN INDICATOR TESTED UPON EXIT FROM DIRECT.
C     KONVRG=-1,A PARAMETER ERROR WAS DETECTED.
C                       K.GT.15   OR  K.LE.0,
C                       DELCAP.LE.0,
C                       RHO.LE.0  OR  RHO.GE.1,
C     KONVRG=0,MAXEV WAS EXCEEDED.MINIMUM WAS NOT FOUND
C     KONVRG GREATER THAN ZERO THEN THIS NUMBER IS THE
C     NUMBER OF EVALUATIONS OF THE FUNCTION.
C
C     MAXEV IS THE MAX. NO. OF EVALUATIONS USER ALLOWS
C     TO FIND THE MINIMUM.
C
C     KN IS AN INDICATOR USED TO OBTAIN OUTPUT
C     KN=-1 OUTPUT OF FUNCTION VALUE AND VARIABLES IS MADE
C     AT ORIGIN,AFTER EACH EXPLOPE MOVE,AFTER EACH PATTERN
C     MOVE, AND AT EXIT.
C     KN=0, NO OUTPUT BY DIRECT
C     KN=1,SAME AS FOR -1 EXCEPT EXPLORE MOVES ARE OMMITED.
C
C


      SUBROUTINE DIRECT (X,K,SPSI,DELCAP,RHO,DELLC,S,KONVRG,
     1MAXEV,KN,BU,BL)
      DIMENSION [illegible](50),PSI(15),PHI(15),SLC(15),X(15),BU(15)
     1,BL(15)
      INTEGER EVAL
      DO 100 I=1,K
```

```
  100 PSI(I)=X(I)
C
      IF(K.GT.15) GO TO 50
      IF(K) 50,50,4
    4 IF(DELCAP) 50,50,5
    5 IF(RHO) 50,50,6
    6 IF(RHO.GE.1.) GO TO 50
      IF(DELLC) 50,50,7
    7 MAXEVL = MAXEV
      IF(MAXEVL) 8,8,9
    8 MAXEVL = 500
C
    9 DO 60 I=1,K
   60 SLC(I) = DELCAP
      SPSI = S(PSI)
      EVAL = 1
C
      IF(KN) 61,1,61
   61 WRITE (6,63) DELCAP,RHO,DELLC,MAXEVL,KN,(I,I=1,K)
   63 FORMAT (14H1DIRECT SEARCH,2X,8HDELCAP=,E15.6,2X,5HRHO
     1 =,E15.6,2X,7HDELLC =,E15.6,2X,8HMAXEVL =,I8,2X,5H KN
     2=,I3//8HC MOVE   ,15H FUNCTION VALUE,3X,3X,I2,6HST VAR,
     34X,3X,I2,6HND VAR,4X, 3X,I2,6HRD VAR,4X,3(3X,I2,6HTH
     4 VAR,4X)/ 26X,6(3X,I2,6HTH VAR,4X)/26X,6(3X,I2,6HTH VA
     5R,4X))
      WRITE ( 6,62) SPSI, (PSI(I),I=1,K)
   62 FORMAT(8HCORIGIN ,E15.7,3X,6E15.6 /(26X,6E15.6))
C
    1 SS = SPSI
      DO 10 I=1,K
   10 PHI(I)= PSI(I)
      ASSIGN 11 TO IBK
      GO TO 40
C
   11 IF(KN) 12,13,13
   12 WRITE ( 6,14) SS,(PHI(I),I=1,K)
   14 FORMAT(8HCEXPLORE,E15.7,3X,6E15.6 /(26X,6E15.6))
C
   13 IF(SS.GE.SPSI) GO TO 3
    2 IF (EVAL.GE.MAXEVL) GO TO 51
C
      DO 20 I=1,K
      IF(SLC(I)) 21,50,22
   21 IF(PHI(I).GT.PSI(I))  SLC(I) =-SLC(I)
      GO TO 23
   22 IF(PHI(I).LT.PSI(I))  SLC(I) = -SLC(I)
   23 THET = PSI(I)
      PSI(I) = PHI(I)
      PHI(I) = 2.*PHI(I) - THET
      PHI(I)=AMIN1(PHI(I),BU(I))
      PHI(I)=AMAX1(PHI(I),BL(I))
   20 CONTINUE
C
      SPSI = SS
      SPHI=S(PHI)
      SS=SPHI
      EVAL = EVAL +1
      ASSIGN 25 TO IBK
C
   40 DO 41 I=1,K
      THET = PHI(I)
      SLCI = SLC(I)
      PHI(I) = THET + SLCI
      PHI(I)=AMIN1(PHI(I),BU(I))
      PHI(I)=AMAX1(PHI(I),BL(I))
      SPHI =S(PHI)
      EVAL = EVAL +1
      IF(SPHI.LT.SS) GO TO 42
      PHI(I) = THET - SLCI
      PHI(I)=AMAX1(PHI(I),BL(I))
      PHI(I)=AMIN1(PHI(I),BU(I))
      SPHI=S(PHI)
```

```
      EVAL = EVAL +1
      IF(SPHI.GE.SS) GO TO 44
      SLC(I)=-SLCI
   42 SS=SPHI
      GO TO 41
   44 PHI(I)=THET
   41 CONTINUE
C
      GO TO IBK,(11,25)
C
   25 IF(KN) 27,28,27
   27 WRITE( 6,29) SS,(PHI(I),I=1,K)
   29 FORMAT(8H PATTERN,E15.7,3X,6E15.6 /(26X,6E15.6))
C
   28 IF(SS.GE.SPSI) GO TO 1
      DO 26 I=1,K
      IF(ABS(PHI(I)-PSI(I)).GT.0.5*ABS(SLC(I))) GO TO 2
   26 CONTINUE
C
    3 IF(DELCAP.LT.DELLC) GO TO 52
      DELCAP = RHO * DELCAP
      DO 30 I=1,K
   30 SLC(I) = RHO * SLC(I)
      GO TO 1
C
   50 KONVRG = -1
      GO TO 53
   51 KONVRG = 0
      GO TO 53
   52 KONVRG = EVAL
   53 IF(KN) 55,54,55
   55 WRITE( 6,56)   KONVRG,SPSI,(PSI(I),I=1,K)
   56 FORMAT(1CHOKONVRG=  ,I)O/8H EXIT    ,E15.7,3X,6E15.6/
     1(26X,6E15.6))
   54 RETURN
      END
C
C
C                         FUNCTION FE
C
C


      FUNCTION FE(X)
      DIMENSION VAL(100),X(100),D(100),Y(60),Z(60),R2(100),
     1MP(100,3),ML(50,5),ELT(100),MAP(20,5),ELTA(20),VALA(20
     2),C(50),Y11(60),Y12(60),Y21(60),Y22(60),VALL(50),
     3ZZ(60,2),PP(60,2)
      COMMON VAL,OMGMIN,OMGMAX,Y,R2,D,ELT,ELTA,VALA,Y11,Y12,
     1VALL,Y21,Y22,Z,ZZ,PP,LIN,NOM,JP,JZ,KEY1,ND,NPL,NN,JI,
     2KI,JO,KO,NAL,KEY2,MP,MAP,JW,NVAR,KEY3,NRES
      NPM=NPL-NRES
      DO 2 J=1,NPM
    2 VAL(J)=X(J)
      CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,3,KEY2,MP,ELT,VAL,
     1MAP,ELTA,VALA,Y11,Y12,VALL,JW,NVAR,KEY3,Y,NP,J11)
      CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,2,KEY2,MP,ELT,VAL,
     1MAP,ELTA,VALA,Y21,Y22,VALL,JW,NVAR,KEY3,Z,NZ,J22)
      DO 14 J=1,60
      IF(Y(1))6,4,6
    4 IF(Z(1))6,8,6
    6 JP=NP-J+1
      JZ=NZ-J+1
      GO TO 16
    8 DO 10 K=1,NP
   10 Y(K)=Y(K+1)
      DO 12 K=1,NZ
   12 Z(K)=Z(K+1)
   14 CONTINUE
   16 CONTINUE
      DO 20 J=1,60
      JJ=JP-J+1
```

```
      IF(Y(JJ))18,20,18
   18 JP=JP-J+1
      GO TO 22
   20 CONTINUE
   22 DO 26 J=1,60
      JJ=JZ-J+1
      IF(Z(JJ))24,26,24
   24 JZ=JZ-J+1
      GO TO 28
   26 CONTINUE
C     CALCULATE ZEROS
   28 CALL MULLER(Y,JP,ZZ)
C     CALCULATE POLES
      CALL MULLER(Z,JZ,PP)
C     CALCULATE FREQUENCY RESPONSE
      CALL FREQQ(LIN,NOM,OMGMIN,OMGMAX,JP,JZ,Y,Z,KEY1,R2)
      FE=0.
      DO 1 I=1,ND
    1 FE=FE+(R2(I)-D(I))**2
      RETURN
      END
```

```
C
C                        THE MODIFIED VERSION OF CALAHAN
C
C
C
C        THE SUBROUTINES ARE LISTED ALPHABETICALLY FOR CONVENIENCE
C
C
C
      SUBROUTINE ASBS(LN,C,G,H,X,NP,MG,I,KEY1,JN,JP,II,NPL,NAL)
      DIMENSION C(50),G(50),H(50),X(60),Z(60),MG(30,5),Y(3,50)
      IF(I-2)101,100,101
100   GO TO(2,3,3,4),KEY1
    3 NP=2*LN-1
      JM=1
      JP=2
      GO TO 5
    4 NP=2*LN-3
      JM=2
      JP=3
      GO TO 5
    2 NP=2*LN+1
      JM=0
      JP=1
5     JN=LN-JM
  101 N=MG(JP,3)
      IF(NP-2)30,30,102
30    IF(NP)31,31,32
32    X(1)=0.
      NP=1
      RETURN
31    X(1)=0.
      NP=1
      RETURN
102   DO 130 J=1,NP
130   X(J)=0.
      Z(1)=1.
      K1=1
      K2=1
      DO 1 J=1,JN
      K=JM+J
      N=MG(K,3)
      K2P=K2+2
      DO 116 K=1,3
      DO 117 I=K1,K2P
117   Y(K,I)=0.
```

```
116   CONTINUE
      IF(C(N))81,80,81
81    N1=2
      N2=2
      DO 110 K=K1,K2
110   Y(3,K+2)=Z(K)*C(N)
      IF(G(N))86,87,86
80    IF(G(N))82,83,82
82    N2=1
86    N1=1
      DO 111 K=K1,K2
111   Y(2,K+1)=Z(K)*G(N)
87    IF(H(N))85,84,85
83    IF(H(N))88,89,88
88    N2=0
85    N1=0
      DO 112 K=K1,K2
112   Y(1,K)=Z(K)*H(N)
84    K1=K1+N1
      K2=K2+N2
      DO 113 K=K1,K2
113   Z(K)=Y(1,K)+Y(2,K)+Y(3,K)
1     CONTINUE
      IF(II)132,133,133
132   DO 131 J=K1,K2
131   X(J)=-Z(J)
      RETURN
133   DO 134 J=K1,K2
134   X(J)=Z(J)
89    RETURN
      END
C
C
      SUBROUTINE DPOLRT (XCOF,COF,M,ROOTR,ROOTI,IER)
C
C        PURPOSE
C           COMPUTES THE REAL AND COMPLEX ROOTS OF A REAL POLYNOMIAL
C
C        USAGE
C           CALL POLRT(XCOF,COF,M,ROOTR,ROOTI,IER)
C
C        DESCRIPTION OF PARAMETERS
C           XCOF  -VECTOR OF M+1 COEFFICIENTS OF THE POLYNOMIAL
C                  ORDERED FROM SMALLEST TO LARGEST POWER
C           COF   -WORKING VECTOR OF LENGTH M+1
C           M     -ORDER OF POLYNOMIAL
```

```
C          ROOTR-RESULTANT VECTOR OF LENGTH M CONTAINING REAL ROOTS
C                   OF THE POLYNOMIAL
C          ROOTI-RESULTANT VECTOR OF LENGTH M CONTAINING THE
C                   CORRESPONDING IMAGINARY ROOTS OF THE POLYNOMIAL
C          IER  -ERROR CODE WHERE
C                   IER=0  NO ERROR
C                   IER=1  M LESS THAN ONE
C                   IER=2  M GREATER THAN 36
C                   IER=3  UNABLE TO DETERMINE ROOT WITH 500 INTERATIONS
C                          ON 8 STARTING VALUES
C                   IER=4  HIGH ORDER COEFFICIENT IS ZERO
C
C       REMARKS
C          LIMITED TO 36TH ORDER POLYNOMIAL OR LESS.
C          FLOATING POINT OVERFLOW MAY OCCUR FOR HIGH ORDER
C          POLYNOMIALS BUT WILL NOT AFFECT THE ACCURACY OF THE RESULTS.
C
C          NONE
C
C       METHOD
C          NEWTON-RAPHSON ITERATIVE TECHNIQUE.  THE FINAL ITERATIONS
C          ON EACH ROOT ARE PERFORMED USING THE ORIGINAL POLYNOMIAL
C          RATHER THAN THE REDUCED POLYNOMIAL TO AVOID ACCUMULATED
C          ERRORS IN THE REDUCED POLYNOMIAL.
C
C    ..................................................................
C
      DIMENSION XCOF(1),COF(1),ROOTR(1),ROOTI(1)
C
C       ...............................................................
C
C       IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C       C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C       STATEMENT WHICH FOLLOWS.
C
      DOUBLE PRECISION XCOF,COF,ROOTR,ROOTI,XO,YO,X,Y,XPR,YPR,UX,UY,V,
     1               YT,XT,U,XT2,YT2,SUMSQ,DX,DY,TEMP,ALPHA
C
C       THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C       APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C       ROUTINE.
C
C       THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C       CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  ABS IN STATEMENTS
C       78 AND 122 MUST BE CHANGED TO DABS.
C
C       ...............................................................
```

```
      IFIT=0
      N=M
      IER=0
      IF(XCOF(N+1))10,25,10
   10 IF(N) 15,15,32
C
C        SET ERROR CODE TO 1
C
   15 IER=1
   20 RETURN
C
C        SET ERROR CODE TO 4
C
   25 IER=4
      GO TO 20
C
C        SET ERROR CODE TO 2
C
   30 IER=2
      GO TO 20
   32 IF(N-36) 35,35,30
   35 NX=N
      NXX=N+1
      N2=1
      KJ1 = N+1
      DO 40 L=1,KJ1
      MT=KJ1-L+1
   40 COF(MT)=XCOF(L)
C
C        SET INITIAL VALUES
C
   45 XO=.00500101
      YO=0.01000101
C
C        ZERO INITIAL VALUE COUNTER
C
      IN=0
   50 X=XO
C
C        INCREMENT INITIAL VALUES AND COUNTER
C
      XO=-10.0*YO
      YO=-10.0*X
C
C        SET X AND Y TO CURRENT VALUE
C
      X=XO
      Y=YO
```

```
      IN=IN+1
      GO TO 59
   55 IFIT=1
      XPR=X
      YPR=Y
C
C        EVALUATE POLYNOMIAL AND DERIVATIVES
C
   59 ICT=0
   60 UX=0.0
      UY=0.0
      V =0.0
      YT=0.0
      XT=1.0
      U=COF(N+1)
      IF(U) 65,130,65
   65 DO 70 I=1,N
      L =N-I+1
      XT2=X*XT-Y*YT
      YT2=X*YT+Y*XT
      U=U+COF(L )*XT2
      V=V+COF(L )*YT2
      FI=I
      UX=UX+FI*XT*COF(L )
      UY=UY-FI*YT*COF(L )
      XT=XT2
   70 YT=YT2
      SUMSQ=UX*UX+UY*UY
      IF(SUMSQ) 75,110,75
   75 DX=(V*UY-U*UX)/SUMSQ
      X=X+DX
      DY=-(U*UY+V*UX)/SUMSQ
      Y=Y+DY
   78 IF(DABS(DY)+DABS(DX)-1.0E-05) 100,80,80
C
C        STEP ITERATION COUNTER
C
   80 ICT=ICT+1
      IF(ICT-500) 60,85,85
   85 IF(IFIT)100,90,100
   90 IF(IN-5) 50,95,95
C
C        SET ERROR CODE TO 3
C
   95 IER=3
      GO TO 20
  100 DO 105 L=1,NXX
      MT=KJ1-L+1
```

```
      TEMP=XCOF(MT)
      XCOF(MT)=COF(L)
  105 COF(L)=TEMP
      ITEMP=N
      N=NX
      NX=ITEMP
      IF(IFIT) 120,55,120
  110 IF(IFIT) 115,50,115
  115 X=XPR
      Y=YPR
  120 IFIT=0
  122 IF(DABS(Y/X)-1.0E-04) 135,125,125
  125 ALPHA=X+X
      SUMSQ=X*X+Y*Y
      N=N-2
      GO TO 140
  130 X=0.0
      NX=NX-1
      NXX=NXX-1
  135 Y=0.0
      SUMSQ=0.0
      ALPHA=X
      N=N-1
  140 COF(2)=COF(2)+ALPHA*COF(1)
  145 DO 150 L=2,N
  150 COF(L+1)=COF(L+1)+ALPHA*COF(L)-SUMSQ*COF(L-1)
  155 ROOTI(N2)=Y
      ROOTR(N2)=X
      N2=N2+1
      IF(SUMSQ) 160,165,160
  160 Y=-Y
      SUMSQ=0.0
      GO TO 155
  165 IF(N) 20,20,45
      END
C
      SUBROUTINE FREQ (NA,NB,A,B,G,W)
      DIMENSION A(60),AE(30),AO(30),AE1(30),AO1(30),B(60)
     1,BE(30),BO(30),BE1(30),BO1(30),G(4)
      CALL PARTS (NA,A,M1,AE,N1,AO,M11,AE1,N11,AO1)
      CALL PARTS (NB,B,M2,BE,N2,BO,M12,BE1,N12,BO1)
 33   EVN=SUM (M1,AE,W)
      ODDN=W*SUM (N1,AO,W)
      EV1N=W*SUM (M11,AE1,W)
      ODD1N=SUM (N11,AO1,W)
      EVD=SUM (M2,BE,W)
```

```
      CDDD=W*SUM (N2,BO,W)
      EV1D=W*SUM (M12,BE1,W)
      ODD1D=SUM (N12,BO1,W)
      TOP=EVN*EVN+ODDN*ODDN
      BOTTOM=EVD*EVD+ODDD*ODDD
      Y=ODDN*EVD-ODDD*EVN
      X=EVN*EVD+ODDD*ODDN
      IF (ABS(X+Y+TOP+BOTTOM) - 1.0E+65)  1,1,2
2     FREQO = W / 6.28318
      WRITE (6,3) W, FREQO
3     FORMAT (//,5X,'DUE TO OVERFLOW, THE FREQUENCY RESPONSE DATA ',/,
     19X,'WILL BE UNRELIABLE FOR',/,20X,'W = ',1PE15.6,'  RADIANS',
     2   ' = ' 1PE15.6, ' HZ')
1     IF (X.EQ. 0.0)   X=1.0E-50
      THETA=57.2957795*ATAN (Y/X)
      IF(X)  111,111,112
111   THETA=THETA-180.
112   AMPL=TOP/BOTTOM
 60   DELAY=(EVD*ODD1D+ODDD*EV1D)/BOTTOM-(EVN*ODD1N+ODDN*EV1N)/TOP
      G(1)=W
      G(2)=AMPL
      G(3)=DELAY
      G(4)=THETA
      RETURN
      END
C
C
      SUBROUTINE FREQQ(LIN,NOM,OMGMIN,OMGMAX,NA,NB,A,B,KEY1,R2)
      DIMENSION A(60),B(60),G(4),R1(100),R2(100),R3(100),R4(100),R(60)
      MAXPTS = 100
      KL=0
      KC=0
      SL=NOM
      RL=0.0
1     W=RL*(OMGMAX-OMGMIN)/SL+OMGMIN
12    WW=6.28318*W
      CALL FREQ (NA,NB,A,B,G,WW)
      KL=KL+1
      R1(KL) =W
      R2(KL) =4.3429448*ALOG(G(2))
22    R3(KL) =G(4)
      R4(KL) =G(3)
      RL=RL+1.0
      IF (KL-MAXPTS) 86,87,87
87    IF(KC)91,89,189
86    IF(OMGMAX-W)87,87,1
```

```
   89 GC TO(93,13),KEY1
  189 GO TO 113
113     WRITE (6,300)
300     FORMAT (1H1,5X, 'CONTINUATION OF ABOVE TABLE AND GRAPH')
        WRITE (6,301)
301     FORMAT(//,2(9X,10HFREQ (HZ)  ,4X,9HGAIN (DB) ,3X, 10HPHASE(DEG) ,
     12X,11HDELAY (SEC) ,2X)//)
        GO TO 91
   13 WRITE(6,8)
8       FORMAT (1H1,2(9X,10HFREQ (HZ) ,4X,9HGAIN (DB) ,3X,10HPHASE(DEG) ,
     12X,11HDELAY (SEC) ,2X)//)
        GO TO 91
128     WRITE (6,300)
        WRITE (6,302)
302     FORMAT(//,2(9X,10HFREQ (HZ)  ,4X,9HOHMS       ,3X,10HPHASE(DEG))//)
        GO TO 91
114     WRITE (6,300)
        WRITE (6,303)
303     FORMAT(//,2(9X,10HFREQ (HZ)  ,4X,9HMHOS       ,3X,10HPHASE(DEG))//)
   91 GO TO 90
   27 WRITE(6,33) (R1(J),R2(J),R3(J),J=1,KL)
      IF (LIN.EQ.1) CALL PLOT(R1,R2,KL)
      WRITE(6,200)
  200 FORMAT(//,10X,'PLOT OF ABOVE TABLE:  Y-AXIS  OHMS',/,32X,
     1    'X-AXIS  FREQ (HZ) ')
      IF (LIN.EQ.1) CALL PLOT(R1,R3,KL)
      WRITE (6,202)
  202 FORMAT(//,10X,'PLOT OF ABOVE TABLE:  Y-AXIS  PHASE (DEG)',
     1     /,32X,'X-AXIS  FREQ (HZ)')
33      FORMAT(2(F20.7,F15.7,F10.3))
        GO TO 93
   90 WRITE (6,9)  (R1(J),R2(J),P3(J),R4(J),J=1,KL)
9       FORMAT(2(F20.7,F15.7,F10.3,F15.7))
C
        CALL PLOT (R1,R2,KL)
        WRITE (6,203)
        CALL PLOT (R1,R3,KL)
        WRITE (6,204)
        CALL PLOT (R1,P4.KL)
        WRITE (6,205)
        GO TO 93
C
C
  203 FORMAT(//,10X,'PLOT OF ABOVE TABLE:  Y-AXIS  GAIN (DB)',
     1     /,32X,'X-AXIS  FREQ (HZ)')
  204 FORMAT(//,10X,'PLOT OF ABOVE TABLE:  Y-AXIS  PHASE (DEG)',
     1     /,32X,'X-AXIS  FREQ (HZ)')
  205 FORMAT(//,10X,'PLOT OF ABOVE TABLE:  Y-AXIS  DELAY (SEC)',
```

```
      1     /,32X,'X-AXIS   FREQ (HZ)')
C
93      IF(OMGMAX-W)94,94,95
95      KL=0
        KC=KC+1
        GO TO 1
94      RETURN
        END
C
C
        SUBROUTINE GROUP(G,C,H,ML,MP,MAP,ELT,ELTA,VAL,VALA,NPL,NAL,NE)
        DIMENSION MP(100,3),ML(50,5),ELT(100),
       1MAP(20,5),ELTA(20),VAL(100),VALA(20),C(50),G(50),
       2H(50)
        REAL IHC / 4HC    /, IHR/4HR    /, IHG /4HG    /
C       MAKE LIST OF ELEMENTS AND NODE NUMBERS IN BOTH CURRENT AND VOLTAGE
C       GRAPH
        KK=0
        NLL=NPL+NAL
C       MAKE PARALLEL RLC ELEMENTS INTO SINGLE TOPOLOGICAL ELEMENT
        DO 120 K=1,NLL
        G(K)=0.
        C(K)=0.
  120   H(K)=0.
C       PICK JTH ELEMENT
        DO 109 J=1,NPL
            IF (ELT(J) .EQ. IHC)  GO TO 197
196     IF(VAL(J))197,195,197
195     VAL(J)=.00001
        WRITE (6,194) J
194     FORMAT(//,10X,4HTHE ,I2,50HTH ELEMENT VALUE HAS BEEN REPLACED BY
       1.00001          )
197     ML(J,1)=MP(J,1)
        ML(J,2)=MP(J,2)
        DO 121 K=1,J
C       COMPARE JTH WITH PRECEDING ELEMENTS (BY NODE NUMBER)
        IF(MP(J,1)-ML(K,1))121,103,121
  103   IF(MP(J,2)-ML(K,2))121,104,121
  104   IF(J-K)111,110,111
  110   JK=J-KK
C       IF NOT SAME AS PRECEDING ELEMENTS,PLACE IN PERMANENT LIST OF ELETS
        ML(JK,1)=MP(J,1)
        ML(JK,2)=MP(J,2)
        ML(JK,3)=JK
      2 ML(JK,4)=MP(J,1)
        ML(JK,5)=MP(J,2)
```

```
      MP(J,3)=JK
C     TEST FOR TYPE OF ELEMENT AND STORE VALUE
    1 IF(ELT(J) .NE. IHR) GO TO 106
  105 G(JK)=1./VAL(J)+G(JK)
      GO TO 109
  106 IF(ELT(J) .NE. IHC) GO TO 108
  107 C(JK)=VAL(J)+C(JK)
      GO TO 109
  108 H(JK)=1./VAL(J)+H(JK)
      GO TO 109
C     IF SAME AS PRECEDING ELEMENT,ADD VALUE TO THAT OF PRECEDING
  111 MP(J,3)=K
      KK=KK+1
         IF (ELT(J) .NE. IHR)  GO TO 116
  115 G(K)=1./VAL(J)+G(K)
      GO TO 109
  116 IF(ELT(J) .NE. IHC) GO TO 118
  117 C(K)=VAL(J)+C(K)
      GO TO 109
  118 H(K)=1./VAL(J)+H(K)
      GO TC 109
  121 CONTINUE
  109 CONTINUE
      NPE=NPL-KK
      IF(NAL)100,99,100
C     TRANSFER ACTIVE ELEMENT NODE NUMBERS TO PERMANENT LIST
  100 DO 98 J=1,NAL
      NM=NPE+J
      ML(NM,1)=MAP(J,1)
      ML(NM,2)=MAP(J,2)
      ML(NM,3)=NM
      ML(NM,4)=MAP(J,3)
      ML(NM,5)=MAP(J,4)
      MAP(J,5)=NM
         IF (ELT(J) .NE. IHG)  GO TO 11
   10 G(NM)=VALA(J)+G(NM)
      GO TO 98
   11 H(NM)=VALA(J)+H(NM)
   98 CONTINUE
   99 NE=NPL+NAL-KK
      RETURN
      END
C

      FUNCTION IGN(NN,LN,MG)
      DIMENSION MG(30,5),ME(20,2),MM(100)
      KI=1
```

```
      K2=2
      K3=1
   19 DO 21 J=1,NN
   21 MM(J)=0
      K=0
      MM(1)=1
    9 KK=K
      DO 7 J=1,LN
      M=MG(J ,K1)
      N=MG(J ,K2)
      IF(MM(N))1,1,3
    1 IF(MM(M))7,7,4
    3 IF(MM(M))14,14,7
    4 ME(N,K3)=-MG(J,3)
      MM(N)=MM(N)+1
   30 K=K+1
      IF(K-LN)7,10,10
   14 ME(M,K3)=MG(J,3)
      MM(M)=MM(M)+1
      GO TO 30
    7 CONTINUE
      IF(KK-K)9,10,9
   10 IF(K1-4)81,80,81
   81 K1=4
      K2=5
      K3=2
      GO TO 19
80    KZ=2
   35 DO 31 J=2,NN
      DO 34 K=2,NN
      JJJ=J
      KKK=K
      IF(IABS (ME(J,1))-IABS (ME(K,2)))32,33,32
   33 IF(J-K)34,31,34
   32 CONTINUE
   31 CONTINUE
      GO TO 55
34    MY=ME(KKK,2)
      ME(KKK,2)=ME(JJJ,2)
      ME(JJJ,2)=MY
      KZ=KZ+1
      GO TO 35
   55 DO 40 J=2,NN
      IF(ME(J,1)-ME(J,2))41,40,41
   41 KZ=KZ+1
   40 CONTINUE
       IGN=(-1)**KZ
      RETURN
```

```
      END
C
C
      SUBROUTINE MAKPOL (N,ROOTR,ROOTI,CR,CI)
      REAL*8  ROOTR(1),ROOTI(1),CR(1),CI(1)
C
C     TWO REAL*8 ARRAYS OF SIZE (N+1) MUST BE FURNISHED
C        FOR THE COEFFICIENTS CR (REAL PART) AND CI (IMAG PART)
C
C     N      NUMBER OF ROOTS
C     RCOTR  ARRAY OF REAL PARTS OF ROOTS
C     ROOTI  ARRAY OF IMAG PARTS OF ROOTS
C     CR     ARRAY OF REAL PARTS OF COEFF
C     CI     ARRAY OF IMAG PARTS OF COEFF
C
C
      CR(N+1) = 1.0
      CI(N+1) = 0.0
      IF ( N.LE. 0)  RETURN
      DO 10 I=1,N
      CR(I) = ROOTR(I)
   10 CI(I) = ROOTI(I)
      CR(N+1) = 1.0
      CI(N+1) = 0.0
      K=N
      M=N-1
      DO 20  L=1,M
      DO 30  I=2,K
      CR(I) = CR(I) + CR(I-1)
   30 CI(I) = CI(I) + CI(I-1)
      K= K-1
      DO 20  I=1,K
      J = I+L
      IF (DABS(ROOTI(J)).LE.1.D-35) ROOTI(J)=0.
      IF (DABS(ROOTR(J)).LE.1.D-35) ROOTR(J)=0.
      IF (DABS(CR(I)).LE.1.D-35) CR(I)=0.
      IF (DABS(CI(I)).LE.1.D-35) CI(I)=0.
      CR(I) = ROOTR(J)*CR(I) - ROOTI(J)*CI(I)
   20 CI(I) = RCOTR(J)*CI(I) + ROOTI(J)*CR(I)
      K=N/2
      K=2*K/(N-K)
      DO 40  I=K,N,2
      CR(I) = -CR(I)
   40 CI(I) = -CI(I)
      RETURN
      END
C
C
```

```
      SUBROUTINE MULLER(ZRO,N8,Z)
C
C        ZRO      COEFF IN ASCENDING ORDER
C        N8    =   ORDER OF POLNOMIAL PLUS ONE
C        Z(I,1)  =   REAL ROOT
C        Z(I,2)  =   CORRESPONDING IMAG PART OF ROOT
C
      DIMENSION ZRO(60),COE(60),Z(60,2)
      N1=N8-1
      IF(N1)1000,1000,1001
C1000 WRITE(6,1119)
1119  FORMAT(//,20X,4HNONE   )
 1000 GO TO 300
1001  DO 1003 J=1,N8
      K=N8-J+1
1003  COE(J)=ZRO(K)
      N2=N1+1
      N4=0
      I=N1+1
   19 IF(COE(I))9,7,9
    7 N4=N4+1
      Z(N4,1)=0.
      Z(N4,2)=0.
      I=I-1
      IF(N4-N1)19,37,19
    9 CONTINUE
   10 AXR=0.8
      AXI=0.
      L=1
      N3=1
      ALP1R=AXR
      ALP1I=AXI
      M=1
      GOTO99
   11 BET1R=TEMR
      BET1I=TEMI
      AXR=0.85
      ALP2R=AXR
      ALP2I=AXI
      M=2
      GOTO99
   12 BET2R=TEMR
      BET2I=TEMI
      AXR=0.9
      ALP3R=AXR
      ALP3I=AXI
```

```
      M=3
      GOTO99
   13 BET3R=TEMR
      BET3I=TEMI
   14 TE1=ALP1R-ALP3R
      TE2=ALP1I-ALP3I
      TE5=ALP3R-ALP2R
      TE6=ALP3I-ALP2I
      TEM=TE5*TE5+TE6*TE6
      IF (TEM .EQ. 0.0)   TEM = 1.0E-35
      TE3=(TE1*TE5+TE2*TE6)/TEM
      TE4=(TE2*TE5-TE1*TE6)/TEM
      TE7=TE3+1.
      TE9=TE3*TE3-TE4*TE4
      TE10=2.*TE3*TE4
      DE15=TE7*BET3R-TE4*BET3I
      DE16=TE7*BET3I+TE4*BET3R
      TE11=TE3*BET2R-TE4*BET2I+BET1R-DE15
      TE12=TE3*BET2I+TE4*BET2R+BET1I-DE16
      TE7=TE9-1.
      TE1=TE9*BET2R-TE10*BET2I
      TE2=TE9*BET2I+TE10*BET2R
      TE13=TE1-BET1R-TE7*BET3R+TE10*BET3I
      TE14=TE2-BET1I-TE7*BET3I-TE10*BET3R
      TE15=DE15*TE3-DE16*TE4
      TE16=DE15*TE4+DE16*TE3
      TE1=TE13*TE13-TE14*TE14-4.*(TE11*TE15-TE12*TE16)
      TE2=2.*TE13*TE14-4.*(TE12*TE15+TE11*TE16)
      TEM=SQRT (TE1*TE1+TE2*TE2)
      IF(TE1),13,113,112
  113 TE4=SQRT (.5*(TEM-TE1))
      IF(TE4.EQ.0.0)TE4=1.E-35
      TE3=.5*TE2/TE4
      GO TO 111
  112 TE3=SQRT (.5*(TEM+TE1))
      IF(TE2)110,200,200
  110 TE3=-TE3
 200  TE3A = TE3
      IF (TE3A .EQ. 0.0)  TE3A = 1.0E-35
      TE4=.5*TE2/TE3A
  111 TE7=TE13+TE3
      TE8=TE14+TE4
      TE9=TE13-TE3
      TE10=TE14-TE4
      TE1=2.*TE15
      TE2=2.*TE16
      IF(TE7*TE7+TE8*TE8-TE9*TE9-TE10*TE10)204,204,205
  204 TE7=TE9
```

```
      TE8=TE10
  205 TEM=TE7*TE7+TE8*TE8
      IF(TEM.EQ.0.0)TEM=1.0E-35
      TE3=(TE1*TE7+TE2*TE8)/TEM
      TE4=(TE2*TE7-TE1*TE8)/TEM
      AXR=ALP3R+TE3*TE5-TE4*TE6
      AXI=ALP3I+TE3*TE6+TE4*TE5
      ALP4R=AXR
      ALP4I=AXI
      M=4
      GO TO 99
   15 N6=1
 38   IF (ABS (HELL)+ABS (BELL)-1.E-20) 18,18,16
   16 TE7=ABS (ALP3R-AXR)+ABS (ALP3I-AXI)
      AXA = AXR
      IF ((AXR .EQ. 0.0) .AND. (AXI .EQ. 0.0))  AXA = 1.0E-35
      IF(TE7/(ABS (AXA)+ABS (AXI))-1.E-7)18,18,17
   17 N3=N3+1
      ALP1R=ALP2R
      ALP1I=ALP2I
      ALP2R=ALP3R
      ALP2I=ALP3I
      ALP3R=ALP4R
      ALP3I=ALP4I
      BET1R=BET2R
      BET1I=BET2I
      BET2R=BET3R
      BET2I=BET3I
      BET3R=TEMR
      BET3I=TEMI
      IF(N3-100)14,18,18
   18 N4=N4+1
      Z(N4,1)=ALP4R
      Z(N4,2)=ALP4I
      N3=0
   41 IF(N4-N1)30,37,37
C  37 WRITE(6,555)
 555  FORMAT (//,2(11X,9HREAL PART,9X,9HIMAG PART,2X))
   37 CALL RTCK (ZRO,N8,Z)
C     WRITE(6,666) (Z(I,1),Z(I,2),I=1,N1)
 666  FORMAT (/,2(1PE22.7,1PE18.7))
      GO TO 300
30    IF(ABS (Z(N4,2))-1.E-5)10,10,31
   31 GO TO(32,10),L
   32 AXR=ALP1R
      AXI=-ALP1I
      ALP1I=-ALP1I
      M=5
```

```
      GC TO 99
   33 BET1R=TEMR
      BET1I=TEMI
      AXR=ALP2R
      AXI=-ALP2I
      ALP2I=-ALP2I
      M=6
      GO TO 99
   34 BET2R=TEMR
      BET2I=TEMI
      AXR=ALP3R
      AXI=-ALP3I
      AIP3I=-ALP3I
      L=2
      M=3
   99 TFMR=COE(1)
      TEMI=0.0
      DC 100 I=1,N1
      TE1=TEMR*AXR-TEMI*AXI
      TEMI=TEMI*AXR+TEMR*AXI
  100 TEMR=      TE1+COE(I+1)
      HELL=TEMR
      BELL=TEMI
   42 IF(N4)102,103,102
  102 DO101I=1,N4
      TEM1=AXR-Z(I,1)
      TEM2=AXI-Z(I,2)
      TE1=TEM1*TEM1+TEM2*TEM2
      TE1A = TE1
      IF (TE1A .EQ. 0.0)  TE1A = 1.0E-35
      TE2=(TEMR*TEM1+TEMI*TEM2)/TE1A
      TEMI=(TEMI*TEM1-TEMR*TEM2)/TE1A
  101 TEMR=TE2
  103 CO TO(11,12,13,15,33,34),M
300   RETURN
      END
C
C
      SUBROUTINE OPT(NML,NEL,KEY1)
      DIMENSION NML(50,5)
      GO TO (4,5,5,6),KEY1
4     NS=1
      GO TO 7
5     NS=2
      GC  TO 7
6     NS=3
```

```
7       K1=1
        K2=2
        GO TO 3
1       K1=4
        K2=5
3       DO 109 J=NS,NEL
        IF(J-NS) 108,109,108
108     KK=J-1
        DO 121 K=NS,KK
        IF(NML(J,K1)-NML(K,K1))1121,103,1121
103     IF(NML(J,K2)-NML(K,K2))121,104,121
1121    IF(NML(J,K1)-NML(K,K2))121,122,121
122     IF(NML(J,K2)-NML(K,K1))121,104,121
104     DO 100 JJ=1,5
        N1=NML(J,JJ)
        LL=J-K-1
        IF(LL)100,100,107
107     DO 101 L=1,LL
        L1=J-L+1
101     NML(L1,JJ)=NML(L1-1,JJ)
100     NML(K+1,JJ)=N1
        GO TO 109
121     CONTINUE
109     CONTINUE
        IF(K1-1)2,1,2
2       RETURN
        END
C
C
        SUBROUTINE PARTS (NA,A,MK,AE,NK,AO,M1K,AE1,N1K,AO1)
        DIMENSION A(60),AE(30),AO(30),AE1(30),AO1(30)
        I=1
        MK=1
        NK=0
        M1K=0
        AE(1)=A(1)
        IF(NA-I)3,3,1
1       I=I+1
        NK=NK+1
        AO(NK)=A(I)
        DUMMY=I-1
        AO1(NK)=DUMMY*A(I)
        IF(NA-I)3,3,2
2       M1K=MK
        MK=MK+1
        I=I+1
```

```
      AE(MK)=A(I)
      DUMMY=I-1
      AE1(M1K)=DUMMY*A(I)
      IF(NA-I)3,3,1
 3    N1K=NK
      RETURN
      END
C
C
      SUBROUTINE PLOT(X,Y,NN)
C
C***  GUAGE INPUT & FIND MAX & MIN FOR X & Y; CALL UTPLOT
C       X - THE X-AXIS COORDINATE
C       Y - THE Y-AXIS COORDINATE
C       NN- THE NUMBER OF POINTS TO BE PLOTTED
C
      DIMENSION X(NN), Y(NN), RANGE(4)
      EQUIVALENCE (RANGE(1),XMAX), (RANGE(2),XMIN),(RANGE(3),YMAX),
     1            (RANGE(4),YMIN)
      IF (NN .GE. 4)  GO TO 200
      WRITE (6,201)
201   FORMAT (10(/),10X,'GRAPHS OF LESS THAN FOUR POINTS ARE NOT PLOTTED
     1.')
      RETURN
200   WRITE (6,300)
  300 FORMAT (1H1)
      XMAX=-1.E20
      XMIN=1.E20
      YMAX=-1.E20
      YMIN=1.E20
      DO 1 I=1,NN
      IF(X(I)-XMAX) 6,6,2
 2    XMAX=X(I)
      YXMAX=Y(I)
 6    IF(X(I)-XMIN) 3,3,7
 3    XMIN=X(I)
      YXMIN=Y(I)
 7    IF(Y(I)-YMAX) 8,8,4
 4    YMAX=Y(I)
      XYMAX=X(I)
 8    IF(Y(I)-YMIN) 5,5,1
 5    YMIN=Y(I)
      XYMIN=X(I)
 1    CONTINUE
      CALL UTPLOT(X,Y,NN,RANGE,1)
      WRITE (6,101) YMAX, XYMAX,YMIN,XYMIN
```

```
101   FORMAT (/,10X,
     1      'MAX Y='1PE12.4,' AT X='E12.4,20X,'MIN Y=' E12.4,' AT X='
     2   E12.4)
      WRITE (6,100) XMAX, YXMAX, XMIN, YXMIN
100   FORMAT (/,10X,
     1      'MAX X='1PE12.4,' AT Y='E12.4,20X,'MIN X=' E12.4,' AT Y='
     2   E12.4)
      RETURN
      END
C
C
      SUBROUTINE RTCK (ZRO,N8,Z)
      REAL*8 XCOF(37),COF(37),ROOTR(36),ROOTI(36),DZR(36),DZI(36),
     1   COF1(37),COF2(37)
      DIMENSION ZRO(1),Z(60,2)
      M=N8-1
      DO 10 I=1,N8
10    XCOF(I) = ZRO(I)
      CALL DPOLRT (XCOF,COF,M,ROOTR,ROOTI,IER)
      IF (IER .EQ. 0)  GO TO 20
      WRITE (6,200)
200   FORMAT (//,' DPOLRT UNABLE TO FACTOR.  NO ROOT CHECK.',//)
      RETURN
C
C     FORM TWO NEW POLYNOMIALS FROM THE TWO SETS OF ROOTS
C
20    DO 21  I=1,M
      DZR(I) = Z(I,1)
21    DZI(I) = Z(I,2)
      CALL MAKPOL (M,DZR,DZI,COF1,COF)
      CALL MAKPOL (M,ROOTR,ROOTI,COF2,COF)
C
C     CALCULATE THE ERROR CRITERIA
C     THE SET OF ROOTS TO BE PRINTED OUT AND USED WILL BE THE  SET WHICH
C          WHEN EXPANDED YIELD THE MOST NEARLY CORRECT COEFFICIENTS WHEN
C          COMPARED (ABS VALUES) TO THE ORIGINAL POLYNOMIAL COEFFICIENTS
C
      ERR1 = 0.0
      ERR2 = 0.0
      DO 30 I=1,N8
      ERR1 =DABS(COF1(I)-XCOF(I)) + ERR1
30    ERR2 =DABS(COF2(I)-XCOF(I)) + ERR2
      IF (ERR2 .GT. ERR1)  GO TO 50
      DO 40 I=1,M
      Z(I,1) = ROOTR(I)
40    Z(I,2) = ROOTI(I)
```

```
50     RETURN
       END
C
C
       SUBROUTINE SORT(ML,NN,NM,NE,NEL,NAL,NALL,KEY1,NML,
      1JI,KI,JO,KO)
       DIMENSION ML(50.5),NML(50,5)
       NALL=NAL
       GC TO(20,21,21,22),KEY1
   20  NM=1
       NFL=NE
       GC TO 23
   21  NM=2
       NEL=NE+1
       GO TO 23
   22  NM=3
       NEL=NE+2
   23  DC 100 J=NM,NEL
       JJ=J+1-NM
       DO 101 K=1,5
  101  ML(J,K)=NML(JJ,K)
  100  CONTINUE
       GO TO(1,2,2,2),KEY1
    1  RETURN
C      REMOVE ELEMENTS IN PARALLEL WITH INPUT
    2  DC 109 J=NM,NEL
       IF(ML(J,1)-JI)110,111,110
  111  IF(ML(J,2)-KI)109,112,109
  110  IF(ML(J,1)-KI)109,113,109
  113  IF(ML(J,2)-JI)109,112,109
C      INSERT LAST ELEMENT IN PLACE OF REMOVED ELEMENT
  112  DO 102 K=1,5
  102  ML(J,K)=ML(NEL,K)
       NEL=NEL-1
       GO TO 2
  109  CONTINUE
       GO TO(1,3,4,4),KEY1
C      INSERT TEST ELEMENT ACRCSS INPUT
    3  ML(1,1)=JI
       ML(1,2)=KI
       ML(1,3)=99
       ML(1,4)=JI
       ML(1,5)=KI
       RETURN
C      REMOVE ELEMENTS IN PARALLEL WITH OUTPUT
    4  DO 115 J=NM,NEL
```

```
      IF(ML(J,4)-JO)117,116,117
  116 IF(ML(J,5)-KO)115,119,115
  117 IF(ML(J,4)-KO)115,120,115
  120 IF(ML(J,5)-JO)115,119,115
C     INSERT LAST ELEMENT IN PLACE OF REMOVED ELEMENT
  119 DO 105 K=1,5
  105 ML(J,K)=ML(NEL,K)
      NEL=NEL-1
      GO TO 4
  115 CONTINUE
      GO TO(1,3,5,6),KEY1
C     INSERT TEST ELEMENT ACROSS INPUT IN CURRENT GRAPH,ACROSS OUTPUT
C     IN VOLTAGE GRAPH
    5 ML(1,1)=JI
      ML(1,2)=KI
      ML(1,3)=98
      ML(1,4)=JO
      ML(1,5)=KO
      NALL=NALL+1
      RETURN
C     INSERT TEST ELEMENT ACROSS INPUT AND OUTPUT
    6 ML(2,1)=JI
      ML(2,2)=KI
      ML(2,3)=97
      ML(2,4)=JI
      ML(2,5)=KI
      ML(1,1)=JO
      ML(1,2)=KO
      ML(1,3)=96
      ML(1,4)=JO
      ML(1,5)=KO
      RETURN
      END
C
C
      FUNCTION  SUM (N,A,W)
      DIMENSION A(30)
      SUM =0.0
      IF(N)3,3,2
    2 SUM =A(N)
      IF(N-1)3,3,4
    4 X=-W*W
      NM1=N-1
      DO 5 I=1,NM1
      K=N-I
    5 SUM =SUM *X+A(K)
```

```
  3     RETURN
        END
C
C
        SUBROUTINE TEST(MG,ML,NEL,NALL,LN,KYOUT,K10)
        DIMENSION MG(30,5),ML(50,5),MM(50)
C       THIS TEST IS MADE BY TRYING TO ADD ONE NEW NODE AT A TIME UNTIL
C       ALL NODES HAVE BEEN ACCOUNTED FOR (SEE TRETST)
  2     K1=1
        K2=2
        GO TO 11
 25     K1=4
        K2=5
 11     DC 21 J=1,NEL
 21     MM(J)=0
        K11=LN
 32     DO 30 J=K10,NEL
        K11=K11+1
        MG(K11,3)=ML(J,3)
        MG(K11,K1)=ML(J,K1)
 30     MG(K11,K2)=ML(J,K2)
        K=1
        M=MG(1,K1)
        N=MG(1,K2)
        MM(M)=1
        MM(N)=1
  9     KK=K
        DC 7 J=2,K11
        M=MG(J,K1)
        N=MG(J,K2)
        IF(MM(N))1,1,3
  1     IF(MM(M))7,7,4
  3     IF(MM(M))4,4,7
  4     MM(M)=MM(M)+1
        MM(N)=MM(N)+1
        K=K+1
        IF(K-LN)7,10,10
  7     CONTINUE
        IF(KK-K)9,8,9
  8     KYOUT=-1
        RETURN
 10     IF(NALL)17,17,110
110     IF(K1-4)25,17,25
 17     KYOUT=1
        RETURN
        END
```

```
C
C
      SUBROUTINE TOPOL(NPL,NAL,NN,JI,KI,JO,KO,KEY1,KEY2,MP,ELT,VAL,
     1MAP,ELTA,VALA,Y1,Y2,VALL,KW,NVAR,KEY3,Y,NP,JP)
      DIMENSION VAL (100)
      DIMENSION MP(100,3),ML(50,5),ELT(100),MAP(20,5),ELTA(20),VALA(20),
     1C(50),G(50),H(50),MG(30,5),NG(50),NML(50,5),            X(60),Y(
     260),Y1(60),Y2(60),VALL(50)
      REAL IH1 /4H-   /, IH2 / 4H+   /
C     IF NETWORK FUNCTION HAS ALREADY BEEN CALCULATED IN BILINEAR FORM,
C     CHANGE ONLY PART OF FUNCTION
      GO TO (6,102),KEY3
102   IF(NV)131,131,130
131   NVV=MP(NVAR,3)
      GO TO 132
130   NVV=MAP(NV,5)
      GC TC 133
132   IF(C(NVV))133,134,133
133   DO 135 J=1,NP
135   Y1(J)=VALL(KW)*Y1(J)/VALL(KW-1)
      GO TO 48
134   DO 136 J=1,NP
136   Y1(J)=VALL(KW-1)*Y1(J)/VALL(KW)
      GO TO 48
C     COMBINE PARALLEL RLC ELEMENTS INTO SINGLE TOPOLOGICAL ELEMENT
6     CALL GROUP(G,C,H,NML,MP,MAP,ELT,ELTA,VAL,VALA,NPL,NAL,NE)
C     INSERT TEST ELEMENTS ACROSS INPUT,OUTPUT
      CALL SORT(ML,NN,NM,NE,NEL,NAL,NALL,KEY1,NML,JI,KI,JO,KO)
      CALL OPT(ML,NEL,KEY1)
      GO TO (103,103,103,103,104),KEY2
C     PRINT ELEMENT NUMBER,CORRESPONDING TOPOLOGICAL ELEMENT NUMBER
  104 WRITE (6,80)
      WRITE (6,105) (J,MP(J,3),J=1,NPL)
80    FORMAT(//,1X,19HELEMENT ASSOCIATION  ,/,1X,8HPHYSICAL  ,1X,11HTOPO
     1LOGICAL ,//)
105   FCRMAT(I5,8X,1HY,I2)
      IF(NAL)106,81,106
  106 WRITE (6,105) (J,MAP(J,5),J=1,NAL)
   81 WRITE (6,82)
82    FORMAT(//,1X,25HTREES,2-TREES,OR 3-TREES  ,//)
103   LN=NN-1
      NV=NVAR-NPL
      KK=0
      NP=2*LN+1
      DO 274 J=1,NP
      Y1(J)=0.
```

```
      Y2(J)=0.
274   Y(J)=0.
      LM=NN-2
      MY=0
      MK=NEL+2-NN
      L=NN-1
      I=1
C     PICK FIRST SET OF ELEMENTS TO BE TESTED AS TREE
      DO 5 K=1,LN
      MAX=MAX+K
      NG(K)=K
      DO 8 J=1,5
8     MG(K,J)=ML(K,J)
5     CONTINUE
      KY1N=0
C     TEST FOR TREE IN CURRENT GRAPH
100   CALL TRETST(NN,LN,KY1N,KYOUT,LL,MG)
      IF(KYOUT)17,17,18
   17 IF(MY)21,20,21
   21 MY=0
      KY1N=0
      GO TO 121
   20 KY1N=-1
C     IF CIRCUIT HAS BEEN FORMED,REMOVE FATAL ELEMENT FROM LIST
121   IF(LL)272,19,272
272   IF(LL-LN)281,19,19
281   IF(KK-I-NEL)282,19,19
282   L=LL+1
      GO TO 50
C     IF NETWORK IS ACTIVE,TEST VOLTAGE GRAPH
18    IF(NALL)233,235,233
235   II=1
      GO TO 899
  233 IF(MY)35,34,35
   34 MY=1
      KY1N=1
      GO TO 100
C     IF NETWORK IS ACTIVE,TEST FOR SIGN OF TREE
   35 MY=0
      KY1N=0
      II=IGN(NN,LN,MG)
899   I=I+1
857   IF(II)870,871,871
870   AA=1H1
      GO TO 872
871   AA=1H2
C     ASSEMBLE TREE-ADMITTANCE PRODUCT
872   CALL ASBS(LN,C,G,H,X,NP,MG,I,KEY1,JN,JP,II,NPL,NAL)
```

```
C       IF TO BE CALCULATED IN BILINEAR FORM,TEST FOR VARIABLE ELEMENT
        IF(NVAR)146,45,146
146     IF(NV)347,347, 46
347     DO 147 K=JP,LN
        IF(MG(K,3)-MP(NVAR,3))147,44,147
147     CONTINUE
        GO TO 45
46      DO 148 K=JP,LN
        IF(MG(K,3)-MAP(NV,5))148,44,148
148     CONTINUE
        GO TO 45
44      DO 144 J=1,NP
144     Y1(J)=Y1(J)+X(J)
        GO TO 149
45      DO 145 J=1,NP
145     Y2(J)=Y2(J)+X(J)
149     GO TO (19,19,19,19,89),KEY2
C       PRINT SYMBOLIC TREE ADMITTANCE PRODUCT
     89 WRITE (6,900) AA, (MG(J,3),J=JP,LN)
900     FORMAT(1X,A1,2X,20(1HY,I2,2X)/(21(1HY,I2,2X)))
C       PERMUTE ELEMENT
19      IF(NG(1)-MK)47,48,48
47      IF(KK-1-NEL)747,50,50
747     K=NG(1)
        DO 265 NY=1,5
    265 MG(1,NY)=ML(K,NY)
        IF(NG(L)-NEL)49,50,50
49      NG(L)=NG(L)+1
        K=NG(L)
        DO 77 J=1,5
77      MG(L,J)=ML(K,J)
        GO TO 100
C       PERMUTE NEXT ELEMENT IN LIST
C
C     EXAMINE THE ORIGINAL IN THIS AREA
C
     50 L = L-1
        NG(L) = NG(L) + 1
        DO 51 K=L,LM
     51 NG(K+1)=NG(K)+1
        KK=NG(L)
        DO 52 J=L,LN
        DO 267 NY=1,5
    267 MG(J,NY)=ML(KK,NY)
52      KK=KK+1
C       IF 2-TREE OR 3-TREE,SEE IF TEST ELEMENTS HAVE BEEN PERMUTED
        GO TO (427,428,428,429),KEY1
428      IF(NG(1)-1)427,427,48
```

```
429    IF(NG(2)-2)427,427,48
427    IF(KK-1-NEL)53,100,100
C      AFTER PERMUTATION,TEST FOR CONNECTED GRAPH
53     CALL TEST(MG,ML,NEL,NALL,LN,KYOUT,KK)
       IF(KYOUT)1050,953,953
1050   IF(L-1)48,48,50
953    L=NN-1
       GO TO 100
C      END TREE TEST,ASSEMBLE AND SHIFT COEFFICIENTS
48     DO 550 J=1,NP
550    Y(J)=Y1(J)+Y2(J)
       IF(JP-1)331,331,555
555    JW=JP-1
       NP=NP+2*JW
       DO 556 J=1,NP
       K=NP-J+1
       KK=K+JW
556    Y(KK)=Y(K)
       DO 557 J=1,JW
       NW=NP-J+1
       Y(J)=0.
557    Y(NW)=0.
  331  RETURN
       END
C
C
       SUBROUTINE TRETST(NN,LN,KY1N,KYOUT,LL,MG)
       DIMENSION MG(30,5),MM(50) ,MMM(50)
C      THIS TEST IS MADE BY TRYING TO ADD ONE NEW NODE AT A TIME UNTIL
C      ALL NODES HAVE BEEN ACCOUNTED FOR
       LL=0
       IF(KY1N)11,2,25
2      K1=1
       K2=2
       GO TO 11
25     K1=4
       K2=5
   11  DO 21 J=1,NN
       MMM(J)=0
   21  MM(J)=0
       K=1
       M=MG(1,K1)
       N=MG(1,K2)
       MM(M)=1
       MM(N)=1
    9  KK=K
```

```
      DO 7 J=2,LN
      M=MG(J,K1)
      N=MG(J,K2)
C     IF ONE NEW NODE IS ADDED,STORE ELEMENT NUMBER
C     IF NO NEW NODE IS ADDED AND ON FIRST PASS,CIRCUIT IS PRESENT,
C     STORE NUMBER OF FATAL CIRCUIT ELEMENT AND RETURN
C     IF TWO NEW NODES ARE ADDED,GO TO NEXT ELEMENT
      IF(MM(N))1,1,3
    1 IF(MM(M))7,7,4
    3 IF(MM(M))4,4,13
   13 IF(KK-1)7,14,15
   15 IF(MG(J,3)-MMM(J))8,7,8
   14 LL=J
      GO TO 8
    4 MM(M)=MM(M)+1
      MM(N)=MM(N)+1
      MMM(J)=MG(J,3)
      K=K+1
      IF(K-LN)7,10,10
   10 KYOUT=1
      RETURN
    7 CONTINUE
      IF(KK-K)9,8,9
    8 KYOUT=-1
      RETURN
      END
C
C
      SUBROUTINE UTPLOT (X  ,Y  ,NDATA,RANGE,KKZ)
      DIMENSION GRID(61,101),XSCALE(6),YSCALE(7)
      DIMENSION X  (1),Y  (1),RANGE(4)
      INTEGER*2  ZERO/1HO/,IYORG(101)
      INTEGER*2  GRID,BLANK,XCHAR,DOT
      DATA  DOT,XCHAR,BLANK  /2H. ,2HX ,2H  /
C
C     GRID IS THE MATRIX USED TO PLOT THE POINTS
C
      DO 8  I=1,101
    8 IYORG(I) = BLANK
      IERR=0
      XMAX=RANGE(1)
      XMIN=RANGE(2)
      YMAX=RANGE(3)
      YMIN=RANGE(4)
C
C     CHECKING X AND Y POINTS,PLOTTING THOSE OUT OF RANGE
```

```
C       AT THE MARGIN
        DC 30 I=1,NDATA,KKZ
        IF(X   (I)-XMAX) 205,205,220
  220   X   ( )=XMAX
        IERR=IERR+1
        GOTO 210
  205   IF(X   (I)-XMIN)203,210,210
  203   X   (I)=XMIN
        IERR=IERR+1
  210   IF(Y   (I)-YMAX)215,215,212
  212   Y   (I)=YMAX
        IERR=IERR+1
        GOTO 30
  215   IF(Y   (I)-YMIN)217,  30,30
  217   Y   (I)=YMIN
        IERR=IERR+1
C
   30 CONTINUE
C
C       PLOTTING X AND Y AXIS , IF NECESSARY
        XRANGE=XMAX-XMIN
        YRANGE=YMAX-YMIN
C
C       BLANKING OUT MATRIX-(GRID)
C
        DC 300 I=1,61
        DO 301 JJ=1,101
301     GRID(I,JJ)=BLANK
  300   CONTINUE
        YTEST=YMAX*YMIN
        XTEST=XMAX*XMIN
        IF(XTEST)1,222,222
  222   IF(YTEST)333,444,444
    1   IYAXIS = 100.*(-XMIN)/XRANGE+1.5
        IYORG(IYAXIS-1) = ZERO
        IYORG(IYAXIS) = DOT
        IYORG(IYAXIS+1) = ZERO
        DO 40 I=1,61
   40   GRID(I,IYAXIS)=DOT
        GO TO 222
  333   IXAXIS=60.*YMAX/YRANGE+1.5
        DO 60 I=1,101
   60   GRID(IXAXIS,I)=DOT
C
C       PLACING POINTS IN THEIR PROPER GRID POSITIONS
C
  444   DO 70 I=1,NDATA,KKZ
        IPTX=60.*(YMAX-Y(I))/YRANGE+1.5
```

```
      IPTY = 100.*(X(I)-XMIN)/XRANGE+1.5
   70 GRID(IPTX,IPTY)=XCHAR
C
C     CCMPUTE PROPER SCALE NUMBERS
C
 8000 XINCR=XRANGE/5.
      YINCR=YRANGE/6.
      XSCALE(1)=XMAX
      YSCALE(1)=YMAX
      DO 80 I=2,5
   80 XSCALE(I)=XSCALE(I-1)-XINCR
      DO 81 I=2,7
   81 YSCALE(I)=YSCALE(I-1)-YINCR
C
C     OUTPUT SECTION WITH GRAPH
      WRITE (6,401)  IYORG
      WRITE (6,17) XSCALE(6),XSCALE(5),XSCALE(4),XSCALE(3),XSCALE(2),
     1     XSCALE(1)
   17 FORMAT(12X,1PE10.3,5(10X,E10.3)/15X,2H**,10(10H+*********),3H+**)
      II=1
      I=0
      DO 101 IK=1,61
      IF(I)91,91,92
   91 WRITE (6,18) YSCALE(II),(GRID(IK,IX),IX=1,101),YSCALE(II)
   18 FORMAT(3X,1PE10.3,4H  + ,101A1,4H +  ,E10.3)
      II=II+1
      GO TO 102
92    IF (IK .NE. IXAXIS)  GO TO 192
      WRITE (6,400)  (GRID(IK,IX),IX=1,101)
400   FORMAT (8X,4H0.00,3X,1H*,1X,101A1,2H *,3X,4H0.00)
      GO TO 102
  192 WRITE (6,19) (GRID(IK,IX),IX=1,101)
   19 FORMAT(15X,1H*,1X,101A1,1X,1H*)
  102 I=I+1
      IF(I-10)101,103,103
  103 I=0
  101 CONTINUE
      WRITE (6,22) XSCALE(6),XSCALE(5),XSCALE(4),XSCALE(3),XSCALE(2),
     1     XSCALE(1)
   22 FORMAT (15X,2H**, 10(10H+*********),3H+**/, 1P
     1  12X,E10.3, 5(10X,E10.3))
      WRITE (6,401)  IYORG
401   FORMAT(17X,101A1)
      IF(IERR) 1000,1000,1001
 1001 WRITE (6,20) IERR
   20 FORMAT(10X 'NUMBER OF POINTS OUT OF RANGE ='  I4)
 1000 RETURN
      END
```

LIST OF REFERENCES

1. Temes, G. C. and Calahan, D. A., "Computer-Aided Network Optimization the State of the Art," Proc. IEEE, v. 55, p. 1832-1863, November, 1967.

2. Wilde, D. J., Optimum Seeking Methods, Englewood Cliffs, New Jersey: Prentice-Hall, 1964.

3. Kirk, D. E., Optimal Control Theory: An Introduction, Prentice-Hall, to be published 1970.

4. Calahan, D. A., "Linear Network Analysis and Realization Digital Computer Programs: An Instruction Manual," University of Illinois Bulletin, v. 62, No. 58.

5. Tuttle, Jr., D. F., Network Synthesis, New York: John Wiley and Sons, Inc., 1958.

6. Weinberg, L., Network Analysis and Synthesis, New York: McGraw-Hill, 1962.

7. Hooke, R. and Jeeves, T. A., " 'Direct Search' Solution of Numerical and Statistical Problems," Journal of Association for Computing Machinery, v. 8, p. 212-229, 1961.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Computer-Aided Network Design by Optimization in the Frequency Domain

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Master's Thesis; December 1969

5. AUTHOR(S) (First name, middle initial, last name)

James Lau

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
December 1969	87	7

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

The filter design problem is considered as an optimization problem. An iterative search technique is employed to adjust the variable network element values to approximate some desired network response, with a minimum of error. Explicit constraints are employed to ensure physical realizability. The design process uses a combination of a modified version of Calahan's network analysis program with a direct search method of minimization developed by Hooke and Jeeves. The result is a procedure which utilizes the circuit designer's experience and knowledge to set up the problem but relieves him of the tedious labor now performed by the high-speed digital computer.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

A-31408

Security Classification

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computer-aided network design						
Network synthesis						
Optimization						
Pattern search						

DD FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821

Security Classification

A-31409