COMPUTER-AIDED CONTROL DESIGN SYSTEMS

A THESIS

PRESENTED IN PARTIAL FULFILMENT

OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF TECHNOLOGY IN INDUSTRIAL

MANAGEMENT AND ENGINEERING AT MASSEY

UNIVERSITY, PALMERSTON NORTH,

NEW ZEALAND.

by

NGUYEN, THI MINH PHUONG

DECEMBER, 1976

SYNOPSIS

This thesis is concerned with the application of interactive computer-aided design techniques. These techniques are graphical and based upon conventional control theory. They are the time-response, frequency-response and root-locus techniques.

The design philosophy based on the above design techniques is that an appropriate compensator is inserted and the overall system performance is studied. The graph representing the system performance after each adjustment of the compensator is adjusted until the performance specifications are met.

The Computer techniques are developed which, having specified the fixed configurations of the original system, permit varied choice of the compensator. For each selection, the control system may be investigated in either open- or closed-loop form. The software for computer-aided control system design is programmed in Algol and operates on the Massey University B6700 computer system.

The program is operated via a Tektronix 4010 visual display terminal in an online interactive mode. The control designer, at the Tektronix console, is able to specify certain designs and evaluate their effectiveness in terms of various graphical analyses produced by the computer and displayed on the screen of the Tektronix.

-ooOoo-

CONTENTS

-ooOoo-

CHAPTER ONE

INTRODUCTION

This thesis is concerned with the application of computer-aided design to classical control theory. In classical control theory, much of the effort of control system design is concerned with the generation of charts for such aspects as time response, root locus and frequency response characteristics. These charts are time consuming to prepare by hand, even though short cut techniques are often available to obtain sketches which closely approximate the actual characteristics.

A digital computer is able to assist in these conventional design procedures by generating the appropriate charts for inspection by the designer, thereby taking much of the manual effort out of the design processes.

This thesis describes the development of an interactive design package. Software for computer-aided control system design has been programmed to operate, via a visual display terminal, in an online interactive mode. The control designer, using the terminal console, is able to specify certain designs and evaluate their effectiveness in terms of the various graphical analyses produced by the computer and displayed on the screen of the terminal.

The computer-aided approaches to control system design are applicable to linear models. Although this assumption of linearity is not exactly valid for most practical systems, it is a realistic approach for three reasons:

1. Many systems are essentially linear under normal operating conditions.
2. Slightly nonlinear systems usually may be approximated with sufficient accuracy by assuming linearity.

3. At present adequate mathematical methods for solving non-linear equations have not been developed.

In this thesis, the system characteristics are determined by time-response, frequency-response and root-locus methods. The time-response method is a direct way of considering the system performance in the time-domain. The output response is derived for given inputs. The direct solutions of the differential equation of the system following a step input are plotted on an "output versus time" graph. The system performance is specified in terms of the information obtainable from this graph.

For the frequency-response methods, the frequency of sinusoidal oscillation of the input signal is varied over a certain range and the resulting frequency-response is studied. The results are plotted on graphs such as the Bode diagrams, Nyquist diagrams and Nichols charts. The root-locus method is a graphical method used to evaluate the roots of the characteristics equation of the system transfer function and determine the influence of the system parameters on these roots. This method provides a very effective way to carry out many design problems with requirements specified in the time-domain.

The design philosophy based on the above graphical approaches is that the graph representing the system performance, obtained by any of the above methods, is shaped and reshaped until the performance specifications are met. To achieve this, appropriate compensation networks are usually required within the control system structure.

Computer techniques have been developed which, having specified the fixed configurations of the control system, permit varied choice of the compensator. For each selection, the control system may be investigated - either in open- or closed-loop form - using

any of the above methods. The software for computer-aided control system design is programmed in Algol and operates on the Massey University B6700 computer system. The interactive terminal is via a Tektronix 4010 visual display unit. The designer, at the Tektronix console, is able to adjust the designs in order to improve the system performance by shaping various graphs displayed on the screen of the Tektronix. Given a particular control system structure, which may be either open-loop or closed-loop, the designer may request the following options:

- A display of the system time response following a step input.
- A Bode diagram display of the system frequency response.
- A Nyquist diagram display of the system frequency response
- A Nichols chart display of the system frequency response.
- A Root-locus plot display of the roots of the system characteristic equation as the system gain varies.

The designer seeks to satisfy all performance specifications by means of educated trial-and-error repetition. Although the design is based on a trial-and-error procedure, the experience of the designer plays an important role in a successful design. An experienced designer will be able to achieve an acceptable system design using fewer trials.

Chapter Two in this thesis describes the analysis and design techniques that have been utilized, namely time-domain, frequency-domain and root-locus techniques. Chapter Three outlines the structure of the software for computer-aided control system design, detailing the methods employed to implement some of the concepts discussed in chapter two, and also specifies how the system is used. Chapter Four presents several illustrative design examples, along with the computer generated graphs.

-ooOoo-

CHAPTER TWO

SYSTEM ANALYSIS AND DESIGN

2.1 INTRODUCTION

This chapter describes some of the analysis and design techniques for linear control systems. These techniques are graphical and based upon conventional control theory.

A computer is able to assist in these design procedures by generating the appropriate charts for inspection by the designer, thereby taking much of the manual effort out of the design processes.

To design a control system to meet given performance specifications, it is necessary to determine the characteristics of the system that is to be controlled either in the time domain or frequency domain. Given such characteristics, several graphical methods are available for analysis and design, namely time, response, root-locus and frequency response methods.

The frequency response methods consist of the Bode, Nyquist and Nichols chart techniques. The design philosophy based on these graphical approaches is that the graph representing the system performance, obtained by any of the above methods, is shaped and reshaped until the performance specifications are met. To achieve this objective, appropriate compensation networks are usually required. The control system configuration that is considered is shown in Figure 2.1 (it is assumed that $G(s)$ and $H(s)$, are fixed configurations over which the designer has no control).

Computer techniques have been developed which, having specified $G(s)$ and $H(s)$, permit varied choice of $G_c(s)$. For each selection, the above system may be investigated - either in open-loop or closed-loop form - using any of the above methods.

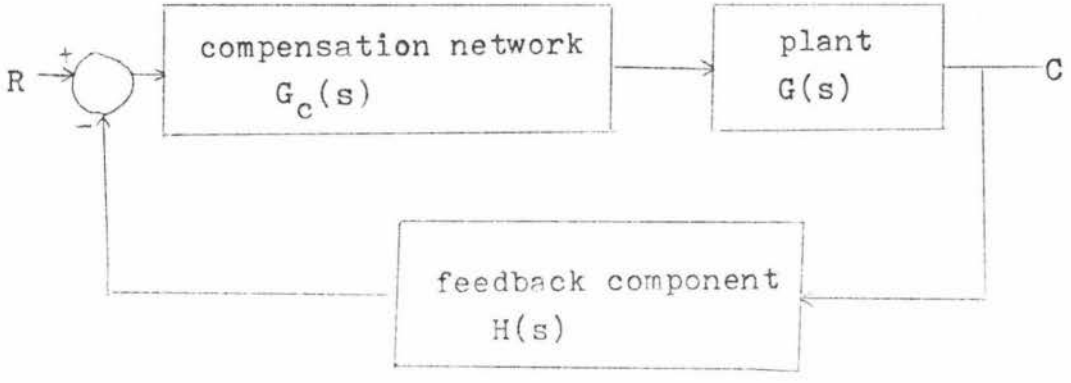


Fig. 2.1 Control system

In the following sections, different design techniques are described with most attention focussed on root-locus and frequency-response design techniques. The time-response technique, though it can be used for control system design with trial and error methods, is more useful for checking time-domain performance following the implementation of a design based upon frequency or root-locus techniques.

2.2 TIME-RESPONSE METHOD

2.2.1 *Time response analysis*

The most direct method of analysis is to consider the system performance in the time domain. The output response is derived for given inputs. The step function is the most useful form of input since control system performance is normally described in terms of step function response.

Direct solutions of the system differential equation following a step input, are plotted on an "output versus time" graph, with the condition that the system is at rest initially. This graph can be broken into two parts, steady-state terms which are directly related to the input, and the transient terms which are either exponential or oscillatory with an envelope of exponential form.

The required transient-response characteristics of a control system following a step input; are commonly specified in terms of time delay, rise time, overshoot and settling time. These time domain specifications are described in Appendix B.

A satisfactory system performance has:

- very small rise time, delay time and settling time
- an overshoot not exceeding about 30 percent
- a subsidence ratio of about 3 : 1. The subsidence ratio is the ratio of the amplitudes of successive cycles in a decaying oscillation and no offset (or steady-state error).

Since overshoot and rise time conflict with each other, high accuracy and good stability are incompatible, and a good design is a compromise between the two. It is therefore desirable that the transient response must be sufficiently fast and sufficiently damped.

2.2.2 *Time response design*

The computer method employing the time response design technique is a trial and arror approach. Given the transfer functions G(s) and H(s) of the fixed configurations of the system in Figure 2.1, a compensator $G_c(s)$ is inserted and the transient response of the overall control system to a unit-step input is then generated.

The resulting response provides information of the system performance, such as overshoot, rise time, delay time, etc.... This information permits the selection of $G_c(s)$ in a trial and error manner, i.e, the designer seeks to satisfy all the performance specifications by means of educated trial and error repetition.

Setting the gain is the first step in adjusting the system for satisfactory performance. In many practical cases, however, the adjustment of the gain alone may not provide sufficient alteration of the system behaviour. As is frequently the case, increasing the gain value will improve the steady-state behaviour but will result in poor stability or even instability. It is then necessary to redesign the system by incorporating additional compensators.

Since the effects of the basic control action, such as proportional, integral and derivative control are known[1], their combination such as proportional-plus-derivative (P.D), proportional-plus-integral (P.I) and proportional-plus-integral-plus-derivative (P.I.D) control actions may be employed to provide various modifications to the behaviour of the original system. The compensator transfer function $G_c(s)$ may take any of the following

forms:

$G_c(s) = Kp(p + \frac{1}{T_I s})$ (proportional-integral control)

$G_c(s) = Kp(1 + T_d s)$ (proportional-derivative control)

$G_c(s) = Kp(1 + \frac{1}{T_I s} + T_d s)$ (proportional-integral-derivative control)

where Kp represents the proportional gain

T_I represents the integral time

T_d represents the derivative time

Kp, T_I and T_d are adjustable.

The selection of $G_c(s)$ may be made as follows:

- If an offset is tolerable within limits, a proportional action may be used since it is simple and least expensive. (This is equivalent to the adjustment of the loop gain.)
- If no offset is tolerable, (P.I) action can be used, but at the expense of a more oscillatory behaviour.
- (P.D) control action increases the stability of the control system by introducing damping.
- (P.I.D.) control action stabilizes the control system and eliminates offset. It also permits increased sensitivity, and consequently increased speed of response.

Overshoot may be reduced by decreasing the value of Kp. This however results in a slower response. Overshoot can also be reduced by increasing T_I at a lesser expense to speed of response. However, overshoot is not lowered by increasing T_d. The speed of response is significantly increased by increasing the value of T_d.[10]

Hence, to decrease overshoot without seriously slowing the response, a combination of changes needs to be made. Similarly,

a better control system response with minimum oscillation, lower overshoot, low rise and response time and with no offset is devised, a series of possible combinations should be tried to reduce K_p slightly and to increase T_I and T_d moderately.

In addition to the above basic control actions in combination, other types of compensation, such as lead, lag and lead-lag compensation may be used to improve the control system behaviour. The particular situation determines the type of the compensation to be used.

- Lead compensation essentially yields an appreciable improvement in transient response and a small improvement in steady-state accuracy (with similar properties to derivative control). The transfer function of a lead compensator is

$$G_c(s) \quad + \frac{s + \frac{1}{T}}{s + \frac{1}{\alpha T}} \qquad \text{(Appendix C)}$$

where $0 < \alpha < 1$, $\alpha = .1$ is a sensible practical value.

Value of T may be adjusted arbitrarily until a satisfactory system performance is reached.

- Lag compensation, on the other hand, yields an appreciable improvement in steady-state accuracy at the expense of increasing the transient response time (with similar properties to integral control).

The transfer function of a lag compensator is

$$G_{c(s)} = \frac{1}{\beta} \frac{(s + \frac{1}{T})}{s + \frac{1}{\beta T}} \qquad \text{(See Appendix C)}$$

Lead-lag compensation combines the characteristics of both lead compensation and lag compensation, i.e. lead-lag compensation yields both fast response, and good static accuracy.

The transfer function of a lead-lag network is:

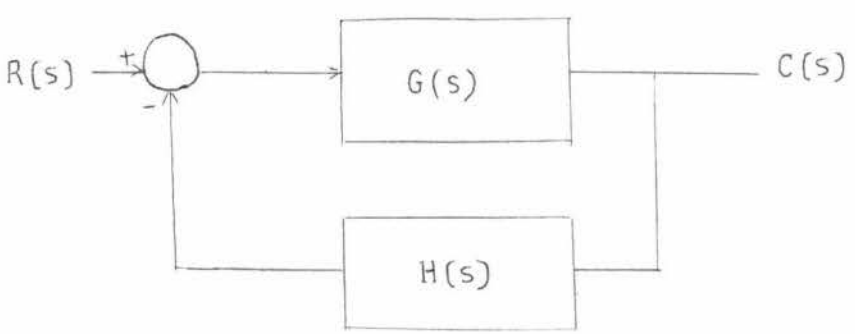
$$G_c(s) = \frac{1}{\beta}\left(\frac{s + \frac{1}{T_1}}{s + \frac{1}{\alpha T_1}}\right)\left(\frac{s + \frac{1}{T_2}}{s + \frac{1}{\beta T_2}}\right) \cdot K_c$$

If α is chosen to be $\frac{1}{\beta}$, then a lead-lag compensator is a combination of lead and lag compensators.

2.3 ROOT-LOCUS METHOD

2.3.1 *Root-locus analysis*

In the root-locus analysis problem, it is important to locate the closed-loop poles in the s-plane. In the design of closed-loop control system, it is desired to adjust the open-loop poles and zeros so as to place the closed-loop poles and zeros at particular locations in the s-plane. A simple method for finding the roots of the characteristic equation has been developed by W R. Evans and used extensively in control engineering[2]. This method, called the root-locus method, is one in which the roots of the characteristic equations are plotted as the open loop gain constant is varied from zero to infinity. Consider the feedback control system given below:



The closed-loop transfer function of this system is

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1 + G(s),\ H(s)}$$

Let the open loop T.F G(s), H(s) be represented by

$$G(s),\ H(s) = \frac{KA(s)}{B(s)} = \frac{K(a_o s^m + a_1 s^{m-1} + \ldots + a_{m-1} s + a_o)}{(b_o s^n + b_1 s^{n-1} + \ldots + b_n s + b_o)}$$

where A(s) and B(s) are the finite polynominals of the complex variable s, $m \leq n$ and K is the open-loop gain factor - the closed-loop transfer function then becomes

(overleaf)

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1 + \frac{KA(s)}{B(s)}}$$

$$+ \frac{G(s), B(s)}{B(s) + KA(s)}$$

The closed-loop poles are the roots of the characteristic equation

$$B(s) + KA(s) = 0 \qquad (2.1)$$

In general the location of these roots in the s-plane changes as the open-loop gain factor K is varied. A locus of these roots plotted in the s-plane as a function of K is called the root-locus.

For K equal to zero, the roots of (2.1) are the roots of the polynominal B(s), which are the same as the poles of the open-loop T.F G(s), H(s).

If K becomes very large, the roots approach those of the polynominal A(s), which are the open-loop zeros. Thus as K is increased from zero to infinity, the loci of the closed-loop poles originate from the open-loop poles and proceed toward and terminate at the open-loop zeros.

For this project, the roots of the characteristic equation are computed and plotted by a digital computer, the rules for constructing root loci from the open-loop roots are therefore not considered.

Stability

The system is unstable for all values of K for which any of the loci pass into the right-hand half of the s-plane, i.e. poles with positive real parts. The system is simple harmonic, in which case the output is a stable amplitude sinusoidal oscillation, for any value of K for which the closed-loop poles are on the imaginary

axis (i.e. poles with zero real parts).

Angle and magnitude condition

In order for a branch of a root-locus to pass through a particular point S_1 in the s-plane, it is necessary that s_1 be a root of the characteristic equation (2.1) for some real value of K.

That is $B(s) + KA(s) = 0$

or $1 + G(s),H(s) = 0$

$$GH(s_1) = \frac{KA(s)}{B(s)} = -1 \qquad (2.2)$$

therefore the complex number GH(s) must have a phase angle of

$$\angle GH(S_1) = \pm 180(2K + 1) \quad (K = 0,1,2.....)$$

This is the angle condition.

In order for S_1 to be the root-locus, it is necessary that Fig (2.2) be satisfied with regard to magnitude in addition to phase angle.

This is the magnitude condition $GH_{(s_1)} = 1$

Dominant closed-loop poles

Those closed-loop poles which have dominant effects upon the transient-response behaviour are called dominant poles. They are the poles nearest to the jω axis.

Damping ratio from the root-locus

The gain factor K required to give a specified damping ζ (or vice-versa) is easily determined from the root-locus. A line is drawn from the origin at an angle of plus or minus Θ with the negative axis, where

$$\Theta = \cos^{-1}\zeta$$

The gain factor at the point of intersection with the root-locus is the required value of K.

2.3.2. *Root-locus design*

Design problem

The root-locus method can be used very effectively in the design of control systems because it graphically illustrates the variation of the closed-loop poles as a function of the gain factor K. In its simplest form, the design is accomplished by choosing a value of K which results in a satisfactory closed-loop behaviour. If it is not possible to meet system specifications in this way, another form of compensation can be added to the system to alter the root-locus in the required manner.

The closed-loop system transient and frequency responses are determined by the location of the closed-loop poles. If the system has only two closed-loop poles and no finite zeros, performance parameters such as percentage overshoot, rise time, and 3db bandwidth are directly related to the damping ratio ζ and undamped natural frequency ω_n of the poles. For higher order systems the relationship of these parameters to the closed-loop poles and zeros becomes more complicated and it is necessary to resort to standard curves or tables for this information. However, in many instances a higher order system may be approximated by a second or a third order system by using a dominant pole-zero approximation[5]. Specifications on allowable steady-state errors usually take the form of a minimum open-loop gain factor, expressed in terms of the error constants. (Appendix A)

Cancellation compensation

If the pole-zero configuration of the plant is such that the system specifications cannot be met by an adjustment of the open-loop gain factor, a cascade compensator, as shown in Fig.2.2 can be added to the system to cancel some or all the poles and zeros of the plant. When poles of the plant are cancelled by zeros of the compensator, the compensator also adds new poles to

the forward loop transfer function.

The philosophy of the compensation technique is then to replace undesirable poles. Once the effects on the root -locus of the addition of poles and/or zeros are fully understood, the locations of the pole(s) and zero(s) of the compensator can be readily determined in order to reshape the root-locus as desired.

Effects of the addition of poles

The addition of a pole to the open-loop transfer function has the effect of pulling the root-locus to the right, tending to lower the system's relative stability and slow down the settling of the response. Figure 2.3 shows examples of root-loci illustrating effects of the addition of a pole to a single pole system and the addition of two poles a single pole system.

Effects of the addition of zeros

The addition of a zero to the open-loop T.F has the effect of pulling the root-locus to the left, tending to make the system more stable and to speed up the settling of the response. Figure 2.4 shows the root-locus plots for the system when a zero is added to the open-loop transfer function.

The procedures for designing a lag-lead compensator may be stated as follows:

(a) From the given performance specifications, determine the desired location for the dominant closed-loop poles.

(b) In order to have the dominant closed-loop poles at the desired locations, calculate the angle contribution ϕ needed from the phase lead position of the lead-lag network.

(c) Using the transfer function of the lead-lag compensator:

$$G_c(s) = \frac{s + \frac{1}{T_1}}{s + \frac{\beta}{T_1}} \cdot \frac{s + \frac{1}{T_2}}{s + \frac{1}{\beta T_2}} \cdot K_c$$

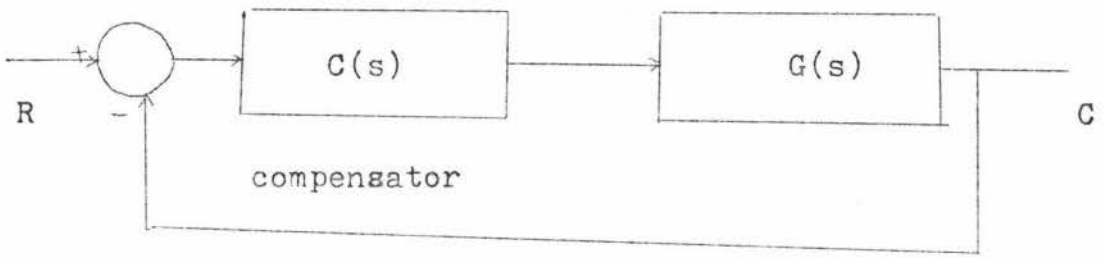


Fig. 2.2 Compensated system

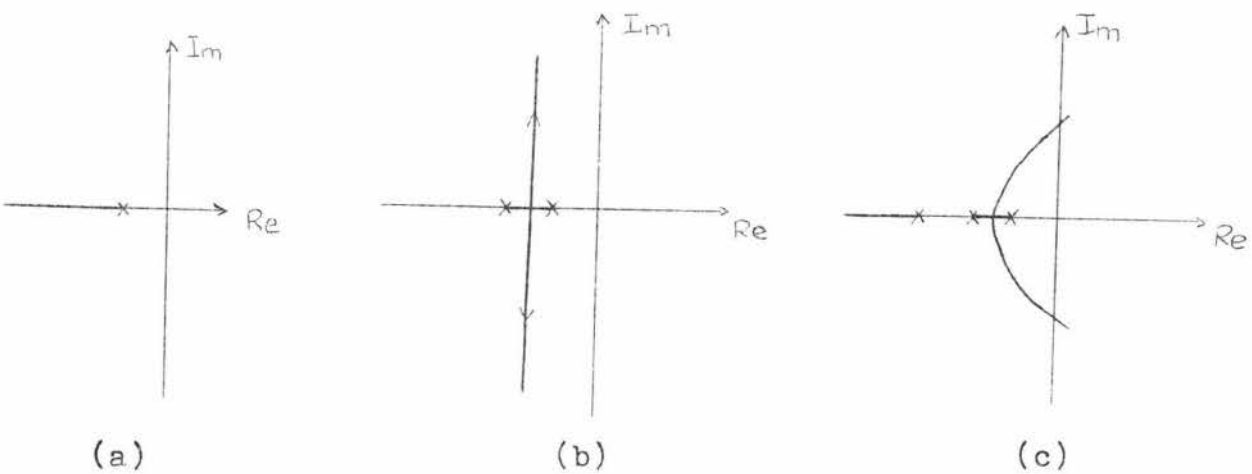


Fig. 2.3

(a) Root-locus plot of a single-pole system
(b) Root-locus plot of a two-pole system
(c) Root-locus plot of a three-pole system

Fig. 2.4
(a) Root-locus of a three-pole system
(b) Root-locus showing the effect of addition of a zero to the system

determine the constant K_c from the requirement on the particular error coefficient specified in the design problem.

(d) For the lead-lag compensator, we choose T_2 sufficiently large so that

$$\frac{s_1 + \frac{1}{T_2}}{s_1 + \frac{1}{\beta T_2}}$$

is approximately unity, where $s = s_1$ in one of the dominant closed-loop poles.

Determine the values of T_1 and β from the requirements that

$$\left| \frac{s_1 + \frac{1}{T_1}}{s_1 + \frac{1}{\beta T_2}} \right| 1k_c G(s_1) = 1$$

and

$$\angle \frac{s_1 + \frac{1}{T_1}}{s_1 + \beta/T_1} = \emptyset$$

(e) Using value of β just determined, choose T_2 so that

$$\left| \frac{s_1 + \frac{1}{T_2}}{s_1 + \frac{1}{\beta T_2}} \right| \# 1$$

and $$0 < \angle \frac{s_1 + \frac{1}{T_2}}{s_1 + \frac{1}{\beta T_2}} < 3^o$$

The value of βT_2, the largest time constant of the lag-lead network, should not be too large to be physically realized.

By choosing T_2 sufficiently large, Fig.(2.3) becomes

$$G_c(s) = \frac{s + \frac{1}{T_1}}{s + \beta/T_1} K_c \qquad (2.4)$$

which is the transfer function of a lead-compensation network.

Steps a,b,c, (with $G_c(s)$ from Fig.(2.4) and d may be followed to design a lead compensator.

Similarly, by choosing T_1 sufficiently large, Fig.(2.3) becomes

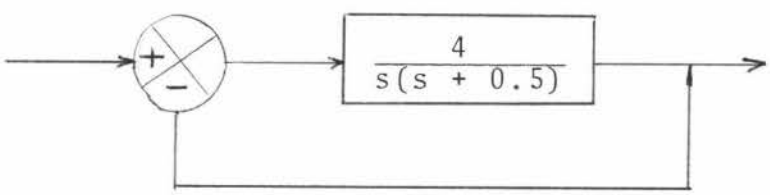
$$G_c(s) = \frac{s + 1/T_2}{s + 1/\beta T_2} K_c \qquad (2.5)$$

which is the transfer function of a lag-compensation network.

To design a lag compensation network, choose β between 1 and 15 ($\beta = 10$ is a good choice) and follow steps a, c (with $G_c(s)$ from Fig.(2,5) and e.

Example: The following example illustrates the design of a lead-lag compensator, following the stated procedures.

Consider the control system shown below



The feed forward transfer function is

$$G(s) = \frac{4}{s(s + 0.5)}$$

It is desired to make the damping ratio equal to 0,5 and to increase the undamped natural frequency to 5 rad/sec and the static velocity error coefficient to 50sec^{-1}. A lead-lag compensator is designed to meet all the performance specifications.

From the performance specifications, the dominant closed-loop poles must be

$$s = -2.5 \pm j\,4.33$$

since

$$\left.\frac{4}{s(s + 0.5)}\right|_{s = -2.5 + j\,4.33} = -235^{o}$$

the phase lead portion of the lead-lag network must contribute 55^{o} so that the root locus passes through the desired location

of the dominant closed-loop poles.

The compensated system will have the transfer function

$$G_c(s)\,G(s) = \frac{s + {}^{1}/_{T_1}}{s + {}^{}/_{T_1}} \quad \frac{s + {}^{1}/_{T_2}}{s + {}^{1}/_{\beta T_2}} \quad K_c\,G(s)$$

The static velocity error coefficient therefore becomes

$$K_V = \lim_{s\to 0} sG_c(s)G(s) = \lim_{s\to 0} sK_cG(s)$$

The requirement on the static velocity error coefficient is

$K_V = 50\text{sec}^{-1}$

Thus

$$K_V = \lim_{s\to 0} \frac{s4K_c}{s(s + 0.5)}$$

$$= 8K_c = 50$$

hence $K_c = 6.25$ and compensated open-loop transfer function becomes

$$G_c(s).G(s) = \left(\frac{s + {}^{1}/_{T_1}}{s + {}^{\beta}/_{T_1}}\right) \left(\frac{s + {}^{1}/_{T_2}}{s + {}^{1}\beta T_2}\right) \left(\frac{25}{s(s + 0.5)}\right)$$

since T_2 is chosen large enough that

$$\frac{s + \frac{1}{T_2}}{s + {}^{1}/_{\beta T_2}} \quad \# 1$$

if it is required that the closed-loop poles lie at $s = -2.5 \pm j4.33$

the magnitude condition becomes

$$\left|G_c(s).G(s)\right|_{s = -2.5 + j4.33} = \left|\frac{s + {}^{1}/_{T_1}}{s + \beta/T_1}\right| \left|\frac{25}{s(s + 0.5)}\right|_{s = -2.5 + j4.3}$$

$$= \left|\frac{s + \frac{1}{T_1}}{s + \beta/T_1}\right| \quad \frac{5}{4.77} = 1$$

and the angle condition becomes

$$\angle \left.\frac{s + {}^{1}/_{T_1}}{s + \beta/T_1}\right|_{s = -2.5 + j4.33} = 55^{o}$$

It is a simple matter to graphically determine the values of T_1 and β that satisfy these magnitude and angle conditions. Referring to Fig.(2.5), we can easily locate points A and B so that

$$\angle APB = 55^{o} \ , \ \frac{\overline{PA}}{\overline{PB}} = \frac{4.77}{5}$$

Graphically, from Fig. (2.5)

$$\overline{AO} = 0.5 \ , \ \overline{BO} = 5$$

Hence

$$-\ {}^{1}/T_1 = -0.5 \ , \ -\ {}^{\beta}/T_1 = -5$$

Thus $T_1 = 2$, $\beta = 10$

Therefore, the phase lead position of the lag-lead network becomes

$$\frac{s + 0.5}{s + 5}$$

For the phase lag position of the lag-lead network, it is required that

$$\left| \frac{s + {}^{1}/T_2}{s + {}^{1}/_{10}T_2} \right|_{s = -2.5 + j4.33} \qquad \# 1$$

$$0 < \left| \frac{s + {}^{1}/T_2}{s + {}^{1}/_{10}T_2} \right|_{s = -2.5 + j4.33} < 3^{o}$$

Choose $T_2 = 10$

Then the transfer function of the lead-lag compensator is

$$G_c(s) = \left(\frac{s + 0.5}{s + 5}\right)\left(\frac{s + 0.1}{s + 0.01}\right) \quad 6.25$$

And the compensated system will have the open-loop transfer function

$$G_c(s) . G(s) = \frac{25\ (s + 0.1)}{s(s + 5)\ (s + 0.01)}$$

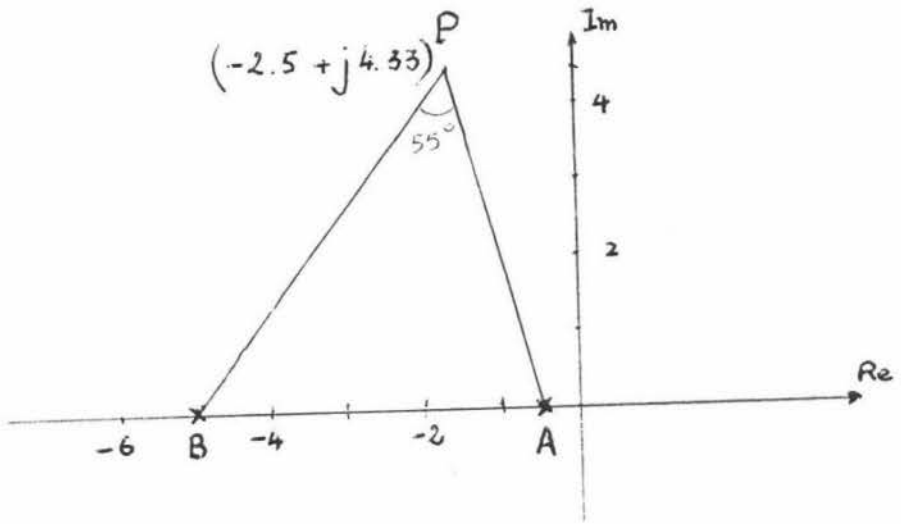


Fig. 2.5

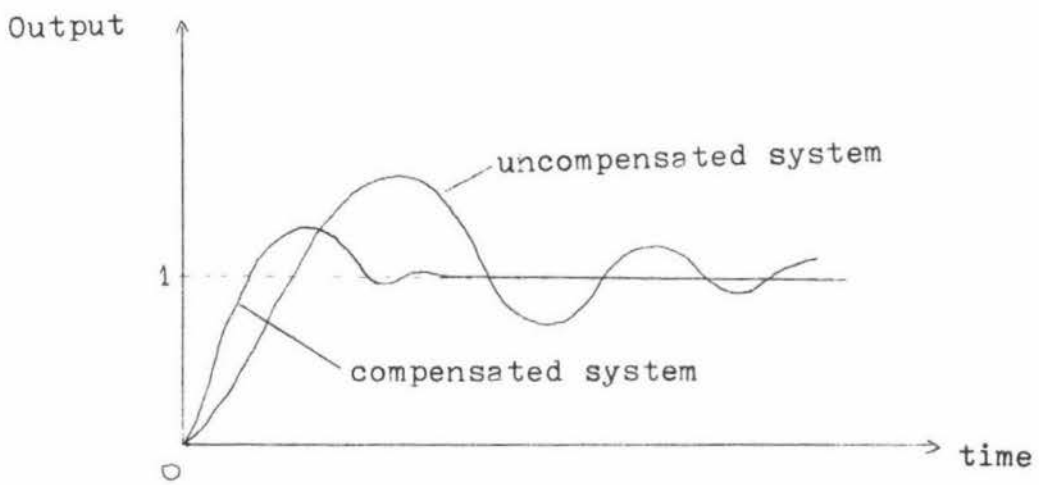


Fig. 2.6 Unit step response of the uncompensate
-d and compensated system

Because the angle contribution of the phase lag portion of the lead-lag network is quite small, there is no appreciable change in the location of the dominant closed-loop poles from the desired location, $s = 2.5j \pm j4.33$. Therefore, the compensated system meets all the required performance specifications.

Thus, given the unalterable plant transfer function G(s), the following procedure is applied to choose the necessary compensating network C(s):

1. Plot the root-locus of the given plant transfer function G(s).
2. Examine the loci to see if it is necessary to "pull" it to the left, making the system more stable and speeding up the settling of the response. To achieve this, the dominant pole(s) which results in oscillation, is approximately cancelled by the compensating network zero(s), replacing it with insignificant pole. The effect is to distinctly alter the root locus, moving the oscillatory poles to the left.
3. If poles with small damping ratios are present in the plant transfer function, they may be cancelled and replaced with poles which have larger damping ratios.

   Although this conpensation technique is simple, the difficulty encountered in applying it is that it is not always apparent what open-loop pole-zero configuration is desirable from the standpoint of meeting specifications on the closed-loop system performance.

Lead-lag compensation

Lead compensation essentially yields an appreciable improvement in transient-response and a small improvement in steady-state accuracy.

Lead compensation on the other hand, yields an appreciable improvement in steady-state accuracy at the expense of increasing

the time response. Lead-lag compensation combines the characteristics of both lead and lag compensation.

The transfer function of the lead-lag network is

$$G_c(s) = \left(\frac{s + 1/T_1}{s + \beta/T_1}\right) \cdot \left(\frac{s + 1/T_2}{s + 1/\beta T_2}\right) \quad K_c \qquad (2.3)$$

2.4 FREQUENCY-RESPONSE METHOD

2.4.1 *Introduction*

The term "frequency-response", means the steady-state response of a system to a sinusoidal input. For frequency-response methods, the most conventional methods available to control engineers for analysis and design of control systems, the frequency of the input signal over a certain range is varied and the resulting frequency-response is studied. The results are plotted on graphs such as Nyquist diagrams, Bode plots and Nichols chart plots. A digital computer is of tremendous assistance in accomplishing the tedious task of plotting point by point, especially when the system transfer function is of high order.

Design in the frequency-domain is simple and straightforward. The frequency-response plot indicates clearly the manner in which the system should be modified in order to obtain the desired transient-response characteristics.

Bode diagrams are convenient to use if we desire a certain phase or gain margin. On the other hand, if a certain value of the resonant peak value M_r is required, the Nyquist or Nichols chart plot is more convenient to use. Usually Nyquist design techniques provide rapid evaluation of stability of the system and they are thus used to supplement other methods.

2.4.2 *Bode design*

Bode design techniques employ graphical representation of the open-loop transfer function $GH(j\omega)$, both plotted as a

function of frequency ω. Logarithmic scales are usually used for the frequency axes because the magnitude and phase angle may be graphed over a greater range of frequencies than with a linear frequency axis. The magnitude $G(j\omega)$ for any value of ω is plotted on a linear scale in decibel (db) unit, where

$$db = 20 \log_{20} G(j\omega)$$

The phase angle, in degrees, is plotted on a linear scale.

Fig. 2.7 shows the Bode representation of a second-order transfer function. From Bode diagrams, information such as gain and phase margins, bandwidth, cut-off frequency, resonant peak and resonant frequency may be obtained. There frequency-domain specifications are described in Appendix B.

Stability of the system

In most cases positive gain and phase margins obtained from the o/p-loop T-F will ensure stability of the closed-loop system. If the open-loop gain is increased by the gain margin the overall gain at 180^{o} phase shift will be unity (odb), and the system is on the verge of instability.

Negative gain and phase margins indicate an unstable system.

For satisfactory performance, the phase margin should be between 30^{o} and 60^{o}, and the gain margin should be greater than 6db.

Bode design.

Design of a control system using Bode techniques entails shaping and reshaping the Bode magnitude and phase angle plots until the system specifications are satisfied. Setting the gain is the first step in shaping the Bode plot for better performance. In many practical cases, however, the adjustment of the gain alone may not provide sufficient alteration of the system behaviour. It is then necessary to shape the Bode plots by adding cascade or feedback compensation in order to alter the

overall behaviour so that the system will behave as desired.

Lead compensation

The lead compensator, presented in Appendix B, has the following Bode form of frequency response function.

$$G_c(j\omega) = \frac{j\omega + 1/T}{j\omega + 1/\alpha T} \qquad (\alpha < 1)$$

The primary function of the lead compensator is to reshape the frequency-response curve to provide sufficient lead angle to offset the excessive phase lag associated with the components of the fixed system.

Assume that a unity feedback system is given. It is desired to satisfy the performance requirements, which are given in terms of phase margin, gain margin, error coefficients, etc. The procedures for designing a lead compensator may be stated as follows:

(a) Determine the open-loop gain to satisfy the requirement on the error coefficients.

(b) Using the gain K thus determined, plot the Bode diagram of the open-loop system and evaluate the phase margin of the uncompensated system.

(c) Determine the necessary phase lead angle ϕ to be added to the system.

(d) Determine the attenuation factor α by use of Eq. $\sin\phi_m = \frac{1-\alpha}{1+\alpha}$ (Appendix C). Determine the frequency at which the magnitude of the uncompensated system is equal to $-20\log_{10}(\frac{1}{\alpha})$ (Appendix C). Select this frequency as the new gain crossover frequency. This frequency corresponds to ω_m and the maximum phase shift ϕ occurs at this frequency.

(e) Determine the corner frequencies of the lead network from

$$\omega = {}^{1}/T \; ; \; \omega = {}^{1}/\alpha T$$

Finally, insert an amplifier with gain equal to $\frac{1}{\alpha}$, or increase the gain of the existing amplifier by a factor of $\frac{1}{\alpha}$.

A design example using lead compensation techniques based on the frequency-response approach is presented in Chapter 4.

Lag compensation

The lag compensation network has the following transfer function

$$G_c(j\omega) = \frac{1}{\beta} \quad \frac{s + {}^{1}/T}{s + {}^{1}/\beta T} \qquad (\beta > 1)$$

The primary function of a lag network is to provide attenuation in the high-frequency range in order to give a system sufficient phase margin. The procedures for designing a lag compensator by the frequency-response approach may be stated as follows:

(It is assumed that the system has unity feedback.)

(a) Determine the open-loop gain such as the requirement on the particular error coefficient is satisfied.

(b) Using the gain thus determined, draw Bode diagram of the uncompensated system and determine the phase and gain margins of the uncompensated system.

(c) If the specifications on the phase and gain margins are not satisfied, then find the frequency point where the phase angle of the open-loop transfer function is equal to 180° plus the required phase margin. The required phase margin is the specified phase margin plus 5° to 12°. (The addition of 5° to 12° compensates for the phase lag of the lag network.) Choose this frequency as the new gain crossover frequency.

(d) Choose the corner frequency $\omega = \frac{1}{T}$ corresponding to the zero of the lag network) one decade below the new gain crossover frequency.

(e) Determine the attenuation necessary to bring the magnitude curve down to odb at the new gain crossover frequency. Noting that this attenuation is $-20 \log_{10}\beta$, determine the value of β. Then the other corner frequency (corresponding to the pole of the lag network) is determined from $\omega = \frac{1}{\beta T}$.

Lead-lag compensation

If the value of α for the lead network must be equal to the reciprocal of the value of β for the lag network, a lead-lag compensator may be designed by combining the individually designed lead and lag compensators. Apart from lead, lag and lead-lag compensators, other conventional control action such as P.I, P.D or P.I.D control action may be employed to compensate for the deficiencies of the system performance.

2.4.4. *Nyquist design*

Nyquist analysis, a frequency response method, is essentially a graphical procedure for determining absolute and relative stability of closed-loop control systems. Information about stability is available directly from a graph of the sinusoidal open-loop transfer function $GH(j\omega)$.

The open-loop transfer function GH(s) may be written in an equivalent form

$$GH(j\omega) = X(\omega) + jY(\omega)$$

A Nyquist plot of $GH(j\omega)$ is a graph of $Y(\omega)$ versus $X(\omega)$ in the finite position of the $G(j\omega)$-plane for $-\alpha \quad \omega \quad +\alpha$

Fig. 2.9 shows the typical Nyquist diagram of

$$GH(j\omega) = \frac{\omega_1^2}{(j\omega)^2 + 2j\omega\ (j\omega) + \omega_n^2}$$

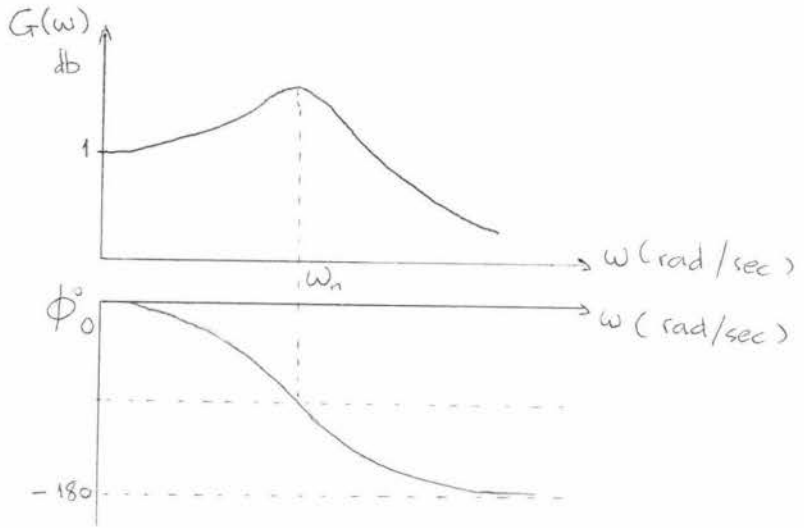


Fig. 2.7 Bode diagram of $\dfrac{\omega_n^2}{(j\omega)^2 + 2\zeta\omega_n(j\omega) + \omega_n^2}$

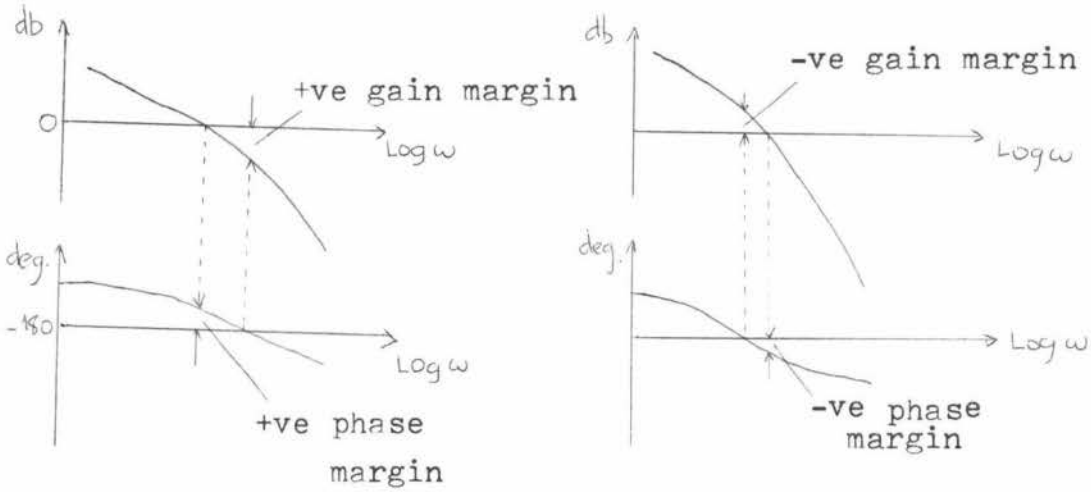


Fig. 2.8 phase and gain margins of stable and unstable system.

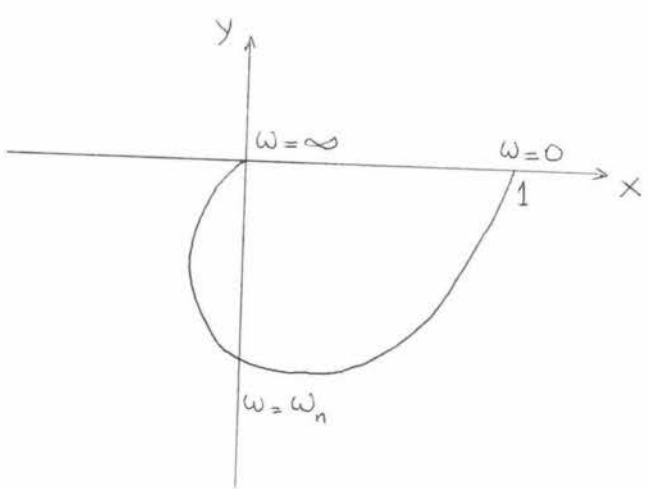


Fig. 2.9 Nyquist diagram of $\dfrac{\omega_n^2}{(j\omega)^2 + 2\zeta\omega_n(j\omega) + \omega_n^2}$

Nyquist stability criterion

The simplified Nyquist stability criterion states that for a closed-loop system to be stable the locus of the open-loop frequency response, plotted on a Nyquist diagram must not enclose the point (-1,jo) as ω varies from zero to infinity. Enclosing the point (-1,o) may be interpreted as passing to the left of the point.

Fig.2.10 shows three systems with similar time constant but different gain constants. When the loop is closed, system (a) is stable, (b) is just stable and (c) is unstable. The complete Nyquist stability criterion states that for a closed-loop system to be stable it is necessary and sufficient that the contour of the open-loop frequency-response $GH(j\omega)$, describes a number of counterclockwise encirclements of the point (-1,jo) as ω varies from $-\alpha$ to $+\infty$, not less than the number of poles of $GH(j\omega)$ with positive real parts.

Let P_o = number of poles with =ve real part of $GH(j\omega)$

N = total number of counterclockwise encirclements of the point (-1,jo)

$N > 0$ for counterclockwise encirclements

$N < 0$ for clockwise encirclements

For the closed-loop system to be stable $N \geqslant P_o$

A system with any clockwise net rotation is always unstable.

Nyquist design

Since it is not easy to generalize methods for the design of lead, lag or lead-lag compensator using Nyquist design technique, this method is normally used to provide rapid evaluation of system stability and to support other methods.

In the following, a method for determining the gain K so that the system will have a specific resonant peak M_p is demonstrated.

Referring to Fig.2.11, the tangent line drawn from the origin to the desired M_p circle has an angle of ψ.

The value of sin ψ is

$$\sin\psi = \left| \frac{\frac{M_p}{M_p^2 - 1}}{\frac{M_p^2}{M_p^2 - 1}} \right| = \frac{1}{M_p}$$

The line drawn from point P, perpendicular to the negative real axis, intersects this axis at the -1 + jo point. Consider the control system shown in Fig. 2.12. The procedure for determining the gain K so that $G(j\omega) = KG_1(j\omega)$ will have a desired value of M_p (where $M_p > 1$) can be summarized as follows:

1. Draw the Nyquist plot of the normalized open-loop transfer function $G_1(j\omega) = \frac{G(j\omega)}{K}$
2. Draw from the origin the line which makes an angle of $\psi = \sin^{-1}(\frac{1}{M_p})$ with the negative real axis.
3. Fit a circle with centre on the negative real axis tangential to both the $G_1(j\omega)$ locus and the line PO.
4. Draw a perpendicular line to the negative real axis from point P, the point of tangency of this circle with the line PO. The perpendicular line PA intersects the negative real axis at point A.
5. In order that the circle just drawn corresponds to the desired M_p circle, point A should be the -1+jo point.

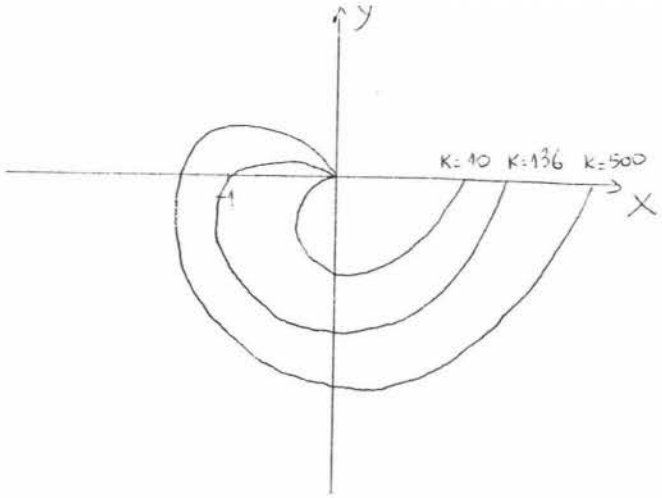


Fig. 2.10 Nyquist diagram of $\dfrac{K}{(1+.2j\omega)(1+j\omega)(1+10j\omega)}$

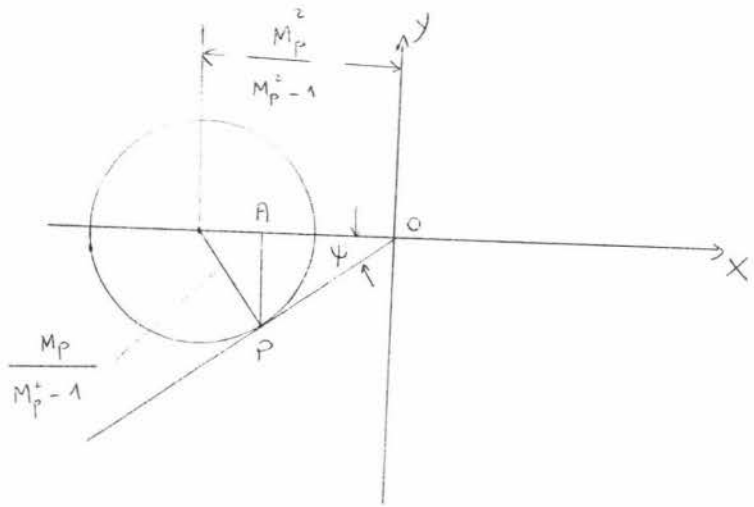


Fig. 2.11 M circle

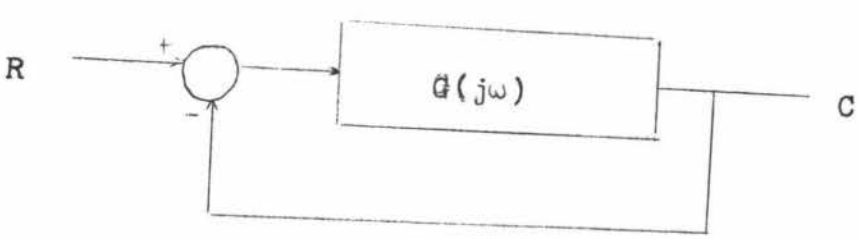


Fig. 2.12 Control system

6. The desired value of the gain K is that value which changes the scale so that point A becomes the -1 + jo point.

   Thus,

$$K_{M_p} = \frac{1}{\overline{OA}}$$

   Note that the resonant frequency ω_p is the frequency of the point at which the circle is tangent to the G (jω) locus. The present procedure may not yield a satisfactory value for ω_p. If this is the case, the system must be compensated in order to increase the value of ω_p without changing the value of M_p.

2.4.5 *Nichols chart design*

The closed-loop transfer function of the control system shown in Figure 2.13 is

$$\frac{C(j\omega)}{R(j\omega)} = \frac{G(j\omega)}{1 + G(j\omega).H(j\omega)}$$

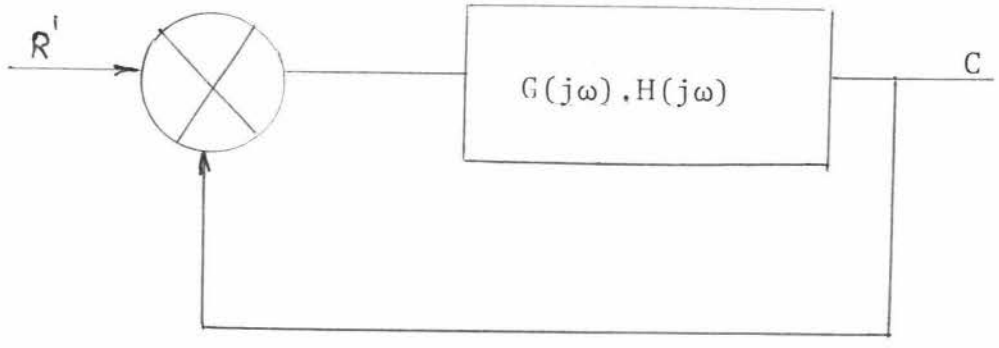
$$= \frac{G(j\omega).H(j\omega)}{1 + G(j\omega).H(j\omega)} \cdot \frac{1}{H(j\omega)}$$

$$= \frac{1}{H(j\omega)} \cdot \frac{C(j\omega)}{R'(j\omega)}$$

with

$$\frac{C(j\omega)}{R'(j\omega)} = \frac{G(j\omega).H(j\omega)}{1 + G(j\omega).H(j\omega)}$$

The term $\frac{C(j\omega)}{R'(j\omega)}$ is the transfer function of a direct feedback system with a forward transfer function of $G(j\omega).H(j\omega)$ as shown below



Put $GH(j\omega) = G_2(j\omega) = G(\cos \phi + j \sin \phi)$

Where G and ϕ are function of ω

Hence the closed-loop transfer function of the direct feedback system is

$$\frac{C(j\omega)}{R'(j\omega)} = \frac{G_2(j\omega)}{1 + G_2(j\omega)}$$

$$= \frac{G(\cos \phi + j \sin \phi)}{1 + G(\cos \phi + j \sin \phi)}$$

Then the closed-loop amplitude of the direct feedback system is

$$M = \frac{G}{\sqrt{(1 + G \cos \phi)^2 + G^2 \sin^2 \phi}}$$

$$= \frac{G}{\sqrt{(1 + 2G \cos \phi + G^2}}$$

$$\therefore \quad M^2 (1 + 2G \cos \phi + G^2) = G^2$$

$$\therefore \quad \cos \phi = \frac{-(M^2 + G^2(M^2-1)}{2M^2G} \qquad 2.6$$

Similarly, the closed-loop phase angle of the direct feedback is

$$\delta = \angle G_c(j\omega) - 1 \angle G_c(j\omega)$$

$$\delta = \phi - \text{TAN}^{-1} \frac{G \sin \phi}{1 + G \cos \phi} \qquad 2.7$$

Equation 2.6 relates M to $\phi(\omega)$ and $G(\omega)$ so that contours for constant M can be plotted as $20 \log_{10} G(\omega)$ against $\phi(\omega)$.

Similarly Eq. 2.7 relates δ with $20 \log_{10} G(\omega)$ and $\phi(\omega)$ so that the constant δ contours can also be drawn.

These constant gain and phase contours are plotted on the db magnitude-degree phase diagram called the Nichols chart.

The magnitude-phase angle of the open-loop transfer function $G(j\omega)\ H(j\omega)$ plotted on a Nichols chart is called a Nichols chart plot of $GH(j\omega)$.

The Nichols chart applies only to direct feedback loops. The chart can be used to find $\frac{C(j\omega)}{R'(j\omega)}$ and multiplying by $\frac{1}{H(j\omega)}$, as illustrated below:

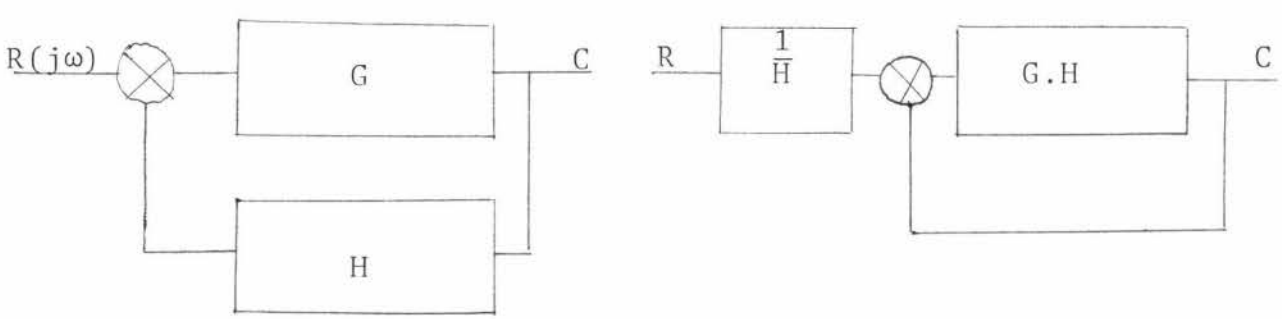


Fig. 2.14 shows the plot of $GH(j\omega)$ superimposed on a Nichols chart.

The Nichols chart is symetric about the -180° axis. The magnitude-phase angle of the closed-loop transfer function of the direct feedback system $\frac{C}{R}(j\omega) + \frac{GH(j\omega)}{1 + GH(j\omega)}$ is obtained directly from the chart as the plot of the points where the graph of $GH(j\omega)$ intersects the graphs of loci of constant $\left|\frac{C}{R}(j\omega)\right|$ and arg $\frac{GH(j\omega)}{1 + GH(j\omega)}$.

If the open-loop locus does not intersect the $M = M_r$ locus but is tangent to it, then the resonant peak value of M of the closed-loop frequency response is given by M_r. The resonant frequency is given by the frequency at the point of tangency. (Note that the) closed-loop plot $\frac{C}{R}$, $(j\omega)$ can be directly generated by the computer

The stability of the system based on the Nichols chart plot is the same as the Bode diagram, i.e.: positive gain and phase margins yield a stable system. Fig. 2.15 shows the gain and phase margins of a stable and an unstable system.

Nichols chart design

Design by analysis in the frequency domain using Nichols chart techniques is performed in the same general manner as the design methods described in previous sections. Appropriate compensation networks are introduced in the forward and/or feedback paths and the behaviour of the resulting system is critically analysed. In this manner, the Nichols chart plot is shaped and reshaped until the performance specifications are met.

Gain-factor compensation using constant amplitude curves

The Nichols chart may be used to determine K_β (for a unity feedback system) for a specified resonant peak M_p(db). The following procedure is applied:

(a) Draw only constant amplitude M_p curve and the magnitude-phase angle plot of $G(j\omega)$ for $K_\beta = 1$ on the Nichols chart.

(b) Trace magnitude-phase angle plot of $G(j\omega)$ on tracing paper.

(c) Slide the plot up or down until it is just tangent to the constant amplitude curve of $M_p$db. The amount of shift in db is the required value of K_β.

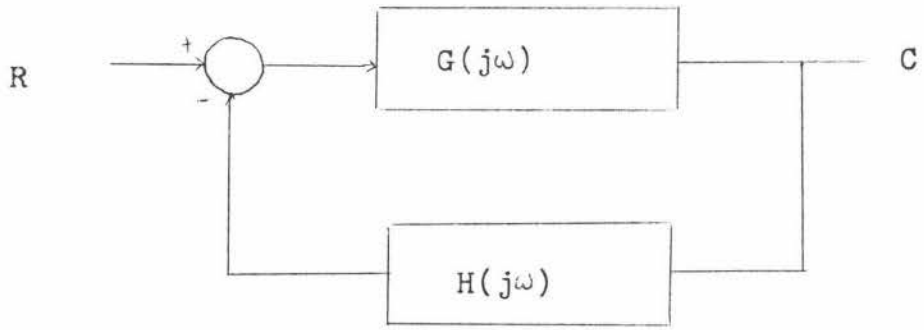


Fig. 2.13 Control system

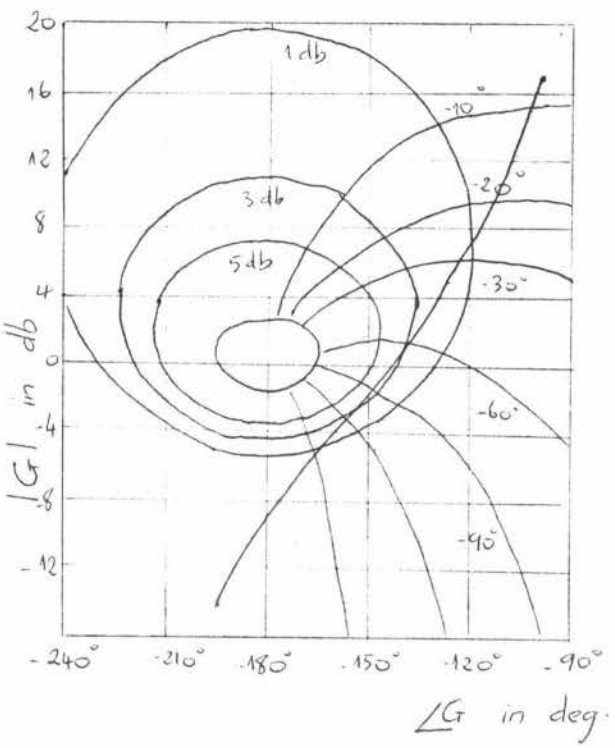


Fig. 2.14 Plot of G(jω).H(jω) superimposed on a Nichols chart

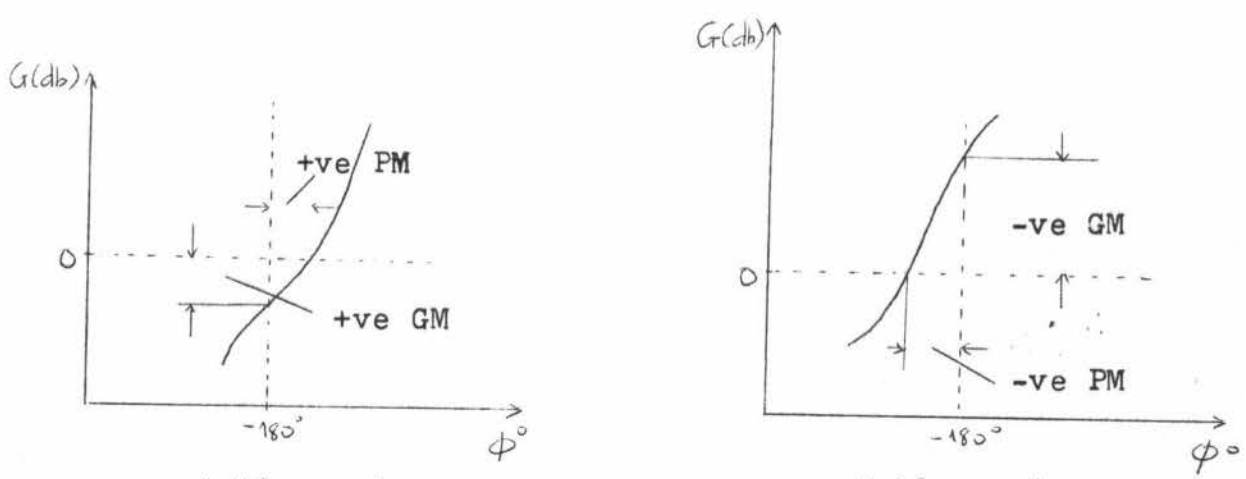


Fig. 2.15 Phase and gain margins of stable and unstable system

Lead compensation

The transfer function for a lead network is $G_c(s) = \dfrac{j\omega + \frac{1}{T}}{j\omega + \frac{1}{\alpha T}}$ where $\alpha<1$.

For some systems in which lead compensation in the forward loop is applicable, appropriate choice of a and b permits an increase in K_β, providing greater accuracy and less sensitivity, without adversely affecting transient performance. The important properties of a lead network compensator are its phase-lead contribution in the low to medium frequency range (the vicinity of resonant frequency ω_p) and its negligible attenuation of high frequencies. If a very large phase-lead is required, several lead networks may be cascaded. Lead compensation generally increases the bandwidth of a system. The use of Nichols chart for selection of phase lead network is particularly suited to problems where the specification is given as a peak M_{max}(db) occurring at a resonant frequency ω_R. Consider the uncompensated system $G(j\omega)$ of a direct feedback control system. The procedure for designing a lead compensator using Nichols chart techniques is as follows:

(a) From the Nichols chart, find the value of phase shift, ϕ, corresponding to the peak of the specified M_{max} contour.

(b) Find the value of phase shift for the uncompensated system when $\omega = \omega_R$ (resonant frequency), say ϕ_R lag.

(c) The required value of phase lead then is

$$\phi_c = \phi_R - \phi_{\text{lead}}$$

If compensation is required, $\phi_R > \phi$.

(d) Determine the attenuation factor α by use of Fig.

$$\sin \phi_M = \frac{1 - \alpha}{1 + \alpha}$$ (Appendix C)

Determine the frequency where the magnitude of the

uncompensated system is equal to $-20 \log_{10}(\frac{1}{\sqrt{\alpha}})$ (Appendix C) Select this frequency as the new gain crossover frequency. This frequency corresponds to ω_m and the maximum phase shift ϕ_m occurs at this frequency.

(e) Determine the corner frequencies of the lead network from

$$\omega = \frac{1}{T}, \quad \omega = \frac{1}{\alpha T}$$

(f) It now remains to find K_c in cascade with the lead network to yield the control system with the required resonant peak M_{max}. Plot $K_c G(j\omega)$ on the Nichols chart with $K_c = 1$. It can now be seen by how much the curve must be moved vertical1 to make the open-loop curve tangential to the M_{max} contour. The vertical movement is $20 \log_{10} K_c$db, hence K_c.

Lag compensation

The lag compensator has the following transfer function:

$$G_c(j\omega) = \frac{1}{\beta} \quad \frac{s + 1/T}{s + 1/\beta T} \qquad (\beta>1)$$

The lag network provides compensation by attenuating the high frequency position of the db magnitude-phase angle plot. Several general effects of lag compensation are:

- The bandwidth of the system is unusually decreased
- The predominant time constant τ of the system is unusually increased, producing a more sluggish system,
- For a given constant, relative stability is improved.

The procedure for using lag compensation in Nichols chart techniques is essentially the same as that for lag compensation in Bode design technique.

Lead-lag compensation

The design of a lead-lag network compensator is based on the combination of the design techniques discussed under lead compensation and lag compensation.

2.5 DISCUSSION

This chapter has described various graphical techniques used in control system analysis and design, namely the time response, root-locus and frequency-response methods.

- The root-locus method is most convenient to use when specifications are given in terms of time domain quantities, such as overshoot, rise time, etc.
- The Bode design technique is the most popular of the frequency response methods if specifications are given in terms of frequency domain quantities, such as gain and phase margins.
- The Nichols chart design technique is used when a certain value of the resonant peak M_r is required.
- The Nyquist design technique provides rapid evaluation of stability of the control system and it is thus used to supplement other methods.

The time response method, though it can be used for control system design with trial and error approaches, is more useful for checking time domain performance following the implementation of a design based upon frequency or root-locus techniques.

However, whatever design technique is used, it is important to note that no single compensation scheme is universally applicable. The particular situation determines the type of compensation that is appropriate. Chapter 3 describes the computer software that has

been written to aid the design procedures described in this chapter. This software is able, given the generalized control system structure of Fig. 2.1, to generate, for either the open- or closed-loop form, any of the specified design charts. The software operates interactively to give the designer wide flexibility of choice in terms of the system structure to be analysed and the technique to be applied. It is planned that the facility will be enhanced to incorporate automatic design procedures for compensation networks.

-ooOoo-

CHAPTER THREE

PROGRAMME DESCRIPTION

3.1 INTRODUCTION, OUTLINE OF PROGRAMME

The software for computer aided control system design has been programmed in Algol and operates on the Massey University B6700 computer system. This chapter outlines the structure of the programme, detailing the methods employed to implement some of the concepts discussed in Chapter 2, and also specifies how the system is used.

The program is operated via a Tektronix 4010 visual display terminal. It is designed to function in an online interactive mode. The control system designer, at the Tektronix console, is able to specify certain designs and evaluate their effectiveness in terms of the various graphical analyses produced by the computer and displayed on the screen of the Tektronix. The designer can then adjust the designs in order to improve performance. Given a particular control system structure, which may be either open-loop or closed-loop, the designer may request the following options:

1. A display of the system time response following a step input.
2. A Bode diagram display of the system frequency response.
3. A Nyquist diagram display of the system frequency response.
4. A Nichols chart display of the system frequency response.
5. A root-locus plot of the roots of the system characteristic equation as the system gain varies.

The designer is able to select the scales and ranges of these displays so that any particular portion of a graph can be isolated and magnified if required.

3.2 EVALUATION OF SYSTEM TRANSFER FUNCTION

The control system is defined with the general structure illustrated in Fig. 3.1. The three transfer functions of Fig. 3.1 are given as the polynomial ratios.

$$G(s) = \frac{G_1(s)}{G_2(s)} = \text{the process transfer function} \qquad (3.1)$$

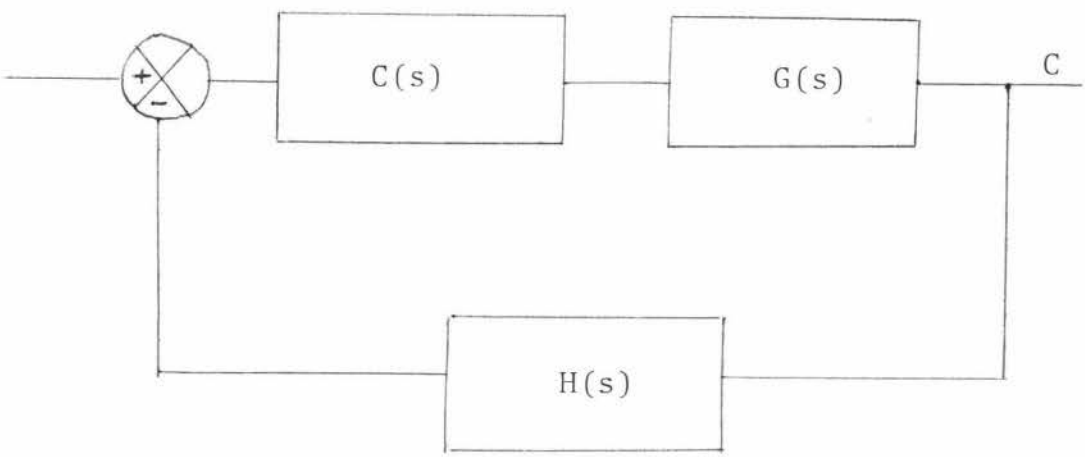


Figure 3.1 Control system

$$H(s) = \frac{H_1(s)}{H_2(s)} = \text{the measurement transfer function} \qquad (3.2)$$

and

$$C(s) = \frac{C_1(s)}{C_2(s)} = \text{the controller transfer function} \qquad (3.3)$$

As mentioned above, the control designer is able to select both open-loop and closed-loop analyses. To achieve this end the program has been designed to be flexible so that the following combinations of the above transfer functions can be studied by request, at the console:

i. open-loop structure

$$G(s),\ K.C(s).G(s) \text{ and } K.C(s).G(s).H(s) \qquad (3.4)$$

ii. the full closed-loop structure of Fig. 3.1 for which the transfer function may be devised as

$$\frac{K\ C_1(s)\cdot G_1(s)\cdot H_2(s)}{C_2(s).G_2(s).H_2(s) + K\ C_1(s)\cdot G_1(s)\cdot H_1(s)} \qquad (3.5)$$

subject to the option chosen, the software is able to manipulate the constituent polynomials to obtain a final transfer function of the form

$$R(s) = \frac{A(s)}{B(s)} \qquad (3.6)$$

which is that analysed.

3.3 FREQUENCY RESPONSE ANALYSIS

Frequency response analysis required evaluation of the following frequency response function

$$R(j\omega) = \frac{A(j\omega)}{B(j\omega)} \qquad (3.7)$$

It is necessary that polynomials $A(j\omega)$ and $B(j\omega)$ and divided into their real and imaginary parts, with

$$A(j\omega) = RA(\omega) + j\ IA(\omega) \qquad (3.8)$$

and

$$B(j\omega) = RB(\omega) + j\ IB(\omega) \qquad (3.9)$$

A procedure has therefore been developed to establish the real and imaginary parts of a general polynomial, of degree N, given the coefficients A_o, A_1, A_2, ... A_n. i.e.,

$$A(j\omega) = A_0(j\omega)^N + A_1(j\omega)^{N-1} + \ldots\ldots\ldots A_{N-1}(j\omega) + A_N \qquad (3.10)$$

$$= A_N + j\omega A_{N-1} - \omega^2 A_{N-2} - j\omega^3 A_{N-3} + \omega^4 A_{N-4} + j\omega^5 A_{N-5} - \omega^6 A_{N-6}$$

$$= (A_N + \omega^4 A_{N-4} + \omega^8 A_{N-8} + \ldots\ldots\ldots) - (\omega^2 A_{N-2} + \omega^6 A_{N-6} + \ldots\ldots\ldots\ldots) + j(\omega A_{N-1} + \omega^5 A_{N-5} + \ldots\ldots\ldots\ldots) - j(\omega^3 A_{N-3} + \omega^7 A_{N-7} + \ldots\ldots\ldots\ldots) \qquad (3.11)$$

Therefore the real and imaginary components are derived as

$$RA(\omega) = (A_N + \omega^4 A_{N-4} + \omega^8 A_{N-8} + \ldots) - (\omega^2 A_{N-2} + \omega^6 A_{N-6} + \ldots) \qquad (3.12)$$

and

$$IA(\omega) = (\omega A_{N-1} + \omega^5 A_{N-5} + \ldots) - (\omega^3 A_{N-3} + \omega^7 A_{N-7} + \ldots) \qquad (3.13)$$

Terms in the above equations are included only if the indices for A are not negative,

$$\text{i.e. } A_{-1} = A_{-2} = \ldots\ldots\ldots\ldots 0$$

A procedure has been developed to derive the real and imaginary components by evaluating the above expressions for RA and IA. Difficulties were encountered in this evaluation because of particular properties of the supplied polynomial arithmetic procedures. This necessitated the development of a slightly less straightforward method of derivation than that indicated above. For the Bode display, the gain is plotted as

$$G = 20 \log_{10}|R(j\omega)| = 20 \log_{10} \sqrt{\frac{RA^2(\omega) + JA^2(\omega)}{RB^2(\omega) + IB^2(\omega)}} \quad \text{db} \qquad (3.14)$$

and the phase is

$$P = \text{TAN}^{-1}\left(\frac{IA(\omega)}{RA(\omega)}\right) - \text{TAN}^{-1}\left(\frac{IB(\omega)}{RB(\omega)}\right) \quad \text{degrees} \qquad (3.15)$$

The above expressions are also used for derivation of the Nichols chart display.

$R(j\omega)$ can also be written as

$$R(j\omega) = X(\omega) + jY(\omega)$$

with

$$X(\omega) = \frac{RA(\omega).RB(\omega) + IA(\omega).IB(\omega)}{RB^2(\omega).IB^2(\omega)} \qquad (3.16)$$

and

$$Y(\omega) = \frac{IA(\omega).RB(\omega) - RA(\omega).IB(\omega)}{RB^2(\omega).IB^2(\omega)} \qquad (3.17)$$

The Nyquist display is a plot of $Y(\omega)$ vs $X(\omega)$ as ω varies.

Constant magnitude and phase contours may be drawn on the Nichols display. These are derived using the expressions:

$$\cos \phi = - \frac{(M^2 + R^2(M^2 - 1))}{2M^2R} \qquad (3.18)$$

and

$$\delta = \phi - \text{TAN}^{-1} \frac{R \sin \phi}{1 + R \cos \phi} \qquad (3.19)$$

R and ϕ, functions of ω, are the magnitude and phase angle of the open loop transfer function of a direct feedback control system ($H(s) = 1$). M and δ are the magnitude and phase angle of the closed-loop transfer function of a direct feedback system. The M and δ contours are plotted on the Nichols display to generate the standard configuration.

3.4 ROOT-LOCUS ANALYSIS

A root-locus display is established by plotting the roots of the characteristic equation extracted from expression 3.5, namely:

$$C_2(s).G_2(s).H_2(s) + K.C_1(s).G_1(s).H_1(s) = 0$$

as K varies. Open-loop poles and zeros are established from the roots of

$$C_2(s).G_2(s).H_2(s) = 0$$

and

$$C_1(s).G_1(s).H_1(s) = 0$$

respectively.

3.5 TIME DOMAIN ANALYSIS

The time-response of the system to a step input is obtained by directly solving differential equations representing the system.

Consider the final transfer function described in equation 3.2

i.e. $$R(s) = \frac{A(s)}{B(s)}$$

or $$\frac{Y(s)}{U(s)} = \frac{A_0 s^M + A_1 s^{M-1} + \ldots + A_M}{B_0 s^N + B_1 s^{N-1} + \ldots + B_N} \qquad (3.20)$$

where $N-1 \geq M$

Equation 3.20 may be written

$$\frac{Y(s)}{U(s)} = \frac{AA_0 s^M + AA_1 s^{M-1} + \ldots AA_M}{s^N + BB_1 s^{N-1} + \ldots + BB_N} \qquad (3.21)$$

where

$$AA_0 = \frac{A_0}{B_0} \quad ; \quad AA_1 = \frac{A_1}{B_0} \quad ; \quad \ldots\ldots AA_M = \frac{A_M}{B_0}$$

and

$$BB_1 = \frac{B_1}{B_0} \quad ; \quad BB_2 = \frac{B_2}{B_0} \quad ; \quad \ldots\ldots BB_N = \frac{B_N}{B_0}$$

From equation 3.1 we obtain

$$Y(s^N + BB_1 s^{N-1} + \ldots + BB_N) = U\left(\frac{AA_0}{s^{N-M}} + \frac{AA_1}{s^{N-(M-1)}} + \ldots + \frac{AA_M}{s^N}\right.$$

or

$$Y = U\left(\frac{AA_0}{s^{N-M}} + \frac{AA_1}{s^{N-(M-1)}} + \ldots + \frac{AA_M}{s^N}\right) - Y\left(\frac{BB_1}{s} + \frac{BB_2}{s^2} + \ldots + \frac{BB_N}{s^N}\right)$$

(3.22)

By introducing additional variables such as the rate of change of output, the same system can be represented by N first-order equations involving N system variables.

The state variable diagram for Eq. 3.22 is shown in Fig. 3.2

From Fig. 3.2 the following system of first-order equations may be written:

$$
\begin{aligned}
Z_1 &= -BB_1 \cdot Z_1 + Z_2 \\
Z_2 &= -BB_2 \cdot Z_1 + Z_3 \\
Z_3 &= -BB_3 \cdot Z_1 + Z_4 \\
&\vdots \\
Z_{N-M} &= -BB_{N-M} \cdot Z_1 + Z_{N-M+1} + AA_0\, U \\
Z_{N-M+1} &= -BB_{N-M+1} \cdot Z_1 + Z_{N-M+2} + AA_1\, U \\
&\vdots \\
Z_{N-1} &= -BB_{N-1} \cdot Z_1 + Z_N + AA_{M-1}\, U \\
Z_N &= -BB_N\, Z_1 + AA_M\, U
\end{aligned}
\qquad (3.23)
$$

Expressing in matrix form

$$
\begin{bmatrix} Z_0 \\ Z_1 \\ Z_2 \\ \vdots \\ Z_{N-M} \\ \vdots \\ Z_{N-1} \\ Z_N \end{bmatrix} =
\begin{bmatrix}
0 & 0 & 0 \ldots\ldots 0 \\
0 & -BB_1 & 1 \ldots\ldots 0 \\
0 & -BB_2 & 0 \ldots\ldots 0 \\
\vdots & \vdots & \vdots \\
AA_0 & -BB_{N-M} & \vdots \\
\vdots & \vdots & \vdots \\
AA_{M-1} & -BB_{N-1} & 1 \\
AA_M & -BB_N & 0 \ldots\ldots 0
\end{bmatrix}
\cdot
\begin{bmatrix} U \\ Z_1 \\ \vdots \\ Z_{N-M} \\ \vdots \\ Z_{N-1} \\ Z_N \end{bmatrix}
$$

$$
\begin{aligned}
Z_0 &= 0 \\
Y &= Z_1
\end{aligned}
\qquad (3.24)
$$

Equations 3.24 are solved numerically using a standard differential equation solving subroutine.

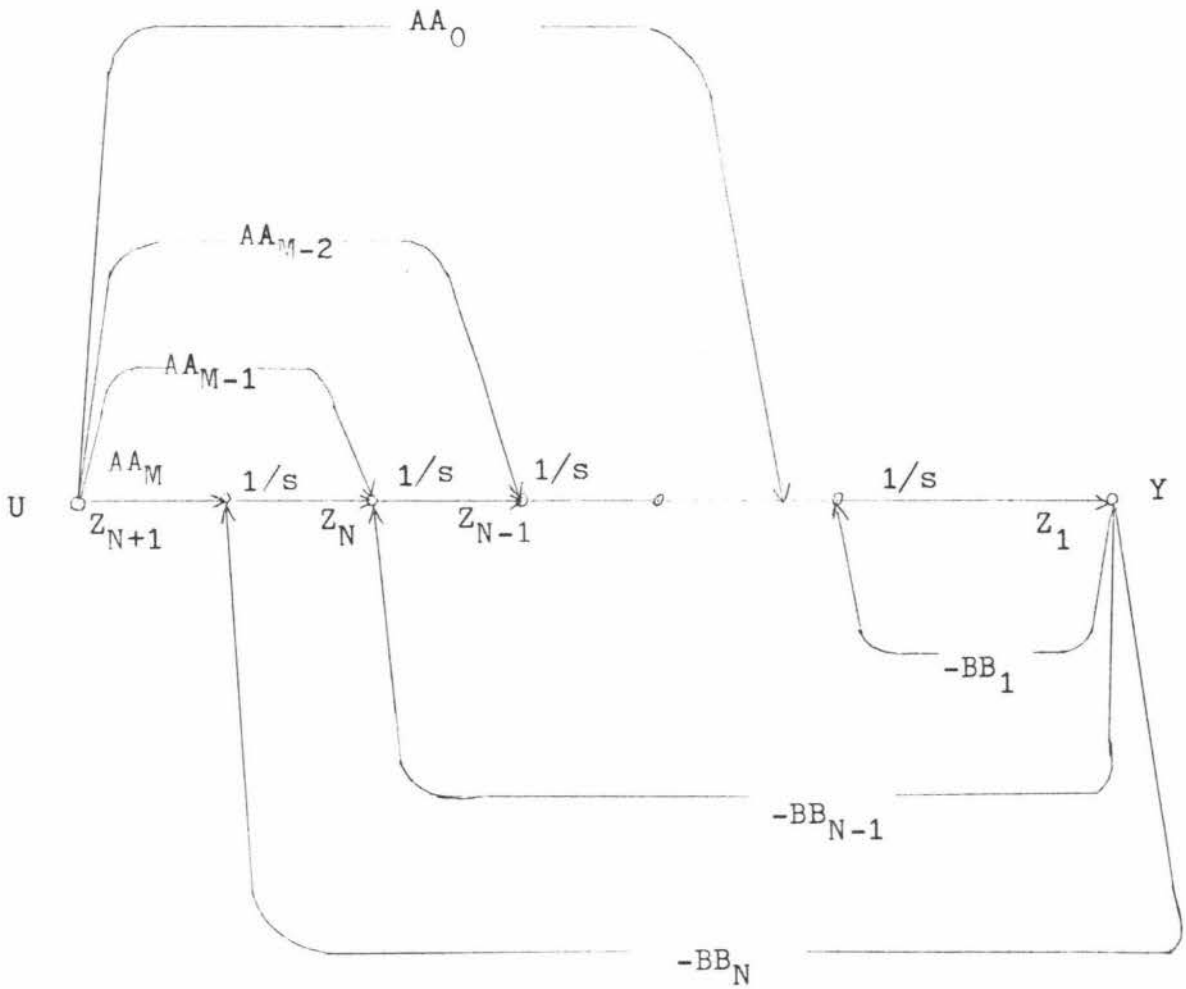


Fig.32 State variable diagram of Eq.(3.22)

3.5 USE OF THE PROGRAM

The program is entered directly into the B6700 by command from the Tektronix terminal. When the program is executed, the following steps are performed in order. (Appendix E illustrates a block flow diagram of the program.)

1. The program asks the user to specify the type of response by printing "TIME OR FREQUENCY RESPONSE?" on the screen.
2. The user types "T" for time response and "F" for frequency response.
3. Assuming "F" is typed in, the program then asks for the required option, i.e. open- or closed-loop system, by printing "OPEN- OR CLOSED-LOOP?"
4. The user types in the appropriate answer from the keyboard, "O/P" for open-loop, and "CLOSED" for closed-loop system.
5. The program asks for system gain and coefficients of the polynomials of the constituent transfer functions, i.e. K, $C_1(s)$, $C_2(s)$, $G_1(s)$, $G_2(s)$, $H_1(s)$ and $H_2(s)$. These values are entered from the keyboard as required
6. The program asks the user to specify the type of frequency-response graph to be displayed by printing "WHICH GRAPH? - BODE, NYQUIST, NICHOLS OR R-LOCUS?"

   (Note: Root-locus is included as a frequency-response plot to facilitate the programming.)
7. The user enters the appropriate answer to the question in step 6 by typing "B" if Bode diagram is required
   "NY" if Nyquist diagram is required
   "NI" if Nichols chart plot is required
   "R" if Root-locus plot is required.
8. After the variables relating to the required graph are entered, the graph is displayed on the screen. Five seconds after the plotting is finished, the program asks if the

user wants to change the scale of the graph by printing "DO YOU WANT TO CHANGE THE SCALE?"

9. If "Y" is answered, steps 8, 9 are repeated.
   If "N" is answered,the program continues to ask if the user wants any other frequency-response plot, by printing "DO YOU WANT ANY MORE GRAPH?"

10. If "Y" is the answer, step 6, 7. 8 9, 10 and repeated.
    If "N" is the answer, the program asks if the user wants to change to any new values, i.e. new $C_1(s)$, $C_2(s)$, $G_1(s)$, $G_2(s)$ $H_2(s)$, $H_1(s)$ and K, by printing "DO YOU WANT TO CHANGE TO ANY NEW VALUES?"

11. If "Y" is typed in, step 5, 6 7, 8, 9, 10,11, are repeated.
    If "N" is typed in, the user is asked "DO YOU WANT ANY MORE FREQUENCY-RESPONSE?"

12. If "Y" is entered, step 3, 4, 5, 6, 7, 8, 9, 0, 11, 12 are repeated.
    If "N" is entered, the user is asked "WHAT ABOUT TIME-RESPONSE?"

13. If "Y", step 3, 4, 5 are performed and followed by steps 14, 15, 16, 17.
    If "N" the program ends.

14. The program asks for the required information to generate the time-response of the system to a step input, such as the value of step-function, the output range and increment, time range and increment.

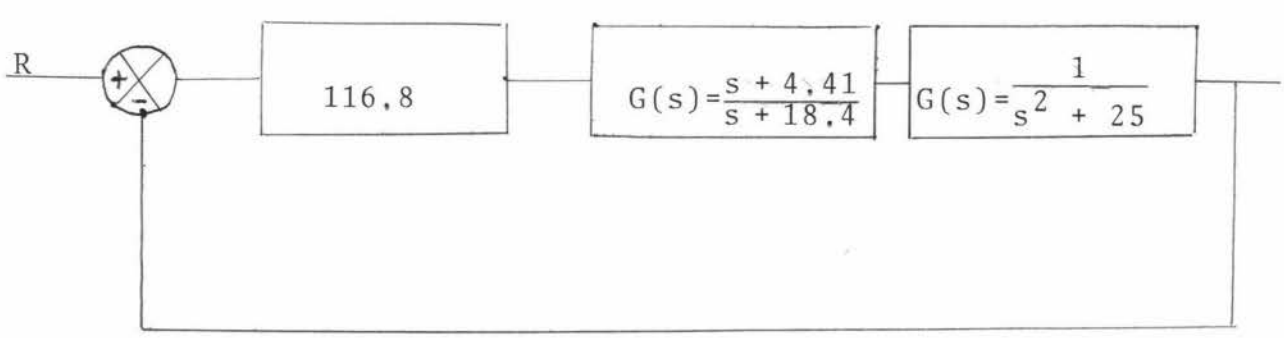
15. After the plotting of the time-response is finished, the user may change the scale of the graph or value of step-input by typing "Y" in answer to the question "DO YOU WANT TO CHANGE THE SCALE?"

16. If "Y" step 14, 15, 16 are repeated.

If "N", the user is asked "DO YOU WANT TO CHANGE TO ANY ANY NEW VALUES?"

17. If "Y" step 13 is repeated.
    If "N" the program ends.
    However, if in step 2, the answer to be typed in is "T" instead of "F", i.e. the time-domain response is required first, the procedure to be followed is:
18. Step 3, 4, 5, 14, 15, 16 are performed.
19. If "N" is answered to question "DO YOU WANT TO CHANGE TO NEW VALUES?" from step 16, steps 20, 21, 22 are performed.
    If "Y", steps 18, 19 are repeated.
20. The program asks "WHAT ABOUT FREQUENCY-RESPONSE?"
21. If "Y" is entered, steps 3, 4, 5, 6 7, 8 9, 10, 11 are performed.
22. If "N" is answered to question "DO YOU WANT ANY MORE FREQUENCY-RESPONSE?" from step 11, program ends
    If "Y" is entered, steps 21, 22 are performed

The following example illustrates the use of the program. Consider the direct feedback control system shown below:



If a Bode diagram of the open-loop transfer function of the system is to be generated, the communication between the computer and the user begins as follows:

C: TIME OR FREQUENCY RESPONSE?

U: "F"

C: O/P OR CLOSED-LOOP?

U: "O/P"

C: VALUE OF K?

U: 116.8

C: DEGREE OF C_1, C_2, G_1, G_2, H_1, H_2?

U: 1,1, 0, 1, 0, 0

C: COEFFICIENTS OF $C_1(s)$?

U: 1, 4.41

C: COEFFICIENTS OF $C_2(s)$?

U: 1, 18.4

C: COEFFICIENT OF $G_1(s)$?

U: 1

C: COEFFICIENTS OF $G_2(s)$?

U: 1. 2. 0

C: COEFFICIENT OF $H_1(s)$?

U: 1

C: COEFFICIENT OF $H_2(s)$?

U: 1

C: WHICH GRAPH? - BODE, NYQUIST, NICHOLS OR ROOT-LOCUS?

U: "B"

C: FREQUENCY RANGE?

U: 1, 100

C: GAIN RANGE?

U: -40, 40

C: PHASE RANGE?

U: -180, 0

C: GAIN, PHASE AND FREQUENCY INCREMENT?

U: 10, 30, 1

C: IS GRID WANTED?

U: "Y"

After the plotting is finished, the communication between the user and the computer continues:

C: DO YOU WANT TO CHANGE THE SCALE?

U: "N"

C: DO YOU WANT MORE GRAPH?

If the user does not want any more graph she should answer

U: "N"

C: DO YOU WANT TO CHANGE TO NEW VALUES?

If the user wants to change the T.F of the system, she should type "Y", and the process starts from the beginning. If she does not with to do so, she can type "N", the computer then answers

C: DO YOU WANT ANY MORE FREQUENCY PLOT?

This gives the user option to change from open-loop to closed-loop T.F. or vice versa. If she does not want any more frequency-response plot of the system, either open- or closed-loop, she may type

U: "N"

C: WHAT ABOUT TIME-RESPONSE?

U: "N"

End of the program.

Other graphs, such as time-response, Nyquist diagram, Nichols chart plot and Root-locus may be generated in the same manner, with different questions to answer.

3.7 DISCUSSION

This chapter has decided the software for computer-aided control system design. Specific design examples are described in Chapter 4. These examples further illustrate aspects of the

design software. A full listing of the programme and details of its constituent procedures are presented in Appendices D and G.

The software development exercise has been taken to a certain stage only because of the limited project time. There are many more options open for anhanced development to further facilitate the design procedure. For example

i. the direct display of the gain and phase margins on the frequency-response charts;

ii. the direct display of frequency at any point on the Nichols chart and Nyquist diagram;

iii. the direct display of the open-loop gain at any point on the root-locus plot

and iv. automatic evaluation of the angle and magnitude condition on the root-locus diagram.

Many other features can be incorporated to make the design procedures more automatic.

-ooOoo-

CHAPTER FOUR

RESULTS

Three illustrative design examples are presented in this chapter. These utilize the interactive computer software. The first example is via the trial-and-error approach. The second example is through analysis, has system performance specifications given in terms of time-domain quantities. The third example, through analysis, has system performance specifications given in terms of frequency-domain quantities.

4.1 DESIGN EXAMPLE VIA TRIAL-AND-ERROR APPROACH

Consider the system shown in Figure 4.1. With no stringent performance specification required, it is desired to compensate the system so that the system has satisfactory relative stability and high-precision closed-loop control.

The following information obtained from the computer-generated graphs indicates that the system is unstable when the loop is closed.

From Figure 1 showing the closed-loop time response to a unit-step function input, the transient terms are seen oscillatory with an increasing exponential envelope.

From Figure 2 showing the Nyquist diagram of the open-loop system, the locus encloses the point $(-1(j\omega))$ as ω is varied from .1 to 20 rad/sec.

From Figure 3, showing the Root-locus of the open-loop system, the loci pass into the R-H half of the s-plane (i.e. poles with positive real parts) when the open-loop gain is increased from zero to 30.

From Figures 4, 5 showing the Nichols chart plot and the Bode diagram of the open-loop system, the gain and phase margins are found to be -8.67db and -16^{o}.

Hence, the system is unstable

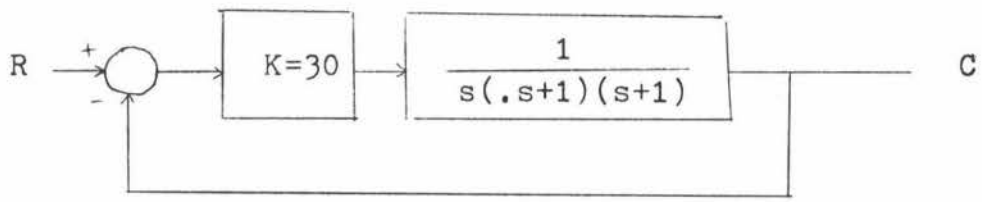


Fig. 4.1 uncompensated system

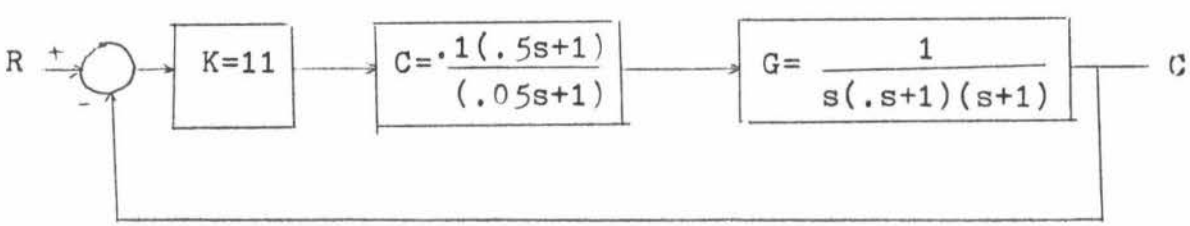


Fig. 4.2 The Lead-compensated system

Reducing the system open-loop gain K is the first step in improving the behaviour of the system. The resulting characteristics of the system are examined and it is found that the system is on the verge of instability for K = 11, as seen in the following graphs.

From Figure 6 showing the time response of the closed-loop system to a unit-step input, the output is a stable amplitude sinusoidal oscillation.

From Figure 7 showing the Nyquist diagram of the open-loop system, the locus passes the point (-1, jo) as ω is varied from .01 to 9 rad/sec.

Figure 8 shows the Root-locus of the O/P-loop system. For the O/P-loop gain K - 11, the poles of the closed-loop T.F. are found on the imaginary axis (i.e. the real parts = 0). The loci will pass into the R H. half of the s-plane if K≥11 and the system is then unstable.

Figure 9 shows the Nichols chart plot of the O/P-loop system. The gain and phase margins are found zero.

Figure 10 shows the Bode diagrams of the O/P-loop system. Gain and phase margins are found zero, as calculated from Figure 9.

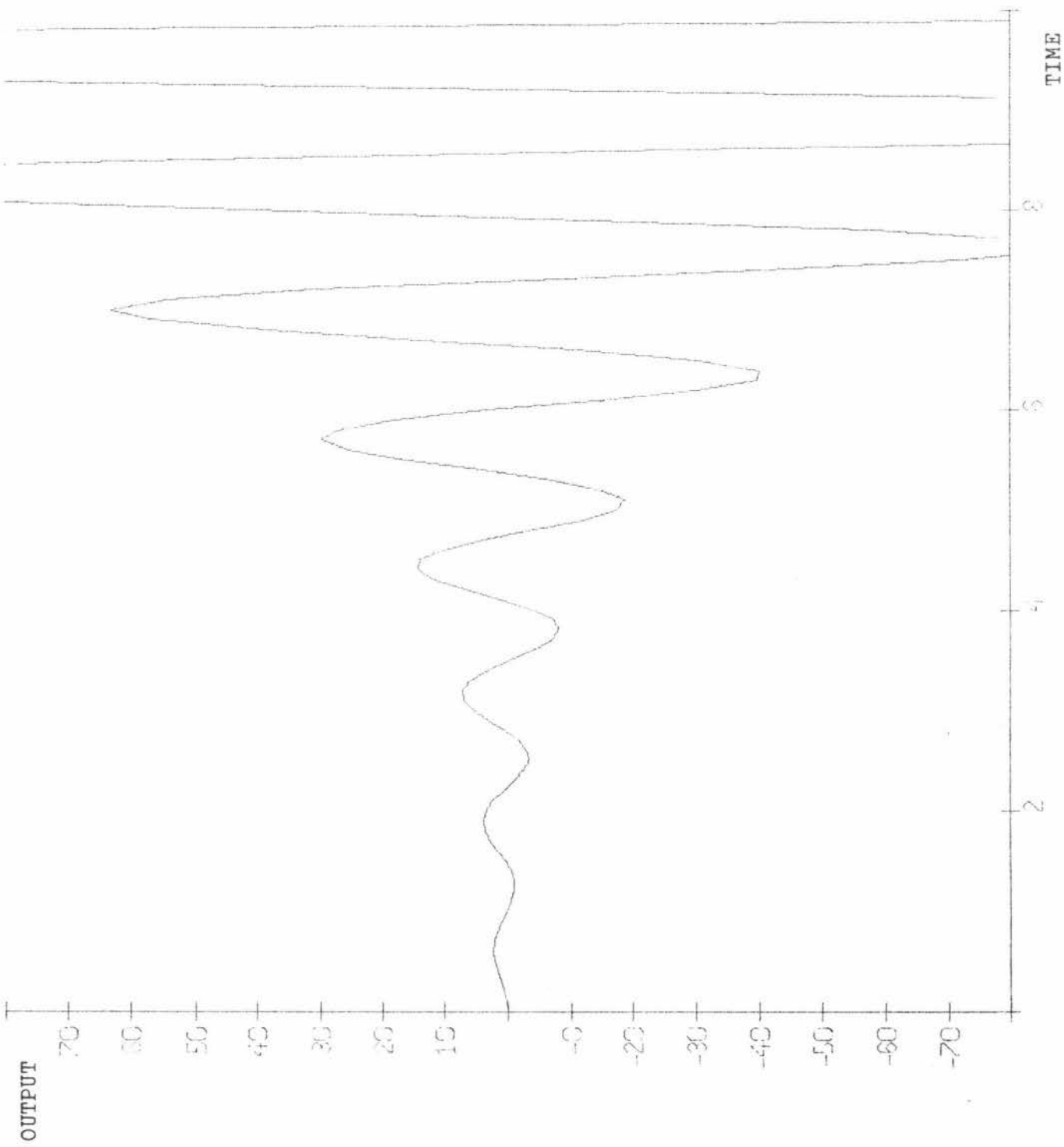
The above information illustrates the'marginal stability' of the system, in which case the output oscillates with a stable amplitude.

The O/P-loop gain K may be further reduced to make the above 'marginally stable' system stable. However, by improving the system stability, the speed of response and therefore the delay and settling time are increased.

Thus, high precision closed-loop control and good stability are incompatible, as seen in Figure 11, Figure 12, Figure 13, and Figure 14.

Resulting quantities measuring the performance of the closed-

FIG. 1



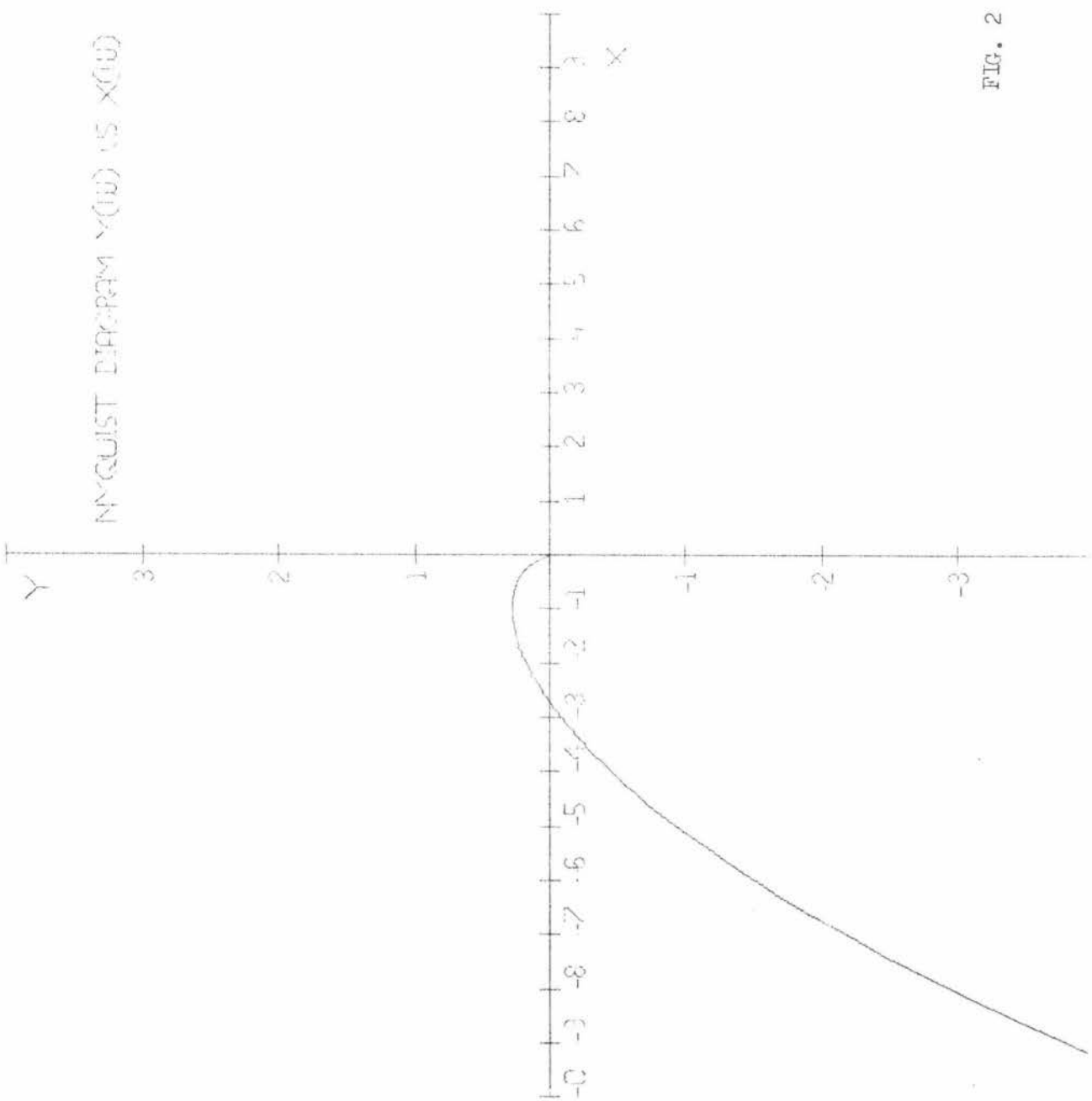
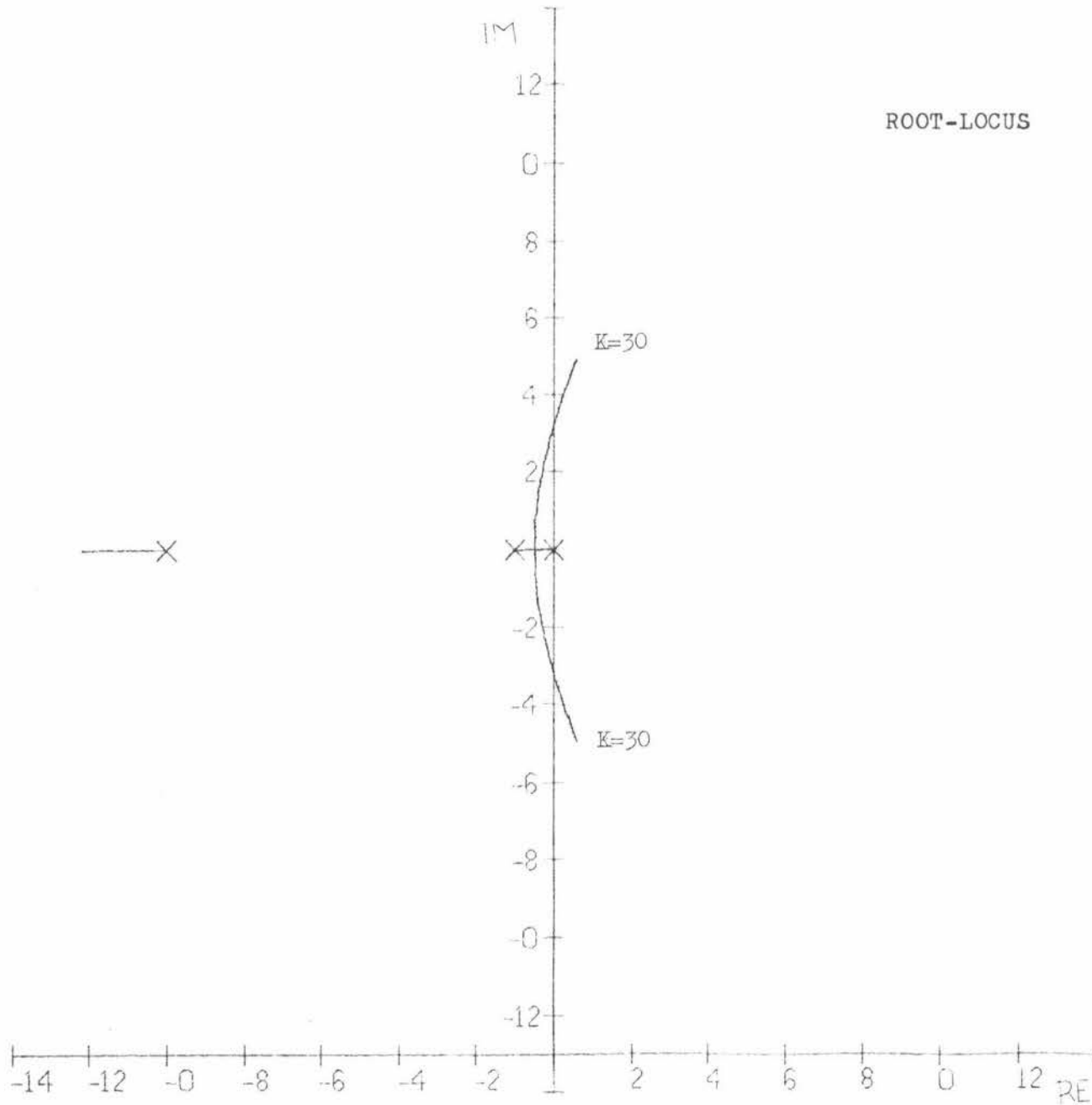


FIG. 2



ROOT-LOCUS

FIG. 3

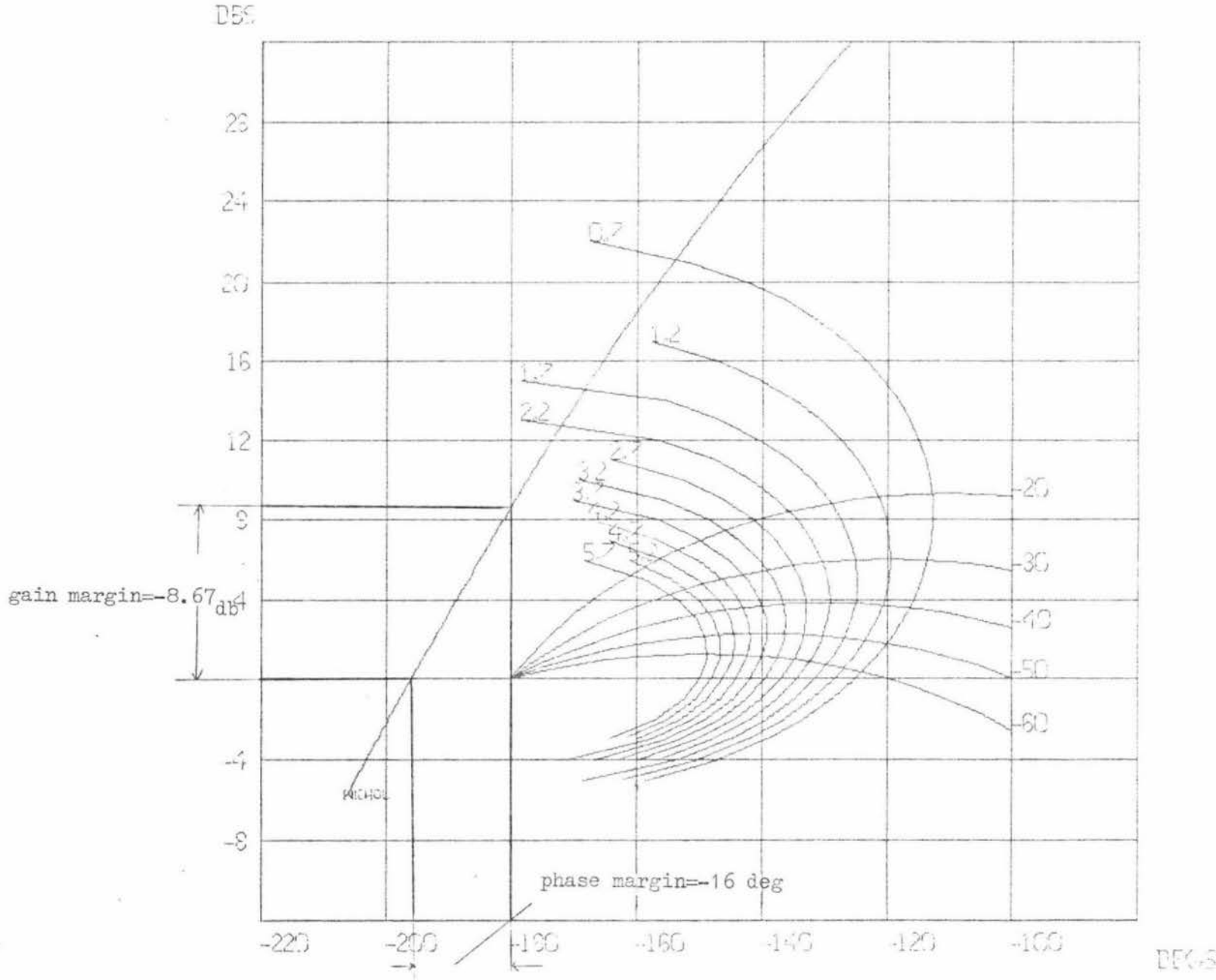

DB
28
24
20
16
12
8
4
-4
-8
0.7
1.2
1.7
2.2
2.7
3.2
-20
-30
-40
-50
-60
gain margin=-8.67 db
phase margin=-16 deg
-220
-200
-180
-160
-140
-120
-100
DEG

FIG. 4

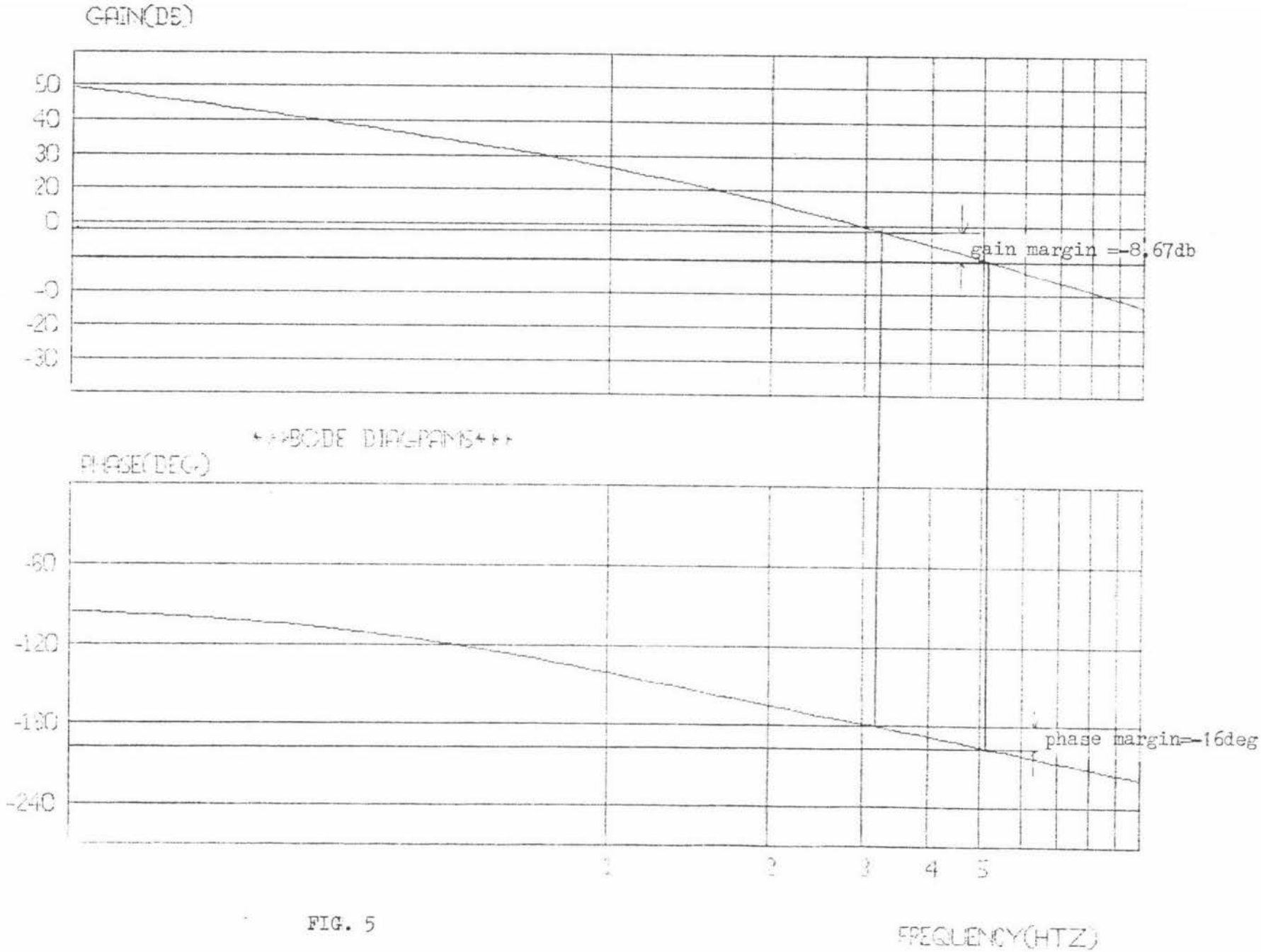

GAIN(DB)
50
40
30
20
0
-0
-20
-30
gain margin =-8.67db
BODE DIAGRAMS
PHASE(DEG)
-60
-120
-180
-240
phase margin=-16deg
1
2
3
4
5
FREQUENCY(HTZ)

FIG. 5

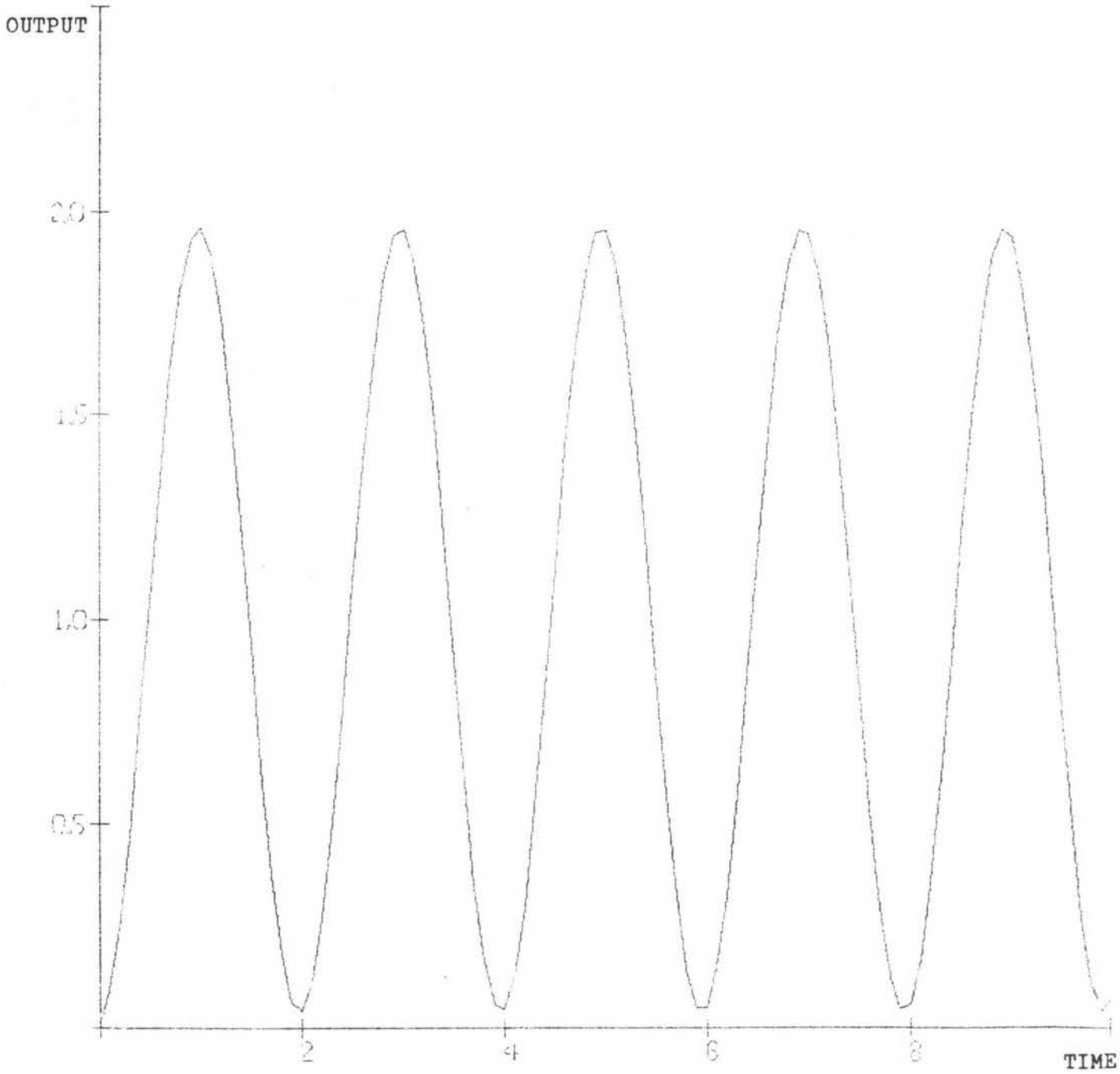


FIG. 6

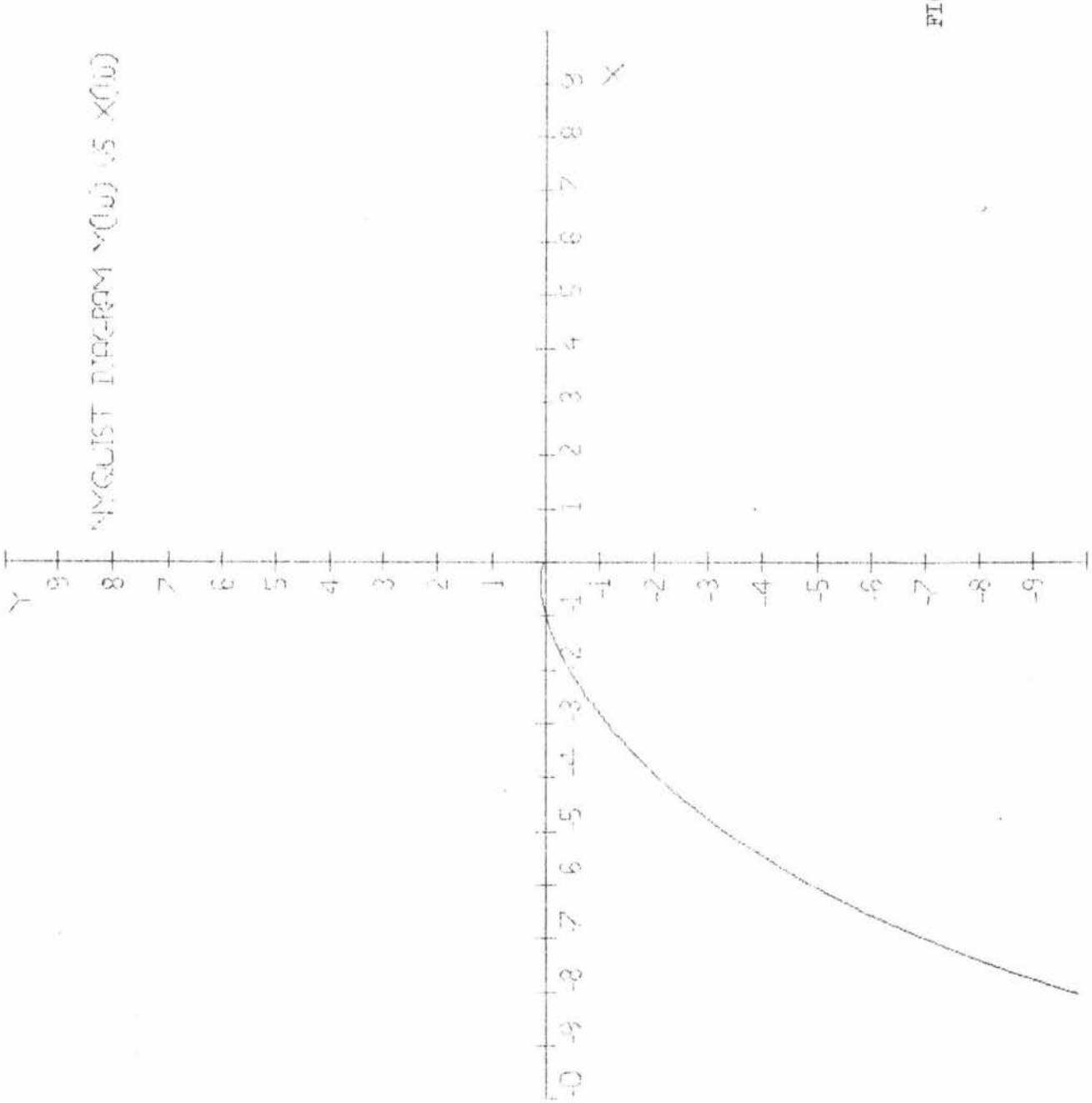


FIG. 7

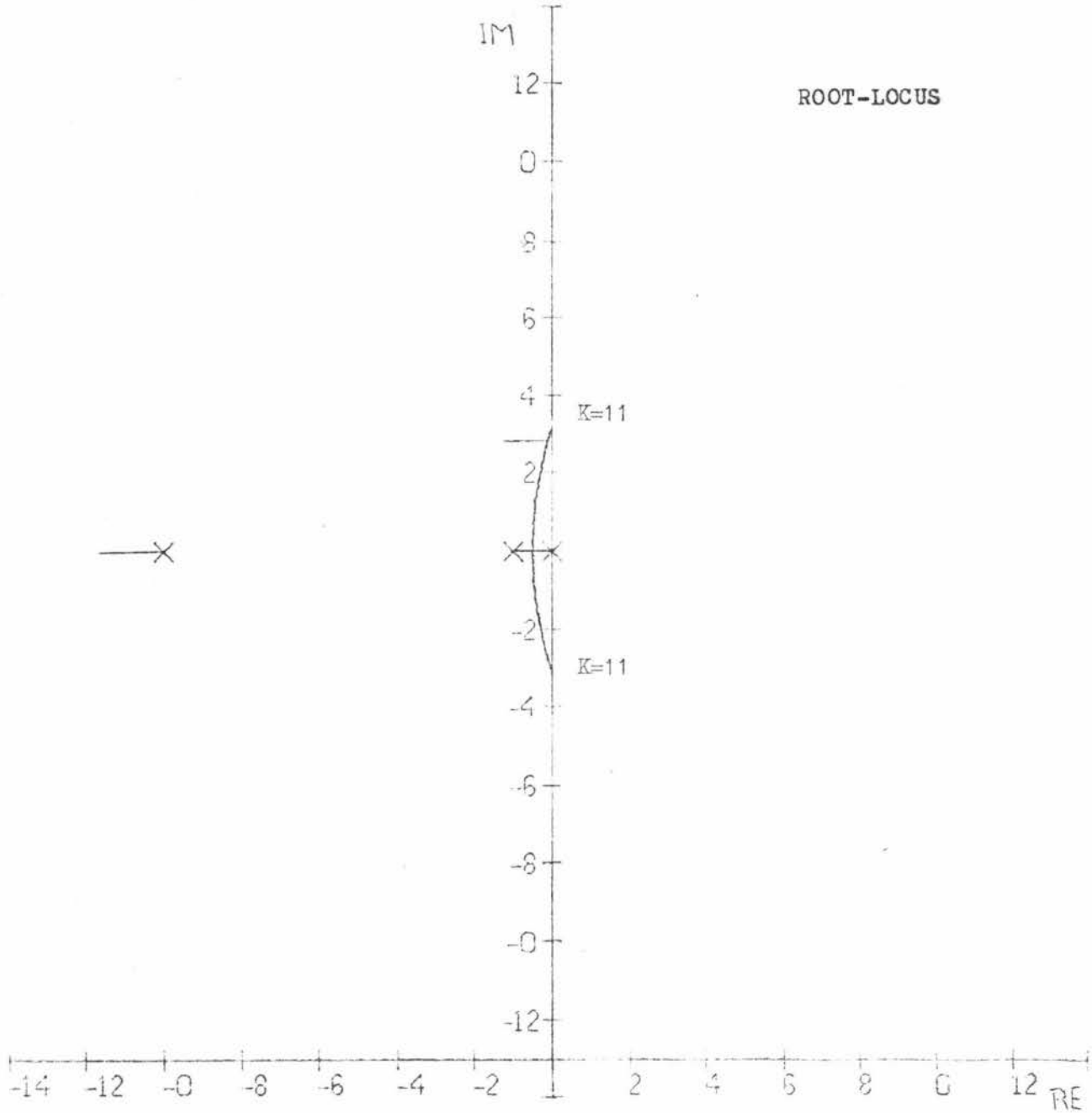


FIG. 8

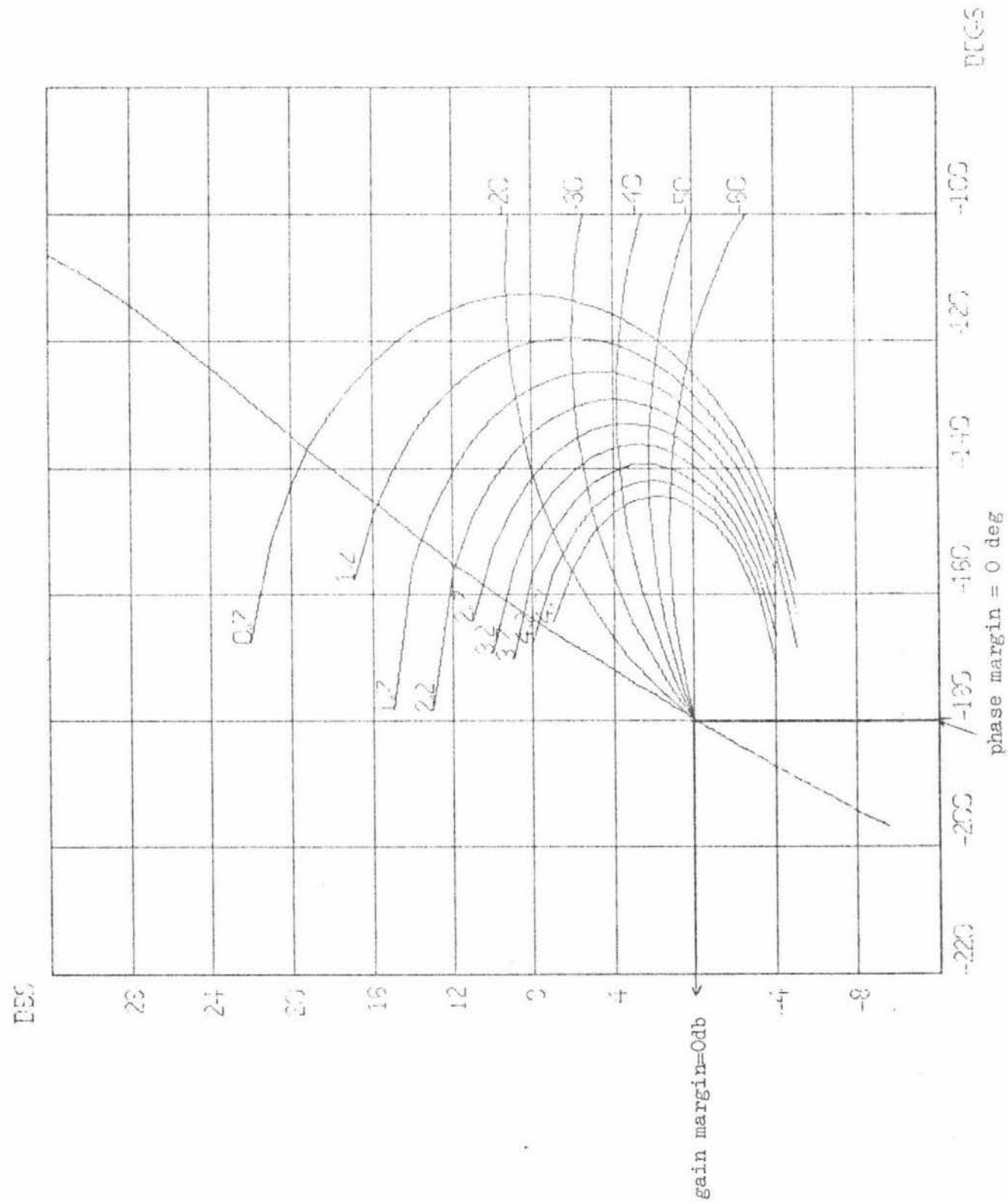

DEGS
23
20
16
12
8
4
-4
-8
gain margin=0db
-120
-140
-160
-180
-200
-220
phase margin = 0 deg
-90
-60
-45
-30
-20
0.7
1.4
2.2
2

FIG. 9

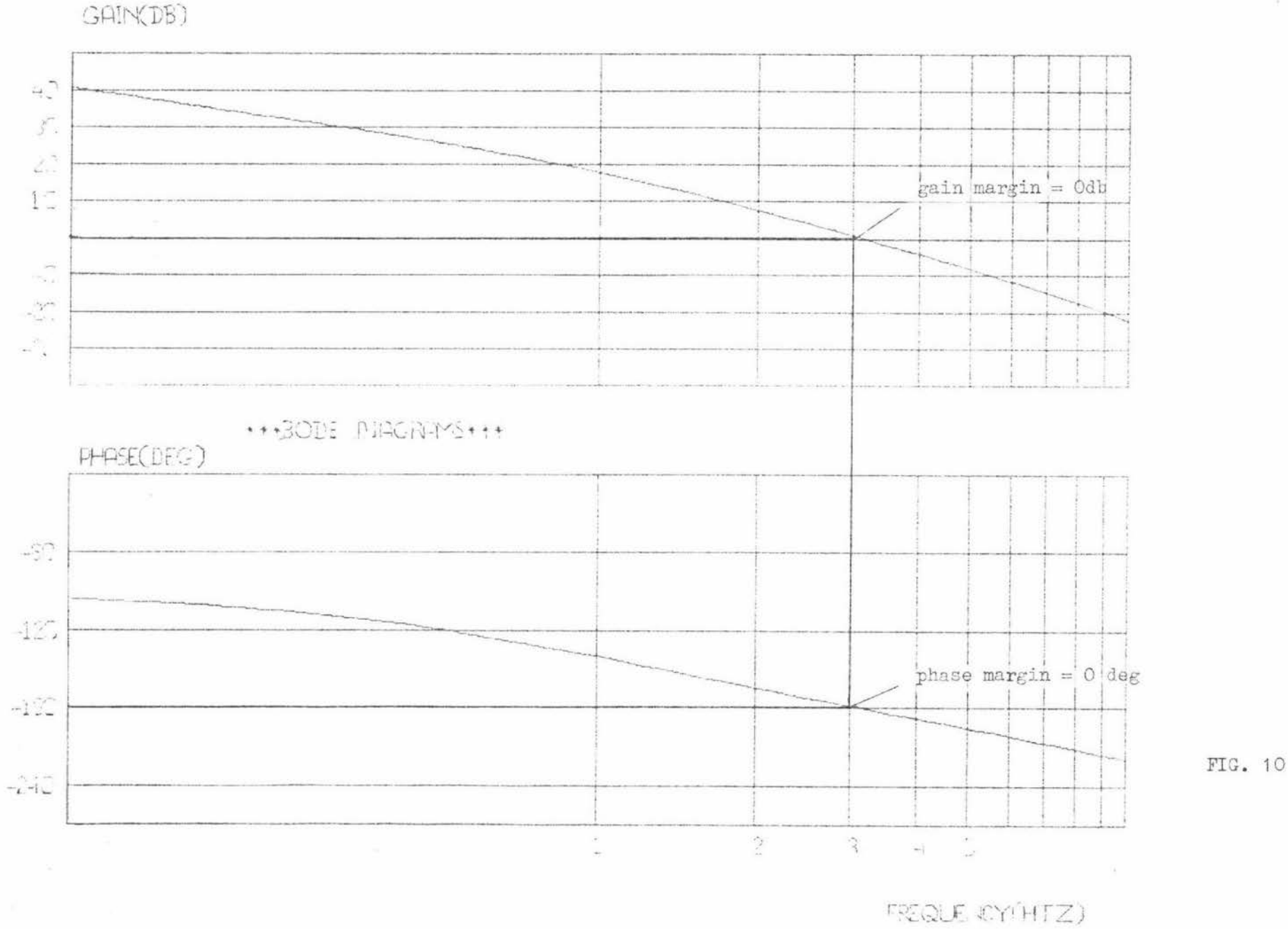

GAIN(DB)
gain margin = 0db
BODE DIAGRAMS
PHASE(DEG)
-90
-135
-180
-240
phase margin = 0 deg
1
2
3
4
5
FREQUENCY(HTZ)

FIG. 10

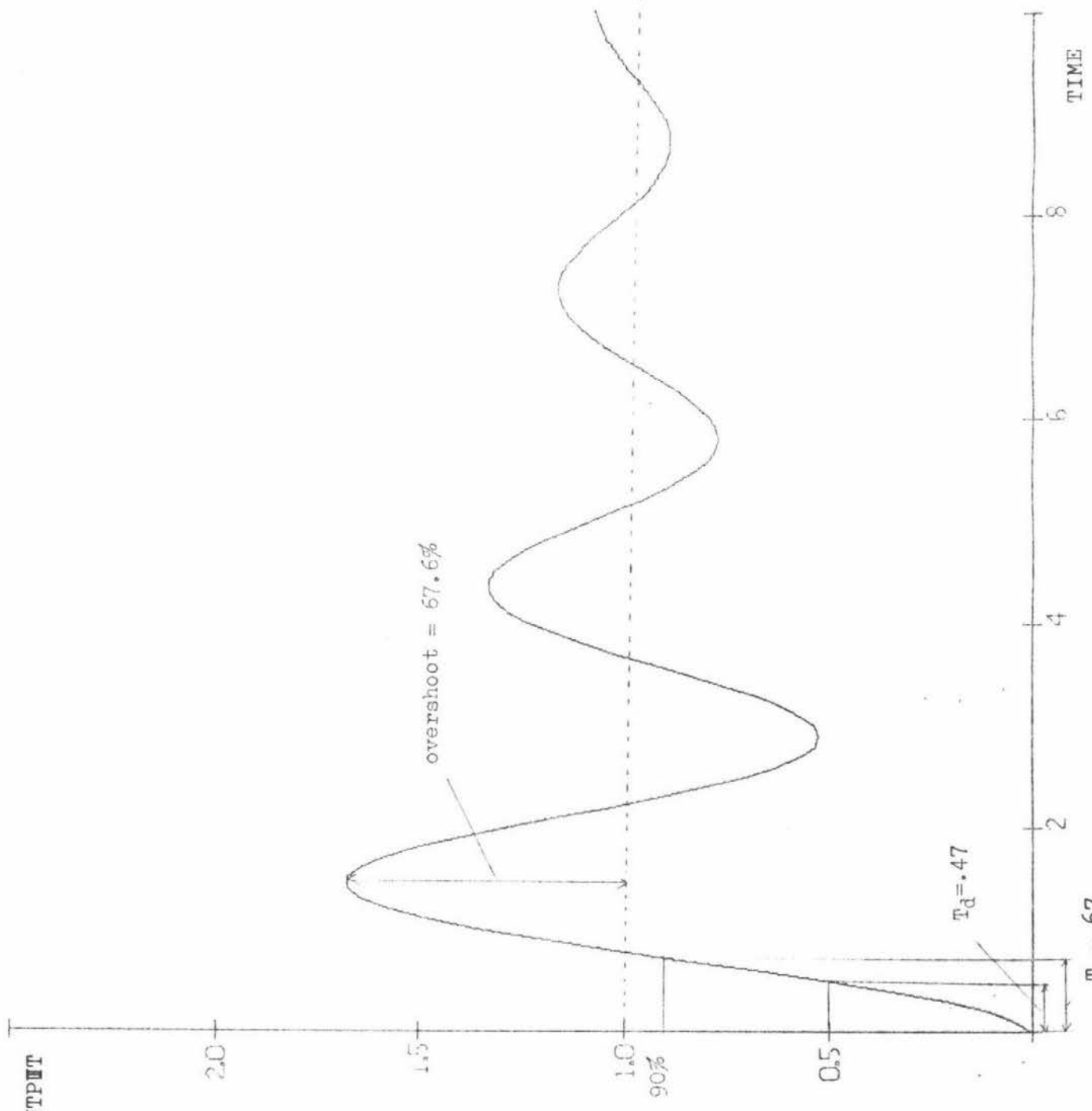


FIG. 11

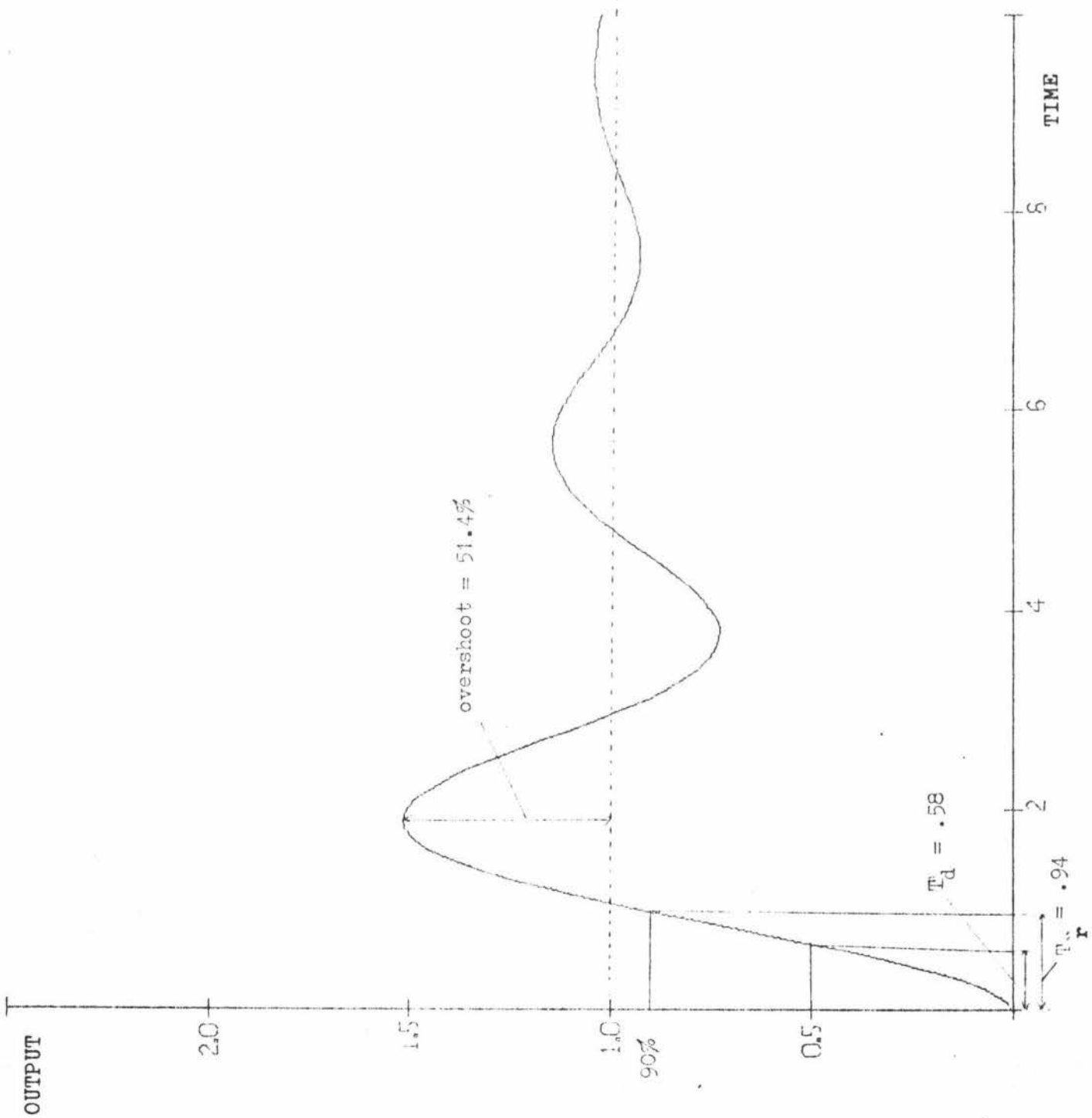


FIG. 12

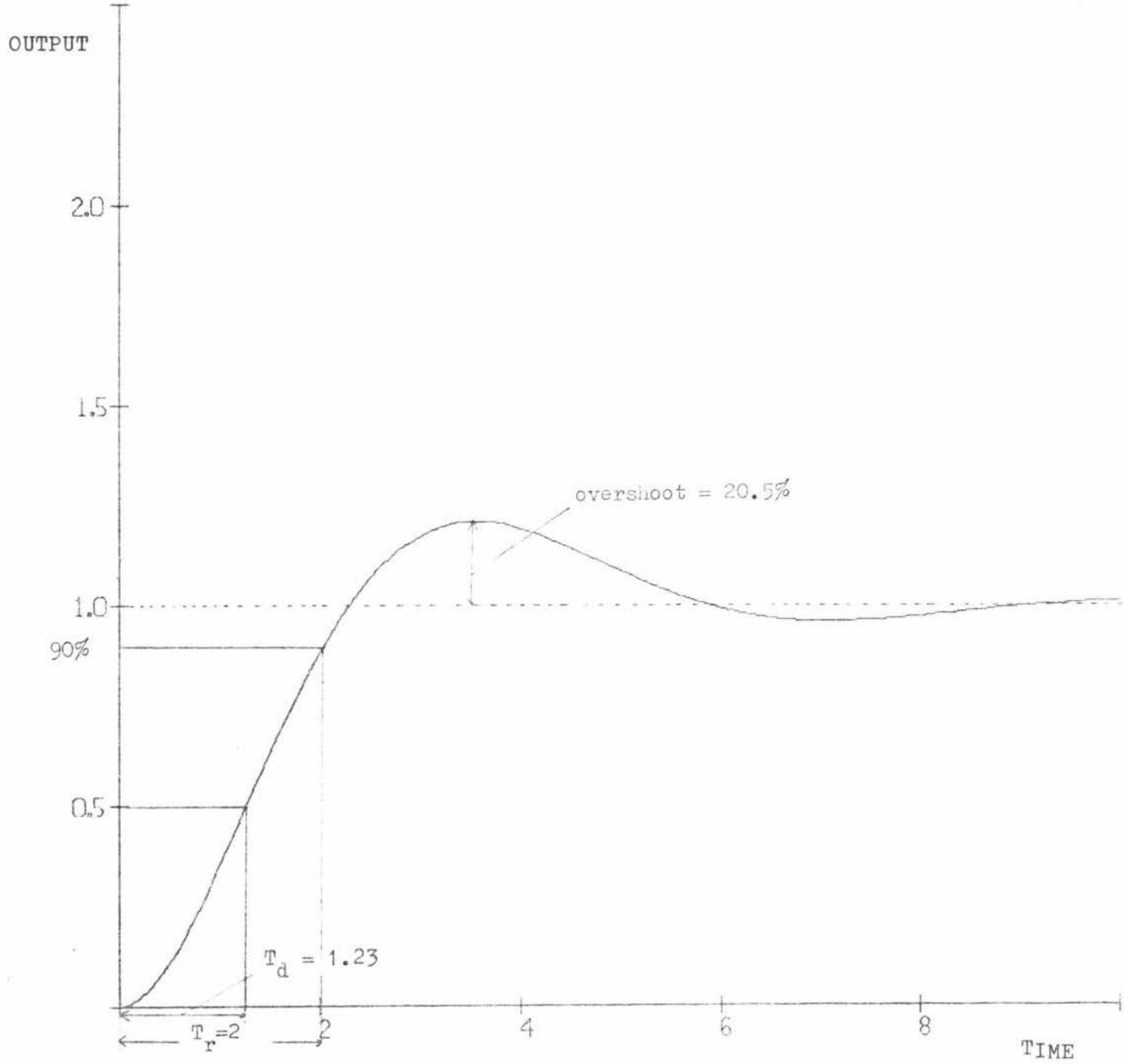


FIG. 13

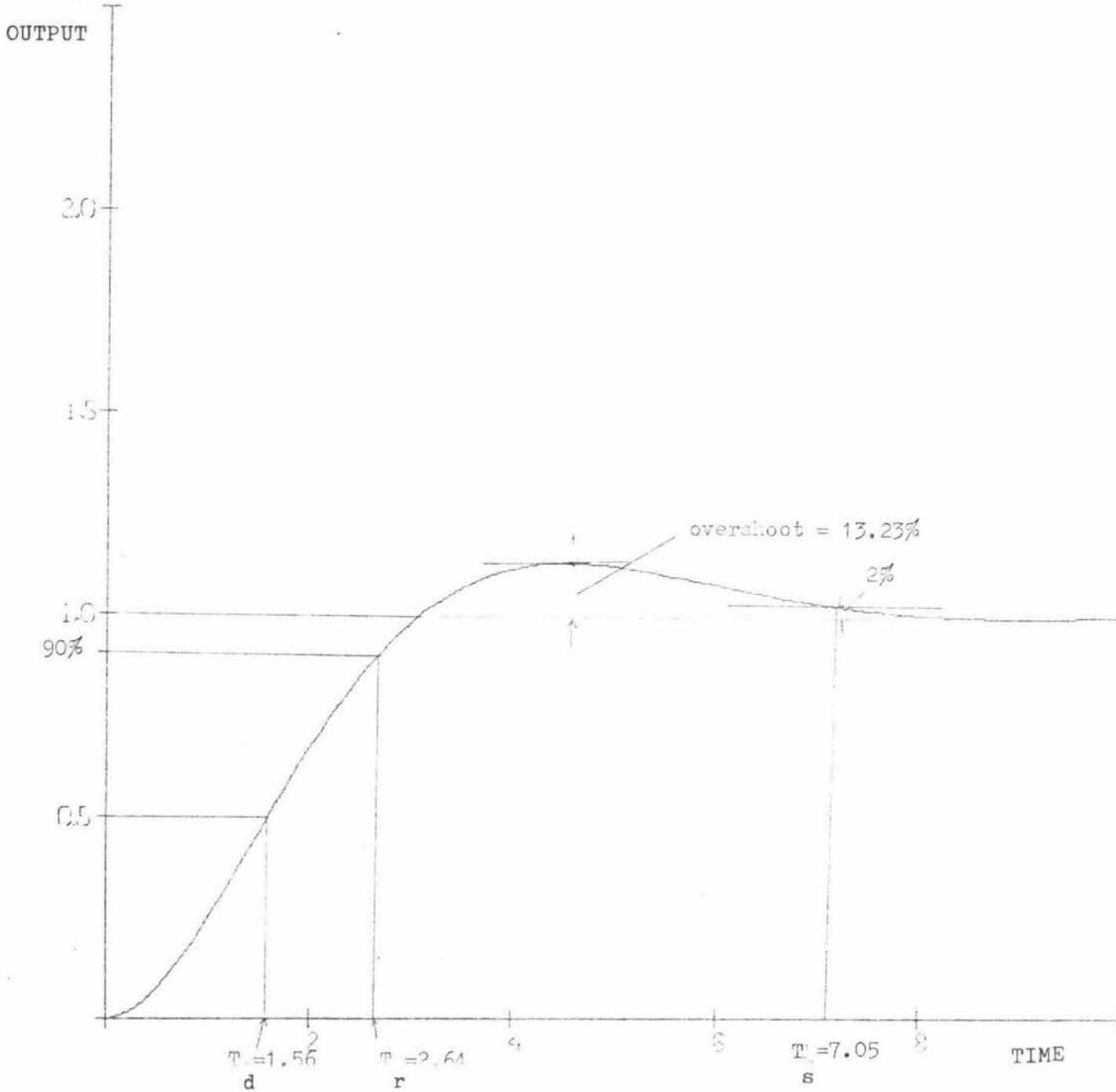


FIG. 14

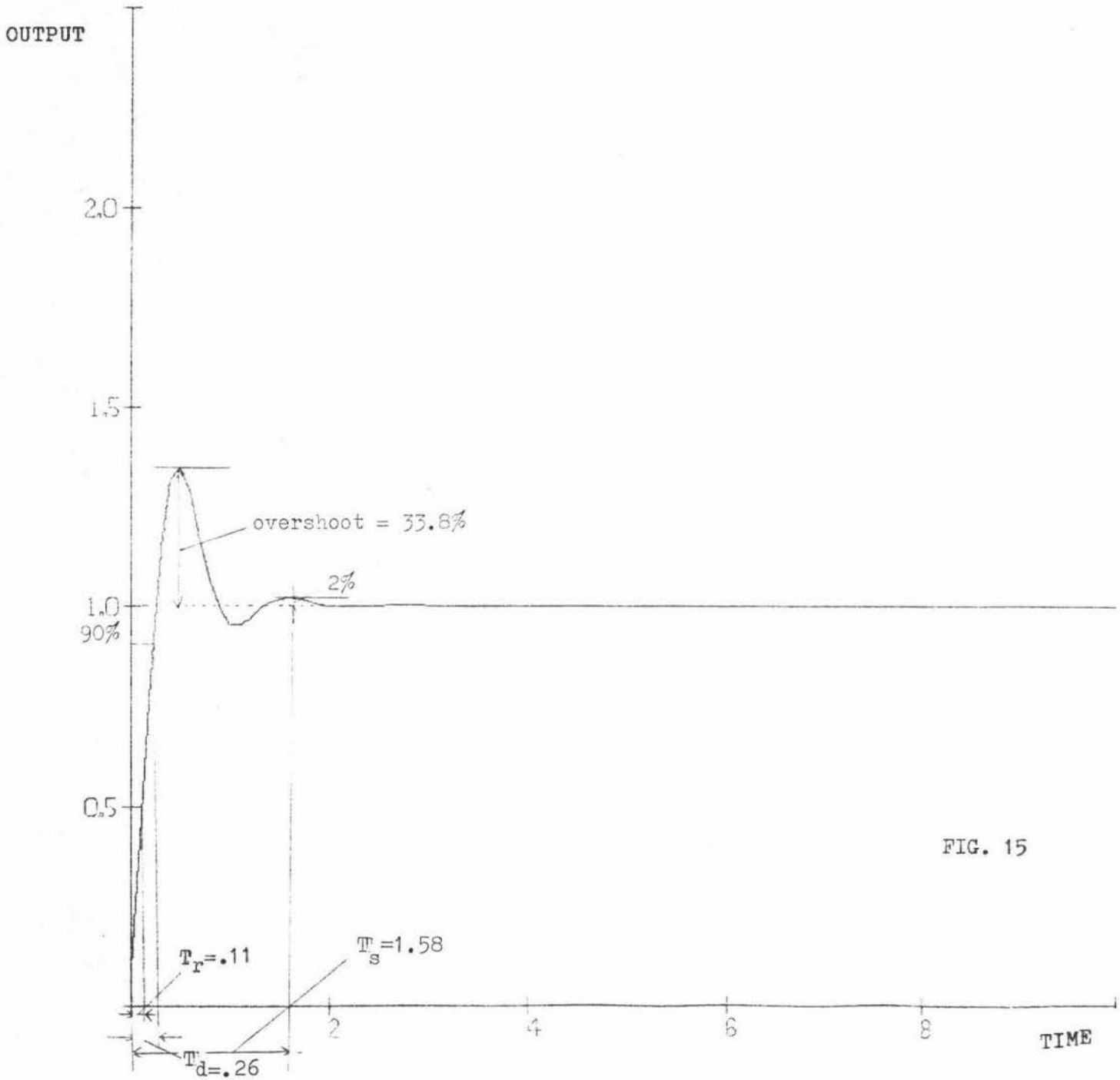

OUTPUT
2.0
1.5
1.0
90%
0.5
overshoot = 33.8%
2%
T_r=.11
T_s=1.58
T_d=.26
2
4
6
8
TIME

FIG. 15

loop system with different values of K are shown in Table 4.1.

Therefore the reduction of the O/P-loop gain alone may not provide sufficient alteration to produce a satisfactory performanc It is then necessary to redesign the system by incorporating additional compensation in order to improve the overall system behaviour.

The best possible design is a compromise between good stability and as high a loop gain as possible. A lead compensation network is added to the system with the O/P-loop gain K = 11, the transfer function of the lead compensator is

$$G_c(s) = \frac{1}{\beta}\left(\frac{1 + sT}{1 + \frac{sT}{\beta}}\right)$$

where β and T are chosen arbitrarily: β = 10 is a sensible practical value and T is chosen as .5. The compensated system is shown in Figure 4.2. System characteristics can be seen in Fig. 15, Fig. 16, Fig. 17, Fig. 18 and Fig. 19.

From Fig. 15 representing the closed-loop time response to a unit-step function input, the following quantities

- overshoot = 33.8%
- delay time = .26secs
- rise time = 11 secs
- settling time = 1.58secs

indicating a satisfactory system performance.

From Fig.16 representing the Nyquist diagram of the O/P-loop system, the locus does not enclose the point (-1,o) as ω is varied from .01 to 30rad/sec,

From Fig.17 representing the Root-locus of the O/P-loop system, the loci do not pass into the R.H half of the s-plane (i.e. there are no poles with +ve real parts) as K is increased from 0 to 11,

From Fig.18 representing the Nichols chart plot of the O/P-loo system, the gain and phase margins are found 13.7dbs and 38.1°.

Table 4.1

	Overshoot	Rise Time	Delay Time	Settling Time	State of Output
Case 1 K = 5	67.6%	.67 secs	.47 secs	$T_1 >> 10$ secs	- poor stability - more oscillation
Case 2 K = 3	51.4%	.94 secs	.58 secs	$10 < T_2 < T_1$	- better stability - less oscillation - slower response
Case 3 K = 1	20.5%	2 secs	1.23 secs	$T_3 < T_2$	- good stability - sluggish response
Case 4 K = .7	13.23%	2.64 secs	1.56 secs	$T_4 = 7.05$ secs	- good stability - more sluggish response

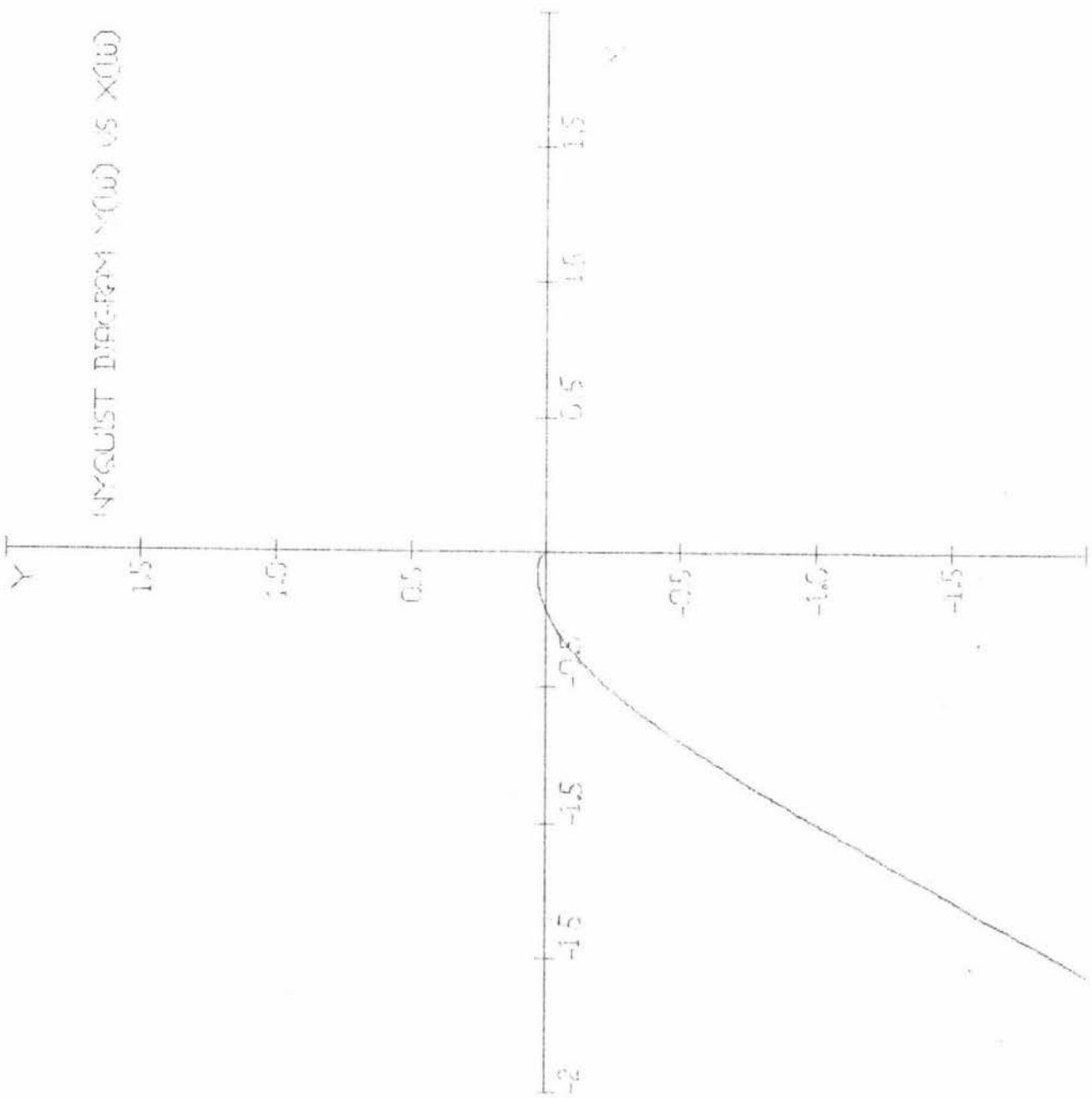


FIG. 16

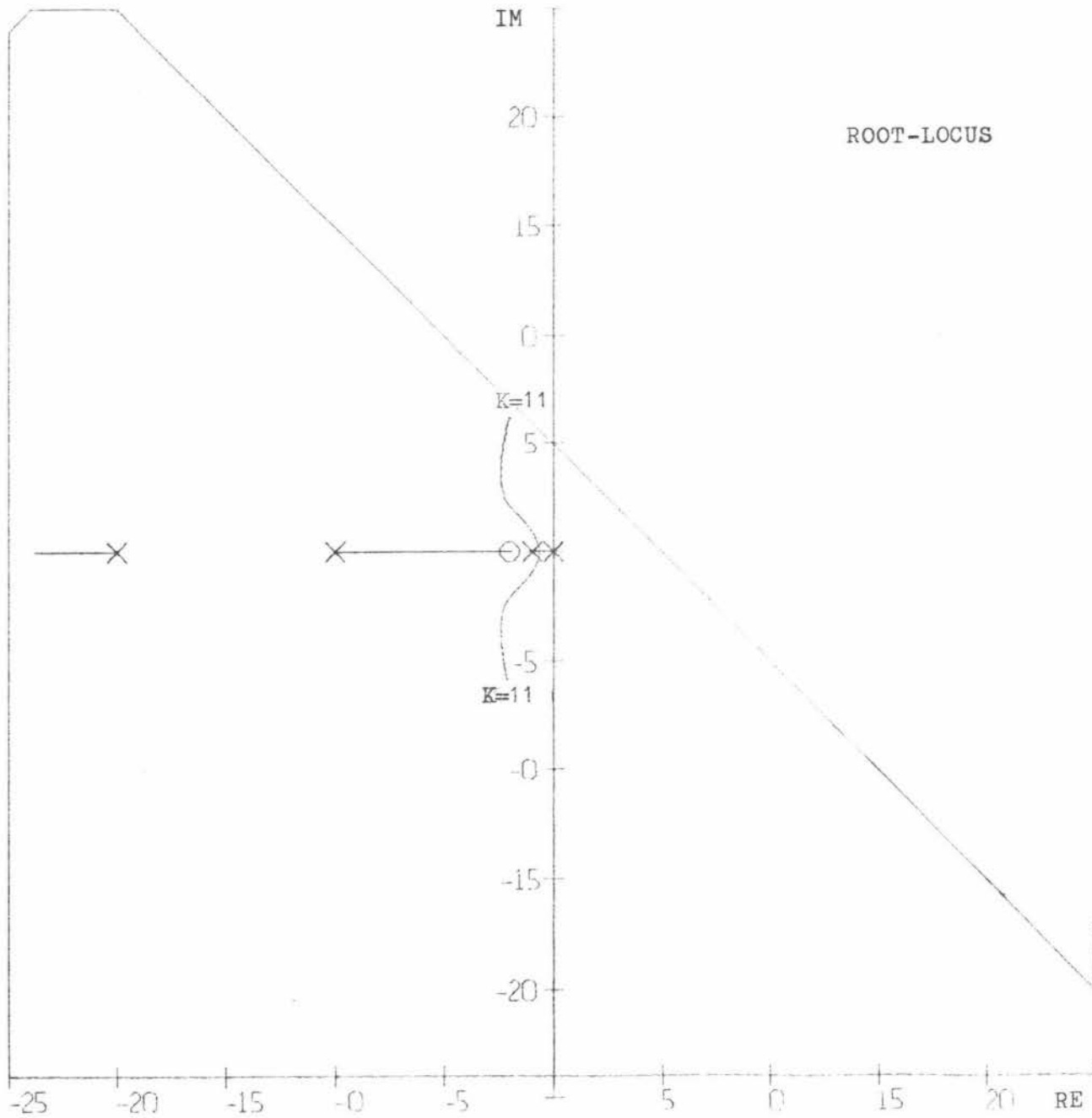


FIG. 17

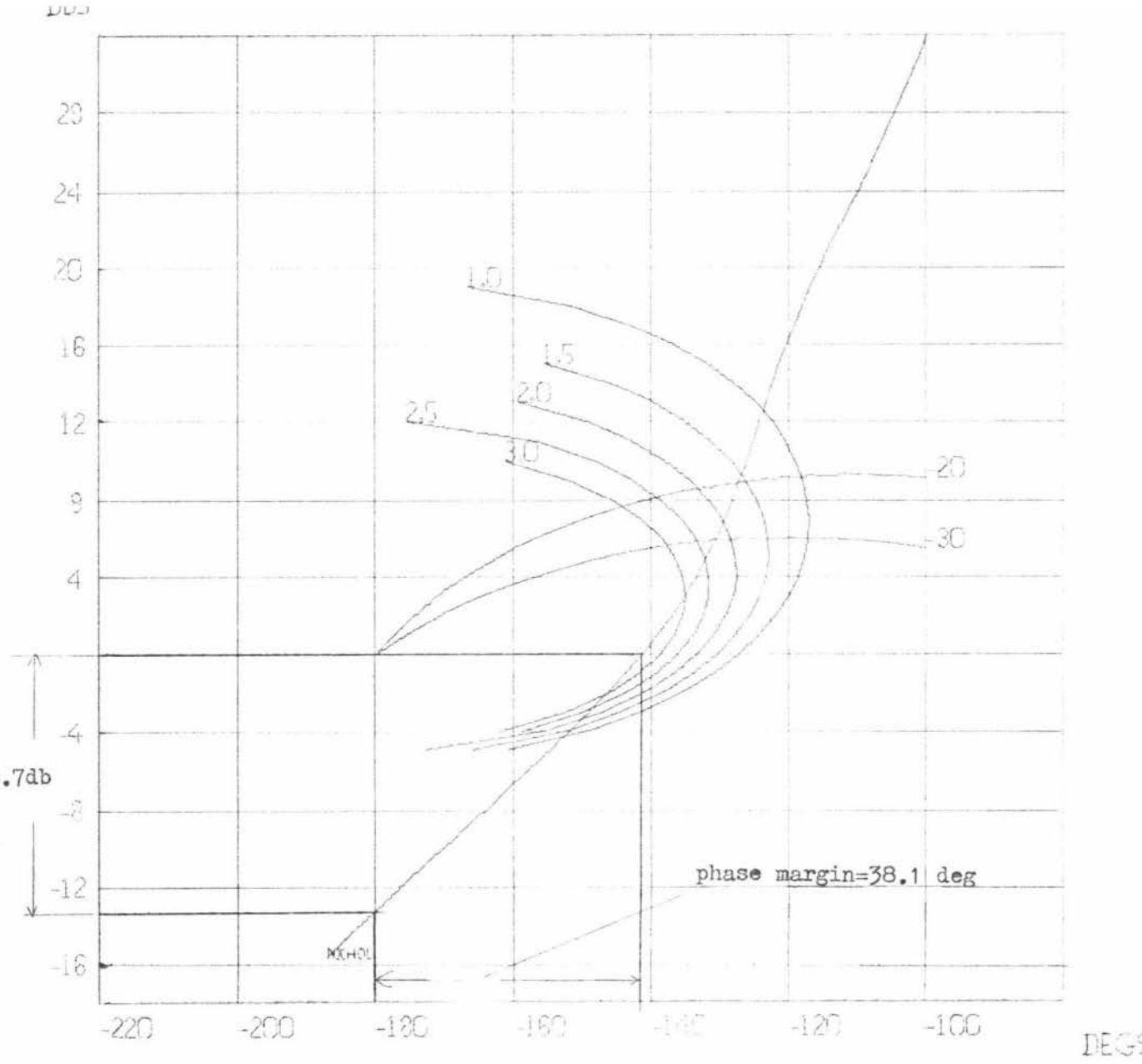

DBS
28
24
20
16
12
9
4
-4
-8
-12
-16
1.0
1.5
2.0
2.5
3.0
20
30
gain margin=13.7db
phase margin=38.1 deg
-220
-200
-180
-160
-140
-120
-100
DEGS

FIG. 18

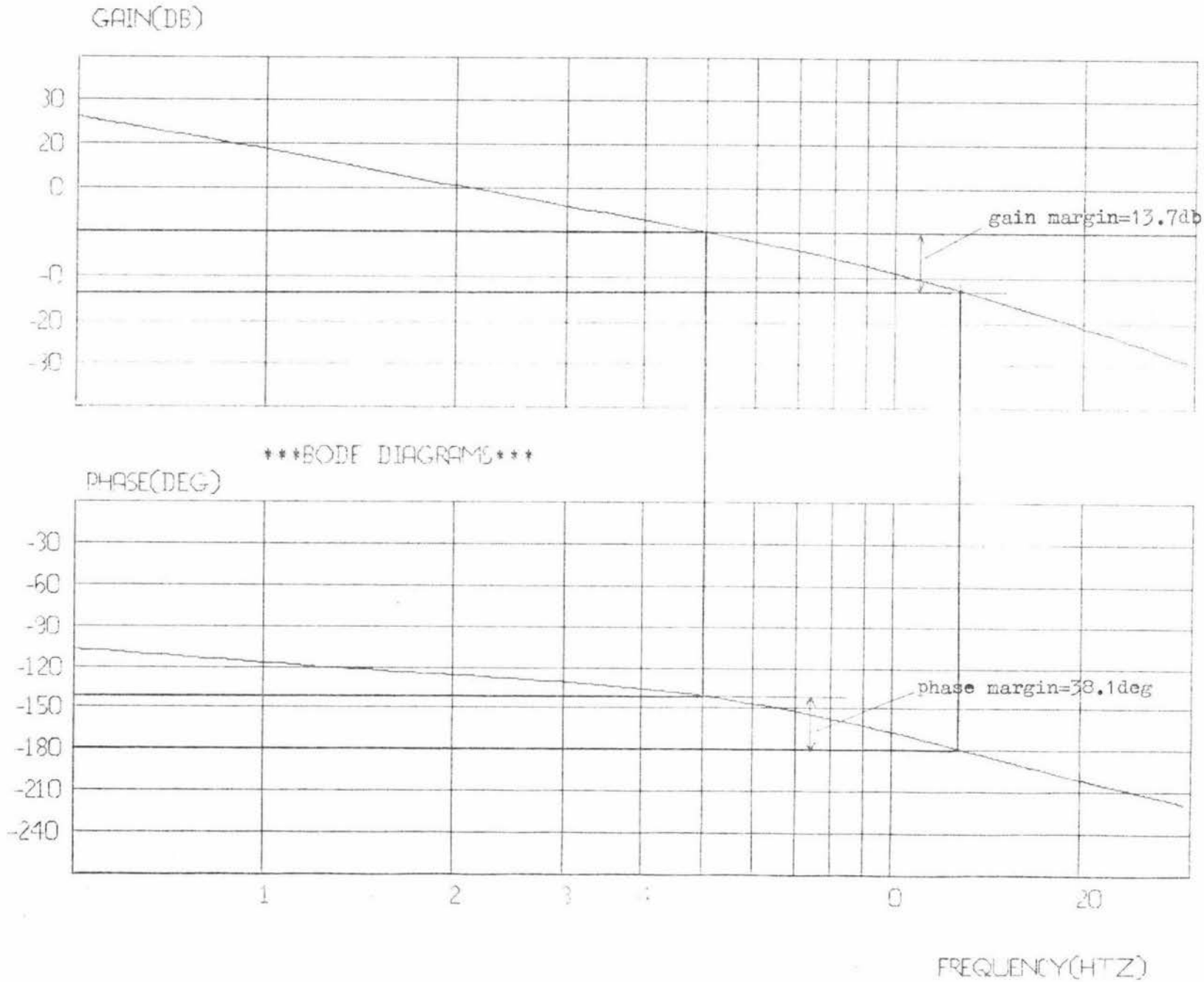

GAIN(DB)
30
20
0
-0
-20
-30
gain margin=13.7db
BODE DIAGRAMS
PHASE(DEG)
-30
-60
-90
-120
-150
-180
-210
-240
phase margin=38.1deg
1
2
3
0
20
FREQUENCY(HTZ)

FIG. 19

From Fig.19 representing the Bode diagrams of the O/P-loop system, gain and phase margins are found to be the same as calculated from Fig.18.

The above information justifies that the compensation network has yielded a satisfactory control system with fast response, no oscillation, negligible delay time, negligible rise time and virtually no offset.

Often, better results can be achieved by trial-and-error adjustment of T and β. Further phase lead can be introduced by using more than one network if necessary.

4.2 DESIGN EXAMPLE USING ROOT-LOCUS METHOD

Consider the control system shown in Fig. 4.3. The feed forward transfer function is

$$G(s) = \frac{4}{s(s + .5)}$$

The root-locus plot for this system is shown in Fig.20. The closed-loop transfer function becomes

$$\frac{C(s)}{R(s)} = \frac{4}{s^2 + .5\,s + 4}$$

$$= \frac{4}{(s + .25 + j\ 1.98)\ (s + .25 - j\ 1.98)}$$

The closed-loop poles are located at

$$s = -\ 0.25 \pm j\ 1.98$$

The damping ratio is .125, the undamped natural frequency is 2 rad/sec. It is desired to make the damping ratio equal to .5 and to increase the undamped natural frequency to 5 rad/sec. An appropriate compensation will be designed to meet these performance specifications.

Since the specifications are given in terms of time-domain quantities, the root-locus approach to design is used.

The closed-loop time response to a unit-step input, shown in Fig.22, indicates that the system has an undesirable transient-response characteristic and it therefore needs some form of compensation in order to produce a more satisfactory system which

will meet the performance specifications.

In the present example, the desired locations of the closed-loop poles are

$$s = -2.5 \pm j\,4.2$$

In some cases, after the root-loci of the original system have been obtained, the dominant closed-loop poles may be moved to the desired location by simple gain adjustment. This is, however, not the case for the present system. Therefore a lead compensator is inserted in the feed forward path.

A general procedure for determining the lead compensator is as follows:

First, find the sum of the angles at the desired location of one of the dominant closed-loop poles with the O/P-loop poles and zeros of the original system, and determine the necessary angle $\emptyset$ to be added so that the total sum of the angles is equal to $\pm 180^{o}(2K+1)$. The lead network must contribute this angle.(If the angle is quite large, then two or more lead networks may be needed.)

If the original system has the open-loop transfer function G(s), then the compensated system will have the open-loop transfer function

$$G_1(s) = \alpha \frac{Ts + 1}{\alpha Ts + 1} \cdot K_c\, G(s)$$

where the first term on the right-hand side corresponds to the lead network, the second term K_c is the gain of the amplifier, and the last term G(s) is the original open-loop transfer function. Note that there are many possible values for T that will yield the necessary angle contribution at the desired closed-loop poles.

The next step is to determine the locations of the pole and zero of the lead network, in other words the value of T. In choosing the value of T, a procedure is introduced to obtain the largest possible value for α so that the additional gain required for the amplifier is as small as possible. First, a horizontal line is drawn passing through point P, the desired location for

one of the dominant closed-loop poles. This is shown as line PA in Fig. 4.4. Also a line is drawn connecting point P and the origin. The angle between the lines PA and PO is disected as shown in Fig.4.4.

The intersection of PC and PD with the negative real axis give the necessary location for the pole and zero of the lead network. The compensator thus designed will make point P on the root-locus of the compensated system. The open-loop gain is determined by means of the magnitude condition.

In the present system, the angle of G(s) at the desired closed-loop pole is

$$\angle \left. \frac{4}{s(s + .5)} \right|_{s = -2.5 + j\,4.2} = 235^{o}$$

Thus for the root-locus to go through the desired closed-loop pole, the lead network must contribute $\phi = 55^{o}$ at this point. By following the foregoing procedure, the pole and zero of the lead network, as shown in Fig.20 are determined to be

pole at s = -9.4, zero at s = -2,6

The lead compensator consisting of the lead network and an amplifier has the transfer function $G_c(s) = \frac{(s + 2.6)}{(s + 9.4)} \cdot K_c$

Thus the open-loop transfer function of the compensated system becomes

$$G_c(s)\ G(s) = \frac{(s + 2.6)}{(s + 9.4)}\ K_c \cdot \frac{4}{s(s + .5)}$$

$$= \frac{K^1(s + 2.6)}{s(s + .5)\ (s + 9.4)} \quad \text{where } K^1 = 4\ K_c$$

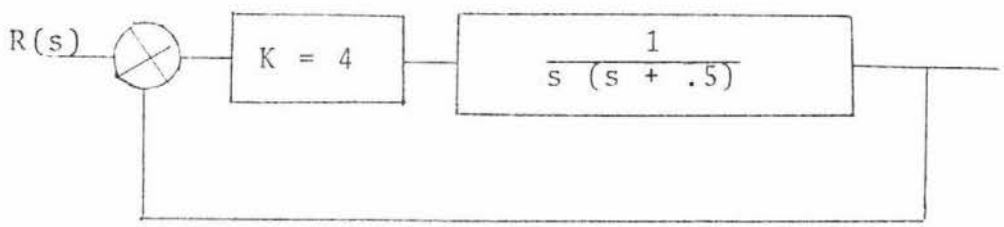


Fig. 4.3 Control system

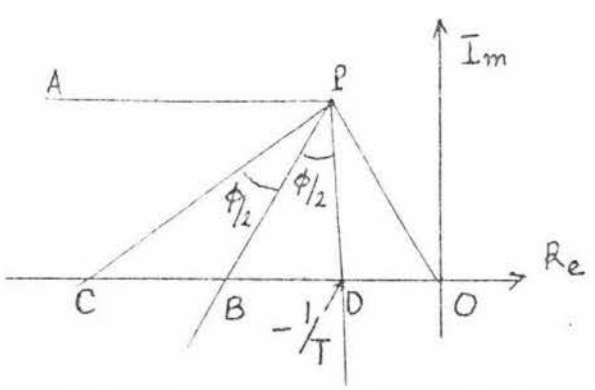


Fig. 4.4 Determination of the pole and zero of a lead network.

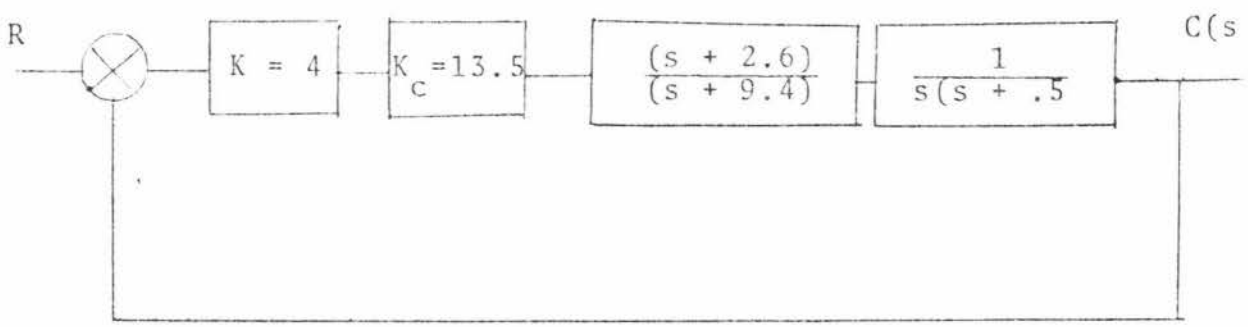


Fig. 4.5 The Compensated System.

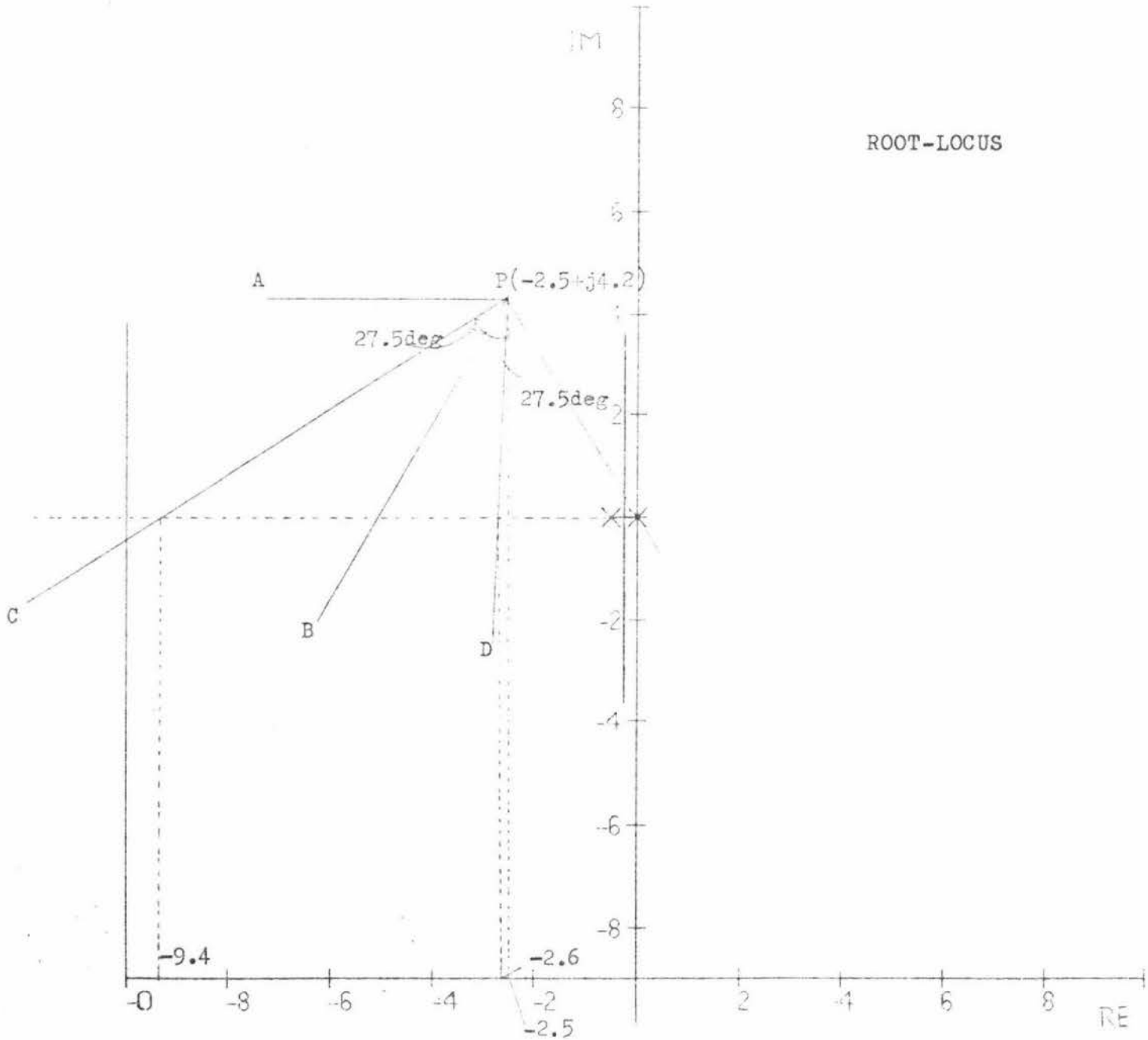
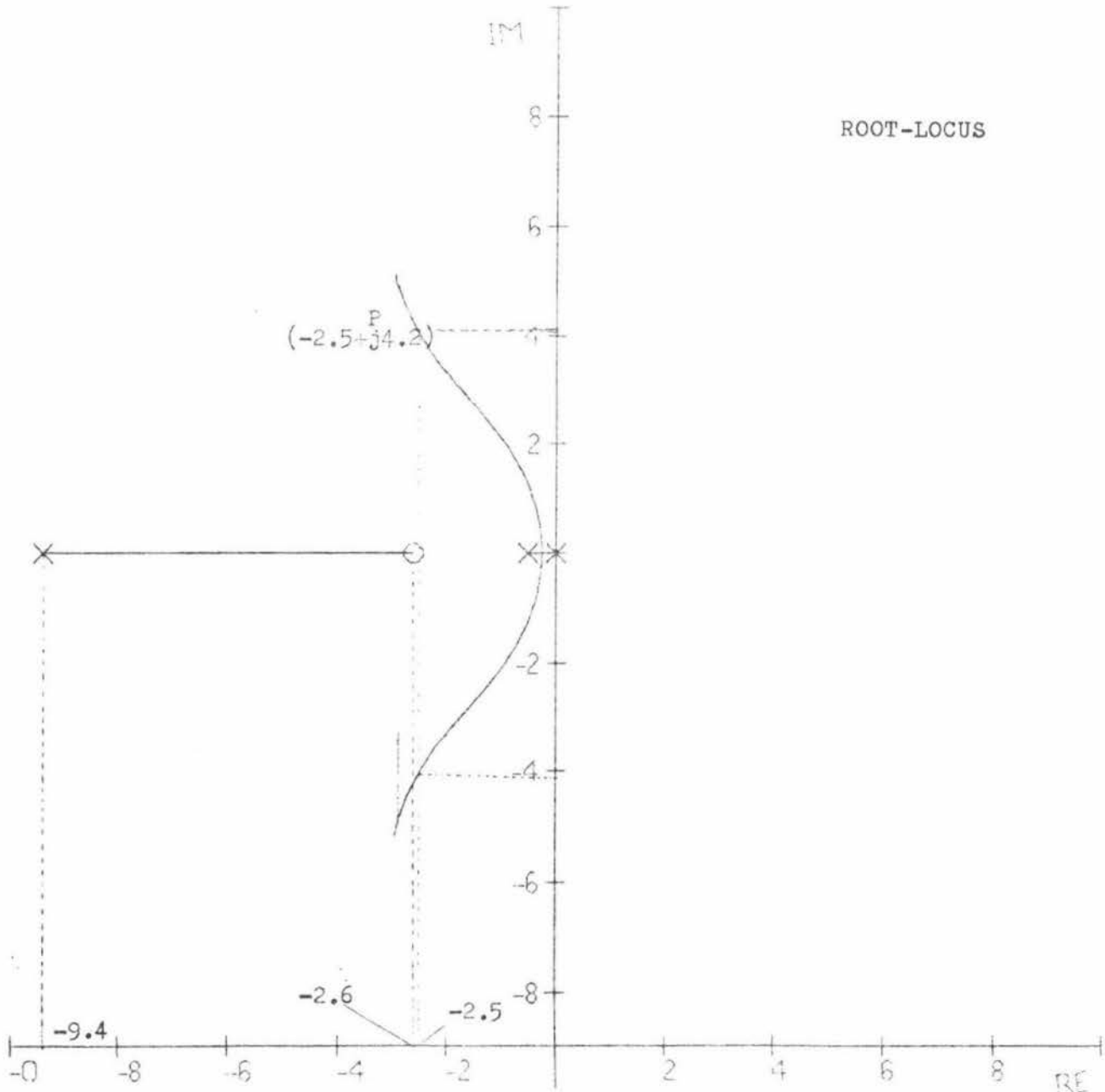

ROOT-LOCUS
IM
RE
8
6
2
-2
-4
-6
-8
-0
-8
-6
-4
-2
2
4
6
8
A
P(-2.5+j4.2)
27.5deg
27.5deg
B
C
D
-9.4
-2.6
-2.5

FIG.20



ROOT-LOCUS

FIG. 21

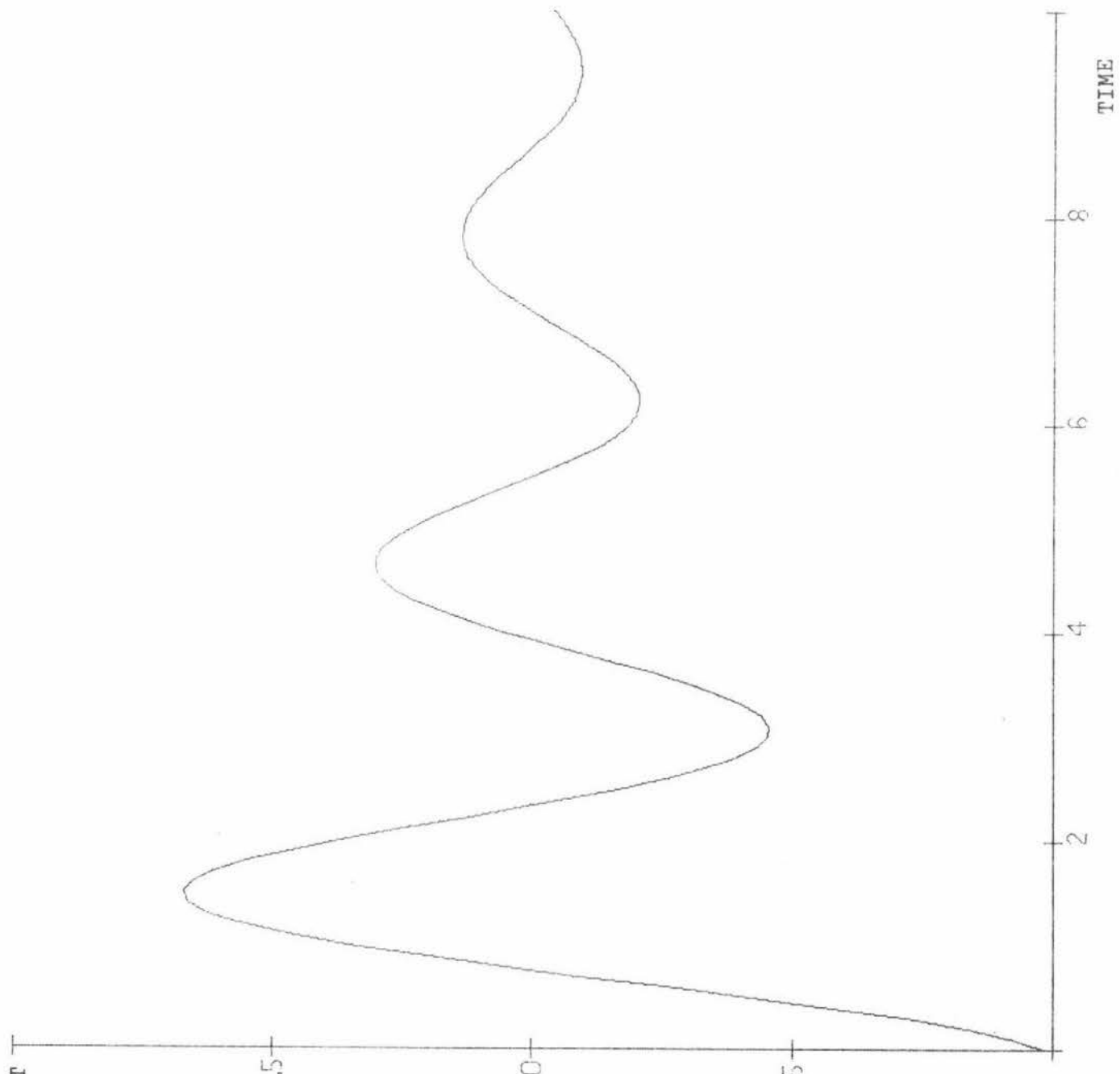


FIG. 22

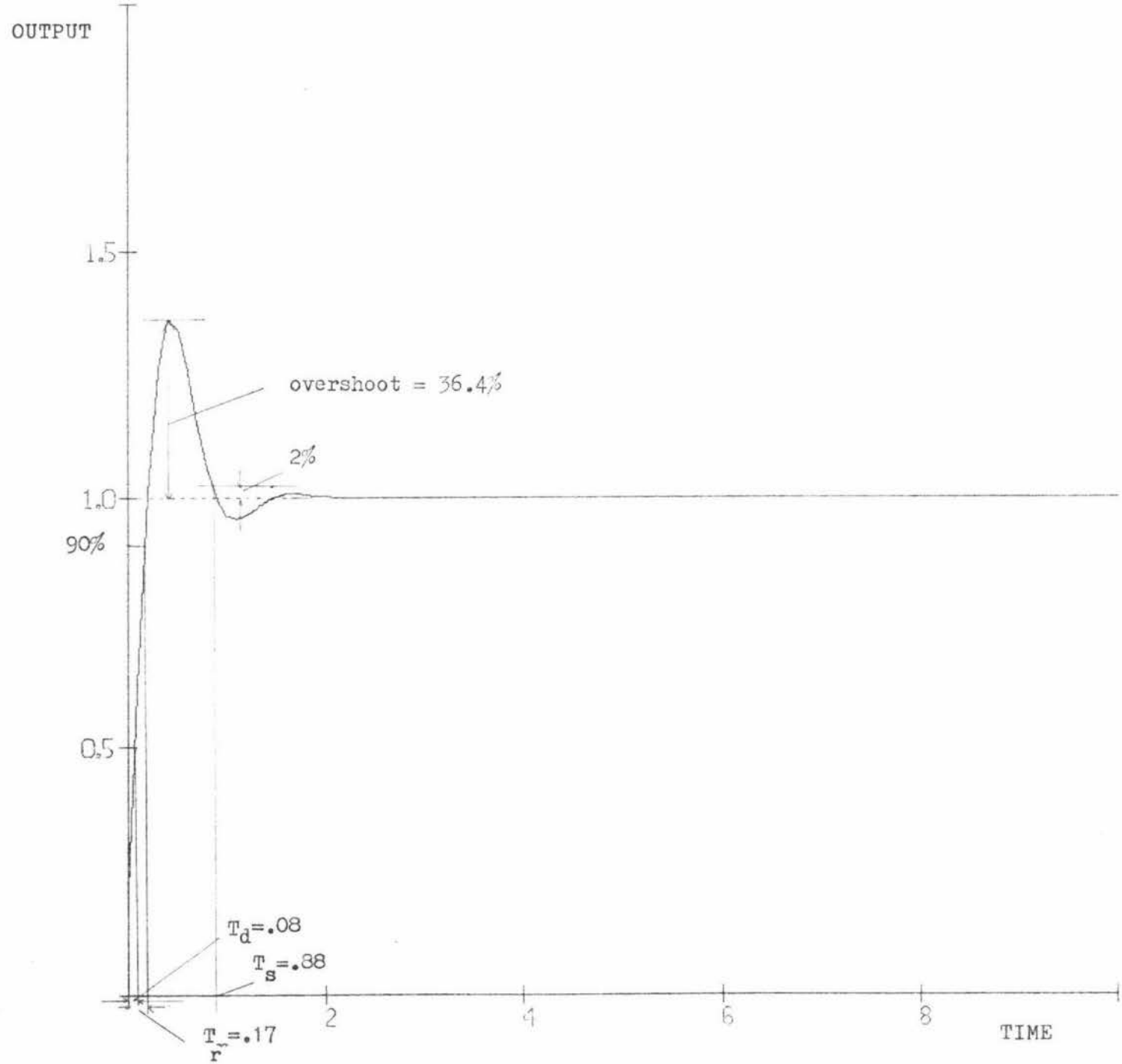


FIG. 23

The gain K is evaluated from the magnitude condition as follows

$$\left| \frac{K^1 (s + 2.6)}{s(s + .5)(s + 9.4)} \right|_{s = -2.5 + j\,4.2} = 1$$

or $K \# 54$

It follows that

$$G_c(s) \cdot G(s) = \frac{54 (s + 2.6)}{s(s + .5)(s + 9.4)}$$

The gain constant K_c of the amplifier is

$$K_c = \frac{K^1}{4} = 13.5$$

The compensated system is shown in Fig. 4.5. Fig. 21 shows that the root-locus of the compensated system passes through the desired closed-loop poles $s = -2.5 \pm j4.2$.

The time response of the closed-loop system to a unit-step input is shown in Fig.23 where the quantities measuring the performance of the system may be obtained, as follows:

- overshoot = 36.4%
- delay time = .008secs
- rise time = .17secs
- settling time = .88secs

It is concluded that the present design is satisfactory since is yields a very fast, almost immediate, response, negligible delay, rise and settling time and virtually no offset. The compensated system is shown in Fig. 4.5.

4.3 DESIGN EXAMPLE USING FREQUENCY-RESPONSE APPROACH

Consider the system shown in Fig. 4.6. The open-loop transfer function is

$$G(s) = \frac{4K}{s(s + 2)}$$

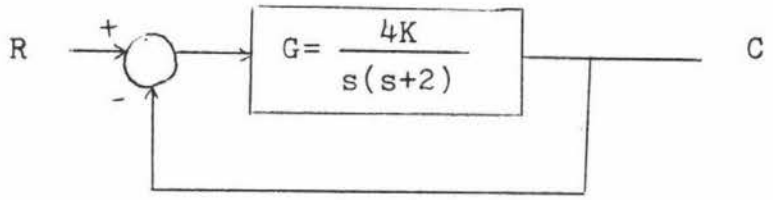


Fig. 4.6 Uncompensated system

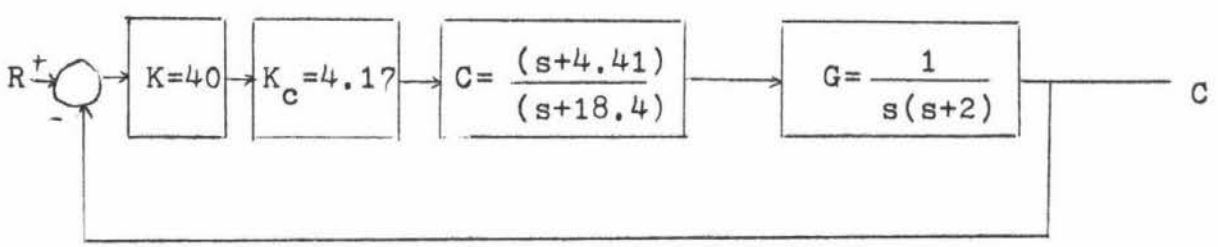


Fig. 4.7 Compensated system

It is desired to find a compensator for the system so that the static velocity error coefficient K_1 is $20sec^{-1}$, the phase margin is at least 50^o, and the gain margin is at least 10db. In the present example, the phase and gain margins have been specified. Bode diagrams will therefore be employed.

The first step in the design is to adjust the gain K to meet the steady-state performance specification or to provide the required static velocity error coefficient.

Since this coefficient is given as $20sec^{-1}$,

$$K_U = \lim_{s\to 0} sG(s) = \lim_{s\to 0} \frac{s4K}{s(s+2)} = 2K = 20$$

$$\text{or } K = 10$$

With K = 10, the system of Fig. 4.8 satisfies the stead-state requirement.

Next plot the diagrams of

$$G(j\omega) = \frac{40}{j\omega\,(j\omega + 2)}$$

Fig 24 shows the magnitude and phase angle curves of $G(j\omega)$. From this plot, the phase and gain margins of the system are found to 17^o and db respectively. A phase margin of 17^o implies that the system is quite oscillatory, as seen in Fig.26. Thus, satisfying the specification on the steady-state yields a poor transient-response performance. The specification calls for a phase margin of at least 50^o. It is thus found that the additional phase lead necessary to satisfy the relative stability requirement is 33^o. In order to achieve a phase margin of 50^o without decreasing the value of K, it is necessary to insert a suitable lead compensator into the system.

Noting that the addition of a lead compensator modifies the magnitude curve in the Bode diagrams, the gain crossover frequency will be shifted to the right. The increased phase lag of $G(j\omega)$,

due to this increase in the gain crossover frequency must be offset. Considering the shift of the gain crossover frequency, assume that ϕ_m, the maximum phase lead required, is approximately 38^o. (This means that 5^o has been added to compensate for the shift in the gain crossover frequency.)

Since

$$\sin \phi_m = \frac{1 - \alpha}{1 + \alpha}$$

$\phi_m = 38^o$ corresponds to $\alpha = .24$. Once the attenuation factor α has been determined on the basis of the required phase lead angle, the next step is to determine the corner frequencies $\omega = \frac{1}{T}$ & $\omega = \frac{1}{(\alpha T)}$ of the lead network. To do so, first note that the maximum phase lead angle ϕ_n occurs at the geometric mean of the two corner frequencies, or $\omega = \frac{1}{\sqrt{\alpha}.T}$.

The amount of the modification in the magnitude curve at $\omega = \frac{1}{\sqrt{\alpha}.T}$ is

$$\left| \frac{1 + j\omega\alpha T}{1 + j\omega\alpha T} \right|_{\omega = \frac{1}{\sqrt{\alpha}.T}} = \left| \frac{1 + j \frac{1}{\sqrt{\alpha}}}{1 + j \alpha \frac{1}{\sqrt{\alpha}}} \right| = \frac{1}{\sqrt{\alpha}}$$

Note that

$$\frac{1}{\sqrt{\alpha}} = \frac{1}{\sqrt{.24}} = \frac{1}{.49} = 6.2 \text{ db}$$

and $G(j\omega) = 6.2$db corresponds to $\omega = 9$ rad/sec, as seen in Fig.24. This frequency is selected to be the new gain crossover frequency ω_c. Noting that this frequency corresponds to $\frac{1}{\sqrt{\alpha}.T}$, or $\omega_c = \frac{1}{\sqrt{\alpha}.T}$,

$$\frac{1}{T} = \sqrt{\alpha}\, \omega_c = 4.41$$

and

$$\frac{1}{\alpha T} = \frac{\omega_c}{\sqrt{\alpha}} = 18.4$$

are obtained.

The lead network thus determined is

$$\frac{s + 4.41}{s + 18.4}$$

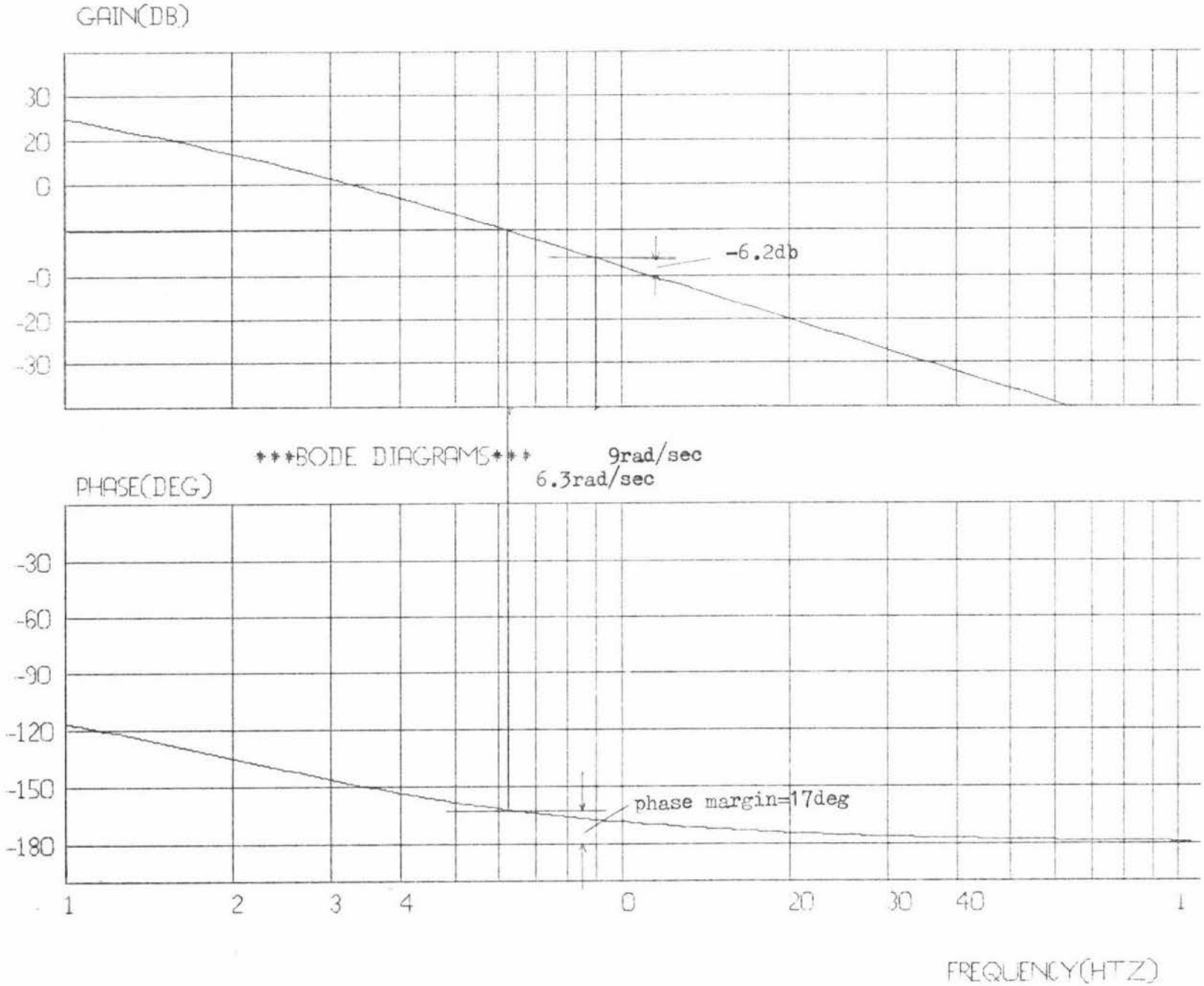

GAIN(DB)
30
20
0
-0
-20
-30
-6.2db
BODE DIAGRAMS
9rad/sec
6.3rad/sec
PHASE(DEG)
-30
-60
-90
-120
-150
-180
phase margin=17deg
1
2
3
4
0
20
30
40
1
FREQUENCY(HTZ)

FIG. 24

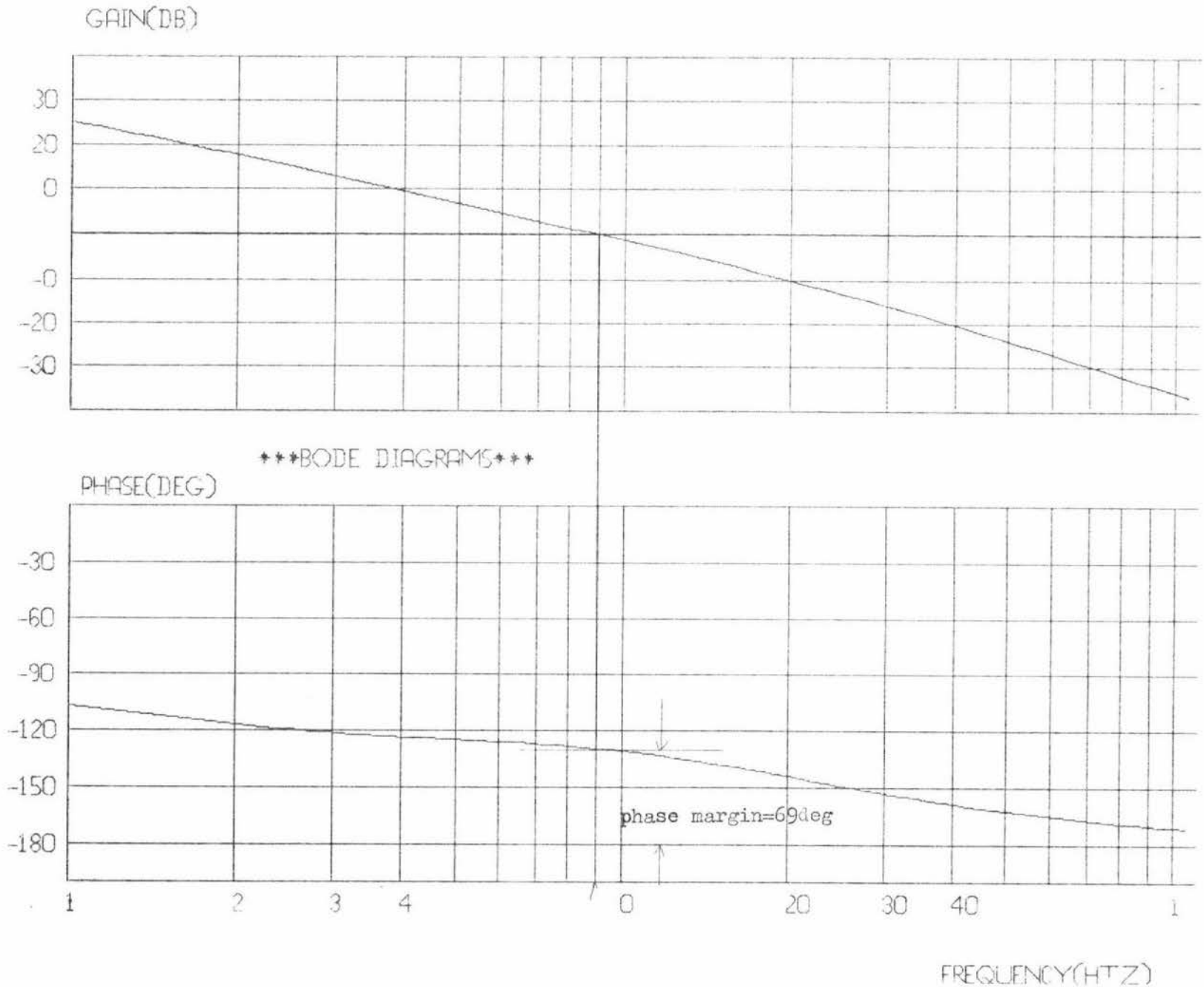

GAIN(DB)
30
20
0
-0
-20
-30
BODE DIAGRAMS
PHASE(DEG)
-30
-60
-90
-120
-150
-180
phase margin=69deg
1
2
3
4
0
20
30
40
1
FREQUENCY(HTZ)

FIG. 25

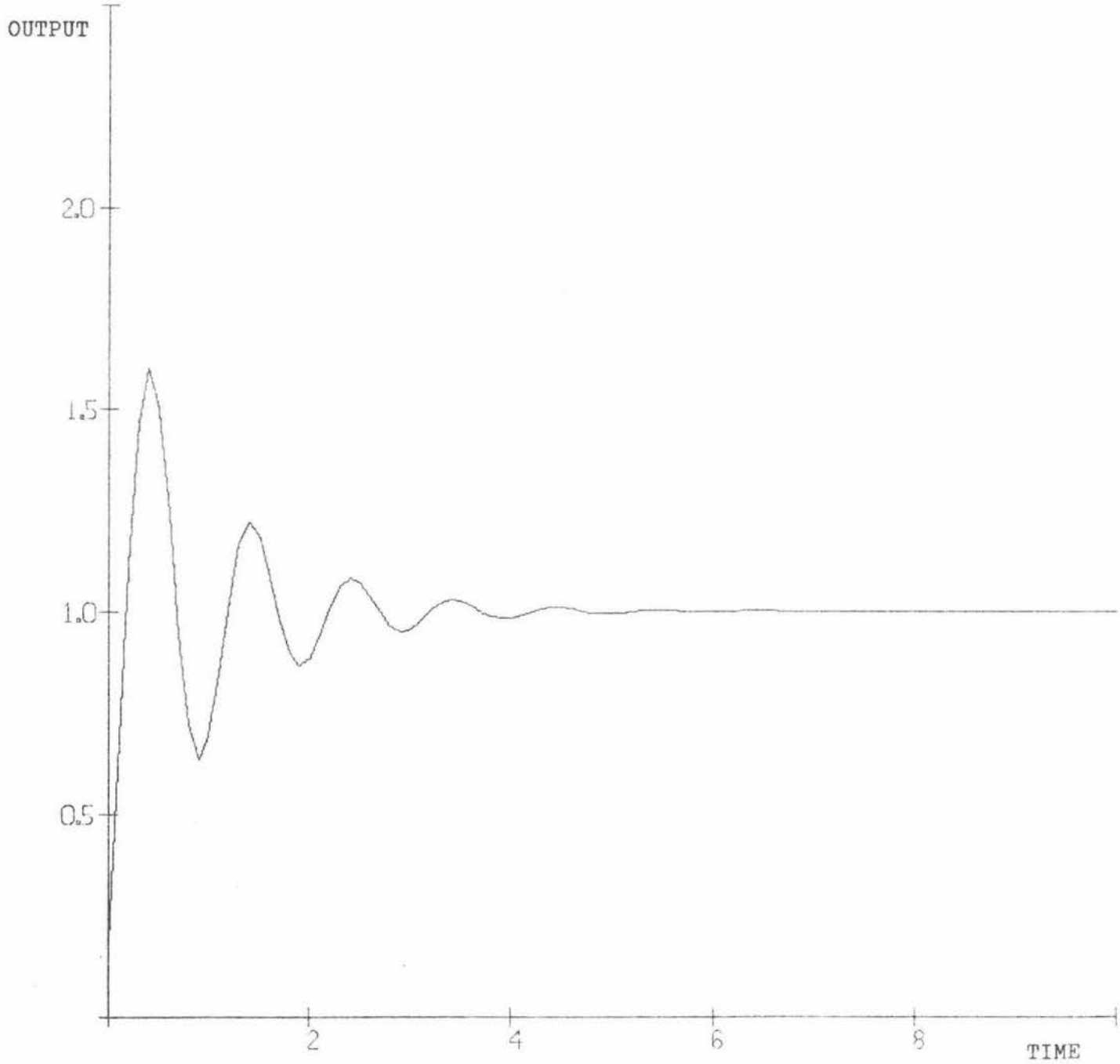


FIG. 26

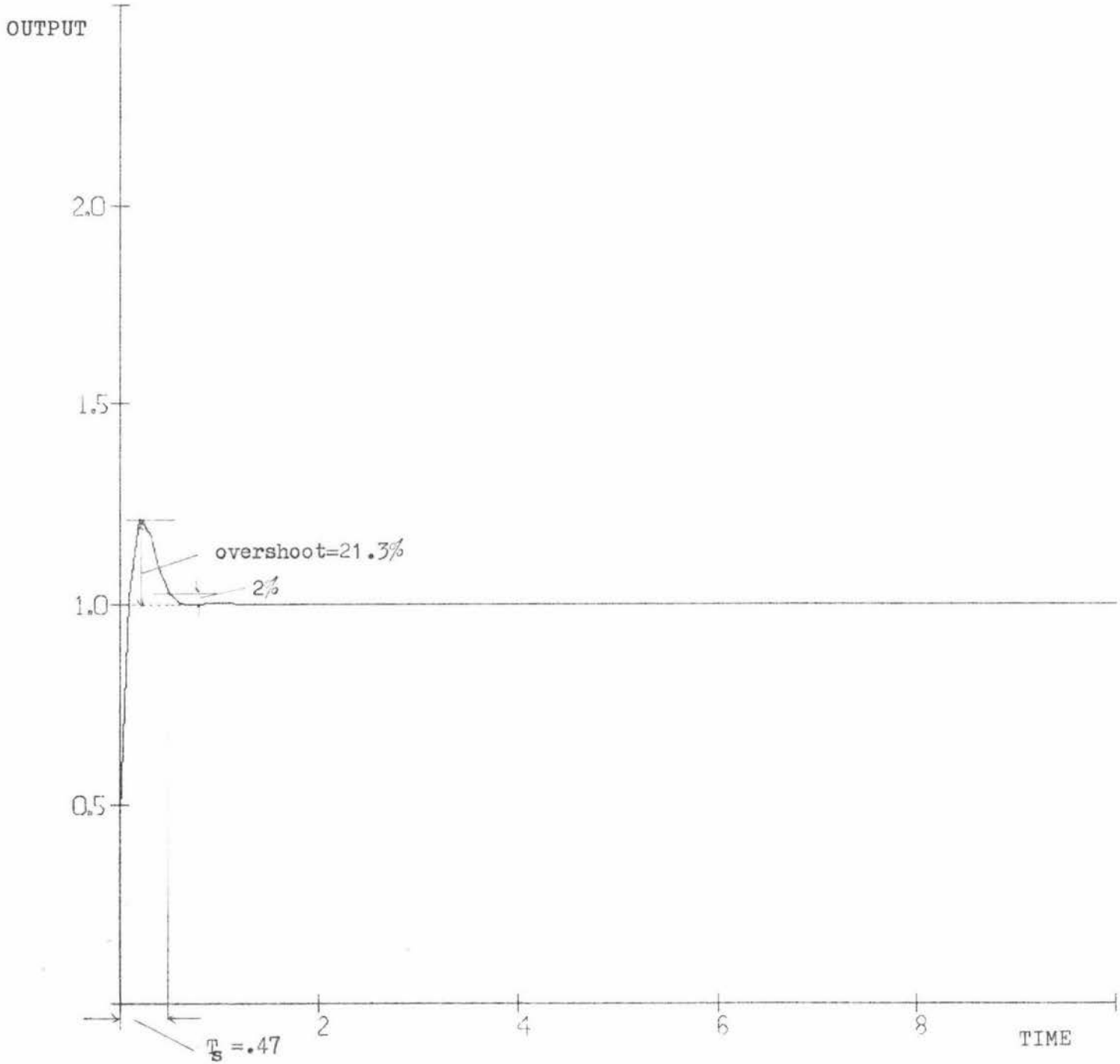


FIG.27

To compensate for the attenuation due to the lead network, the amplifier gain is increased by a factor of $\frac{1}{.24} = 4.17$. (If this were not done, the required static velocity error coefficient could not be realized.) Then the transfer function of the compensator which consists of the lead network and the amplifier becomes

$$G_c(s) = (4.17)\ \frac{s + 4.41}{s + 18.4}$$

The compensated system has the following open-loop transfer functic

$$G_c(s)\ G(s) = (4.17)\left(\frac{(s + 4.41}{(s + 18.4)}\right)\left(\frac{40}{s(s + 2)}\right)$$

Fig.25 shows the magnitude and phase-angle curves for the compensated system. The lead compensator causes the gain crossover frequency to increase from 6.3rad/s (Fig.24) to 9 rad/sec. The increase in this frequency means an increase in bandwidth. This implies an increase in the speed of response which can be seen from Fig.27 showing the compensated system closed-loop time response to a unit-step input.

With an overshoot of 21.3%, zero delay time, negligible rise time and .47secs settling time, the system can be seen to have a much more satisfactory performance in comparison with the original system, shown in graph 26.

From Fig.25, the phase and gain margins are seen to be 69^{o} and + ∞db, respectively. The compensated system shown in Fig.4.7 therefore meets both the steady-state and the relative-stability requirements.

4.4 DISCUSSION

The design examples in this chapter illustrate the use of the computer-aided system design techniques. The system software has been developed which, having specified the control system configurations G(s) and H(s), permits varied

choice of $G_c(s)$ using trial-and-error analysis approaches.

The results from this chapter show how the typical design techniques described in chapter 2 are implemented in the control system design with the assistance of a digital computer. Different graphs representing the system performance in either time- or frequency-domain are exactly generated and analysed to obtain information essential for the compensation of the system deficiencies.

The trial-and-error method, although it is seen less direct and involving many trials, is useful in designing systems with no set performance specifications.

The root-locus approach is seen very convenient in designing systems with performance specifications given in terms of time-domain quantities, such as damping ratio, natural frequency, etc.

The Bode technique is seen very effective in producing systems with performance specifications given in terms of frequency domain quantities, such as gain and phase, bandwidth, etc...

Whatever design technique may be used, it is important to note that a unique system will not be yielded. In fact many systems may satisfy the given specifications. An optimal choice among the many possibilities may be made from such considerations as projected overall performance, cost, space and weight.

The computer system developed contributes a major part in the control system design. It helps the designer to avoid much of the numerical drudgery necessary for the checking of the system performance after each adjustment of the compensator parameters. The information about the overall-system performance is manually calculated from the computer-generated graphs. The present computer system therefore may be further developed to provide complete knowledge of the system performance, i.e. the gain and phase margins, overshoot, delay time, rise time, etc... may be obtained directly from the computer. Automatic design of the

compensation networks may also be incorporated to enhance the facility.

-ooOoo-

CHAPTER FIVE

CONCLUSION AND SUGGESTIONS FOR FURTHER STUDY

The software for computer aided control system design facilitates the design via analysis or trial-and-error approaches. The control designer, at the Tektronix console, is able to specify certain designs and evaluate their effectiveness in terms of the various graphical analyses produced by the computer and displayed on the screen of the Tektronix. If performance specifications are given in terms of time-domain quantities, such as overshoot, rise time, etc..., the root-locus approach to design is very powerful to use. If performance specifications are given in terms of frequency-domain quantities, such as gain and phase margins, the Bode method is the most popular to use. However, if a certain value of the resonant peak is required, the Nichols chart technique is more convenient to use. The Nyquist technique provides rapid evaluation of stability of the control system, it is therefore used to supplement other methods. The time-response method is normally used for checking to see whether or not the designed system satisfies the requirements in the time-domain, though a satisfactory performance may be reached using this method.

Hence, for complete quantitative information of a system, both the differential-equation and the transfer-function representations are desirable. The transfer-function for ease in design, the differential equation for exact performance data on several important characteristics of the system.

The software development exercise has not yet been completed because of the limited project time. Many other features can be incorporated to further facilitate the design procedures and to make them more automatic. (Section 3.7.)

Although the classical approaches to control system design

have many outstanding advantages, they have numerous shortcomings. Empiricism, trial-and-error design, and a lack of a fundamental theory combine to make conventional control more an art than a science.[5]

The root-locus and frequency-response methods, which essentially consist of the loop-gain adjustment and the design of appropriate compensators, are quite useful but are limited to idealized and relatively simple control systems, such as single-input-single-output linear time-invariant systems. Such classical design approaches suffer from severe limitations and difficulties when applied to the design of multiple-input-multiple-output and time-varying systems. In other words, for the exact requirements of the more complex automatic control problems in modern technology, classical methods are entirely inadequate. This is where "modern control" enters the picture. Modern control theory utilizes time-domain system description (state variables) rather than transfer-function methods. Extensive use is made of digital computing. The methods of modern control have been developed in many instances by applied and pure mathematics, with the result that much of the written presentation is very formal and quite inaccessible to most control engineers. By applying modern control theory, the designer is able to start from a performance index, together with constraints imposed on the system, and to proceed to design a stable system by a completely analytical procedure. The advantage of design based upon such control theory is that it enables the designer to produce a control system which is optional with respect to the performance index considered.

In addition, the concept of adaptive control systems has also been introduced to cope with the problem where the plant is normally exposed to varying environments so that plant

parameters change from time to time.

However, it is important to note that if performance specifications are given in terms of the time-domain or frequency-domain quantities considered in this thesis, modern design techniques are not convenient, and the classical techniques are useful for the control system design.

-ooOoo-

REFERENCES

1. GORDON BULL: *Computational Methods and Algol*. George G. Harrap Co. Ltd., London 1966.

2. MARTIN HEALEY: *Principles of Automatic Control, 3rd ed.*, The English University Press Ltd., 1975.

3. OGATA KATSUHIKO: *Modern Control Engineering*. Prentice-Hall, 1970.

4. HALE, F.J.: *Introduction to Control Systems analysis and Design*. Prentice-Hall, 1973.

5. DISTEFANO III et al.: *Feedback and Control Systems*, McGraw-Hill, 1967.

6. AUSLANDER, D.M.: *Introducing Systems and Control*. McGraw-Hill, 1974.

7. KOPPEL, LOWELL, B.: *Introduction to Control Theory with Application to Process Control*. Prentice-Hall, 1968.

8. MASSEY COMPUTER UNIT: *Massey Plotter Package*. Computer User Note Number 26, 1975.

9. THALER, G.J. and BROWN, R.G.: *Analysis and Design of Feedback Control Systems, 2nd ed.*, McGraw-Hill, 1960.

10. ERNEST F. JOHNSON: *Automatic Process Control*. McGraw-Hill Chemical Engineering Series, 1967.

11. BURROUGHS CORPORATION: *Burroughs B 6700/B 7700 Algol Language Reference Manual*, 1974.

12. BURROUGHS CORPORATION: *Burroughs Large Systems Numerals Numerical Analysis Program Library*, 1974.

13. ELGERD, O.I.: *Control System Theory*. McGraw-Hill, 1968.

14. MASSEY COMPUTER UNIT: *Guide to B 6700 C and B*. Computer User Note Number 37, 1975.

-ooOoo-

ACKNOWLEDGEMENTS

Grateful acknowledgement for guidance and assistance is made to my supervisor, Dr D.J. Sandoz, Department of Industrial Management and Engineering, Massey University.

The help given by the Massey Computer Unit Staff is greatly appreciated. Special thanks are due to Dr E.A. Drawneek and Mr N. James for their assistance in de-bugging and running programmes.

APPENDIX A

ERROR COEFFICIENTS

2.3.1. *Static error coefficients*

i. Steady-state errors

Consider system shown in Fig. 2.13. The closed-loop T.F. is

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1 + G(s)H(s)}$$

The transfer-function between the actuating error signal and the input signal r(f) is

$$\frac{E(s)}{R(s)} = 1 - \frac{C(s)H(s)}{R(s)} = \frac{1}{1 + G(s)H(s)}$$

where the actuating error e(t) if the difference between the input signal and the feedback signal. The final value theorem provides a convenient way to find the steady-state performance of a stable system. Since E(s) is

$$E(s) = \frac{1}{1 + G(s)H(s)} R(s)$$

The steady-state actuating error is

$$e_{s_s} = \lim_{t \to \alpha} e(f) = \lim_{s \to o} \frac{s\ R(s)}{1 + G(s)H(s)}$$

The static error coefficients defined in the following are figures of merit of control systems.

ii. Static position error coefficient k_p

The steady-state error of the system for a unit-step input is

$$e_{s_s} = \lim_{s \to o} \frac{s}{1 + G(s)H(s)} \frac{1}{s}$$

$$= \frac{1}{1 + G(o)H(o)}$$

The static position error coefficient k_p is defined by

$$k_p = \lim_{s \to o} G(s)H(s) = G(o)H(o)$$

Thus, the steady-state error in terms of the static position error coefficient k_p is given by

$$e_{s_s} = \frac{1}{1 + k_p}$$

iii. Static velocity error coefficient k_v

The steady-state error of the system with a unit-ramp input is given by

$$e_{s_s} = \lim_{s \to o} \frac{s}{1 + G(s)H(s)} \frac{1}{s^2}$$

$$= \lim_{s \to o} \frac{1}{s\, G(s)H(s)}$$

The static velocity error coefficient k_v is defined by

$$k_v = \lim_{s \to o} sG(s)H(s)$$

Thus the steady-state error is

$$e_{s_s} = \frac{1}{k_v}$$

iv. Static acceleration error coefficient k_a

The steady-state error of the system with a unit-parabol input (acceleration input) which is defined by

$$r(t) = \frac{t^2}{2} \quad \text{for } t \geq o$$

$$= o \quad \text{for } t < o$$

is given by $e_{s_s} = \lim_{s \to o} \frac{s}{1 + G(s)H(s)} \frac{1}{s^3}$

$$= \lim_{s \to o} \frac{1}{s^2 G(s)H(s)}$$

The static acceleration error coefficient k_c is defined by the equation

$$k_a = \lim_{s \to o} s^2 G(s)H(s)$$

The steady-state error is then

$$e_{s_s} = \frac{1}{k_a}$$

The error coefficients k_p, k_u, and k_a describe the ability of a system to reduce or eliminate steady-state error. Therefore they are indicative of the steady-state performance. The higher the coefficients, the smaller the steady-state error. Therefore, it is generally desirable to increase the error coefficients, whil maintaining the transient response within an acceptable range.

2.3.2 *Dynamic error coefficients*

The dynamic errors provide some information about how the error varies with time. We shall now introduce dynamic error coefficients to describe the dynamic error and we shall limit our systems to unity-feedback ones. By dividing the numerator polynomial of E(s)/R(s) by its denominator polynomial, E(s)/R(s) can be expanded into a series in ascending powers of s as follows $\frac{E(s)}{R(s)} = \frac{1}{1 + G(s)} = \frac{1}{k_1} + \frac{1}{k_2} s + \frac{1}{k_3} s^2 + \ldots\ldots$

The coefficients k_1, k_2, k_3, ... are defined to be the dynamic error coefficients. Namely,

k_1 = dynamic position error coefficient

k_2 = dynamic velocity error coefficient

k_3 = dynamic acceleration error coefficient

An advantage of the dynamic error coefficients becomes clear when E(s) is written on the following form:

$$E(s) = \frac{1}{k_1} R(s) + \frac{1}{k_2} sR(s) + \frac{1}{k_3} s^2R(s) + \ldots\ldots\ldots$$

The region of convergence of this series is the neighbourhood of s = o. This corresponds to t = in the time domain. The corresponding time solution or the steady-state error is given, assuming all initial conditions are zero and neglecting impulses at t = o, as follows

$$\lim_{t-\alpha} e(t) = \lim_{t-\alpha} \left[\frac{1}{k_1} r(t) + \frac{1}{k_2} \dot{r}(t) + \frac{1}{k_3} \ddot{r}(t) + \ldots\ldots\right]$$

The steady-state error due to the input function and its derivatives can thus be given in terms of the dynamic error coefficients.

-ooOoo-

APPENDIX B

PERFORMANCES SPECIFICATIONS

1. *Frequency-domain specifications*

Frequency-domain specifications are usually stated in the following terms:

a. gain-margin GM, a measure of relative stability, is defined as the log modulus (db) of the system when the phase shift is 180^{o}, denoted positive for negative db.
On the Nyquist diagram, the gain margin is given directly by $-20\log_{10}M$ where M is the value of G GH(jω) when the curve cuts the negative axis.

b. phase-margin PM, a measure of relative stability, is defined as 180^{o} minus the phase angle at the frequency where the gain is unity (odb).
On the Nyquist diagram, the phase margin is found by drawing a circle of unity radius about thr origin and drawing the radius from the centre to the point where the curve GH(jω) cuts the unit circle,
The phase margin is then the angle between this radius and the negative X(ω) axis, positive below the axis and negative above. Gain and phase margins Bode diagrams and Nyquist diagrams are shown in Fig, B-1.

c. bandwidth BW, a measure of the speed of response of a system, is defined as that range of frequencies over which the magnitude ratio does not differ by more than -3db from its value at a specified frequency. For many feedback control systems this frequency is zero,

d. resonant peak M_p, a measure of relative stability, is the maximum value of the magnitude of the closed-loop frequency

response.

e. resonant frequency ω_p, is the frequency at which M_p occurs. Bandwidth, resonant peak and resonant frequency are illustrated in Fig. B.2.

2. *Time-response specifications*

Time-response specifications are usually stated in terms of the unit-step function response. Typical specifications are:

a. Overshoot is the maximum difference between the transient and steady-state solutions for a unit-step function input. It is a measure of relative stability and is often represented as a percentage of the final value of the output. The following four specifications are measures of the speed of reaponse.

b. Delay time Td is often defined as the time required for the response to a unit-step function input to reach 50 percent of its final value.

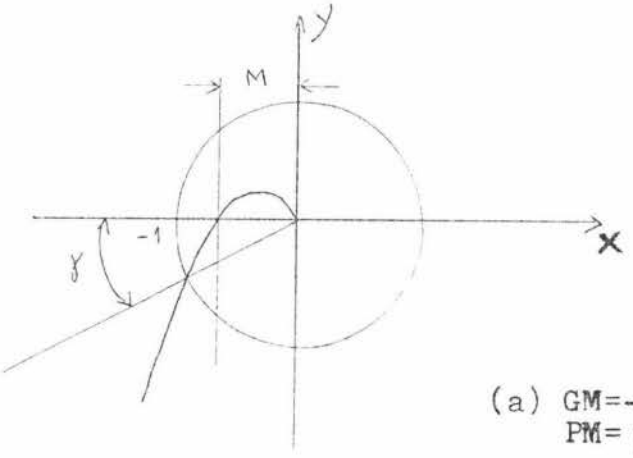
c. Rise-time Tr is customarily defined as the time required fo the response to a unit-step function input to rise from]0 to 90 percent of its final value.

d. Settling time Ts is defined as the time required for the response to a unit-step function input to reach and remain within a specified percentage (frequently 2 or 5 percent) of its final value.

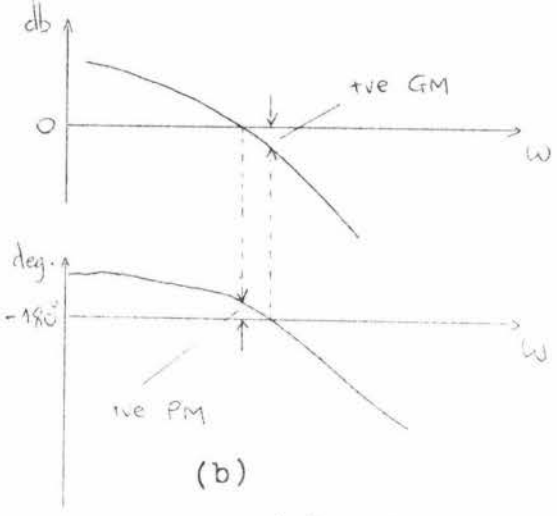
e. Subsidence ratio is the ratio of the amplitudes of successive cycles in a decaying oscillation.

The above specifications are illustrated in Fig. B.3.

-ooOoo-



(a) GM=-20logM
PM= γ



(b)

Fig. B.1 (a) Gain and phase margin on Nyquist diagram
(b) Gain and phase margin on Bode diagram

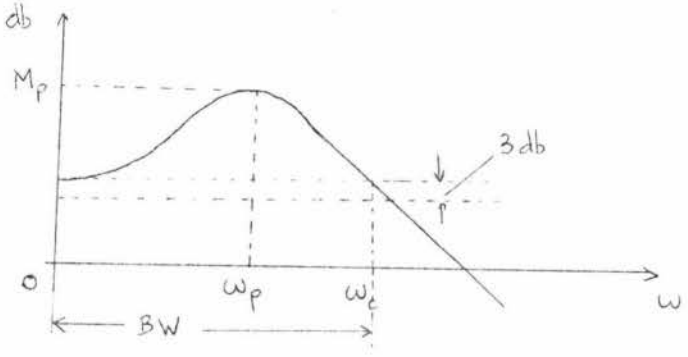


Fig. B.2 Bandwidth, resonant peak & resonant frequency

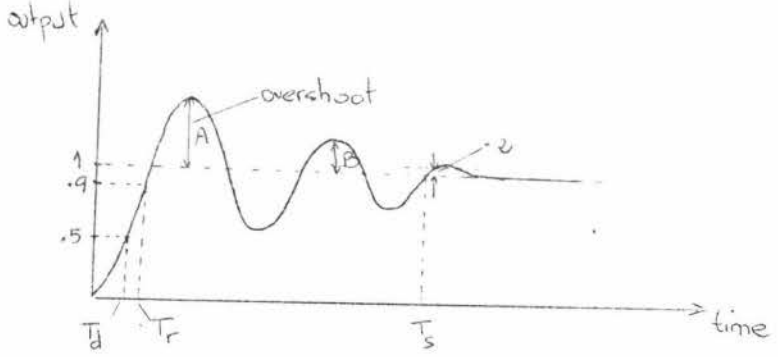


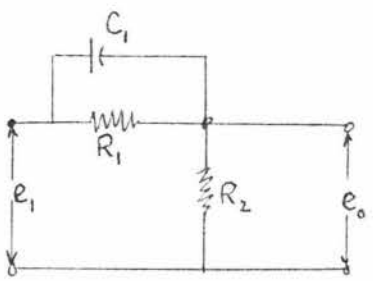
Fig. B.3 Illustration of performance specifications

APPENDIX C

COMPENSATION NETWORKS

1. *Lead compensation*

a. Lead network A schematic diagram of an electrical lead network is shown below



Define $R_1C = T$, $\frac{R_2}{R_1 + R_2} = \alpha < 1$

Then the transfer function for the above network may be derived as

$$\frac{E_o(s)}{E_i(s)} = \alpha \frac{Ts + 1}{\alpha Ts + 1} = \frac{s + \frac{1}{T}}{s + \frac{1}{\alpha T}}$$

b. Characteristics of lead network

Fig. C_1 shows the Nyquist diagram of

$$\alpha \frac{j\omega T + 1}{j\omega \alpha T + 1} \quad (0 < \alpha < 1)$$

For a given value of α, the angle between the positive real axis and the tangent line drawn from the origin to the semicircle gives the maximum phase lead angle, ϕ_m. Define the frequency at the tangent point to be ω_m. From Fig. C_1, the phase angle at $\omega = \omega_m$ is

$$\sin \phi_m = \frac{1 - \alpha}{1 + \alpha} \qquad (C_1)$$

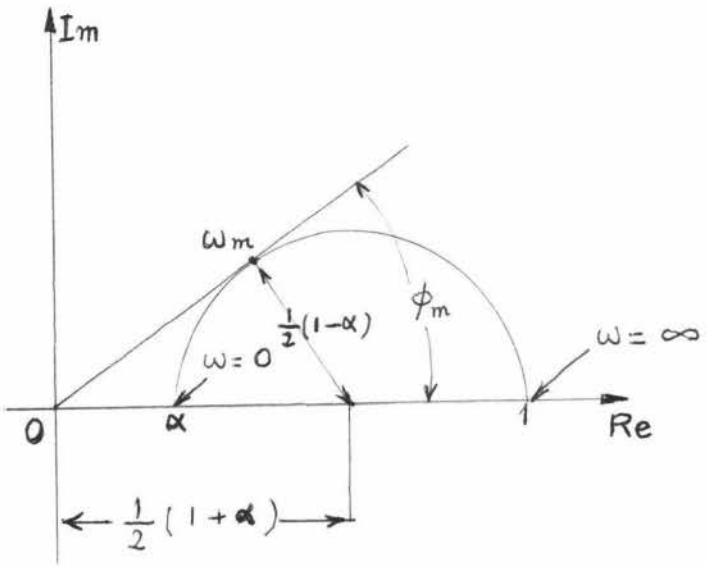


Fig.C1 Nyquist plot of $\alpha \cdot \dfrac{j\,T + 1}{j\omega\alpha T + 1}$

$(0 < \alpha < 1)$

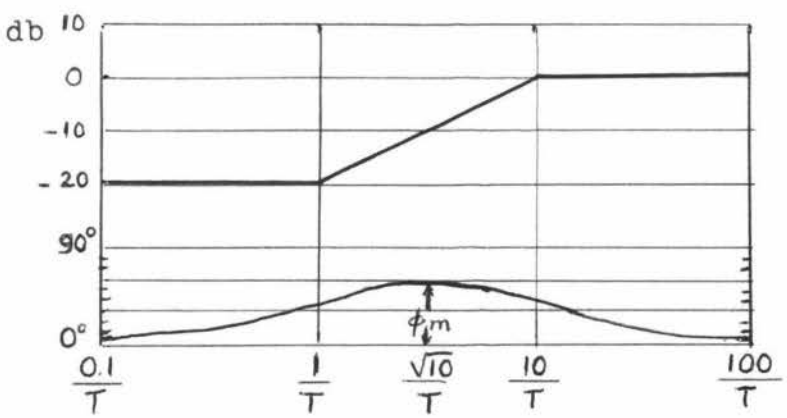


Fig.C2 Bode diagram of a lead network

Fig. C_2 shows the Bode diagram of a lead network when $\alpha = .1$. The corner frequencies for the lead network are $\omega = \frac{1}{T}$ and $\omega = \frac{1}{\alpha T}$. By examining Fig. C_2, it is seen that ω_m is the geometric mean of the two corner frequencies, or

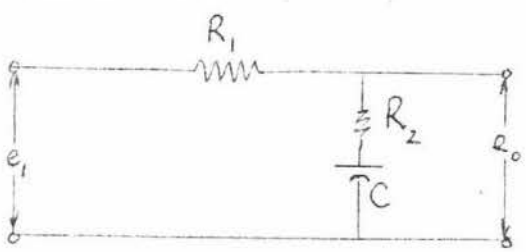
$$\log \omega_m = \frac{1}{2} \left[\log \frac{1}{T} + \log \frac{1}{\alpha T} \right]$$

Hence $$\omega_m = \frac{1}{\alpha T}$$

As seen from Fig. C_2, the lead network is basically a high-pass filter, therefore an additional gain elsewhere is needed to increase the low frequency gain.

2. *Lag compensation*

a. Lag network A schematic diagram of an electrical lag network is shown below



Define $R_2C = T$, $\frac{R_1 + R_2}{R_2} = \beta$ 1

Then the transfer function for the network may be derived as

$$\frac{E_o}{E_i}(s) = \frac{1}{\beta} \left(\frac{s + \frac{1}{T}}{s + \frac{1}{\alpha T}} \right)$$

b. Characteristics of lag network

Fig. C_3 shows the Bode diagram of a lag network when $\beta = 10$. The corner frequencies of the lag network are $\omega = \frac{1}{T}$ & $\omega = \frac{1}{\beta T}$

As seen from Fig. C_3, the lag network is essentially a

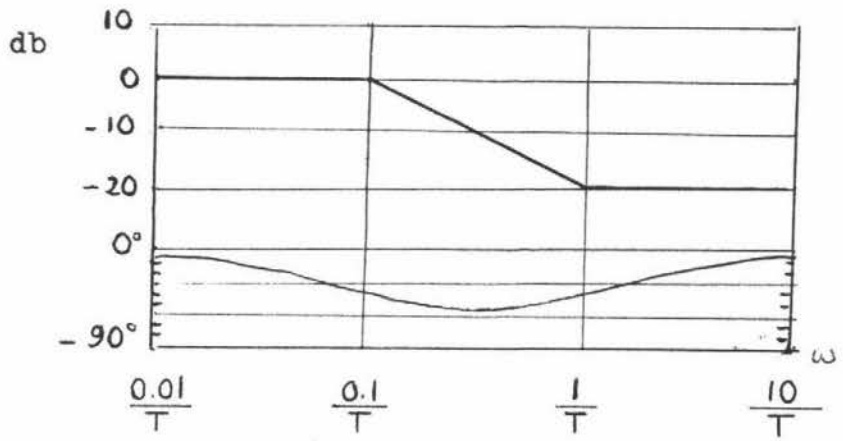


Fig.C3 Bode diagram of a lag network

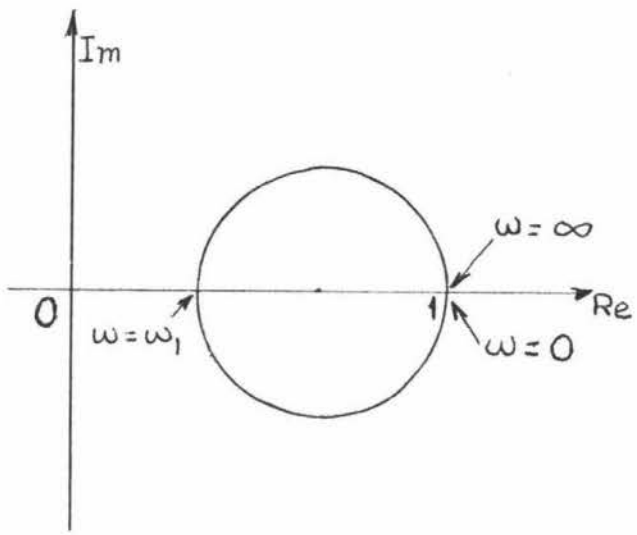


Fig.C4 Nyquist diagram of a lead-lag network

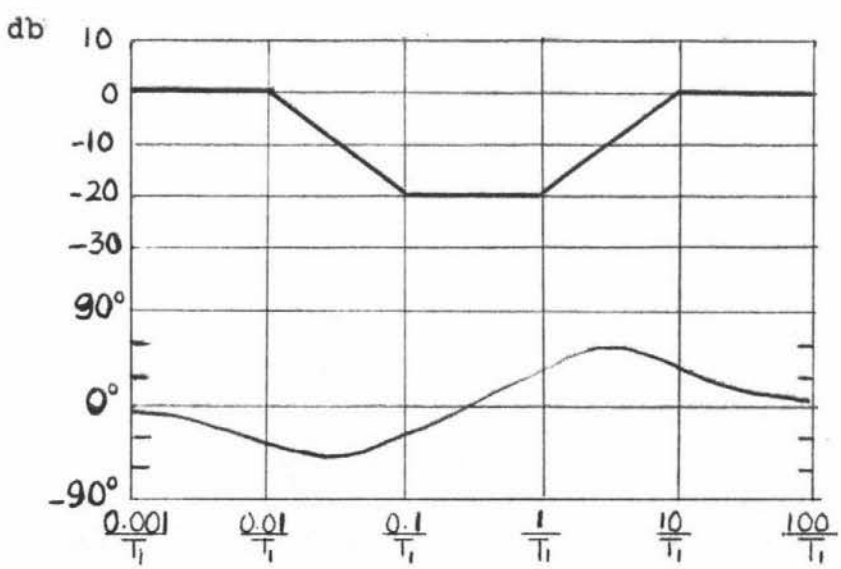


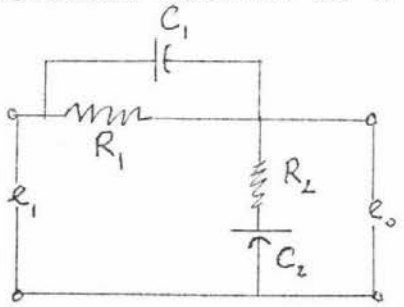
Fig.C5 Bode diagram of a lead-lag network

low-pass filter.

3. *Lead-lag compensation*

a. Lead-lag network

The electrical network of a lead-lag network is shown below:



The transfer function of this network is

$$\frac{E_o(s)}{E_i(s)} = \frac{(s + \frac{1}{T})(s + \frac{1}{T_2})}{(s + \frac{\beta}{T_1})(s + \frac{1}{\beta T_2})}$$

with R_1, C_1, $= T_1$, R_2, C_2, $= T_2$

$$R_1,C_1, + R_2, C_2, + R_1, C_2 = \frac{T_1}{\beta} + \beta T_2 \quad (\beta \quad 1)$$

b. Characteristics of lead-lag network

Fig. C_4 shows the Nyquist diagram of a lead-lag network. It can be seen that for $0 < \omega < \omega_1$, the network acts as a lag network, while for $\omega_1 < \omega < \alpha$ it acts as a lead network. The frequency at which the phase angle is zero is

$$\omega_1 = \frac{1}{\sqrt{T_1 T_2}}$$

Fig. C_5 shows the Bode diagram of a lead-lag network when $\beta = 10$ and $T_2 = 10T_1$.

-ooOoo-

APPENDIX D

CONSTITUENT PROCEDURES

The program consists of the following constituent procedures:

1. POLYADD which adds two polynomials
2. POLYMUL which multiplies two polynomials
3. REALPOLYZEROFINDE which finds all the zeros of a real polynomial
4. DIFEQNS which solves a system of first order differential equations. (The above procedures are system procedures[8].)
5. I_1 which calculates the imaginary part of polynomial with order less than three, as ω varies.
6. R_1 which calculates the real part of polynomial with order less than three as ω varies.
7. I_2 which calculates the imaginary part of polynomial with order greater than or equal to three, as ω varies.
8. R_2 which calculates the real part of polynomial with order greater than or equal to three, as ω varies.
9. TIMEDOMAIN which generates the time-response of the system, either open- or closed-loop, to a unit-step input. This procedure provides flexible features, such as the interactive changeability of scales and ranges of the display.
10. BODE which generates the Bode diagrams of the system, either open- or closed-loop, to a unit-step input. Associated features such as scales and ranges of the display may be changed and grid lines also may be drawn.
11. NYQUIST which generates the Nyquist diagram of the system, either in open- or closed-loop form. Similar flexible

features provided by BODE procedure are also available here.

]2. NICHOLS which generates the Nichols chart plot of the open-loop system superimposed on the Nichols chart. This procedure provides similar features as in BODE procedure.

]3. RLOCUS which generates the root-locus of the system. Scales and ranges of the display and final value of the open-loop gain k are changeable.

]4. READCGH1 which manipulates the constituent transfer functions $C_1(s)$, $C_2(s)$, $G_1(s)$, $G_2(s)$, $H_1(s)$, $H_2(s)$ and the open-loop gain to produce the system open-loop transfer function in a general form $\frac{A(s)}{B(s)}$. The coefficients of the constituent transfer functions and the open-loop gain are entered from the keyboard as requested.

]5. READCGH2 which manipulates the constituent transfer functions $C_1(s)$, $C_2(s)$, $G_1(s)$, $G_2(s)$, $H_1(s)$, $H_2(s)$ and the open-loop gain to produce the system closed-loop transfer function in a general form $\frac{A(s)}{B(s)}$.

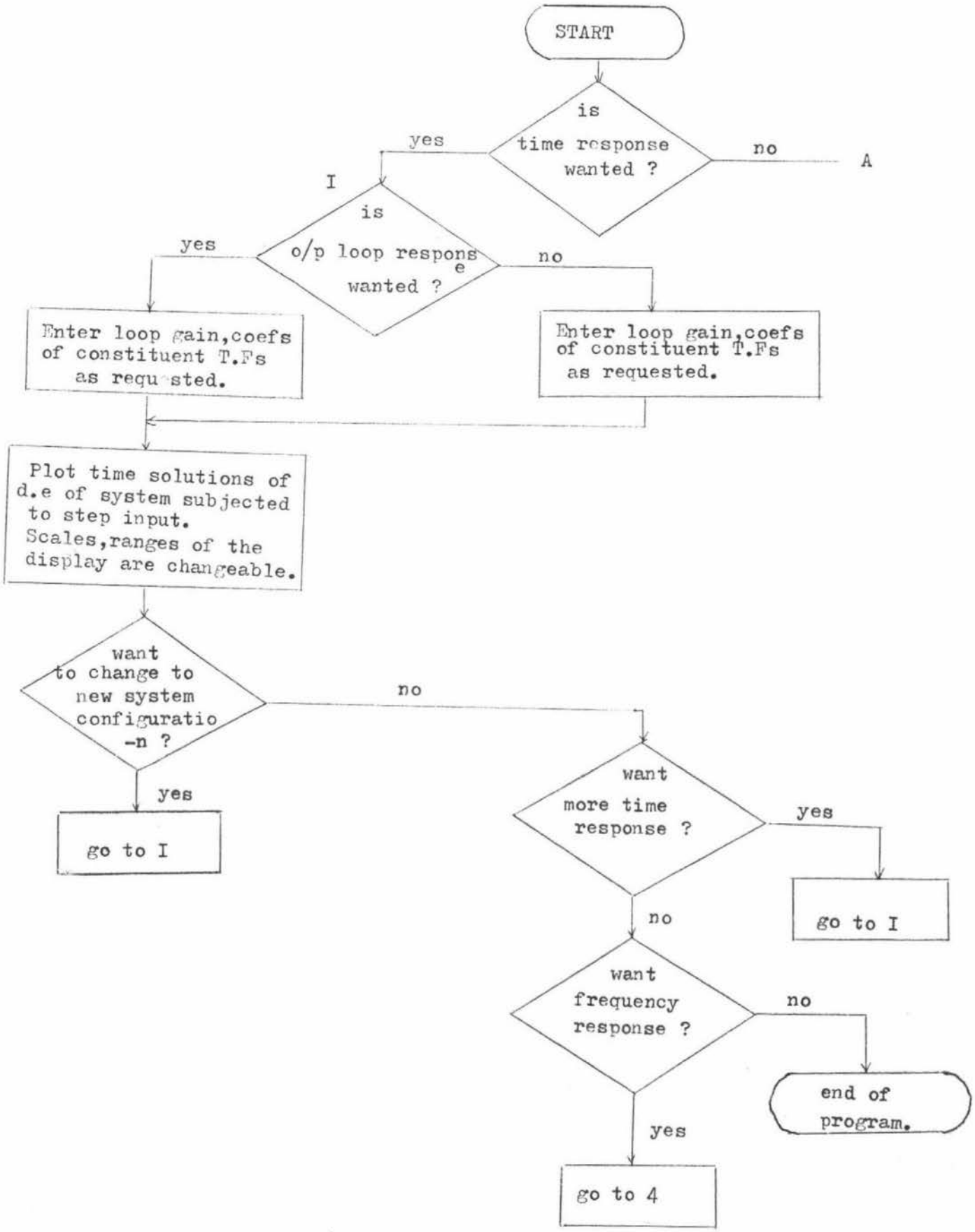
In addition, the plotting procedures used in the program are STARTP, DEVICE, YRANGE, XRANGE, EDGESR, FRAME, AT, LINETO, SYSMSEL, SYMSIZ, PUTSYM, WRTARY, TXT, FIXEDN, TXTSIZ, TXTDIR, SUBFRM, ORGRM, XAXIS, YAXIS, MODE, CHARMD and ENP.

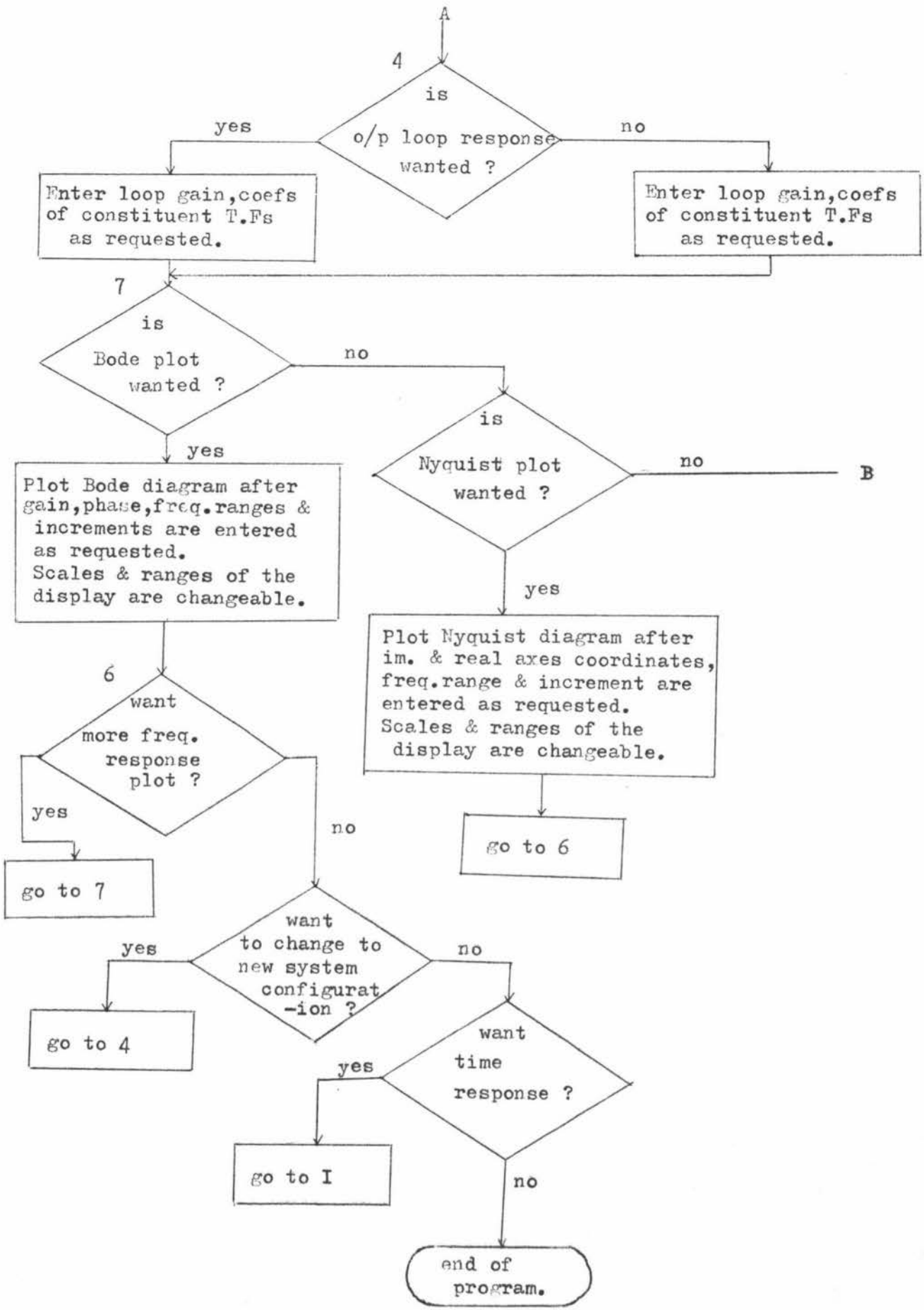
The functions and other details of these procedures may be obtained from the Massey plotter package manual[9].

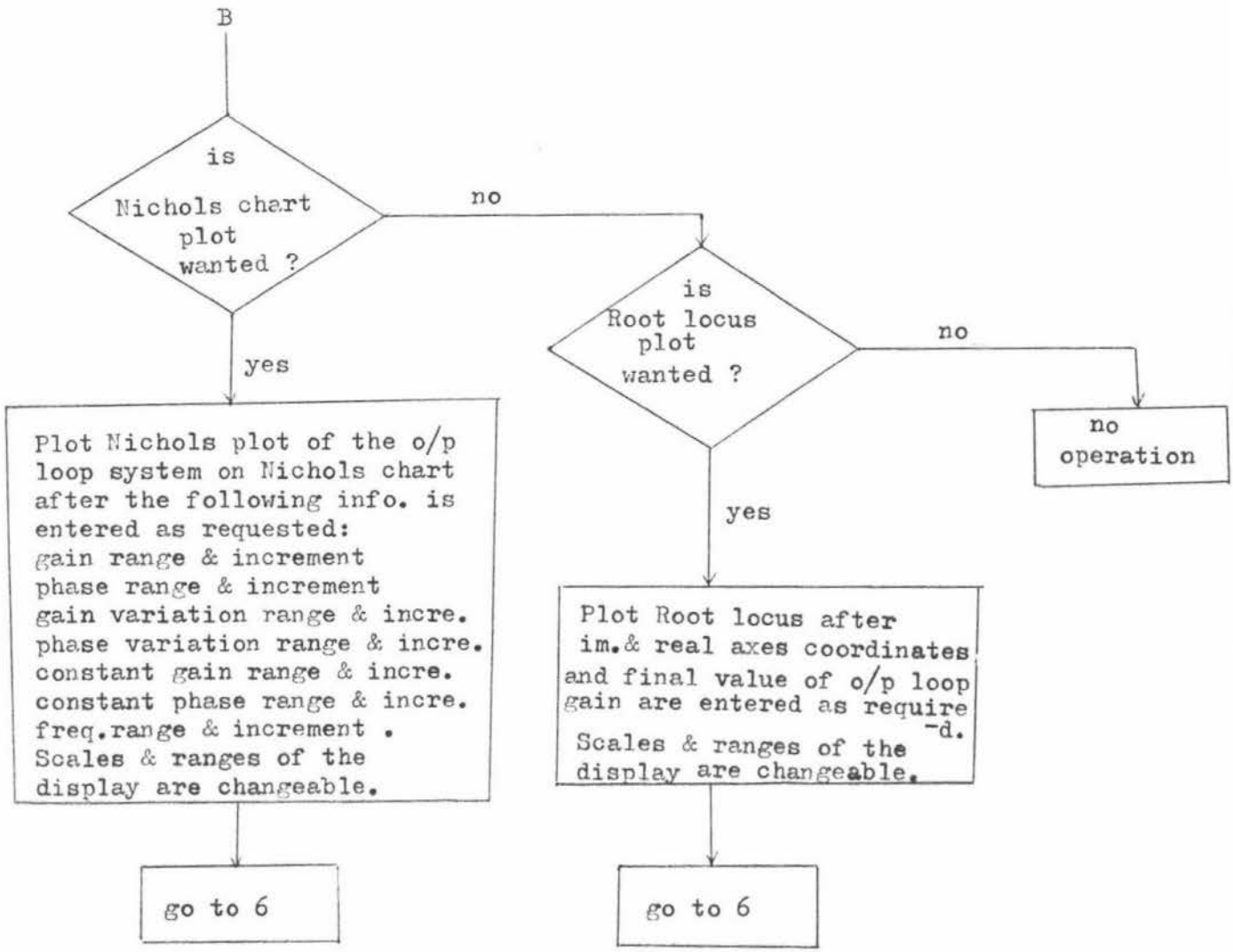
-ooOoo-

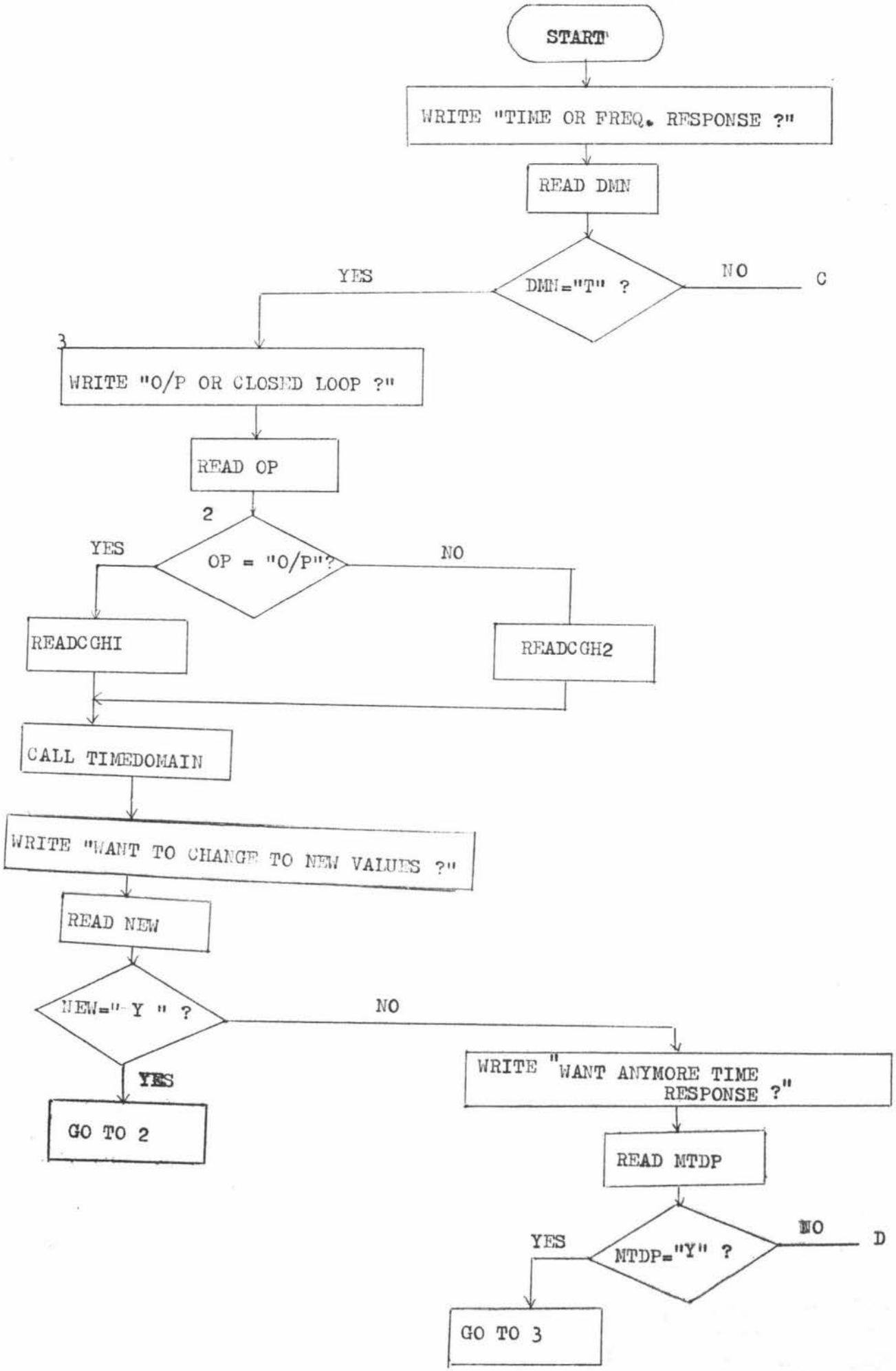
APPENDIX E

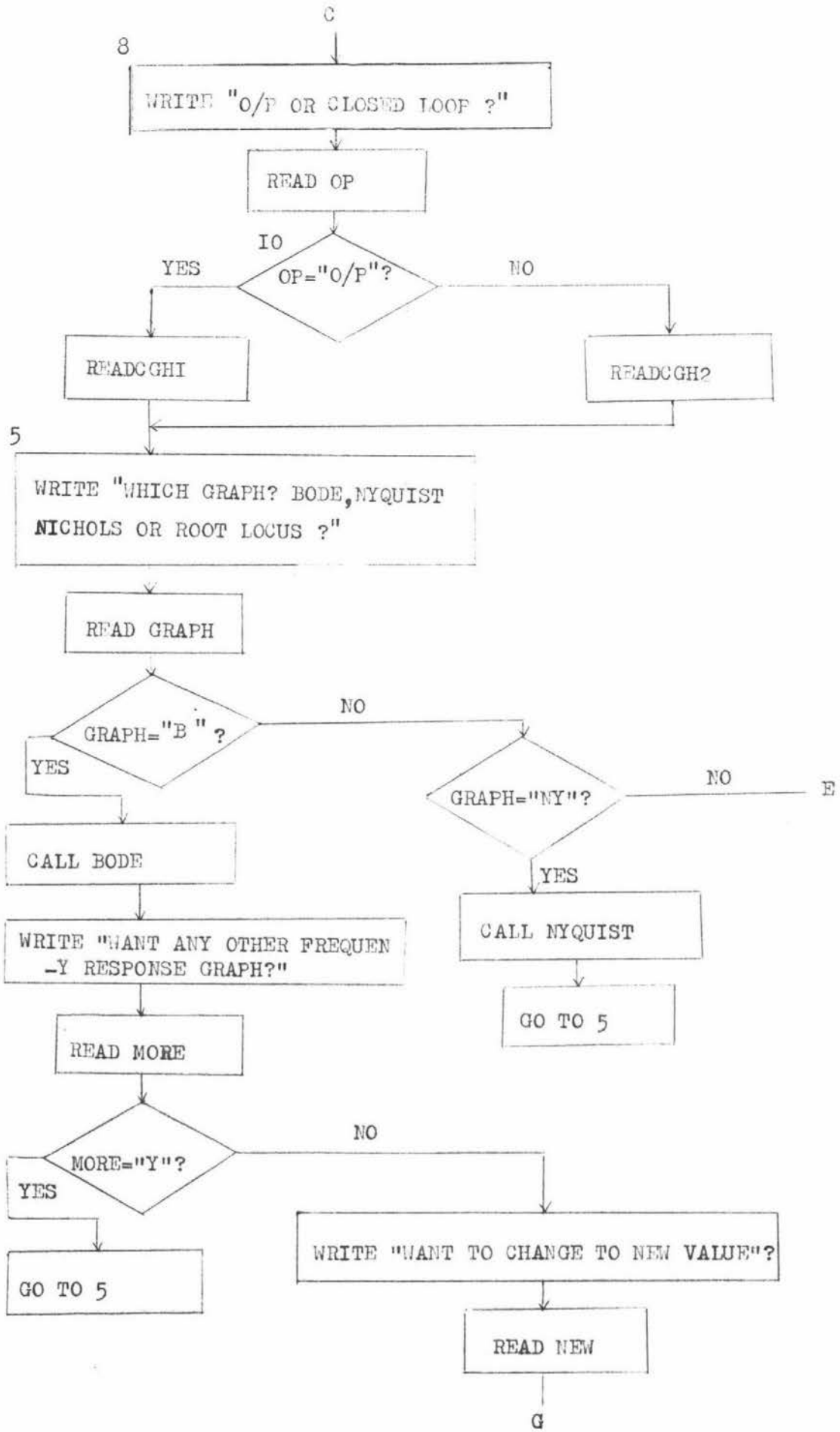
FLOWCHART

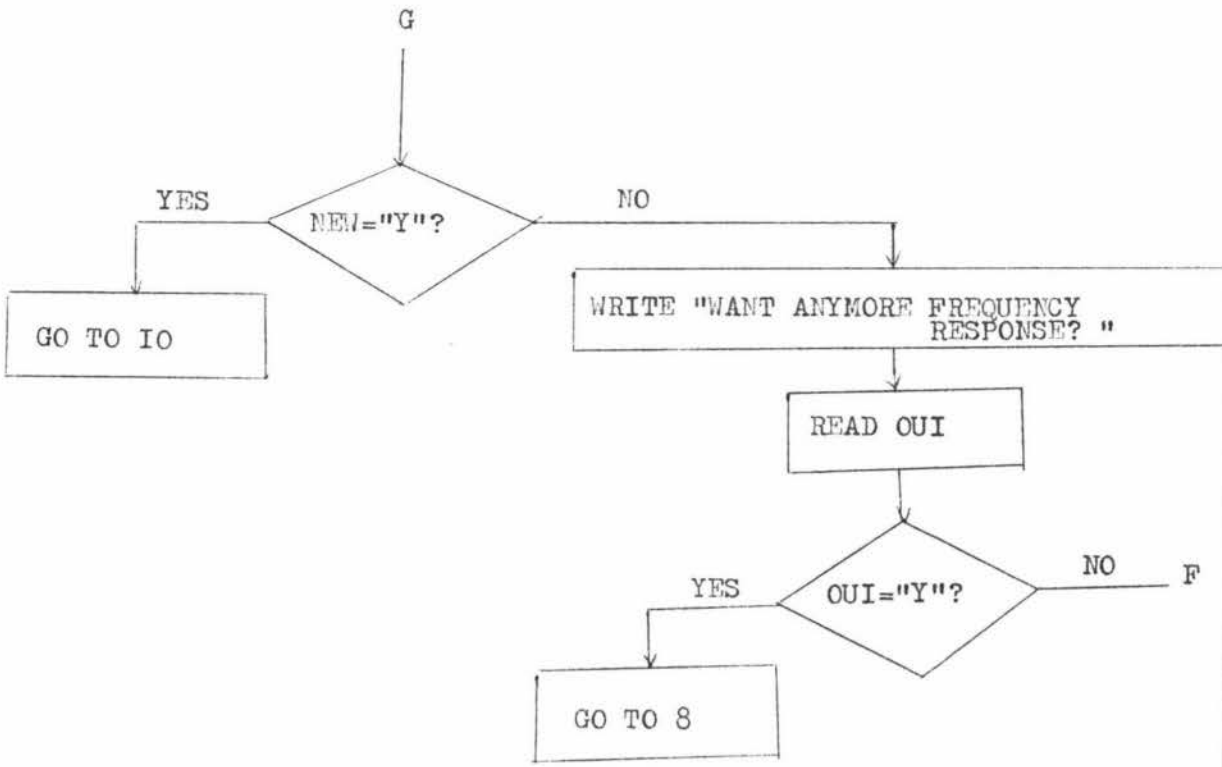


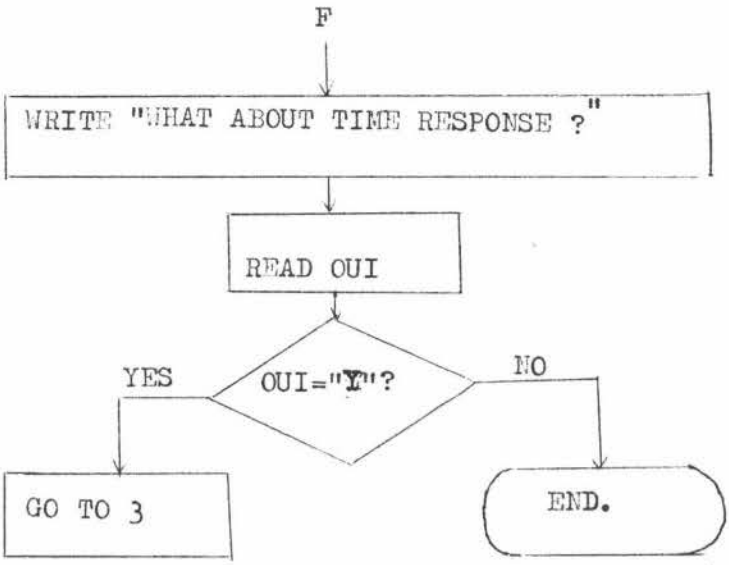

A
4
is o/p loop response wanted ?
yes
no
Enter loop gain,coefs of constituent T.Fs as requested.
Enter loop gain,coefs of constituent T.Fs as requested.
7
is Bode plot wanted ?
no
yes
is Nyquist plot wanted ?
no
B
yes
Plot Bode diagram after gain,phase,freq.ranges & increments are entered as requested. Scales & ranges of the display are changeable.
Plot Nyquist diagram after im. & real axes coordinates, freq.range & increment are entered as requested. Scales & ranges of the display are changeable.
6
want more freq. response plot ?
yes
no
go to 7
go to 6
want to change to new system configurat -ion ?
yes
no
go to 4
want time response ?
yes
no
go to I
end of program.

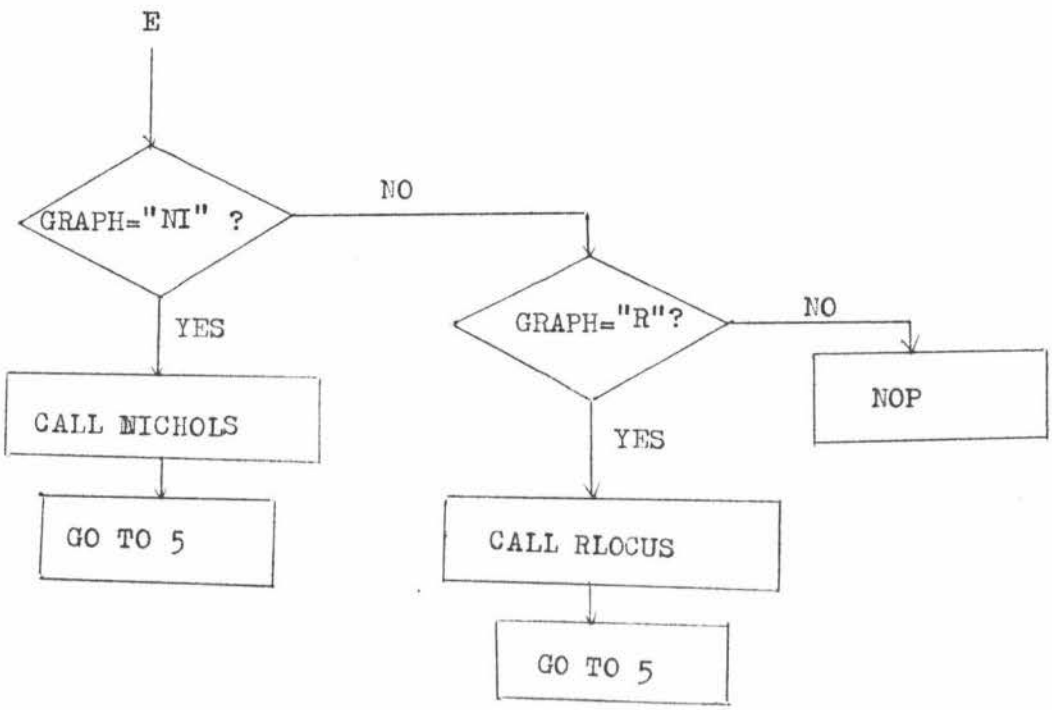

B
is Nichols chart plot wanted ?
no
yes
is Root locus plot wanted ?
no
yes
no operation
Plot Nichols plot of the o/p loop system on Nichols chart after the following info. is entered as requested:
gain range & increment
phase range & increment
gain variation range & incre.
phase variation range & incre.
constant gain range & incre.
constant phase range & incre.
freq.range & increment .
Scales & ranges of the display are changeable.
Plot Root locus after im.& real axes coordinates and final value of o/p loop gain are entered as required.
Scales & ranges of the display are changeable.
go to 6
go to 6

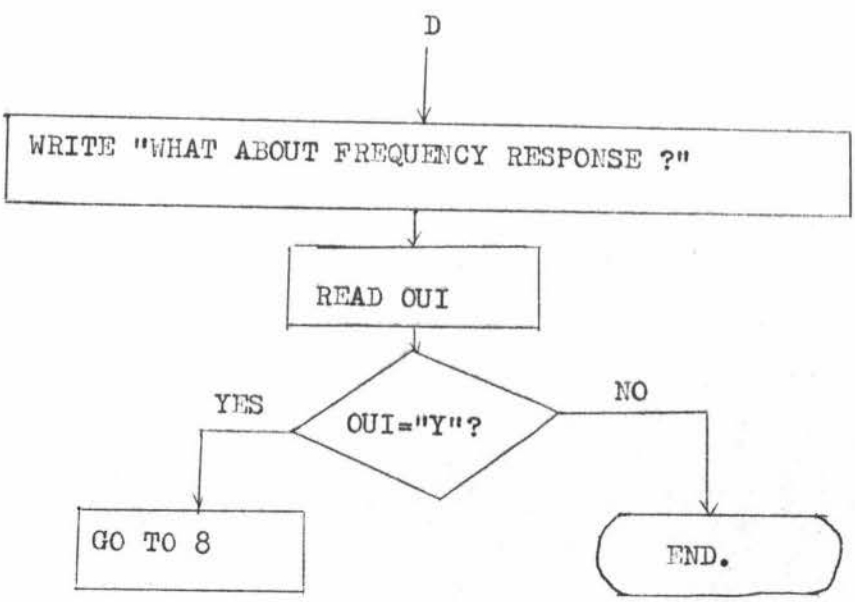

START
WRITE "TIME OR FREQ. RESPONSE ?"
READ DMN
DMN="T" ?
YES
NO
C
3
WRITE "O/P OR CLOSED LOOP ?"
READ OP
2
OP = "O/P"?
YES
NO
READCGH1
READCGH2
CALL TIMEDOMAIN
WRITE "WANT TO CHANGE TO NEW VALUES ?"
READ NEW
NEW="Y" ?
NO
YES
GO TO 2
WRITE "WANT ANYMORE TIME RESPONSE ?"
READ MTDP
MTDP="Y" ?
YES
NO
D
GO TO 3


C
8
WRITE "O/P OR CLOSED LOOP ?"
READ OP
IO
OP="O/P"?
YES
NO
READCGHI
READCGH2
5
WRITE "WHICH GRAPH? BODE,NYQUIST NICHOLS OR ROOT LOCUS ?"
READ GRAPH
GRAPH="B "?
NO
YES
GRAPH="NY"?
NO
E
YES
CALL BODE
CALL NYQUIST
WRITE "WANT ANY OTHER FREQUEN -Y RESPONSE GRAPH?"
GO TO 5
READ MORE
MORE="Y"?
NO
YES
WRITE "WANT TO CHANGE TO NEW VALUE"?
GO TO 5
READ NEW
G


G
NEW="Y"?
YES
NO
GO TO IO
WRITE "WANT ANYMORE FREQUENCY RESPONSE? "
READ OUI
OUI="Y"?
YES
NO
F
GO TO 8


F
WRITE "WHAT ABOUT TIME RESPONSE ?"
READ OUI
OUI="Y"?
YES
NO
GO TO 3
END.


E
GRAPH="NI" ?
NO
YES
GRAPH="R"?
NO
YES
CALL NICHOLS
NOP
GO TO 5
CALL RLOCUS
GO TO 5


D
WRITE "WHAT ABOUT FREQUENCY RESPONSE ?"
READ OUI
OUI="Y"?
YES
NO
GO TO 8
END.

```
  100     BEGIN                                                                                    00000100
  150      $SET LIST                                                                               00000150
  200     $RESET SINGLE                                                                            00000200
  300     $INCLUDE"PLOT/IN/ALGOL"                                                                  00000300
  400     $BIND POLYADD,POLYMUL FROM MATHLIB/=                                                     00000400
  500     REAL DC1,DC2,DG1,DG2,DH1,DP1,DP2,DA,DB,DPOLY1,DPOLY2;                                    00000500
  600     REAL DCG1,DCG2,DPOL1,DPOL2,J,DH2;                                                        00000600
  700     REAL DMM,MTDP,QI,DKCG1;                                                                  00000700
  800      ARRAY U,INB[0:0];                                                                       00000800
  900     REAL GRAPH,OP,MORE,ANY,NEW,OUI,W;                                                        00000900
 1000     LABEL LAB5,LAB6,LAB7,LAB8,LAB9;                                                          00001000
 1100     PROCEDURE POLYMUL(N1DEG,COEF1,N2DEG,COEF2,NRDEG,COEFR);                                  00001100
 1200     VALUE N1DEG,N2DEG;                                                                       00001200
 1300     REAL N1DEG,N2DEG,NRDEG;                                                                  00001300
 1400     ARRAY COEF1,COEF2,COEFR[0];                                                              00001400
 1500     EXTERNAL;                                                                                00001500
 1600     PROCEDURE POLYADD(N1DEG,COEF1,N2DEG,COEF2,NRDEG,COEFR);                                  00001600
 1700     VALUE N1DEG,N2DEG;                                                                       00001700
 1800     REAL N1DEG,N2DEG,NRDEG;                                                                  00001800
 1900     ARRAY COEF1,COEF2,COEFR[0];                                                              00001900
 2000     EXTERNAL;                                                                                00002000
 2100     REAL PROCEDURE R1(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];FORWARD;                     00002100
 2200     REAL PROCEDURE I1(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];FORWARD;                     00002200
 2300     REAL PROCEDURE I(W,Q,POLY);VALUE W,Q;REAL Q;REAL W;                                      00002300
 2400     ARRAY POLY[0];FORWARD;                                                                   00002400
 2500     REAL PROCEDURE R(W,Q,POLY);VALUE W,Q;REAL Q;REAL W;                                      00002500
 2600     ARRAY POLY[0];FORWARD;                                                                   00002600
 2700     REAL PROCEDURE R2(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];FORWARD;                     00002700
 2800     REAL PROCEDURE I2(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];FORWARD;                     00002800
 2900     REAL PROCEDURE R1(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];                             00002900
 3000     CASE P OF                                                                                00003000
 3100          BEGIN                                                                               00003100
 3200     R1:=A[0];                                                                                00003200
 3300     R1:=A[1];                                                                                00003300
 3400     R1:=A[2]*A[0]*W**2;                                                                      00003400
 3700          END;                                                                                00003700
 3800     REAL PROCEDURE I1(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];                             00003800
 3900     CASE P OF                                                                                00003900
 4000          BEGIN                                                                               00004000
 4100     I1:=0;                                                                                   00004100
 4200     I1:=A[0]*W;                                                                              00004200
 4300     I1:=A[1]*W;                                                                              00004300
 4600          END;                                                                                00004600
 4700     REAL PROCEDURE R(W,Q,POLY);VALUE W,Q;REAL Q;REAL W;ARRAY POLY[0];                        00004700
 4800          BEGIN                                                                               00004800
 4900     REAL REP,REN;                                                                            00004900
 5000     REP:=REN:=0;                                                                             00005000
 5100     FOR J:=0 STEP 4 UNTIL Q DO REP:=REP+POLY[J]*W**(Q-J);                                    00005100
 5200     FOR J:=2 STEP 4 UNTIL Q DO REN:=REN+POLY[J]*W**(Q-J);                                    00005200
 5300     R:=REP-REN;                                                                              00005300
 5400          END;                                                                                00005400
 5500     REAL PROCEDURE I(W,Q,POLY);VALUE W,Q;REAL Q;REAL W;ARRAY POLY[0];                        00005500
 5600          BEGIN                                                                               00005600
 5700     REAL IMP,IMN;                                                                            00005700
 5800     IMP:=IMN:=0;                                                                             00005800
 5900     FOR J:=3 STEP 4 UNTIL Q DO IMP:=IMP+POLY[J]*W**(Q-J);                                    00005900
 6000     FOR J:=1 STEP 4 UNTIL Q DO IMN:=IMN+POLY[J]*W**(Q-J);                                    00006000
 6100     I:=IMP-IMN;                                                                              00006100
```

```
      END;                                                                                                  00006200
REAL PROCEDURE R2(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];                                                00006300
CASE P MOD 4 OF                                                                                             00006400
      BEGIN                                                                                                 00006500
R2:=R(W,P,A);                                                                                               00006600
R2:=-I(W,P,A);                                                                                              00006700
R2:=-R(W,P,A);                                                                                              00006800
R2:=I(W,P,A);                                                                                               00006900
      END;                                                                                                  00007000
REAL PROCEDURE I2(W,P,A);VALUE W,P;REAL P;REAL W;ARRAY A[0];                                                00007100
CASE P MOD 4 OF                                                                                             00007200
      BEGIN                                                                                                 00007300
I2:=I(W,P,A);                                                                                               00007400
I2:=R(W,P,A);                                                                                               00007500
I2:=-I(W,P,A);                                                                                              00007600
I2:=-R(W,P,A);                                                                                              00007700
      END;                                                                                                  00007800
PROCEDURE TIMEDOMAIN(DA,DB,AA,BB);VALUE DA,DB;REAL DA,DB;ARRAY AA,BB[0];                                     00007900
  BEGIN                                                                                                     00008000
 VALUE ARRAY T1("OUTPUT"),T2("TIME");                                                                       00008100
 INTEGER I;                                                                                                 00008200
  REAL XI2,XF1,XI1,XF2,X1,X2,XI,Y1,Y2,YI,HI1,SC,QI;                                                         00008300
  ARRAY YSTART[0:10];                                                                                       00008400
 LABEL FAIL;                                                                                                00008500
PROCEDURE DIFEQNS(SP,XFX,N,XI,XF,Y,HI,EPS,ERR,ERRLBL);                                                      00008600
VALUE SP,N,XI,XF,HI,EPS,ERR;PROCEDURE XFX;                                                                  00008700
 INTEGER N;REAL SP,XI,XF,HI,EPS,ERR;                                                                        00008800
 LABEL ERRLBL;ARRAY Y[0];FORWARD;                                                                           00008900
 PROCEDURE FUNCT(X,Z,DZ);                                                                                   00009000
 VALUE X;REAL X;ARRAY Z,DZ[0];                                                                              00009100
 BEGIN                                                                                                      00009200
  DZ[0]:=0;                                                                                                 00009300
  FOR I:=1 STEP 1 UNTIL DB DO                                                                               00009400
    BEGIN                                                                                                   00009500
 IF (DB-I)<=DA THEN                                                                                         00009600
      BEGIN                                                                                                 00009700
   IF I=DB THEN DZ[I]:=-Z[1]*BB[I]+AA[I-(DB-DA)]*QI                                                         00009800
    ELSE DZ[I]:=-Z[1]*BB[I]+Z[I+1]+AA[I-(DB-DA)]*QI;                                                        00009900
      END                                                                                                   00010000
    ELSE DZ[I]:=-Z[1]*BB[I]+Z[I+1];                                                                         00010100
  END;                                                                                                      00010200
 END;                                                                                                       00010300
PROCEDURE      DIFEQNS(SP,XFX,N,XI,XF,Y,HI,EPS,ERR,ERRLBL);                                                 00010400
 VALUE         SP,N,XI,XF,HI,EPS,ERR;                                                                     00010500
PROCEDURE      XFX;                                                                                         00010600
 INTEGER       N;                                                                                         00010700
 REAL          SP,XI,XF,HI,EPS,ERR;                                                                       00010800
 LABEL         ERRLBL;                                                                                    00010900
ARRAY          Y[0] ;                                                                                     00011000
COMMENT*XXXX;                                                                                               00011100
COMMENT        PURPOSE                                                                                      00011200
                  SOLVE A SYSTEM OF FIRST ORDER ORDINARY DIFFERENTIAL                                       00011300
                  EQUATIONS                                                                                 00011400
COMMENT*YYYY                                                                                                00011500
               DESCRIPTION OF PARAMETERS                                                                    00011600
                  SP      - A POSITIVE VALUE FOR SP WILL CAUSE OUTPUT AT                                    00011700
                            INCREMENTS OF SP IN ADDITION TO PROGRAM                                         00011800
                            CONTROLLED OUTPUT.  IF SP = 0 ONLY PROGRAM                                      00011900
                            CONTROLLED OUTPUT WILL OCCUR                                                    00012000
                  XFX     - PROCEDURE TO GENERATE THE RIGHT-HAND-SIDE OF                                    00012100
                            THE DIFFERENTIAL EQUATION. IT MUST HAVE A                                       00012200
```

```
12300                          HEADING OF THE FORM                                  00012300
12400                               "PROCEDURE     NAME(X,Y,DY)                     00012400
12500                                VALUE  X       REAL  X                        00012500
12600                                ARRAY  Y,DY[0]"                               00012600
12700                          WHERE GIVEN THE VALUE OF THE INDEPENDENT             00012700
12800                          VARIABLE X AND THE VALUES OF THE DEPENDENT           00012800
12900                          VARIABLES AT X, STORED IN Y[1:N], NAME RETURNS       00012900
13000                          THE VALUES OF THE DERIVATIVES AT X IN DY[1:N]        00013000
13100                     N    - NUMBER OF DIFFERENTIAL EQUATIONS IN THE SYSTEM     00013100
13200                          N SHOULD BE LESS THAN OR EQUAL TO 10                 00013200
13300                     XI   - INITIAL VALUE OF THE INDEPENDENT VARIABLE         00013300
13400                     XF   - FINAL VALUE OF THE INDEPENDENT VARIABLE.          00013400
13500                          XF MAY BE LESS THAN XI IN WHICH CASE                 00013500
13600                          HI MUST BE NEGATIVE                                  00013600
13700                     Y    - ARRAY CONTAINING THE INITIAL VALUES OF THE        00013700
13800                          DEPENDENT VARIABLES, WHICH MUST BE STORED IN         00013800
13900                          Y[1] THRU Y[N].  THESE ARE NOT PRESERVED.            00013900
14000                          Y MUST BE DIMENSIONED [0:N]                          00014000
14100                     HI   - INITIAL STEPSIZE.  HI MAY BE NEGATIVE, AND        00014100
14200                          MUST SATISFY ABS(XF-XI)/(2*15) ? ABS(H)              00014200
14300                     EPS  - ERROR TOLERANCE(1.0@-10 LEQ EPS LEQ 1.0@-2)       00014300
14400                     ERR  - PARAMETER TO SPECIFY WHICH CONVERGENCE            00014400
14500                          CRITERION TO USE                                     00014500
14600                          ERR = "SE" - STANDARD ERROR                          00014600
14700                          ERR = "RE" - RELATIVE ERROR                          00014700
14800                          ERR = "AE" - ABSOLUTE ERROR                          00014800
14900                     ERRLBL - A LABEL IN THE CALLING PROGRAM TO WHICH A       00014900
15000                          RETURN WILL BE MADE IN CASE OF AN INPUT ERROR        00015000
15100                          OR IN CASE THE ITERATION FAILS TO CONVERGE           00015100
15200                PARAMETERS RETURNED                                           00015200
15300                     Y    - CONTAINS THE SOLUTION VALUES AT THE FINAL         00015300
15400                          PROGRAM SOLUTION POINT.                              00015400
15500                METHOD                                                        00015500
15600                     RATIONAL EXTRAPOLATION METHOD OF R.BULIRSCH AND J.STOER  00015600
15700                     WHICH HAS BEEN FOUND TO BE SUPERIOR TO ALL OTHER         00015700
15800                     METHODS BOTH IN SPEED AND ACCURACY FOR SMALL SYSTEMS OF  00015800
15900                     DIFFERENTIAL EQUATIONS, SAY N LESS THAN 5.               00015900
16000                REFERENCE                                                     00016000
16100                     DIFEQNS IS AN ADAPTION OF THE FORTRAN PACKAGE DESUB      00016100
16200                     WHICH IS VOL 2, ISSUE 1, OF THE NUMERICAL MATHEMATICS    00016200
16300                     PROGRAM LIBRARY PROJECT, BELL TELEPHONE LABORATORIES,    00016300
16400                     MURRAY HILL, NEW JERSEY ;                                00016400
16500     COMMENT*ZZZZ;                                                           00016500
16600           BEGIN                                                             00016600
16700     COMMENT        INITIALIZATION OF MACHINE DEPENDENT CONSTANTS            00016700
16800                     ZOT  - REAL NUMBERS ARE CONSIDERED TO BE EQUAL IF        00016800
16900                          THE DIFFERENCE BETWEEN THEM IS LESS THAN             00016900
17000                          ZOT = 2*(-33)                                        00017000
17100                     ZOTUP - THE CONSTANT USED IN THE STEP HALVING DECISION   00017100
17200                          IN DESUB BASED ON COMPARING ZOTUP*DT WITH C.         00017200
17300                          ZOTUP = 2*33                                         00017300
17400      COMMENT        PARAMETERS FOR NUMERICAL IMPLEMENTATION OF THE ALGORITHM 00017400
17500                     HDIV - THE CONSTANT USED IN FINDING HMIN, THE MINIMUM    00017500
17600                          VALUE THE STEPSIZE IS ALLOWED TO ATTAIN SUCH         00017600
17700                          THAT HMIN = H/HDIV                                   00017700
17800                          HDIV = 1024                                          00017800
17900                     EMIN - THE LOWER BOUND ON EPS                            00017900
18000                          EMIN = 1.0@-10                                       00018000
18100                     EMAX - THE UPPER BOUND ON EPS                            00018100
18200                          EMAX = 1.0@-2 ;                                      00018200
18300     COMMENT        (XEND - XSTART) / P2 IS USED AS LOWER BOUND               00018300
```

```
18400                 FOR INITIAL STEPSIZE HI.  P2 IS SET AT 2 * 15. ;                    00018400
18500     COMMENT    THE ORDER OF EXTRAPOLATION JM IS SET AT 6. ;                         00018500
18600      COMMENT    THE MAXIMUM NO. OF DIFFERENTIAL EQUATIONS IN THE SYSTEM             00018600
18700                 IS SET AT 10.  THIS LIMIT MAY BE INCREASED BY THE USER            00018700
18800                 BUT THE PROGRAM MAY NOT BE EFFICIENT FOR LARGER SYSTEMS ;         00018800
18900      DEFINE     ZQI    = 1.164153218270-10 #,                                     00018900
19000                 ZQTOP  = 8589934592.0 #,                                          00019000
19100                 HDIV   = 1024.0 #,                                                00019100
19200                 EMIN   = 1.0@-10 #,                                               00019200
19300                 EMAX   = 1.0@-2 #,                                                00019300
19400                 P2     = 32768.0 #,                                               00019400
19500                 JM     = 6 #,                                                     00019500
19600                 NMAX   = 10 #;                                                    00019600
19700     COMMENT     ERROR INFORMATION PARAMETERS ;                                    00019700
19800      REAL       EX,ER,EH;                                                         00019800
19900       INTEGER   NE;                                                               00019900
20000     COMMENT     OUTPUT PARAMETERS ;                                               00020000
20100      REAL       X,XR,T1;                                                          00020100
20200       INTEGER   NP;                                                               00020200
20300     COMMENT     VARIOUS LOGICAL VARIABLES ;                                       00020300
20400      BOOLEAN    STYPE,KONVF,RE,SE,AE;                                             00020400
20500     COMMENT     OTHER VARIABLES ;                                                 00020500
20600     REAL        TTL,H,HMIN,HMAX,HP,HQ,HR,XP,XPMX,XT,FH ;                          00020600
20700     INTEGER     KOUNT,I,J ;                                                       00020700
20800     COMMENT     WORKING ARRAYS ;                                                  00020800
20900      ARRAY      S,R,YA,YR,YL,YM,DY,DZ,SA[0:N], D[0:7],                            00020900
21000                 DT[0:N,0:7], YG,YH,SG[0:8,0:N];                                   00021000
21100     LABEL       L1,L2,L3,L4,L5,EXIT ;                                             00021100
21200                                                                                   00021200
21300      FILE       PRINTER(KIND=PRINTER, BUFFERS=2, MAXRECSIZE=22);                  00021300
21400      FORMAT     TYME(7/"ELAPSED PROCESSOR TIME IN SECONDS = ",F10.4);             00021400
21500     PROCEDURE   ERROR(XFX,XCURNT,ERRNUM,PRNT,ERRLBL);                             00021500
21600      VALUE      XCURNT,ERRNUM;                                                    00021600
21700     PROCEDURE   XFX;                                                              00021700
21800     REAL        XCURNT ;                                                          00021800
21900     INTEGER     ERRNUM ;                                                          00021900
22000     BOOLEAN     PRNT ;                                                            00022000
22100     LABEL       ERRLBL;                                                           00022100
22200     COMMENT     PURPOSE                                                           00022200
22300                    PRINT THE CURRENT VALUES OF X AND ARRAY Y AND THE              00022300
22400                    APPROPRIATE ERROR MESSAGE, WHENEVER AN ERROR OCCURS            00022400
22500                 DESCRIPTION OF PARAMETERS                                         00022500
22600                    XCURNT - CURRENT VALUE OF X                                    00022600
22700                    ERRNUM - NUMBER INDICATING THE ERROR OCCURRED                  00022700
22800                    PRNT   - BOOLEAN VARIABLE WHOSE VALUE IS TRUE IF               00022800
22900                             VALUES OF X,Y ETC.                                    00022900
23000                 OTHER PROCEDURES REQUIRED                                         00023000
23100                    XFX ;                                                          00023100
23200           BEGIN                                                                   00023200
23300     LABEL       L1 ;                                                              00023300
23400                 IF NOT PRNT THEN GO TO L1 ;                                       00023400
23500                 XFX(XCURNT,Y,DY) ;                                                00023500
23600     L1:      BEGIN                                                                00023600
23700     FORMAT      ERRTLE(/X30,"ERROR"/X30"*****"/) ;                                00023700
23800     SWITCH FORMAT SWFMT :=                                                        00023800
23900     (/"*****      NO CONVERGENCE IN ABOVE STEP TO",X19,"*****"/                   00023900
24000       "*****      X = ",R12.6,"     WITH    H = ",R12.6," *****"/                 00024000
24100       "*****      THE LIMITING ERROR IS ",R12.6," IN EQUATION ",I2,"*****"),      00024100
24200     (/"*****",X17,"N LSS 0  OR N GTR 10",X18,"*****"),                            00024200
24300     (/"*****",X15,"EPS LSS @-10  OR EPS GTR @-2",X15,"*****"),                   00024300
24400     (/"*****",X15,"H I (XEND - XSTART) LSS 0",X15,"*****"),                       00024400
```

```
24500     (/"*****",X15,"H = 0  OR  XEND = XSTART",X16,"*****"),                                  00024500
24600     (/"*****",X1,"H LSS (XEND - XSTART)/2*15  OR  H GTR (XEND - XSTART)",                   00024600
24700       X1,"*****")';                                                                         00024700
24800                 WRITE(PRINTER,ERRTLE);                                                    00024800
24900                 IF ERRNUM = 0 THEN                                                        00024900
25000                 WRITE(PRINTER,SWFMT[ERRNUM],EX,EH,ER,N)                                   00025000
25100                 ELSE                                                                      00025100
25200                 WRITE(PRINTER,SWFMT[ERRNUM]);                                             00025200
25300                 GO TO ERRLBL ;                                                            00025300
25400           END   ;                                                                         00025400
25500           END   ERROR ;                                                                   00025500
25600     PROCEDURE   DESUB(X,Y,H,HMIN,REDIF);                                                  00025600
25700     VALUE       HMIN,REDIF;                                                               00025700
25800     REAL        X,H,HMIN;                                                                 00025800
25900     ARRAY       Y[0];                                                                     00025900
26000     BOOLEAN     REDIF;                                                                    00026000
26100        BEGIN                                                                              00026100
26200     COMMENT         PERFORMS A SINGLE INTEGRATION STEP. WHEN THE PROCEDURE                00026200
26300       IS EXITED,X AND THE ARRAY Y WILL CONTAIN THE MOST RECENTLY COMPUTED                 00026300
26400       VALUES FOR THE INDEPENDENT AND DEPENDENT VARIABLES ;                               00026400
26500     COMMENT         DESUB IS BASED ON THE ALGOL PROCEDURE DIFFSYS OF                      00026500
26600       R. BULIRSCH AND J. STOER, NUMERISHE MATHEMATIK 8,1-13(1966) AND                    00026600
26700       INCORPORATES THE MODIFICATIONS GIVEN IN SUBROUTINE DESUB OF THE                    00026700
26800       BELL TELEPHONE LABORATORIES DESUB PACKAGE ;                                        00026800
26900     REAL            A,B,B1,C,G,U,V,TA,FC;                                               00026900
27000     INTEGER         I,J,K,KK,JJ,L,M,SR,JMAX,JR,JS,N1;                                   00027000
27100     LABEL           RESETH,RESTART1,RESTART2,EXIT;                                      00027100
27200     BOOLEAN         KONV,BO,BH;                                                         00027200
27300     COMMENT         ALTERNATE INITIALIZATION IN CASE A SPECIFIED POINT HAS              00027300
27400       BEEN OVERSTEPPED SO THAT A RETURN MUST BE MADE TO THE PREVIOUS POINT;             00027400
27500           N1 := N+1;                                                                    00027500
27600           IF REDIF THEN                                                                 00027600
27700           BEGIN                                                                         00027700
27800              JMAX := JM+3;                                                              00027800
27900              WRITE(Y[*],N1,YA[*]);                                                      00027900
28000              GO TO RESTART1;                                                            00028000
28100           END;                                                                          00028100
28200     COMMENT         INITIALIZATION ;                                                    00028200
28300     COMMENT         FOR AN EXTRAPOLATION OF ORDER JM, JM+1 UNEXTRAPOLATED               00028300
28400       APPROXIMATIONS ARE REQUIRED. THREE MORE ARE ALLOWED IN ATTEMPTING                 00028400
28500       TO ACHIEVE CONVERGENCE ;                                                          00028500
28600           JMAX := JM+3;                                                                 00028600
28700     COMMENT         SAVE THE INITIAL VALUES FOR THE DEPENDENT VARIABLES AND             00028700
28800       THE ERROR TEST VECTOR FOR THE STEP ;                                            00028800
28900           WRITE(YA[*],N1,Y[*]);                                                         00028900
29000           WRITE(SA[*],N1,S[*]);                                                         00029000
29100     COMMENT         USE THE FUNCTION ROUTINE TO OBTAIN THE INITIAL SLOPES.              00029100
29200                                    DZ = DY/DX ';                                        00029200
29300           XFX(X,Y,DZ);                                                                  00029300
29400     COMMENT         THE LOGICAL VARIABLE BH DETERMINES WHETHER THE STEP SIZE            00029400
29500       HAS BEEN HALVED. INITIALLY FALSE.                                               00029500
29600       LATER BH IS FALSE IF THE STEPSIZ IS CUT BY A FACTOR NOT 2 ;                     00029600
29700     RESTART1:   BH := FALSE;                                                            00029700
29800     COMMENT         PRESET THE CONVERGENCE SUCCESS FLAG TRUE ;                          00029800
29900           KONVF := TRUE;                                                                00029900
30000     COMMENT         ADVANCE THE INDEPENDENT VARIABLE BY THE STEPSIZE H ;                00030000
30100     RESTART2:   A := H+X;                                                               00030100
30200     COMMENT         SET THE SWITCH FOR THE FIRST SET OF D COEFFICIENTS ;                00030200
30300           BO := FALSE ;                                                                 00030300
30400     COMMENT         INITIALIZE THE H SEQUENCE: H/M, H/JR, H/JS ;                        00030400
30500           M := 1;    JR := 2;    JS := 3;                                               00030500
```

```
30600      COMMENT      JJ IS THE INDEX FOR THE ARRAY OF VALUES SAVED IN CASE                00030600
30700        THE INTERVAL MUST BE HALVED ;                                                  00030700
30800              JJ := -1;                                                                00030800
30900      COMMENT      INTEGRATION STEP: MID-POINT RULE + EXTRAPOLATION ;                   00030900
31000              FOR J := 0 STEP 1 UNTIL JMAX DO                                          00031000
31100              BEGIN                                                                    00031100
31200      COMMENT      SET THE VALUES OF THE EXTRAPOLATION COEFFICIENTS TO                  00031200
31300        THEIR CORRECT VALUES FOR THIS EXTRAPOLATION STEP:                              00031300
31400        D[1]:=16/9, D[3]:=64/9, D[5]:=256/9 OR D[1]:=9/4, D[3]:=9, D[5]:=36 ;          00031400
31500                IF BO THEN                                                           00031500
31600                BEGIN                                                                00031600
31700                   D[1] := 1.77777777778;                                            00031700
31800                   D[3] := 7.11111111111;                                            00031800
31900                   D[5] := 28.44444444444;                                           00031900
32000                END                                                                  00032000
32100                ELSE                                                                 00032100
32200                BEGIN                                                                00032200
32300                   D[1] := 2.25;                                                     00032300
32400                   D[3] := 9.0;                                                      00032400
32500                   D[5] := 36.0;                                                     00032500
32600                END;                                                                 00032600
32700      COMMENT      IF THE ORDER OF THE EXTRAPOLATION STEP BEING COMPUTED            00032700
32800        IS LESS THAN JM/2, SET KONV FALSE ;                                          00032800
32900                IF J LSS JM/2 THEN KONV := FALSE ELSE KONV := TRUE ;                 00032900
33000      COMMENT      IF J IS GREATER THAN JM RESTRICT THE ORDER OF                    00033000
33100        EXTRAPOLATION TO JM AND ADJUST THE LAST EXTRAPOLATION COEFFICIENT ;          00033100
33200                IF J GTR JM THEN                                                     00033200
33300                BEGIN                                                                00033300
33400                   L := JM;  D[L] := 4.0 * D[L-2];                                    00033400
33500      COMMENT      DISCOURAGE THE STEP-INCREASING FACTOR FC BY A FACTOR OF           00033500
33600        SQRT(2) SINCE CONVERGENCE WAS NOT OBTAINED IN JM EXTRAPOLATIONS ;            00033600
33700                   FC := .707106781187 * FC;                                         00033700
33800                END                                                                  00033800
33900                ELSE                                                                 00033900
34000      COMMENT      THE NUMBER, J, OF EXTRAPOLATIONS HAS NOT EXCEEDED JM.            00034000
34100        FIND D(J) = (H DIVIDED BY H/M)*2.                                            00034100
34200        ADJUST THE FACTOR, FC, USED TO ADJUST THE STEPSIZE FOR THE NEXT STEP         00034200
34300        TO BE TAKEN ;                                                                00034300
34400                BEGIN                                                                00034400
34500                   L := J;   D[L] := M*M;                                             00034500
34600                   FC := (JM-J)/6.0 + 1.0;                                           00034600
34700                END;                                                                 00034700
34800      COMMENT      MODIFIED MIDPOINT RULE USED TO FIND FIRST VALUE FOR THIS         00034800
34900        EXTRAPOLATION STEP ;                                                         00034900
35000                M := M+M;    G := H/M;    B := G+G;                                  00035000
35100      COMMENT      IF THE STEPSIZE HAS BEEN HALVED AND IF THE ORDER OF THE          00035100
35200        EXTRAPOLATION STEP DOES NOT EXCEED THAT FOR WHICH PREVIOUSLY COMPUTED        00035200
35300        VALUES WERE SAVED, THEY CAN BE RESTORED ;                                    00035300
35400                IF BH AND J LSS JMAX-1 THEN                                          00035400
35500                BEGIN                                                                00035500
35600                   WRITE(YM[*,J],N1,YH[J,*]);                                        00035600
35700                   WRITE(YL[*,J],N1,YG[J,*]);                                        00035700
35800                   WRITE(S[*],N1,SG[J,*]);                                           00035800
35900                END                                                                  00035900
36000                ELSE                                                                 00036000
36100                BEGIN                                                                00036100
36200      COMMENT      OTHERWISE THE STARTING VALUES FOR THE MIDPOINT RULE              00036200
36300        MUST BE COMPUTED ;                                                           00036300
36400                   KK := .5 * (M-2);                                                 00036400
36500                   WRITE(YL[*,J],N1,YA[*]);                                          00036500
36600                   FOR I := 1 STEP 1 UNTIL N DO                                      00036600
```

```
                    YM[I] := YA[I] + DZ[I]*G;                                   00036700
                  WRITE(S[*],N1,SA[*]);                                         00036800
COMMENT         THE MEMBER OF THE H SEQUENCE BEING USED BY THE MIDPOINT         00036900
  INTEGRATION RULE IS H/M. COMPUTE TO THE END OF THE STEP CONTINUOUSLY           00037000
  UPDATING THE VECTOR,S,CONTAINING THE MAXIMUM VALUE COMPUTED UP TO             00037100
  NOW FOR EACH DEPENDENT VARIABLE ;                                             00037200
                  M := M-1;                                                     00037300
                  FOR K := 1 STEP 1 UNTIL M DO                                  00037400
                  BEGIN                                                         00037500
                    XFX(X + K*G,YM,DY);                                         00037600
                    FOR I := 1 STEP 1 UNTIL N DO                                00037700
                    BEGIN                                                       00037800
                      U := YL[I] + DY[I]*B;                                     00037900
                      YL[I] := YM[I];                                           00038000
                      YM[I] := U;                                               00038100
                      S[I] := MAX(ABS(U),S[I]);                                 00038200
                    END;                                                        00038300
COMMENT           IN CASE THE INTERVAL MUST BE HALVED NEXT TIME, SAVE THE       00038400
  VALUES AT HALFWAY ALONG (KK = M/2 - 1) THE STEP UNLESS K=2 ;                  00038500
                    IF K=KK AND K NEQ 2 THEN                                    00038600
                    BEGIN                                                       00038700
                      JJ := JJ+1;                                               00038800
                      WRITE(YH[JJ,*],N1,YM[*]);                                 00038900
                      WRITE(YG[JJ,*],N1,YL[*]);                                 00039000
                      WRITE(SG[JJ,*],N1,S[*]);                                  00039100
                    END;                                                        00039200
                  END  KLOOP;                                                   00039300
                END;                                                            00039400
                XFX(A,YM,DY);                                                   00039500
                FOR I := 1 STEP 1 UNTIL N DO                                    00039600
                BEGIN                                                           00039700
COMMENT           V IS USED TO SAVE THE VALUE OBTAINED BY THE MIDPOINT RULE     00039800
  USING THE PREVIOUS MEMBER OF THE H SEQUENCE (THE FIRST TIME THROUGH           00039900
  THIS VALUE IS UNDEFINED, BUT IT IS NOT USED SINCE L IS LESS THAN 1);          00040000
                  V := DT[I,0];                                                 00040100
COMMENT           COMPUTE THE FINAL VALUE OBTAINED FOR THIS MEMBER OF THE       00040200
  H SEQUENCE BY THE MODIFIED MIDPOINT RULE ;                                    00040300
                  TA := C := DT[I,0] := .5 * (DY[I]*G + YL[I] + YM[I]);         00040400
COMMENT           IF THE VALUE JUST COMPUTED BY THE MIDPOINT RULE SHOWS         00040500
  A LARGE JUMP FROM THE PREVIOUS VALUE, HALVE THE INTERVAL ;                    00040600
                  IF ABS(V)*ZOTUP LSS ABS(C)  AND  ABS(H) NEQ HMIN              00040700
                    AND  J GTR .5*JM  THEN GO TO RESETH;                        00040800
COMMENT           PERFORM THE L STEPS FOR THE CURRENT LTH ORDER                  00040900
  EXTRAPOLATION STEP. IF THE DENOMINATIONATOR OF THE RATIONAL FUNCTION           00041000
  GOES TO ZERO AT ANY STEP,SET DT AT THAT STEP TO ITS VALUE JUST BEFOR;         00041100
                  FOR K:= 1 STEP 1 UNTIL L DO                                   00041200
                  BEGIN                                                         00041300
                    B1 := D[K]*V;                                               00041400
                    B := B1-C;                                                  00041500
                    U := V;                                                     00041600
                    IF B NEQ 0 THEN                                             00041700
                    BEGIN                                                       00041800
                      B := (C-V)/B;                                             00041900
                      U := C*B;                                                 00042000
                      C := B1*B;                                                00042100
                    END;                                                        00042200
                    V := DT[I,K];                                               00042300
                    DT[I,K] := U;                                               00042400
                    TA := U+TA;                                                 00042500
                  END  KLOOP;                                                   00042600
COMMENT           EACH DEPENDENT VARIABLE IS CHECKED SEPARATELY FOR             00042700
```

```
42800          CONVERGENCE ;                                                                    00042800
42900                       IF RE THEN S[I] := ABS(Y[I])  ELSE                                  00042900
43000                       IF SE THEN S[I] := MAX(ABS(TA),S[I]);                               00043000
43100                       R[I] := ABS(Y[I]-TA);                                               00043100
43200                       Y[I] := TA;                                                         00043200
43300                       S[I] := MAX(S[I],EPS);                                              00043300
43400                       IF R[I] GTR EPS*S[I] THEN KONV := FALSE;                            00043400
43500                  END  ILOOP;                                                              00043500
43600                  IF KONV THEN GO TO EXIT;                                                 00043600
43700          COMMENT      RESET THE EXTRAPOLATION COEFFICIENTS ;                              00043700
43800                  D[2] := 4.0;   D[4] := 16.0;                                             00043800
43900          COMMENT      FLIP THE SWITCH FOR THE NEXT SET OF COEFFICIENTS ;                  00043900
44000                  BO := NOT BO;                                                            00044000
44100          COMMENT      TAKE THE NEXT MEMBER OF THE H SEQUENCE ;                            00044100
44200                  M := JR;   JR := JS;   JS := M+M;                                        00044200
44300          COMMENT      AND GO BACK FOR THE NEXT EXTRAPOLATION ;                            00044300
44400             END  JLOOP;                                                                   00044400
44500          COMMENT      IF, AFTER ALL THE EXTRAPOLATIONS ALLOWED, CONVERGENCE HAS           00044500
44600            NOT BEEN ACHIEVED, ATTEMPT TO HALVE H SO THAT THE SAVED VALUES CAN           00044600
44700            BE USED (SET BH TRUE FOR THIS PURPOSE). IF HALVING H MAKES IT LESS           00044700
44800            THAN HMIN, SET H:=HMIN. IN THIS CASE THE SAVED VALUES CANNOT BE USED.        00044800
44900            IF H HAD ALREADY BEEN AT HMIN, CONVERGENCE CANNOT BE ACHIEVED FOR            00044900
45000            THIS HMIN AND THIS EPS CRITERION. SET KONVF FALSE ;                          00045000
45100                  BH := NOT BH;                                                          00045100
45200          RESETH: IF ABS(H) LEQ HMIN THEN                                              00045200
45300                     BEGIN   KONVF := FALSE;  GO TO EXIT;   END;                       00045300
45400                  H := .5*H;                                                             00045400
45500                  IF ABS(H) GEQ HMIN THEN GO TO RESTART2;                                00045500
45600                  H := SIGN(H) * HMIN;                                                   00045600
45700                  GO TO RESTART1;                                                        00045700
45800          COMMENT      WHETHER OF NOT CONVERGENCE HAS BEEN ACHIEVED SET A NEW          00045800
45900            SUGGESTED STEPSIZE FOR THE NEXT STEP AND ASSIGN THE END OF STEP            00045900
46000            VALUE TO THE INDEPENDENT VARIABLE ;                                        00046000
46100          EXIT:   H := FC*H;                                                           00046100
46200                  X := A;                                                              00046200
46300             END  DESUB;                                                               00046300
46400          COMMENT      MAIN PROCEDURE BEGINS HERE ;                                    00046400
46500                  T1 := TIME(2);                                                       00046500
46600          COMMENT      SET ERROR CRITERION PARAMETER ;                                 00046600
46700                  IF ERR = "RE" OR ERR = "0000RE" OR ERR = "    RE"                    00046700
46800                  OR ERR = "RE0000" OR ERR = "RE    " THEN RE := TRUE                00046800
46900                  ELSE                                                                 00046900
47000                  IF ERR = "SE" OR ERR = "0000SE" OR ERR = "    SE"                    00047000
47100                  OR ERR = "SE0000" OR ERR = "SE    " THEN SE := TRUE                00047100
47200                  ELSE                                                                 00047200
47300                  BEGIN                                                                00047300
47400                     AE := TRUE;                                                       00047400
47500                     FOR I := 1 STEP 1 UNTIL N DO S[I] := 1.0;                         00047500
47600                  END;                                                                 00047600
47700          COMMENT      SET SPECIFIED POINTS FLAG FOR OUTPUT AND CHECK SP ;             00047700
47800                  NP := IF SP = 0 THEN 0 ELSE 1 ;                                      00047800
47900                  KOUNT := 0 ;                                                         00047900
48000                  IF SP * (XF - XI) LSS 0 THEN SP := SP * SIGN(XF-XI) ;                00048000
48100          COMMENT      CHECK PARAMETERS AND PERFORM INITIALIZATION ;                    00048100
48200                  STYPE := TRUE ;                                                      00048200
48300                  H := HI ;                                                            00048300
48400                  IF N LEQ 0 OR N GTR NMAX THEN ERROR(XFX,XI,1,TRUE,ERRLBL);           00048400
48500                  IF EPS LSS EMIN OR EPS GTR EMAX THEN ERROR(XFX,XI,2,TRUE,            00048500
48600                                                             ERRLBL);                  00048600
48700                  TTL := XF - XI ;                                                     00048700
48800                  IF TTL * H LSS 0 THEN ERROR(XFX,XI,3,TRUE,ERRLBL) ELSE IF            00048800
```

```
48900                   TTL * H = 0 THEN ERROR(XFX,XI,4,TRUE,ERRLBL);                    00048900
49000                IF (H / TTL) * P2 LSS 1 OR H / TTL GTR 1                            00049000
49100                   THEN ERROR(XFX,XI,5,TRUE,ERRLBL);                                00049100
49200                KONVF := TRUE ;                                                     00049200
49300                FOR I := 1 STEP 1 UNTIL N DO S[I] := ABS(Y[I]) ;                    00049300
49400                HMIN := ABS(H/HDIV) ;                                               00049400
49500                HMAX := ABS(TTL) ;                                                  00049500
49600                HP := 0 ;                                                           00049600
49700                XP := X := XI ;                                                     00049700
49800     COMMENT    BEGIN SOLUTION OF DIFFERENTIAL EQUATIONS                            00049800
49900                THE RIGHTMOST 21 BITS OF THE STEPSIZE H ARE ZEROED OUT              00049900
50000                IN AN ATTEMPT TO REDUCE THE BUILD UP OF ROUND OFF ERROR.            00050000
50100                THIS LEAVES H WITH APPROXIMATELY 5 NON ZERO DECIMAL                 00050100
50200                DIGITS ;                                                            00050200
50300      L1:       IF X NEQ XI THEN                                                    00050300
50400                H.[20:21] := 0 ;                                                    00050400
50500                IF NP = 0 AND NOT STYPE THEN GO TO L3 ;                             00050500
50600                XPMX := XP - X ;                                                    00050600
50700                FH := XPMX/H ;                                                      00050700
50800                IF FH GTR ZOT THEN GO TO L3 ;                                       00050800
50900              J := IF ABS(FH) GTR ZOT THEN 1 ELSE 0 ;                               00050900
51000                CASE J OF                                                           00051000
51100             BEGIN                                                                  00051100
51200                BEGIN                                                               00051200
51300                   WRITE(YR[*],N+1,Y[*]);                                           00051300
51400                   HQ := HP ;                                                       00051400
51500                   XR := X ;                                                        00051500
51600                END ;                                                            00051600
51700                BEGIN                                                            00051700
51800                   HR := HQ := XPMX + HP ;                                          00051800
51900                   XR := XT ;                                                       00051900
52000                   DESUB(XR,YR,HR,HQ,TRUE);                                         00052000
52100                   HQ := XR - XT ;                                                  00052100
52200                END ;                                                            00052200
52300             END ;                                                                  00052300
52400                   STYPE := TRUE ;                                                 00052400
52500                   XFX(XR,YR,DY) ;                                                 00052500
52600                   STYPE := FALSE ;                                                00052600
52700                   IF KONVF THEN                                                   00052700
52800             BEGIN                                                                  00052800
52900                   IF (XF - XR)/TTL LEQ 0 THEN GO TO EXIT ;                        00052900
53000                   KOUNT := KOUNT + 1 ;                                            00053000
53100                   XP := XI + KOUNT * SP ;                                         00053100
53200                   IF (XP - XF)/H GTR 0 THEN XP := XF ;                            00053200
53300                   GO TO L1                                                        00053300
53400             END   ELSE GO TO L4 ;                                                 00053400
53500     COMMENT       OUTPUT RESULTS AT A NATURAL STEP ;                              00053500
53600     L3:           IF ABS((X-XR)/H) LEQ ZOT THEN GO TO L2 ;                        00053600
53700                   XFX(X,Y,DY) ;                                                   00053700
53800     COMMENT       PERFORM A STEP IN SOLUTION OF DIFFERENTIAL EQUATIONS ;          00053800
53900     L2:           IF (XF - X)/TTL LEQ 0 THEN GO TO EXIT ;                         00053900
54000                   IF ABS(H) LSS HMIN THEN H := HMIN * SIGN(H) ELSE                00054000
54100                   IF ABS(H) GTR HMAX THEN H := HMAX * SIGN(H) ;                   00054100
54200                   IF (XF-X-H)/TTL LSS 0 THEN H := XF - X ;                        00054200
54300                   XT := X ;                                                       00054300
54400                   DESUB(X,Y,H,HMIN,FALSE);                                        00054400
54500                   HP := X - XI ;                                                  00054500
54600                   IF KONVF THEN GO TO L1 ELSE                                     00054600
54700             BEGIN                                                                  00054700
54800                   J := 0 ;                                                        00054800
54900                   GO TO L5                                                        00054900
```

```
                END  ;
       L4:           IF KONVF THEN GO TO EXIT ;
                     J := 1 ;
       L5:           ER :=0 ;
                     FOR I := 1 STEP 1 UNTIL N DO
                BEGIN
       LABEL         L6;
                     IF  ER * S[I] GEQ R[I] THEN GO TO L6 ;
                     ER := R[I] / S[I] ;
                     NE := I ;
       L6:      END  ;
                     CASE J OF
                BEGIN
                    BEGIN
                           EH := HP ;
                           EX := X ;
                       ERROR(XFX,X,0,TRUE,ERRLBL);
                    END  ;
                    BEGIN
                           EH := HQ ;
                           EX := XR ;
                       ERROR(XFX,XR,0,FALSE,ERRLBL);
                    END  ;
                END  ;
         EXIT:  BEGIN END;
            END  DIFEQNS;
         STARTP;
        XRANGE(-250,1250);YRANGE(-250,1250);
         SUBFRM(0,0,1000,1000);
         DEVICE(3,1,0,0);
         WRITE(FILE97,<"INITIAL CONDITIONS?">);
         READ(FILE97,/,FOR I:=1 STEP 1 UNTIL DB DO YSTART[I]);
         WRITE(FILE97,<"FINAL TIME VALUE AND INCREMENT?">);
         READ(FILE97,/,XF1,XI2);
         WRITE(FILE97,<"VALUE OF INPUT STEP?">);
         READ(FILE97,/,QI);
         WRITE(FILE97,<"TIME RANGE AND INCREMENT?">);
         READ(FILE97,/,X1,X2,XI);
        WRITE(FILE97,<"OUTPUT RANGE AND INCREMENT?">);
        READ(FILE97,/,Y1,Y2,YI);
         AT(90,1000);TXTDIR(0);WRTARY(1,T1);
        AT(1000,5);TXTDIR(0);WRTARY(1,T2);
        XRANGE(X1,X2);YRANGE(Y1,Y2);XAXIS(Y1,XI);YAXIS(X1,YI);
        NOLINE;
        HI1:=XI2/10;
        FOR XI1:=-XI2 STEP XI2 UNTIL (XF1-XI2) DO
          BEGIN
        XF2:=XI1+XI2;
       DIFEQNS(0,FUNCT,DB,XI1,XF2,YSTART,HI1,@-3,"RE",FAIL);
          CHARMD;
        LINETO(XF2,YSTART[1]);
         END;
         WHEN(10);
        WRITE(FILE97,<"WANT TO CHANGE THE SCALE?">);
        READ(FILE97,/,SC);
        IF SC="Y" THEN TIMEDOMAIN(DA,DB,AA,BB);
        ENDP;
       FAIL: END;
         COMMENT **** PROCEDURE BODE ***;
          PROCEDURE BODE(W,DA,DB,A,B);VALUE W,DA,DB;REAL DA,DB;
       REAL W;ARRAY A[0],B[0];
```

```
61000          BEGIN                                                                    00061000
61100            FILE EE1(KIND=DISK,MAXRECSIZE=40);                                     00061100
61200            FILE E1(KIND=DISK,MAXRECSIZE=40);                                      00061200
61300            FILE E2(KIND=DISK,MAXRECSIZE=40);                                      00061300
61400       ARRAY Y8[0:100000];                                                         00061400
61500       INTEGER I;                                                                  00061500
61600      REAL SC,GR,X,Y,COF,S,G;                                                      00061600
61700         REAL LF,FI,HF,LG,HG,LP,HP,GI,PI;                                          00061700
61800        VALUE ARRAY T1("***BODE DIAGRAMS***");                                     00061800
61900        VALUE ARRAY T2("GAIN(DB)");                                                00061900
62000        VALUE ARRAY T3("PHASE(DEG)");                                              00062000
62100        VALUE ARRAY T4("FREQUENCY(HTZ)");                                          00062100
62200       PROCEDURE MGRAPH1;FORWARD;                                                  00062200
62300       PROCEDURE MGRAPH2;FORWARD;                                                  00062300
62400       PROCEDURE MGRAPH3;FORWARD;                                                  00062400
62500       PROCEDURE MGRAPH4;FORWARD;                                                  00062500
62600       PROCEDURE PGRAPH1;FORWARD;                                                  00062600
62700       PROCEDURE PGRAPH2;FORWARD;                                                  00062700
62800       PROCEDURE PGRAPH3;FORWARD;                                                  00062800
62900       PROCEDURE PGRAPH4;FORWARD;                                                  00062900
63000       PROCEDURE THREE(W,Y8);VALUE W;REAL W;ARRAY Y8[0];FORWARD;                   00063000
63100       PROCEDURE MGRAPH1;                                                          00063100
63200        BEGIN                                                                      00063200
63300       FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                         00063300
63400                                 BEGIN                                            00063400
63500      G:=((R1(W,DA,A)**2+I1(W,DA,A)**2)/(R1(W,DB,B)**2+I1(W,DB,B)**2));           00063500
63600       IF G>0 THEN  BEGIN                                                         00063600
63700        Y:=10*LOG(G);                                                            00063700
63800       X:=W;                                                                       00063800
63900       IF Y<=HG OR Y>=LG THEN                                                      00063900
64000             BEGIN                                                              00064000
64100          CHARMD;                                                               00064100
64200          LINETO(X,Y);                                                          00064200
64300        WRITE(E1,/,X,Y);                                                        00064300
64400             END;                                                               00064400
64500             END;                                                               00064500
64600        END;                                                                    00064600
64700        END;                                                                    00064700
64800       PROCEDURE MGRAPH2;                                                       00064800
64900       BEGIN                                                                    00064900
65000       FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                      00065000
65100                                 BEGIN                                         00065100
65200      G:=((R1(W,DA,A)**2+I1(W,DA,A)**2)/(R2(W,DB,B)**2+I2(W,DB,B)**2));        00065200
65300       IF G>0 THEN  BEGIN                                                      00065300
65400        Y:=10*LOG(G);                                                          00065400
65500       X:=W;                                                                   00065500
65600       IF Y<=HG OR Y>=LG THEN                                                  00065600
65700             BEGIN                                                           00065700
65800          CHARMD;                                                            00065800
65900          LINETO(X,Y);                                                       00065900
66000        WRITE(E1,/,X,Y);                                                     00066000
66100             END;                                                            00066100
66200             END;                                                            00066200
66300        END;                                                                 00066300
66400       END;                                                                 00066400
66500       PROCEDURE MGRAPH3;                                                   00066500
66600        BEGIN                                                               00066600
66700       FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                  00066700
66800                                 BEGIN                                      00066800
66900      G:=((R2(W,DA,A)**2+I2(W,DA,A)**2)/(R1(W,DB,B)**2+I1(W,DB,B)**2));     00066900
67000       IF G>0 THEN  BEGIN                                                   00067000
```

```
67100       Y:=10*LOG(G);                                                                00067100
67200      X:=W;                                                                         00067200
67300      IF Y<=HG OR Y>=LG THEN                                                        00067300
67400            BEGIN                                                                  00067400
67500        CHARMD;                                                                    00067500
67600        LINETO(X,Y);                                                               00067600
67700       WRITE(E1,/,X,Y);                                                            00067700
67800            END;                                                                   00067800
67900            END;                                                                   00067900
68000      END;                                                                         00068000
68100      END;                                                                         00068100
68200      PROCEDURE MGRAPH4;                                                           00068200
68300       BEGIN                                                                       00068300
68400      FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                          00068400
68500                                    BEGIN                                        00068500
68600     G:=((R2(W,DA,A)**2+I2(W,DA,A)**2)/(R2(W,DB,B)**2+I2(W,DB,B)**2));           00068600
68700      IF G>0 THEN  BEGIN                                                          00068700
68800      Y:=10*LOG(G);                                                               00068800
68900      X:=W;                                                                       00068900
69000      IF Y<=HG OR Y>=LG THEN                                                      00069000
69100            BEGIN                                                                00069100
69200        CHARMD;                                                                  00069200
69300        LINETO(X,Y);                                                             00069300
69400       WRITE(E1,/,X,Y);                                                          00069400
69500            END;                                                                 00069500
69600            END;                                                                 00069600
69700      END;                                                                        00069700
69800      END;                                                                        00069800
69900      PROCEDURE PGRAPH1;                                                          00069900
70000      BEGIN                                                                       00070000
70100      REAL NNT,DNT;                                                               00070100
70200      Y8[0]:=0;                                                                   00070200
70300      FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                         00070300
70400      BEGIN                                                                       00070400
70500      I:=*+1;                                                                     00070500
70600      NNT:=(I1(W,DA,A)/R1(W,DA,A))-(I1(W,DB,B)/R1(W,DB,B));                       00070600
70700      DNT:=1+(I1(W,DA,A)/R1(W,DA,A))*(I1(W,DB,B)/R1(W,DB,B));                     00070700
70800     Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                  00070800
70900      IF Y8[I]<=0 THEN                                                            00070900
70902        BEGIN                                                                     00070902
70904      WRITE(E2,/,I,W,Y8[I]);                                                      00070904
70906      THREE(W,Y8);                                                                00070906
70908        END ELSE                                                                  00070908
70910        BEGIN                                                                     00070910
70912      Y8[I]:=Y8[I]-180;                                                           00070912
70914      WRITE(E2,/,I,W,Y8[I]);                                                      00070914
70916      THREE(W,Y8);                                                                00070916
70918        END;                                                                      00070918
71100      END;                                                                        00071100
71200      END;                                                                        00071200
71300     PROCEDURE PGRAPH2;                                                           00071300
71400       BEGIN                                                                      00071400
71500     REAL NNT,DNT;                                                                00071500
71600     Y8[0]:=0;                                                                    00071600
71700      FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                         00071700
71800      BEGIN                                                                       00071800
71900      I:=*+1;                                                                     00071900
72000      NNT:=(I1(W,DA,A)/R1(W,DA,A))-(I2(W,DB,B)/R2(W,DB,B));                       00072000
72100      DNT:=1+(I1(W,DA,A)/R1(W,DA,A))*(I2(W,DB,B)/R2(W,DB,B));                     00072100
72200     Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                  00072200
72210      IF Y8[I]<=0 THEN                                                            00072210
```

```
72220       BEGIN                                                                                00072220
72300      WRITE(E2,/,I,W,Y8[I]);                                                                00072300
72400      THREE(W,Y8);                                                                          00072400
72410       END ELSE                                                                             00072410
72420       BEGIN                                                                                00072420
72421      Y8[I]:=Y8[I]-180;                                                                     00072421
72422      WRITE(E2,/,I,W,Y8[I]);                                                                00072422
72424      THREE(W,Y8);                                                                          00072424
72426       END;                                                                                 00072426
72500      END;                                                                                  00072500
72600      END;                                                                                  00072600
72700     PROCEDURE PGRAPH3;                                                                     00072700
72800      BEGIN                                                                                00072800
72900      REAL NNT,DNT;                                                                        00072900
73000      Y8[0]:=0;                                                                            00073000
73100      FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                                  00073100
73200      BEGIN                                                                                00073200
73300        I:=*+1;                                                                            00073300
73400        NNT:=(I2(W,DA,A)/R2(W,DA,A))-(I1(W,DB,B)/R1(W,DB,B));                              00073400
73500        DNT:=1+(I2(W,DA,A)/R2(W,DA,A))*(I1(W,DB,B)/R1(W,DB,B));                            00073500
73600       Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                         00073600
73700      IF Y8[I]<=0 THEN                                                                     00073700
73702       BEGIN                                                                               00073702
73704      WRITE(E2,/,I,W,Y8[I]);                                                               00073704
73706      THREE(W,Y8);                                                                         00073706
73708       END ELSE                                                                            00073708
73710       BEGIN                                                                               00073710
73712      Y8[I]:=Y8[I]-180;                                                                    00073712
73714      WRITE(E2,/,I,W,Y8[I]);                                                               00073714
73716      THREE(W,Y8);                                                                         00073716
73718       END;                                                                               00073718
73900        END;                                                                              00073900
74000        END;                                                                              00074000
74100       PROCEDURE PGRAPH4;                                                                 00074100
74200        BEGIN                                                                             00074200
74300        REAL NNT,DNT;                                                                     00074300
74400        Y8[0]:=0;                                                                         00074400
74500        FOR W:=LF STEP W*1.1/20 UNTIL HF DO                                               00074500
74600        BEGIN                                                                             00074600
74700        I:=*+1;                                                                           00074700
74800        NNT:=(I2(W,DA,A)/R2(W,DA,A))-(I2(W,DB,B)/R2(W,DB,B));                             00074800
74900        DNT:=1+(I2(W,DA,A)/R2(W,DA,A))*(I2(W,DB,B)/R2(W,DB,B));                           00074900
75000       Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                        00075000
75100      IF Y8[I]<=0 THEN                                                                    00075100
75102       BEGIN                                                                              00075102
75104      WRITE(E2,/,I,W,Y8[I]);                                                              00075104
75106      THREE(W,Y8);                                                                        00075106
75108       END ELSE                                                                           00075108
75110       BEGIN                                                                              00075110
75112      Y8[I]:=Y8[I]-180;                                                                   00075112
75114      WRITE(E2,/,I,W,Y8[I]);                                                              00075114
75116      THREE(W,Y8);                                                                        00075116
75118       END;                                                                               00075118
75300        END;                                                                              00075300
75400        END;                                                                              00075400
75500      PROCEDURE THREE(W,Y8);VALUE W;REAL W;ARRAY Y8[0];                                   00075500
75600      BEGIN                                                                               00075600
77900       X:=W;                                                                              00077900
78000      IF Y8[I]<=Y8[I-1] AND Y8[I]<=0 THEN                                                 00078000
78100        BEGIN                                                                            00078100
78200      CHARMD;LINETO(X,Y8[I]);                                                             00078200
```

```
78300       WRITE(EE1,/,X,Y8[I]);                                                                00078300
78400         END                                                                                00078400
78500       ELSE IF Y8[I]>=Y8[I-1] AND Y8[I]<=0 THEN                                             00078500
78600         BEGIN                                                                              00078600
78700       Y8[I]:=-180+Y8[I];                                                                   00078700
78800         IF Y8[I]<=Y8[I-1] THEN                                                             00078800
78900           BEGIN                                                                            00078900
79000         CHARMD;LINETO(X,Y8[I]);                                                          00079000
79100         WRITE(EE1,7,X,Y8[I]);                                                            00079100
79200           END                                                                              00079200
79300         ELSE BEGIN                                                                         00079300
79400       Y8[I]:=-180+Y8[I];                                                                   00079400
79500         CHARMD;LINETO(X,Y8[I]);                                                          00079500
79600         WRITE(EE1,/,X,Y8[I]);                                                            00079600
79700           END;                                                                             00079700
79800         END;                                                                               00079800
81400       END;                                                                                 00081400
81500     DEFINE                                                                                 00081500
81600           GFDEC1=                                                                          00081600
81700       XRANGE(-250,1250);YRANGE(-250,1250);                                                 00081700
81800     W:=LF;AT(110,1000);TXTDIR(0);WRTARY(1,T2);                                             00081800
81900       AT(250,540);TXTDIR(0);WRTARY(1,T1);                                                  00081900
82000       SUBFRM(100,600,1000,970);YRANGE(LG,HG);                                            00082000
82100         XRANGE(LF,HF);                                                                   00082100
82200         GRID;                                                                            00082200
82300         YAXIS(LF,GI);                                                                    00082300
82400         MODE(3);                                                                         00082400
82500         GRID;                                                                            00082500
82600         XAXIS(0,-FI);NOLINE#,                                                            00082600
82700           GFDEC2=                                                                        00082700
82800       XRANGE(-250,1250);YRANGE(-250,1250);                                               00082800
82900     W:=LF;AT(110,1000);TXTDIR(0);WRTARY(1,T2);                                           00082900
83000       AT(250,540);TXTDIR(0);WRTARY(1,T1);                                                00083000
83100     SUBFRM(100,600,1000,970);YRANGE(LG,HG);                                              00083100
83200     XRANGE(LF,HF);YAXIS(LF,GI);MODE(3);XAXIS(0,-FI);NOLINE#,                             00083200
83300           PFDEC1=                                                                        00083300
83400       ORGFRM;MODE(0);XRANGE(-250,1250);YRANGE(-250,1250);                                00083400
83500         W:=LF;AT(110,510);TXTDIR(0);WRTARY(1,T3);                                        00083500
83600       AT(800,5);TXTDIR(0);WRTARY(1,T4);                                                  00083600
83700       SUBFRM(100,110,1000,500);YRANGE(LP,HP);XRANGE(LF,HF);                              00083700
83800         GRID;                                                                            00083800
83900         YAXIS(LF,PI);                                                                    00083900
84000         MODE(3);                                                                         00084000
84100         GRID;                                                                            00084100
84200         XAXIS(-300,FI);NOLINE#,                                                          00084200
84300           PFDEC2=                                                                        00084300
84400       ORGFRM;MODE(0);XRANGE(-250,1250);YRANGE(-250,1250);                                00084400
84500         W:=LF;AT(110,510);TXTDIR(0);WRTARY(1,T3);                                        00084500
84600       AT(800,5);TXTDIR(0);WRTARY(1,T4);                                                  00084600
84700       SUBFRM(100,110,1000,500);YRANGE(LP,HP);XRANGE(LF,HF);                              00084700
84800     YAXIS(LF,PI);MODE(3);XAXIS(-300,FI);NOLINE#,                                         00084800
84900           BODESCALE=                                                                     00084900
85000       WRITE(FILE97,<"FREQUENCY RANGE?">);                                                00085000
85100       READ(FILE97,/,LF,HF);                                                              00085100
85200       WRITE(FILE97,<"GAIN RANGE?">);                                                     00085200
85300       READ(FILE97,/,LG,HG);                                                              00085300
85400       WRITE(FILE97,<"PHASE RANGE?">);                                                    00085400
85500       READ(FILE97,/,LP,HP);                                                              00085500
85600       WRITE(FILE97,<"GAIN,PHASE AND FREQUENCY INCREMENT?">);                             00085600
85700       READ(FILE97,/,GI,PI,FI)#,                                                          00085700
85800           CHANGEBODESCALE=                                                               00085800
```

```
85900       WHEN(10);
86000      WRITE(FILE97,<"DO YOU WANT TO CHANGE THE SCALE?">);
86100      READ(FILE97,/,SC);
86200      IF SC="Y" THEN BODE(W,DA,DB,A,B)#;
86300     COF:=180/3.14157;
86400       STARTP;
86500     DEVICE(3,1,0,0);EDGESR(1.312);
86600     IF DA<3 THEN
86700                BEGIN
86800          IF DB<3 THEN
86900                     BEGIN
87000       LABEL BODE1;
87100      BODESCALE;
87200          WRITE(FILE97,<"IS GRID WANTED?">);
87300          READ(FILE97,/,GR);
87400     FRAME("GRAPH");
87500     IF GR="Y" THEN
87600                          BEGIN
87700                       GFDEC1;
87800      MGRAPH1;
87900      PFDEC1;
88000      PGRAPH1;
88100      ENDP;
88200      CHANGEBODESCALE;
88300     END
88400      ELSE
88500      BEGIN
88600      GFDEC2;
88700      MGRAPH1;
88800      PFDEC2;
88900      PGRAPH1;
89000       ENDP;
89100      CHANGEBODESCALE;
89200       END;
89300     END
89400       ELSE
89500                     BEGIN
89600      BODESCALE;
89700          WRITE(FILE97,<"IS GRID WANTED?">);
89800          READ(FILE97,/,GR);
89900     FRAME("GRAPH");
90000     IF GR="Y" THEN
90100                          BEGIN
90200                       GFDEC1;
90300        MGRAPH2;
90400         PFDEC1;
90500         PGRAPH2;
90600         ENDP;
90700         CHANGEBODESCALE;
90800        END
90900        ELSE
91000         BEGIN
91100        GFDEC2;
91200         MGRAPH2;
91300         PFDEC2;
91400         PGRAPH2;
91500          ENDP;
91600          CHANGEBODESCALE;
91700        END;
91800         END;
91900     END
```

```
92000       ELSE
92100                 BEGIN
92200       IF DB<4 OR DB=4 THEN
92300                   BEGIN
92400         LABEL BODE1;
92500          WRITE(FILE97,<"IS GRID WANTED?">);
92600          READ(FILE97,/,GR);
92700     FRAME("GRAPH");
92800     IF GR="Y" THEN
92900                       BEGIN
93000                     GFDEC1;
93100         MGRAPH3;
93200         PFDEC1;
93300          PGRAPH3;
93400         ENDP;
93500          CHANGEBODESCALE;
93600       END
93700      ELSE
93800       BEGIN
93900       GFDEC2;
94000       MGRAPH3;
94100        PFDEC2;
94200       PGRAPH3;
94300          ENDP;
94400        CHANGEBODESCALE;
94500      END;
94600     END
94700     ELSE
94800                     BEGIN
94900      BODESCALE;
95000         WRITE(FILE97,<"IS GRID WANTED?">);
95100          READ(FILE97,/,GR);
95200     FRAME("GRAPH");
95300     IF GR="Y" THEN
95400                       BEGIN
95500                     GFDEC1;
95600         MGRAPH4;
95700         PFDEC1;
95800          PGRAPH4;
95900          ENDP;
96000          CHANGEBODESCALE;
96100         END
96200         ELSE
96300         BEGIN
96400         GFDEC2;
96500       MGRAPH4;
96600         PFDEC2;
96700         PGRAPH4;
96800           ENDP;
96900           CHANGEBODESCALE;
97000           END;
97100            END;
97200         END;
97300          END;
97400     COMMENT **** PROCEDURE NYQUIST ****;
97500     PROCEDURE NYQUIST(W,DA,DB,A,B);VALUE W,DA,DB;REAL DA,DB;
97600     REAL W;ARRAY A,B[0];
97700          BEGIN
97800           FILE EMI(KIND=DISK,MAXRECSIZE=14);
97900     REAL LF,HF,FI,LY,HY,YI,LX,HX,XI,DEM;
98000        VALUE ARRAY T1("NYQUIST DIAGRAM Y(W) VS X(W)");
```

```
98100        VALUE ARRAY T2("Y");                                                                                00098100
98200        VALUE ARRAY T3("X");                                                                                00098200
98300     REAL SC,GR,G,X,Y;                                                                                      00098300
98400     DEFINE                                                                                                 00098400
98500            NYQUISTSCALE=                                                                                   00098500
98600     WRITE(FILE97,<"YRANGE AND Y INCREMENT?">);                                                             00098600
98700           READ(FILE97,/,LY,HY,YI);                                                                        00098700
98800           WRITE(FILE97,<"XRANGE AND X INCREMENT?">);                                                      00098800
98900           READ(FILE97,/,LX,HX,XI);                                                                        00098900
99000     WRITE(FILE97,<"FREQUENCY RANGE AND INCREMENT?">);                                                      00099000
99100           READ(FILE97,/,LF,HF,FI)#,                                                                       00099100
99200            CHANGENYQUISTSCALE=                                                                            00099200
99300       WHEN(10);                                                                                           00099300
99400     WRITE(FILE97,<"DO YOU WANT TO CHANGE THE SCALE?">);                                                    00099400
99500              READ(FILE97,/,SC);                                                                           00099500
99600     IF SC="Y" THEN NYQUIST(W,DA,DB,A,B)#,                                                                  00099600
99700              NFDEC1=                                                                                       00099700
99800      W:=LF;AT(530,900);TXTDIR(0);WRTARY(1,T1);                                                           00099800
99900       AT(510,1000);TXTDIR(0);WRTARY(1,T2);                                                               00099900
100000      AT(950,450);TXTDIR(0);WRTARY(1,T3);                                                                00100000
100100         YRANGE(LY,HY);XRANGE(LX,HX);GRID;YAXIS(0,YI);GRID;                                              00100100
100200       XAXIS(0,XI);NOLINE#,                                                                              00100200
100300              NFDEC2=                                                                                     00100300
100400      W:=LF;AT(530,900);TXTDIR(0);WRTARY(1,T1);                                                          00100400
100500       AT(520,1000);TXTDIR(0);WRTARY(1,T2);                                                              00100500
100600       AT(950,430);TXTDIR(0);WRTARY(1,T3);                                                               00100600
100700         YRANGE(LY,HY);XRANGE(LX,HX);YAXIS(0,YI);XAXIS(0,XI);NOLINE#,                                    00100700
100800              NPLOT=                                                                                      00100800
100900     IF X<=HX OR X>=LX THEN                                                                               00100900
101000     IF Y<=HY OR Y>=LY THEN                                                                               00101000
101100          CHARMD;                                                                                         00101100
101200        LINETO(X,Y);WRITE(EMI,/,W,X,Y)#;                                                                  00101200
101300        STARTP;                                                                                           00101300
101400      XRANGE(-250,1250);YRANGE(-250,1250);                                                                00101400
101500       SUBFRM(0,0,1000,1000);                                                                            00101500
101600     DEVICE(3,1,0,0);                                                                                     00101600
101700     IF DA<3 THEN                                                                                         00101700
101800              BEGIN                                                                                       00101800
101900        IF DB<3 THEN                                                                                      00101900
102000                 BEGIN                                                                                    00102000
102100        LABEL NYQUIST2;                                                                                   00102100
102200       NYQUISTSCALE;                                                                                      00102200
102300     WRITE(FILE97,<"IS GRID WANTED?">);                                                                   00102300
102400     READ(FILE97,/,GR);                                                                                   00102400
102500     FRAME("GRAPH");                                                                                      00102500
102600     IF GR="Y" THEN                                                                                       00102600
102700                          BEGIN                                                                           00102700
102800        NFDEC1;GO TO NYQUIST2;                                                                            00102800
102900                          END                                                                             00102900
103000            ELSE                                                                                          00103000
103100                          BEGIN                                                                           00103100
103200                   NFDEC2;GO TO NYQUIST2;                                                                 00103200
103300                          END;                                                                            00103300
103400       NYQUIST2:                    BEGIN                                                                00103400
103500           FOR W:=LF STEP FI UNTIL HF DO                                                                  00103500
103600                                        BEGIN                                                             00103600
103700     DEM:=R1(W,DB,B)**2+I1(W,DB,B)**2;                                                                    00103700
103800     IF DEM NEQ 0 THEN BEGIN                                                                              00103800
103900     X:=(R1(W,DA,A)*R1(W,DB,B)+I1(W,DA,A)*I1(W,DB,B))/DEM;                                                00103900
104000     Y:=(I1(W,DA,A)*R1(W,DB,B)-R1(W,DA,A)*I1(W,DB,B))/DEM;                                                00104000
104100     NPLOT;                                                                                               00104100
```

```
104200                        END;                                                    00104200
104300                                END;                                            00104300
104400    ENDP;CHANGENYQUISTSCALE;                                                    00104400
104500                                    END;                                        00104500
104600                        END                                                     00104600
104700    ELSE                                                                        00104700
104800                        BEGIN                                                   00104800
104900        LABEL NYQUIST2;                                                         00104900
105000       NYQUISTSCALE;                                                            00105000
105100    WRITE(FILE97,<"IS GRID WANTED?">);                                          00105100
105200    READ(FILE97,/,GR);                                                          00105200
105300    FRAME("GRAPH");                                                             00105300
105400    IF GR="Y" THEN                                                              00105400
105500                        BEGIN                                                   00105500
105600       NFDEC1;GO TO NYQUIST2;                                                   00105600
105700                        END                                                     00105700
105800            ELSE                                                              00105800
105900                        BEGIN                                                   00105900
106000                  NFDEC2;GO TO NYQUIST2;                                        00106000
106100                        END;                                                    00106100
106200       NYQUIST2:                    BEGIN                                     00106200
106300          FOR W:=LF STEP FI UNTIL HF DO                                        00106300
106400                                     BEGIN                                     00106400
106500    DEM:=R2(W,DB,B)**2+I2(W,DB,B)**2;                                           00106500
106600    IF DEM NEQ 0 THEN     BEGIN                                                 00106600
106700    X:=(R1(W,DA,A)*R2(W,DB,B)+I1(W,DA,A)*I2(W,DB,B))/DEM;                       00106700
106800    Y:=(I1(W,DA,A)*R2(W,DB,B)-R1(W,DA,A)*I2(W,DB,B))/DEM;                       00106800
106900    NPLOT;                                                                      00106900
107000                         END;                                                 00107000
107100                                END;                                           00107100
107200    ENDP;CHANGENYQUISTSCALE;                                                    00107200
107300                                    END;                                       00107300
107400                        END;                                                    00107400
107500                  END                                                         00107500
107600    ELSE                                                                        00107600
107700                  BEGIN                                                       00107700
107800      IF DB<3 THEN                                                            00107800
107900                        BEGIN                                                   00107900
108000        LABEL NYQUIST2;                                                       00108000
108100       NYQUISTSCALE;                                                          00108100
108200    WRITE(FILE97,<"IS GRID WANTED?">);                                        00108200
108300    READ(FILE97,/,GR);                                                        00108300
108400    FRAME("GRAPH");                                                           00108400
108500    IF GR="Y"  THEN                                                           00108500
108600                                    BEGIN                                     00108600
108700       NFDEC1;GO TO NYQUIST2;                                                 00108700
108800                                    END                                       00108800
108900            ELSE                                                            00108900
109000                                    BEGIN                                   00109000
109100                  NFDEC2;GO TO NYQUIST2;                                    00109100
109200                                    END;                                    00109200
109300        NYQUIST2:                   BEGIN                                   00109300
109400          FOR W:=LF STEP FI UNTIL HF DO                                     00109400
109500                                     BEGIN                                  00109500
109600    DEM:=R1(W,DB,B)**2+I1(W,DB,B)**2;                                         00109600
109700    IF DEM NEQ 0 THEN          BEGIN                                          00109700
109800    X:=(R2(W,DA,A)*R1(W,DB,B)+I2(W,DA,A)*I1(W,DB,B))/DEM;                     00109800
109900    Y:=(I2(W,DA,A)*R1(W,DB,B)-R2(W,DA,A)*I1(W,DB,B))/DEM;                     00109900
110000    NPLOT;                                                                    00110000
110100                         END;                                               00110100
110200                                END;                                         00110200
```

```
110300     ENDP;CHANGENYQUISTSCALE;
110400                             END;
110500                           END
110600       ELSE
110700                           BEGIN
110800         LABEL NYQUIST2;
110900       NYQUISTSCALE;
111000     WRITE(FILE97,<"IS GRID WANTED?">);
111100     READ(FILE97,/,GR);
111200     FRAME("GRAPH");
111300     IF GR="Y" THEN
111400                             BEGIN
111500       NFDEC1;GO TO NYQUIST2;
111600                             END
111700        ELSE
111800                             BEGIN
111900                  NFDEC2;GO TO NYQUIST2;
112000                             END;
112100       NYQUIST2:             BEGIN
112200          FOR W:=LF STEP FI UNTIL HF DO
112300                                BEGIN
112400     DEM:=R2(W,DB,B)**2+I2(W,DB,B)**2;
112500     IF DEM NEQ 0 THEN  BEGIN
112600     X:=(R2(W,DA,A)*R2(W,DB,B)+I2(W,DA,A)*I2(W,DB,B))/DEM;
112700     Y:=(I2(W,DA,A)*R2(W,DB,B)-R2(W,DA,A)*I2(W,DB,B))/DEM;
112800     NPLOT;
112900                           END;
113000                                 END;
113100     ENDP;CHANGENYQUISTSCALE;
113200                             END;
113300                           END;
113400                        END;
113500         END;
113600      COMMENT **** PROCEDURE NICHOLS ****;
113700     PROCEDURE NICHOLS(W,DA,DB,A,B);VALUE W,DA,DB;REAL DA,DB;
113800     REAL W;ARRAY A,B[0];
113900         BEGIN
114000     FILE O(KIND=DISK,MAXRECSIZE=40);
114100     FILE CHI(KIND=DISK,MAXRECSIZE=40);
114200     FILE PRS(KIND=DISK,MAXRECSIZE=40);
114300     FILE GRS(KIND=DISK,MAXRECSIZE=40);
114400     VALUE ARRAY T1("DBS"),T2("DEGS");
114500      ARRAY Y8[0:10000];
114600      REAL NNT,DNT;
114700      INTEGER I;
114800     REAL ML,MH,MI,PHAL,PHAH,PHAI,FAS;
114900     REAL M,F,P1,G,B1,GI,FI,SC,GR;
115000     REAL X,Y,COF,ICOF;
115100     REAL LF,HF,M1,G1;
115200     REAL GAIN1,GAIN2,GAININ;
115300     REAL FA1,FA2,FAI;
115400     REAL LG,HG,LP,HP,PI;
115500     REAL GAIN,P,F1;
115600     PROCEDURE SIX(Y8);ARRAY Y8[0];FORWARD;
115700     PROCEDURE SIX(Y8);ARRAY Y8[0];
115800      BEGIN
115900      IF GAIN>0 THEN
116000       BEGIN
116100         Y:=10*LOG(GAIN);
118300      IF Y8[I]<=Y8[I-1] AND Y8[I]<=0 THEN
118400        BEGIN
```

```
118500       CHARMD;LINETO(Y8[I],Y);                                                                    00118500
118600       WRITE(CHI,7,Y8[I],Y);                                                                      00118600
118700        END                                                                                       00118700
118800      ELSE IF Y8[I]>=Y8[I-1] AND Y8[I]<=0 THEN                                                    00118800
118900       BEGIN                                                                                     00118900
119000      Y8[I]:=-180+Y8[I];                                                                         00119000
119100        IF Y8[I]<=Y8[I-1] THEN                                                                   00119100
119200         BEGIN                                                                                   00119200
119300       CHARMD;LINETO(Y8[I],Y);                                                                  00119300
119400       WRITE(CHI,7,Y8[I],Y);                                                                    00119400
119500        END                                                                                     00119500
119600       ELSE BEGIN                                                                               00119600
119700      Y8[I]:=-180+Y8[I];                                                                         00119700
119800       CHARMD;LINETO(Y8[I],Y);                                                                  00119800
119900      WRITE(CHI,7,Y8[I],Y);                                                                      00119900
120000           END;                                                                                00120000
120100       END;                                                                                     00120100
121600       END;                                                                                     00121600
121800      END;                                                                                      00121800
121900          DEFINE                                                                               00121900
122000                 NICHOL=                                                                       00122000
122100     TXTSIZ(0);TXTDIR(0);TXT("NICHOL")#,                                                        00122100
122200                 PHASECONTOURS=                                                                00122200
122300     G1:=TAN(P-F)/(SIN(P)-COS(P)*TAN(P-F));                                                     00122300
122400     IF G1>0 THEN                                                                               00122400
122500                 BEGIN                                                                         00122500
122600     G:=20*LOG(G1);                                                                             00122600
122700     X:=P*COF;Y:=G;CHARMD;LINETO(X,Y);                                                          00122700
122800     WRITE(PRS,/,F,G1,X,Y);                                                                     00122800
122900                 END#,                                                                         00122900
123000                  GAINCONTOURS=                                                                00123000
123100      M1:=10**(M/20);G1:=10**(G/20);                                                            00123100
123200     B1:=-(M1*M1+G1*G1*(M1*M1-1))/(2*M1*M1*G1);                                                 00123200
123300     IF B1<=1 AND B1>=-1 THEN                                                                   00123300
123400                 BEGIN                                                                         00123400
123500     X:=-ARCCOS(B1)*COF;Y:=G;CHARMD;LINETO(X,Y);WRITE(GRS,/,M,X,Y);                             00123500
123600                 END#;                                                                         00123600
123700     STARTP;DEVICE(3,1,0,0);                                                                    00123700
123800       XRANGE(-250,1250);YRANGE(-250,1250);                                                     00123800
123900     COF:=180/3.14157;                                                                          00123900
124000     ICOF:=3.14157/180;                                                                         00124000
124100     WRITE(FILE97,<"GAIN RANGE AND INCREMENT?">);                                               00124100
124200     READ(FILE97,/,LG,HG,GI);                                                                   00124200
124300     WRITE(FILE97,<"PHASE RANGE AND INCREMENT?">);                                              00124300
124400     READ(FILE97,/,LP,HP,PI);                                                                   00124400
124500     WRITE(FILE97,<"GAIN RANGE VARIATION AND INCREMENT?">);                                     00124500
124600     READ(FILE97,/,GAIN1,GAIN2,GAININ);                                                         00124600
124700      WRITE(FILE97,<"PHASE RANGE VARIATION AND INCREMENT?">);                                   00124700
124800      READ(FILE97,/,FA1,FA2,FAI);                                                               00124800
124900     WRITE(FILE97,<"RANGE AND INCREMENT OF CONSTANT GAIN?">);                                   00124900
125000     READ(FILE97,/,ML,MH,MI);                                                                   00125000
125100     WRITE(FILE97,<"RANGE AND INCREMENT OF CONSTANT PHASE?">);                                  00125100
125200     READ(FILE97,/,PHAL,PHAH,PHAI);                                                             00125200
125300     WRITE(FILE97,<"FREQUENCY RANGE AND INCREMENT?">);                                          00125300
125400     READ(FILE97,/,LF,HF,FI);                                                                   00125400
125500     FRAME("GRAPH");                                                                            00125500
125600     AT(30,1000);WRTARY(1,T1);AT(1000,30);WRTARY(1,T2);                                         00125600
125700        SUBFRM(80,80,980,980);                                                                  00125700
125800     YRANGE(LG,HG);XRANGE(LP,HP);GRID;YAXIS(LP,GI);GRID;XAXIS(LG,PI);                           00125800
125900     NOLINE;                                                                                    00125900
126000     FOR M:=ML STEP MI UNTIL MH DO                                                              00126000
```

```
126100                BEGIN                                                                  00126100
126200      NOLINE;                                                                          00126200
126300      FOR G:=GAIN1 STEP GAININ UNTIL GAIN2 DO                                          00126300
126400                    BEGIN                                                            00126400
126500          GAINCONTOURS;                                                              00126500
126600                    END;                                                             00126600
126700      TXTDIR(0);                                                                     00126700
126800      FLOATN(3,1,M);                                                                 00126800
126900                END;                                                                 00126900
127000      NOLINE;                                                                        00127000
127100      NOLINE;                                                                        00127100
127200      FOR F:=PHAL*ICOF STEP PHAI*ICOF UNTIL PHAH*ICOF DO                             00127200
127300                BEGIN                                                              00127300
127400          NOLINE;                                                                  00127400
127500      FOR P:=FA1*ICOF STEP FAI*ICOF UNTIL FA2*ICOF DO                                00127500
127600                    BEGIN                                                          00127600
127700              PHASECONTOURS;                                                       00127700
127800                    END;                                                           00127800
127900      F1:=F*COF;TXTDIR(0);FIXEDN(4,F1);                                              00127900
128000                END;                                                               00128000
128100       IF DA<3 THEN                                                              00128100
128200         BEGIN                                                                   00128200
128300          IF DB<3 THEN                                                           00128300
128400           BEGIN                                                                 00128400
128500              W:=LF;NOLINE;                                                      00128500
128600            FOR W:=LF STEP FI UNTIL HF DO                                        00128600
128700                                  BEGIN                                          00128700
128800       GAIN:=((R1(W,DA,A)**2+I1(W,DA,A)**2)/(R1(W,DB,B)**2+I1(W,DB,B)**2));     00128800
128900        I:=I+1;                                                                  00128900
129000        Y8[0]:=0;                                                                00129000
129100       NNT:=(I1(W,DA,A)/R1(W,DA,A))-(I1(W,DB,B)/R1(W,DB,B));                     00129100
129200       DNT:=1+(I1(W,DA,A)/R1(W,DA,A))*(I1(W,DB,B)/R1(W,DB,B));                   00129200
129300      Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                00129300
129400       IF Y8[I]<=0 THEN BEGIN                                                    00129400
129500       WRITE(U,/,W,GAIN,Y8[I]);                                                  00129500
129600       SIX(Y8);                                                                  00129600
129700                          END ELSE                                               00129700
129800         BEGIN                                                                   00129800
129900       Y8[I]:=-180+Y8[I];                                                        00129900
130000       WRITE(U,/,W,GAIN,Y8[I]);                                                  00130000
130100       SIX(Y8);                                                                  00130100
130200         END;                                                                    00130200
130300         END;                                                                    00130300
130400         NICHOL;                                                                 00130400
130500      END                                                                        00130500
130600       ELSE IF DB>=4 THEN                                                        00130600
130700          BEGIN                                                                  00130700
130800              W:=LF;NOLINE;                                                      00130800
130900            FOR W:=LF STEP FI UNTIL HF DO                                        00130900
131000                                  BEGIN                                          00131000
131100       GAIN:=((R1(W,DA,A)**2+I1(W,DA,A)**2)/(R2(W,DB,B)**2+I2(W,DB,B)**2));     00131100
131200        I:=I+1;                                                                  00131200
131300        Y8[0]:=0;                                                                00131300
131400       NNT:=(I1(W,DA,A)/R1(W,DA,A))-(I2(W,DB,B)/R2(W,DB,B));                     00131400
131500       DNT:=1+(I1(W,DA,A)/R1(W,DA,A))*(I2(W,DB,B)/R2(W,DB,B));                   00131500
131600      Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                00131600
131700       IF Y8[I]<=0 THEN BEGIN                                                    00131700
131800       WRITE(U,/,W,GAIN,Y8[I]);                                                  00131800
131900       SIX(Y8);                                                                  00131900
132000                          END ELSE                                               00132000
132100         BEGIN                                                                   00132100
```

```
132200      Y8[I]:=-180+Y8[I];                                                                  00132200
132300      WRITE(U,/,W,GAIN,Y8[I]);                                                            00132300
132400      SIX(Y8);                                                                            00132400
132500        END;                                                                              00132500
132600       END;                                                                               00132600
132700       NICHOL;                                                                            00132700
132800     END;                                                                                 00132800
132900     END                                                                                  00132900
133000      ELSE                                                                                00133000
133100      IF DA>=4 THEN                                                                       00133100
133200       BEGIN                                                                              00133200
133300      IF DB<3 THEN                                                                        00133300
133400       BEGIN                                                                              00133400
133500            W:=LF;NOLINE;                                                                 00133500
133600        FOR W:=LF STEP FI UNTIL HF DO                                                     00133600
133700                                  BEGIN                                                 00133700
133800      GAIN:=((R2(W,DA,A)**2+I2(W,DA,A)**2)/(R1(W,DB,B)**2+I1(W,DB,B)**2));               00133800
133900      I:=*+1;                                                                             00133900
134000       Y8[0]:=0;                                                                          00134000
134100        NNT:=(I2(W,DA,A)/R2(W,DA,A))-(I1(W,DB,B)/R1(W,DB,B));                             00134100
134200        DNT:=1+(I2(W,DA,A)/R2(W,DA,A))*(I1(W,DB,B)/R1(W,DB,B));                           00134200
134300       Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                        00134300
134400      IF Y8[I]<=0 THEN BEGIN                                                              00134400
134500      WRITE(U,/,W,GAIN,Y8[I]);                                                            00134500
134600      SIX(Y8);                                                                            00134600
134700                        END ELSE                                                          00134700
134800       BEGIN                                                                              00134800
134900      Y8[I]:=-180+Y8[I];                                                                  00134900
135000      WRITE(U,/,W,GAIN,Y8[I]);                                                            00135000
135100      SIX(Y8);                                                                            00135100
135200        END;                                                                              00135200
135300        END;                                                                              00135300
135400       NICHOL;                                                                            00135400
135500       END                                                                                00135500
135600       ELSE                                                                               00135600
135700       IF DB>=4 THEN                                                                      00135700
135800        BEGIN                                                                             00135800
135900            W:=LF;NOLINE;                                                                 00135900
136000        FOR W:=LF STEP FI UNTIL HF DO                                                     00136000
136100                                  BEGIN                                                 00136100
136200      GAIN:=((R2(W,DA,A)**2+I2(W,DA,A)**2)/(R2(W,DB,B)**2+I2(W,DB,B)**2));               00136200
136300      I:=*+1;                                                                             00136300
136400        Y8[0]:=0;                                                                         00136400
136500        NNT:=(I2(W,DA,A)/R2(W,DA,A))-(I2(W,DB,B)/R2(W,DB,B));                             00136500
136600        DNT:=1+(I2(W,DA,A)/R2(W,DA,A))*(I2(W,DB,B)/R2(W,DB,B));                           00136600
136700       Y8[I]:=ARCTAN(NNT/DNT)*COF;                                                        00136700
136800      IF Y8[I]<=0 THEN BEGIN                                                              00136800
136900      WRITE(U,/,W,GAIN,Y8[I]);                                                            00136900
137000      SIX(Y8);                                                                            00137000
137100                        END ELSE                                                          00137100
137200       BEGIN                                                                              00137200
137300      Y8[I]:=-180+Y8[I];                                                                  00137300
137400      WRITE(U,/,W,GAIN,Y8[I]);                                                            00137400
137500      SIX(Y8);                                                                            00137500
137600        END;                                                                              00137600
137700       END;                                                                               00137700
137800       NICHOL;                                                                            00137800
137900       END;                                                                               00137900
138000     END;                                                                                 00138000
138100       WHEN(10);                                                                          00138100
138200     WRITE(FILE97,<"DO YOU WANT TO CHANGE THE SCALE?">);                                  00138200
```

```
138300     READ(FILE97,/,SC);                                                                  00138300
138400     IF SC="Y" THEN NICHOLS(W,DA,DB,A,B);                                                00138400
138500           END;                                                                          00138500
138600      COMMENT **** PROCEDURE RLOCUS ****;                                                00138600
138700     PROCEDURE RLOCUS(DPOLY1,DPOLY2,POLY1,POLY2);                                        00138700
138800     VALUE DPOLY1,DPOLY2;REAL DPOLY1,DPOLY2;                                             00138800
138900     ARRAY POLY2,POLY1[0];                                                               00138900
139000           BEGIN                                                                         00139000
139100     FILE E(KIND=DISK,MAXRECSIZE=40);                                                    00139100
139200     REAL X,Y,SC,GR,K1,KI,J,K;                                                           00139200
139300     REAL LY,HY,YI,LX,HX,XI;                                                             00139300
139400     VALUE ARRAY T1("**ROOT LOCI**");                                                    00139400
139500      PROCEDURE REALPOLYZEROFINDER(DEGREE,P,ZEROR,ZEROI);                                00139500
139600     VALUE DEGREE;INTEGER DEGREE;ARRAY P,ZEROR,ZEROI[0];FORWARD;                         00139600
139700     PROCEDURE      REALPOLYZEROFINDER(DEGREE,P,ZEROR,ZEROI);                            00139700
139800     VALUE          DEGREE ;                                                             00139800
139900     INTEGER        DEGREE ;                                                             00139900
140000     ARRAY          P,ZEROR,ZEROI[0];                                                    00140000
140100     COMMENT        PURPOSE                                                              00140100
140200                                                                                         00140200
140300                       FIND ALL THE ZEROS OF A REAL POLYNOMIAL P                         00140300
140400                                                                                         00140400
140500                    DESCRIPTION OF PARAMETERS                                            00140500
140600                                                                                         00140600
140700                       DEGREE - DEGREE OF THE POLYNOMIAL                                 00140700
140800                       P      - ARRAY OF COEFFICIENTS OF THE POLYNOMIAL, THE             00140800
140900                                COEFFICIENT OF Z*J  BEING STORED IN P[N-J]               00140900
141000                                DIMENSION P[0:DEGREE]                                    00141000
141100                       ZEROR  - ARRAY OF REAL PART OF ZEROS OF THE POLYNOMIAL,          00141100
141200                                THE REAL PART OF THE K-TH ZERO BEING STORED              00141200
141300                                IN ZEROR[K]                                              00141300
141400                                DIMENSION  ZEROR[0:DEGREE]                               00141400
141500                       ZEROI  - ARRAY OF IMAGINARY PARTS OF ZEROS OF THE                 00141500
141600                                POLYNOMIAL, THE IMAGINARY PART OF K-TH ZERO              00141600
141700                                BEING STORED IN ZEROI[K]                                 00141700
141800                                DIMENSION  ZEROI[0:DEGREE]                               00141800
141900                                                                                         00141900
142000                    METHOD                                                               00142000
142100                                                                                         00142100
142200                       A THREE-STAGE VARIABLE-SHIFT ITERATION DUE TO M.A.                00142200
142300        JENKINS AND J.F. TRAUB IS USED.  THE ZEROES ARE CALCULATED ONE OR                00142300
142400        TWO AT A TIME IN ROUGHLY INCREASING ORDER OF MAGNITUDE USING REAL                00142400
142500        ARITHMETIC WHEREVER POSSIBLE.  FAILURE OF THE ITERATION TO                       00142500
142600        CONVERGE FOR A ROOT WILL CAUSE AN ERROR MESSAGE TO BE PRINTED AND                00142600
142700        THE PROCEDURE EXITED WITH THE ZEROES ALREADY FOUND STORED IN THE                 00142700
142800        ARRAYS ZEROR AND ZEROI.  FAILURES WILL BE RARE.                                  00142800
142900                                                                                         00142900
143000                    REFERENCE                                                            00143000
143100                                                                                         00143100
143200                       THIS PROCEDURE IS DUE TO M.A. JENKINS AND APPEARS IN              00143200
143300      "THREE-STAGE VARIABLE-SHIFT ITERATIONS FOR THE SOLUTION OF                         00143300
143400       POLYNOMIAL EQUATIONS WITH A POSTERIORI ERROR BOUNDS FOR THE ZEROES"               00143400
143500       TECHNICAL REPORT NO. CS 138, PP 125-133, AUGUST 1969,                            00143500
143600       COMPUTER SCIENCE DEPT., STANFORD UNIVERSITY, STANFORD,CALIF 94305 ;              00143600
143700         BEGIN                                                                          00143700
143800      COMMENT        INITIALIZATION OF MACHINE DEPENDENT CONSTANTS ;                    00143800
143900      DEFINE         BASE = 8 #, % FLOATING POINT ARITHMETIC IS BASE 8                  00143900
144000                     PRECISION = 13 #, % NO. OF OCTAL DIGITS IN NUMBER                  00144000
144100                                       % REPRESENTATION                                  00144100
144200                     EXPRNGHI = 63 #,  % UPPER EXPONENT RANGE                          00144200
144300                     LOEXPRNG = -51 #, % LOWER EXPONENT RANGE                          00144300
```

```
144400                  ROUNDED = TRUE #, % ARITHMETIC IS ROUNDED                                  00144400
144500                  ETA = 7.27595761420@-12 #, % RELATIVE REPRESENTATION ERROR                 00144500
144600                  INFINITY = 4.31359146672@68 #, % LARGEST FLOATING PT. NO.                  00144600
144700                  SMALLESTNO = 8.75811540210@-47 #, % SMALLEST FL. PT. NO.                   00144700
144800                  ZETA = 5.49755813890@11 #, % ZETA = BASE**PRECISION                        00144800
144900                  CONSTL = 3.65185988877@-34 #,                                              00144900
145000   %                       CONSTL = ZETA * SMALLESTNO * BASE                                 00145000
145100                  LNBASE = 2.07944154167 #, %   LN(8)                                       00145100
145200                  MRE = 7.27595961420@-12#, % MAX. RELATIVE ROUNDOFF ERROR                   00145200
145300                                            % FOR FL. PT. ADDITION AND MULT.                 00145300
145400                  ARE = 7.27595761420@-12 #,% ARE SAME ON B6700                              00145400
145500                  W1 =  1.45519152284@-11 #, % W1 = MRE+ARE                                  00145500
145600                  W2 =  7.27595761420@-12 #, % W2 = MRE                                      00145600
145700                  W3 =  6.54836185280@-11 #, % W3 = 5*MRE + 4*ARE                            00145700
145800                  W4 =  5.09317032993@-11 #, % W4 = 5*MRE + 2*ARE                            00145800
145900                  W5 =  1.45519152284@-11 #; % W5 = 2*MRE                                    00145900
146000    REAL          M1,M2,M3,M4,SR,SI,A,B,C,D,E,F,G,H,A1,A2,A3,A6,A7,                          00146000
146100                  MP,U,V,DU,DV,UI,VI,SVU,SVV;                                                00146100
146200    INTEGER       N,TYPE,ND,RANDOMSTART;                                                     00146200
146300    ARRAY         QP[0:DEGREE], K,QK,SVK[0:DEGREE-1];                                        00146300
146400   LABEL          NEXTZERO,FAILURE,EXIT;                                                     00146400
146500   FILE           PRINTER (KIND =  7 ,BUFFERS = 1,MAXRECSIZE = 15);                          00146500
146600   FORMAT         ERR("FAILURE TO CONVERGE OCCURRED ON ",I2,"-TH ZERO");                     00146600
146700   REAL PROCEDURE CMOD(A,B) ;                                                                00146700
146800   VALUE          A,B ;                                                                      00146800
146900   REAL           A,B ;                                                                      00146900
147000   COMMENT        COMPLEX MODULUS OF (A,B) ;                                                 00147000
147100            BEGIN                                                                            00147100
147200                  A := ABS(A) ;                                                              00147200
147300                  B := ABS(B) ;                                                              00147300
147400                  CMOD := IF A LSS B THEN SQRT(1+(A/B)**2) * B ELSE                          00147400
147500                          IF A GTR B THEN SQRT(1+(B/A)**2) * A ELSE                          00147500
147600                          1.41421356237 * A;      COMMENT   SQRT(2) ;                        00147600
147700            END   CMOD ;                                                                     00147700
147800   PROCEDURE      CDIVIDE(AR,AI,BR,BI,CR,CI) ;                                               00147800
147900   VALUE          AR,AI,BR,BI ;                                                              00147900
148000   REAL           AR,AI,BR,BI,CR,CI ;                                                        00148000
148100   COMMENT        COMPLEX DIVISION - (CR,CI) = (AR,AI)/(BR,BI) ;                             00148100
148200            BEGIN                                                                            00148200
148300   REAL           D,R ;                                                                      00148300
148400                  IF BR = 0 AND BI = 0 THEN CR := CI := INFINITY ELSE                        00148400
148500                  IF ABS(BR) LSS ABS(BI) THEN                                                00148500
148600            BEGIN                                                                            00148600
148700                  R := BR/BI ;                                                               00148700
148800                  D := BR * R + BI ;                                                         00148800
148900                  CR := (AR * R + AI)/D ;                                                    00148900
149000                  CI := (AI * R - AR)/D ;                                                    00149000
149100            END                                                                              00149100
149200                  ELSE                                                                       00149200
149300            BEGIN                                                                            00149300
149400                  R := BI/BR ;                                                               00149400
149500                  D := BI * R + BR ;                                                         00149500
149600                  CR := (AI * R + AR)/D ;                                                    00149600
149700                  CI := (AI - AR * R)/D ;                                                    00149700
149800            END                                                                              00149800
149900            END   CDIVIDE ;                                                                  00149900
150000   PROCEDURE      PRESCALE(N,P) ;                                                            00150000
150100   VALUE          N ;                                                                        00150100
150200   INTEGER        N ;                                                                        00150200
150300   ARRAY          P[0] ;                                                                     00150300
150400   COMMENT        SCALES THE POLYNOMIAL TO AVOID OVERFLOW AND UNDERFLOW IF                   00150400
```

```
150500                  POSSIBLE ;                                                                  00150500
150600            BEGIN                                                                             00150600
150700      REAL        MAXM,MINM,S,SCALE;                                                        00150700
150800      INTEGER     L,I ;                                                                     00150800
150900                  MAXM := 0 ;                                                               00150900
151000                  MINM := INFINITY ;                                                        00151000
151100                  FOR I := 0 STEP 1 UNTIL N DO                                              00151100
151200            BEGIN                                                                           00151200
151300                  S := ABS(P[I]) ;                                                          00151300
151400                  MAXM := MAX(MAXM,S) ;                                                     00151400
151500                  IF S NEQ 0 THEN MINM := MIN(MINM,S) ;                                     00151500
151600            END ;                                                                           00151600
151700                  IF MINM LSS CONSTL OR MAXM GTR BASE ** (EXPRNGHI DIV 2)                   00151700
151800                  THEN                                                                      00151800
151900            BEGIN                                                                           00151900
152000                  S := CONSTL/MINM ;                                                        00152000
152100                  SCALE := IF S GTR 1 THEN (IF INFINITY/S LSS MAXM THEN 1.0                 00152100
152200                  ELSE S) ELSE (1/SQRT(MAXM)) * (1/SQRT(MINM)) ;                            00152200
152300                  L := LN(SCALE)/LNBASE ;                                                   00152300
152400                  IF L NEQ 0 THEN                                                           00152400
152500            BEGIN                                                                           00152500
152600                  S := BASE ** L ;                                                          00152600
152700                  FOR I := 0 STEP 1 UNTIL N DO                                              00152700
152800                  P[I] := P[I] * S ;                                                        00152800
152900            END                                                                             00152900
153000            END                                                                             00153000
153100            END   PRESCALE ;                                                                00153100
153200      REAL PROCEDURE CAUCHYBOUND(N,P) ;                                                     00153200
153300      VALUE       N ;                                                                       00153300
153400      INTEGER     N ;                                                                       00153400
153500      ARRAY       P[0] ;                                                                    00153500
153600      COMMENT     THIS PROCEDURE COMPUTES A LOWER BOUND ON THE MODULUS OF                   00153600
153700                  THE ZEROS OF THE POLYNOMIAL ;                                             00153700
153800            BEGIN                                                                           00153800
153900      INTEGER     I ;                                                                       00153900
154000      LABEL       L ;                                                                       00154000
154100      ARRAY       Q,PT[0:N] ;                                                               00154100
154200      REAL        X,XM,DX,F,DF,BND ;                                                        00154200
154300                  FOR I := 0 STEP 1 UNTIL N - 1 DO PT[I] := ABS(P[I]) ;                     00154300
154400                  PT[N] := -ABS(P[N]) ;                                                     00154400
154500      COMMENT     INITIAL GUESS FOR LOWER BOUND ;                                           00154500
154600                  X := EXP((LN(-PT[N]) - LN(PT[0]))/N) ;                                    00154600
154700      COMMENT     IF THE NEWTON STEP AT ZERO IS BETTER USE IT AS THE                        00154700
154800                  INITIAL GUESS ;                                                           00154800
154900                  IF PT[N-1] NEQ 0 THEN                                                     00154900
155000            BEGIN                                                                           00155000
155100                  XM := -PT[N]/PT[N-1] ;                                                    00155100
155200                  X := MIN(X,XM) ;                                                          00155200
155300            END ;                                                                           00155300
155400      COMMENT     CHOP THE INTERVAL (0,X) UNTIL  F LEQ 0 ;                                  00155400
155500      L:          XM := X * 0.1;                                                            00155500
155600                  F := PT[0] ;                                                              00155600
155700                  FOR I := 1 STEP 1 UNTIL N DO F := F * XM + PT[I] ;                        00155700
155800                  IF F GTR 0 THEN                                                           00155800
155900            BEGIN                                                                           00155900
156000                  X := XM ;                                                                 00156000
156100                  GO TO L                                                                   00156100
156200            END ;                                                                           00156200
156300      COMMENT     USE NEWTON RAPHSON TO FIND THE LOWER BOUND ;                              00156300
156400                  DX := X ;                                                                 00156400
156500                  FOR X := X WHILE ABS(DX/X) GTR  .005 DO                                   00156500
```

```
156600              BEGIN                                                                                   00156600
156700                   Q[0] := PT[0] ;                                                                    00156700
156800                   FOR I := 1 STEP 1 UNTIL N DO Q[I] := Q[I-1] * X + PT[I] ;                          00156800
156900                   F := Q[N] ;                                                                        00156900
157000                   DF := Q[0] ;                                                                       00157000
157100                   FOR I := 1 STEP 1 UNTIL N-1 DO DF := DF * X + Q[I] ;                               00157100
157200                   DX := F/DF ;                                                                       00157200
157300                   X := X - DX                                                                        00157300
157400              END  ;                                                                                  00157400
157500              END  CAUCHYBOUND := X ;                                                                 00157500
157600         END  CAUCHYBOUND ;                                                                           00157600
157700    PROCEDURE      QUADRATIC(A,B,C,SZR,SZI,LZR,LZI) ;                                                00157700
157800    VALUE          A,B,C ;                                                                           00157800
157900    REAL           A,B,C,SZR,SZI,LZR,LZI ;                                                           00157900
158000    COMMENT        FINDS THE ROOTS OF THE QUADRATIC A!Z*2+2!B!Z+C, AVOIDING                          00158000
158100                   UNNECESSARY OVERFLOW ;                                                            00158100
158200              BEGIN                                                                                   00158200
158300    REAL           D,E ;                                                                             00158300
158400                   IF A = 0 THEN                                                                     00158400
158500              BEGIN                                                                                   00158500
158600                   SZR := SZI := LZI := 0 ;                                                          00158600
158700                   LZR := IF B = 0 THEN 0 ELSE -C/(2*B)                                              00158700
158800              END                                                                                     00158800
158900                   ELSE                                                                              00158900
159000                   IF C = 0 AND B = 0 THEN SZR := SZI := LZR := LZI := 0 ELSE                         00159000
159100              BEGIN                                                                                   00159100
159200                   IF ABS(B) GEQ ABS(C) THEN                                                         00159200
159300                BEGIN                                                                                00159300
159400                   E := 1 - (A/B) * (C/B) ;                                                          00159400
159500                   D := SQRT(ABS(E)) * ABS(B)                                                         00159500
159600                END                                                                                  00159600
159700                   ELSE                                                                              00159700
159800                BEGIN                                                                                00159800
159900                   E := B * (B/ABS(C)) - (IF C GTR 0 THEN A ELSE -A) ;                                00159900
160000                   D := SQRT(ABS(E)) * SQRT(ABS(C))                                                   00160000
160100              END  ;                                                                                  00160100
160200                   IF E GEQ 0 THEN                                                                   00160200
160300                BEGIN                                                                                00160300
160400                   LZR := (IF B GEQ 0 THEN -B-D ELSE -B+D)/A ;                                        00160400
160500                   SZR := IF LZR = 0 THEN 0 ELSE (C/LZR)/A ;                                          00160500
160600                   LZI := SZI := 0                                                                   00160600
160700                END                                                                                  00160700
160800                   ELSE                                                                              00160800
160900                BEGIN                                                                                00160900
161000                   SZR := LZR := -B/A ;                                                              00161000
161100                   SZI := ABS(D/A) ;                                                                 00161100
161200                   LZI := - SZI                                                                      00161200
161300                END  ;                                                                               00161300
161400              END                                                                                     00161400
161500              END QUADRATIC ;                                                                         00161500
161600    BOOLEAN PROCEDURE PEVALANDTESTREAL(SR,QP) ;                                                      00161600
161700    VALUE          SR ;                                                                              00161700
161800    REAL           SR ;                                                                              00161800
161900    ARRAY          QP[0];                                                                            00161900
162000    COMMENT        EVALUATES P AT SR BY SYNTHETIC DIVISION PUTTING THE                               00162000
162100                   QUOTIENT POLYNOMIAL IN QP.  AN UPPERBOUND ON THE ROUNDOFF                         00162100
162200                   ERROR IS COMPUTED.  THE PROCEDURE IS TRUE IF THE MODULUS                          00162200
162300                   OF THE POLYNOMIAL VALUE IS OF THE SAME ORDER OF MAGNITUDE                         00162300
162400                   AS THE BOUND ON THE ROUNDOFF ERROR ;                                              00162400
162500              BEGIN                                                                                   00162500
162600    REAL           E,MS ;                                                                            00162600
```

```
162700     INTEGER          I ;                                                                  00162700
162800                      QP[0] := P[0] ;                                                      00162800
162900                      FOR I := 1 STEP 1 UNTIL N DO QP[I] := QP[I-1] * SR + P[I];           00162900
163000                      MS := ABS(SR) ;                                                      00163000
163100                      E := ABS(QP[0])* W1/W2 ;                                             00163100
163200                      FOR I := 1 STEP 1 UNTIL N DO E := E * MS + ABS(QP[I]) ;              00163200
163300                      MP := ABS(QP[N]) ;                                                   00163300
163400                      PEVALANDTESTREAL := MP LEQ 20 * (E * W2 - MP * W1)                   00163400
163500           END        PEVALANDTESTREAL ;                                                   00163500
163600     BOOLEAN PROCEDURE COMPLEXPEVALANDTEST ;                                             00163600
163700     COMMENT          EVALUATES P AT (SR,SI) USING THE MORE GENERAL FORM OF               00163700
163800                      SYNTHETIC DIVISION WHICH DIVIDES OUT A QUADRATIC FACTOR.            00163800
163900                      THE QUOTIENT POLYNOMIAL IS QP AND THE REMAINDER IS                  00163900
164000                      A * Z + B.  THE PROCEDURE IS TRUE IF THERE TWO ZEROS OF P           00164000
164100                      WHICH FORM A QUADRATIC FACTOR CLOSE TO Z * 2 + U * Z + V.           00164100
164200                      SEPARATE TESTS ARE MADE FOR A PAIR OF COMPLEX CONJUGATE             00164200
164300                      ZEROS AND FOR TWO REAL ZEROS ;                                      00164300
164400           BEGIN                                                                         00164400
164500     REAL             E,MS ;                                                             00164500
164600     ARRAY            GARB[0:N] ;                                                        00164600
164700     BOOLEAN          B1 ;                                                               00164700
164800     INTEGER          I ;                                                                00164800
164900                      QP[0] := P[0] ;                                                    00164900
165000                      QP[1] := P[1] - QP[0] * U ;                                        00165000
165100                      FOR I := 2 STEP 1 UNTIL N DO                                       00165100
165200                      QP[I] := P[I] - QP[I-1] * U - QP[I-2] * V ;                        00165200
165300                      A := QP[N] ;                                                       00165300
165400                      B := QP[N-1] ;                                                     00165400
165500                      QUADRATIC(1,U/2,V,M1,M2,M3,M4) ;                                   00165500
165600                      IF M1 = M3 THEN                                                    00165600
165700           BEGIN                                                                         00165700
165800                      MS := SQRT(V) ;                                                    00165800
165900                      E := ABS(QP[0]) * W4 / W3 ;                                        00165900
166000                      MP := - M1 * B;                                                    00166000
166100                      FOR I := 1 STEP 1 UNTIL N-1 DO                                     00166100
166200                      E := E * MS + ABS(QP[I]) ;                                         00166200
166300                      E := E * MS + ABS(A + MP) ;                                        00166300
166400                      E := E * W3 - W4 * (ABS(A+MP) + ABS(B) * MS) + W5*ABS(MP);         00166400
166500                      MP := CMOD(A + MP, M2 * B) ;                                       00166500
166600                      B1 := MP LEQ 20 * E                                                00166600
166700           END                                                                           00166700
166800                      ELSE                                                               00166800
166900                      B1 := PEVALANDTESTREAL(M1,GARB) AND                                00166900
167000                            PEVALANDTESTREAL(M3,GARB) ;                                  00167000
167100                      MP := CMOD(A,B) ;                                                  00167100
167200                      COMPLEXPEVALANDTEST := B1                                          00167200
167300           END COMPLEXPEVALANDTEST ;                                                     00167300
167400     PROCEDURE        DEFLATE(I) ;                                                       00167400
167500     VALUE            I ;                                                                00167500
167600     INTEGER          I ;                                                                00167600
167700     COMMENT          DEFLATE STORES THE ZERO OR ZEROS FOUND BY THE PROGRAM,             00167700
167800                      REPLACES P WITH THE DEFLATED POLYNOMIAL, AND REDUCES N ;           00167800
167900           BEGIN                                                                         00167900
168000                      IF I =2 THEN                                                       00168000
168100           BEGIN                                                                         00168100
168200                      ZEROR[ND] := M1 ;                                                  00168200
168300                      ZEROI[ND] := M2 ;                                                  00168300
168400                      ZEROR[ND + 1] := M3 ;                                              00168400
168500                      ZEROI[ND + 1] := M4 ;                                              00168500
168600                      N := N - 2                                                         00168600
168700           END                                                                           00168700
```

```
168800                  ELSE
168900            BEGIN
169000                  ZEROR[ND] := SR ;
169100                  ZEROI[ND] := 0 ;
169200                  N := N - 1
169300            END   ;
169400                  WRITE(P[*],N+1,QP[*]);
169500                  GO TO NEXTZERO ; COMMENT   IN MAIN ROUTINE ;
169600            END   DEFLATE ;
169700      PROCEDURE   CALCULATECOEFF ;
169800      COMMENT     EVALUATES K(S) AND CALCULATES THE COEFFICIENTS NEEDED TO
169900                  COMPUTE THE THE NEXT K POLYNOMIAL ;
170000            BEGIN
170100      INTEGER     I ;
170200                  QK[0] := K[0] ;
170300                  QK[1] := K[1] - QK[0] * U ;
170400                  FOR I := 2 STEP 1 UNTIL N-1 DO
170500                  QK[I] := K[I] - QK[I-1] * U - QK[I-2] * V ;
170600                  C := QK[N-1] ;
170700                  D := QK[N-2] ;
170800                  TYPE := IF ABS(C) LEQ ABS(K[N-1]) * 10 * ETA THEN 3 ELSE
170900                  IF ABS(D) GEQ ABS(C) THEN 2 ELSE 1 ;
171000                  IF TYPE = 1 THEN
171100            BEGIN
171200                  E := A/C ;
171300                  F := D/C ;
171400                  G := U * E ;
171500                  H := V * B ;
171600                  A3 := A2 := A * E + G * B + H * (B/C) ;
171700                  A1 := B - A * (D/C) ;
171800                  A6 := A7 := G * D + H * F + A
171900            END
172000                  ELSE IF TYPE = 2 THEN
172100            BEGIN
172200                  E := A/D ;
172300                  F := C/D ;
172400                  G := U * B ;
172500                  H := V * B ;
172600                  A3:= A2 := A * E + E * G + H * (B/D) ;
172700                  A1 := B * F - A ;
172800                  A6 := A7 := A * F + U * A + H
172900            END   ;
173000            END   CALCULATECOEFF ;
173100      PROCEDURE   NEXTESTIMATES ;
173200      COMMENT     COMPUTES THE NEXT U AND V USING THE RESULTS OF
173300                  CALCULATECOEFF ;
173400            BEGIN
173500      REAL        A4,A5,B1,B2,C1,C2,C3,C4,TEMP ;
173600                  IF TYPE = 3 THEN UI := VI := 0 ELSE
173700            BEGIN
173800                  IF TYPE = 1 THEN
173900            BEGIN
174000                  A4 := U * B + H * F + A ;
174100                  A5 := V * (U * F) + U * D + C
174200            END
174300                  ELSE
174400            BEGIN
174500                  A4 := A * F + G * F + H ;
174600                  A5 := C * F + U * C + V * D
174700            END   ;
174800                  B1 := - K[N-1]/P[N] ;
```

```
174900                  B2 := -(K[N-2] + P[N-1] * B1)/P[N] ;                                                    00174900
175000                  C1 := V * B2 * A1 ;                                                                     00175000
175100                  C2 := B1 * A6 ;                                                                         00175100
175200                  C3 := B1 * (B1 * A2) ;                                                                  00175200
175300                  C4 := C1 - C2 - C3 ;                                                                    00175300
175400                  TEMP := B1 * A4 + A5 - C4 ;                                                             00175400
175500                  IF TEMP = 0 THEN UI := VI := 0 ELSE                                                     00175500
175600              BEGIN                                                                                       00175600
175700                  UI := U - (U * (C3 + C2) + V * (B1 * A1 + B2 * A6))/TEMP ;                              00175700
175800                  VI := V * (C4/TEMP) + V                                                                 00175800
175900              END                                                                                         00175900
176000              END                                                                                         00176000
176100              END  NEXTESTIMATES ;                                                                        00176100
176200      PROCEDURE    NEXTKPOLY ;                                                                            00176200
176300      COMMENT      CALCULATES THE NEXT K POLYNOMIAL USING THE RESULTS FROM                                00176300
176400                   CALCULATECOEFF ;                                                                       00176400
176500              BEGIN                                                                                       00176500
176600      INTEGER      I ;                                                                                    00176600
176700                   IF TYPE LSS 3 THEN                                                                     00176700
176800              BEGIN                                                                                       00176800
176900                   IF ABS(A1) LEQ ABS(IF TYPE = 1 THEN B ELSE A) * ETA * 10                               00176900
177000                   THEN                                                                                   00177000
177100              BEGIN                                                                                       00177100
177200                   FOR I := N-1 STEP -1 UNTIL 2 DO                                                        00177200
177300                   K[I] := QK[I-2] * (A3/QP[0]) - QP[I-1] * (A7/QP[0]) ;                                  00177300
177400                   K[1] := -A7 ;                                                                          00177400
177500                   K[0] := 0                                                                              00177500
177600              END                                                                                         00177600
177700                   ELSE                                                                                   00177700
177800              BEGIN                                                                                       00177800
177900                   A7 := A6/A1 ;                                                                          00177900
178000                   A3 := A2/A1 ;                                                                          00178000
178100                   FOR I := N-1 STEP -1 UNTIL 2 DO                                                        00178100
178200                   K[I] := QK[I-2] * A3 - QP[I-1] * A7 + QP[I] ;                                          00178200
178300                   K[1] := -QP[0] * A7 + QP[1] ;                                                          00178300
178400                   K[0] := QP[0]                                                                          00178400
178500              END                                                                                         00178500
178600              END                                                                                         00178600
178700                   ELSE                                                                                   00178700
178800              BEGIN                                                                                       00178800
178900                   FOR I := N-1 STEP -1 UNTIL 2 DO                                                        00178900
179000                   K[I] := QK[I-2] ;                                                                      00179000
179100                   K[1] := K[0] := 0                                                                      00179100
179200              END                                                                                         00179200
179300              END  NEXTKPOLY ;                                                                            00179300
179400      PROCEDURE    NOSHIFTPROCESS ;                                                                       00179400
179500      COMMENT      INITIALIZES K AND COMPUTES 5 NO SHIFT K POLYNOMIALS ;                                  00179500
179600              BEGIN                                                                                       00179600
179700      INTEGER      I,J ;                                                                                  00179700
179800      REAL         A,C,T ;                                                                                00179800
179900                   FOR I := 0 STEP 1 UNTIL N-1 DO K[I] := P[I] * (N-I)/N ;                                00179900
180000                   A := P[N] ;                                                                            00180000
180100                   B := P[N-1] ;                                                                          00180100
180200                   FOR J := 1 STEP 1 UNTIL 5 DO                                                           00180200
180300              BEGIN                                                                                       00180300
180400                   C := K[N-1] ;                                                                          00180400
180500                   IF ABS(C) LEQ ABS(B) * ETA * 10 THEN                                                   00180500
180600              BEGIN                                                                                       00180600
180700                   FOR I := N-1 STEP -1 UNTIL 1 DO K[I] := K[I-1] ;                                       00180700
180800                   K[0] := 0                                                                              00180800
180900              END                                                                                         00180900
```

```
181000                  ELSE                                                                    00181000
181100              BEGIN                                                                       00181100
181200                  T := - A/C ;                                                            00181200
181300                  FOR I := N-1 STEP -1 UNTIL 1 DO K[I] := K[I-1] * T + P[I];              00181300
181400                  K[0] := P[0]                                                            00181400
181500              END                                                                         00181500
181600              END  J                                                                      00181600
181700              END  NOSHIFTPROCESS ;                                                       00181700
181800      PROCEDURE    CPLXITERATION ;                                                        00181800
181900      COMMENT      ATTEMPTS 10 STEPS WITH THE VARIABLE-SHIFT PROCESS FOR A                00181900
182000                   QUADRATIC FACTOR ;                                                     00182000
182100              BEGIN                                                                       00182100
182200      REAL         OMP ;                                                                  00182200
182300      INTEGER      J ;                                                                    00182300
182400      LABEL        EXIT ;                                                                 00182400
182500                   IF COMPLEXPEVALANDTEST THEN DEFLATE(2) ;                               00182500
182600                   FOR J := 1 STEP 1 UNTIL 10 DO                                          00182600
182700              BEGIN                                                                       00182700
182800                   IF J GTR 2 AND MP GTR 10 * OMP THEN GO TO EXIT ;                       00182800
182900                   OMP := MP ;                                                            00182900
183000                   CALCULATECOEFF ;                                                       00183000
183100                   NEXTKPOLY ;                                                            00183100
183200                   CALCULATECOEFF ;                                                       00183200
183300                   NEXTESTIMATES ;                                                        00183300
183400                   IF VI NEQ 0 THEN                                                       00183400
183500              BEGIN                                                                       00183500
183600                   U := UI ;                                                              00183600
183700                   V := VI                                                                00183700
183800              END  ;                                                                      00183800
183900                   IF COMPLEXPEVALANDTEST THEN DEFLATE(2) ;                               00183900
184000              END  J ;                                                                    00184000
184100      EXIT:   END  CPLXITERATION ;                                                        00184100
184200      PROCEDURE    REALITERATION ;                                                        00184200
184300      COMMENT      ATTEMPTS 10 STEPS WITH THE VARIABLE-SHIFT PROCESS FOR A                00184300
184400                   REAL ZERO ;                                                            00184400
184500              BEGIN                                                                       00184500
184600      INTEGER      I,J ;                                                                  00184600
184700      REAL         T,OMP,KS ;                                                             00184700
184800      LABEL        EXIT ;                                                                 00184800
184900                   IF PEVALANDTESTREAL(SR,QP) THEN DEFLATE(1) ;                           00184900
185000                   FOR J := 1 STEP 1 UNTIL 10 DO                                          00185000
185100              BEGIN                                                                       00185100
185200                   IF J GTR 2 THEN                                                        00185200
185300              BEGIN                                                                       00185300
185400                   IF ABS(T) < 0.01 * ABS(SR-T) AND MP > OMP THEN                         00185400
185500              BEGIN                                                                       00185500
185600      COMMENT      A CLUSTER OF ZEROS NEAR THE REAL AXIS HAS BEEN ENCOUNTERED             00185600
185700                   WE ATTEMPT TO FIND A QUADRATIC FACTOR ;                                00185700
185800                   U := -2 * SR ;                                                         00185800
185900                   V := SR * SR ;                                                         00185900
186000                   CPLXITERATION ;                                                        00186000
186100      COMMENT      RETURNS ONLY IF NO QUADRATIC FACTOR IS FOUND ;                         00186100
186200                   GO TO EXIT                                                             00186200
186300              END                                                                         00186300
186400                   ELSE                                                                   00186400
186500                   IF MP GTR 10*OMP THEN GO TO EXIT                                       00186500
186600              END  ;                                                                      00186600
186700                   OMP := MP ;                                                            00186700
186800                   QK[0] := K[0] ;                                                        00186800
186900                   FOR I := 1 STEP 1 UNTIL N-1 DO                                         00186900
187000                   QK[I] := QK[I-1] * SR + K[I] ;                                         00187000
```

```
187100                  IF ABS(QK[N-1]) LEQ ABS(K[N-1]) * 10 * ETA THEN                                 00187100
187200              BEGIN                                                                               00187200
187300                  FOR I := N-1 STEP -1 UNTIL 1 DO K[I] := QK[I-1] ;                               00187300
187400                  K[0] := 0                                                                       00187400
187500              END                                                                                 00187500
187600                  ELSE                                                                            00187600
187700              BEGIN                                                                               00187700
187800                  T := -QP[N]/QK[N-1] ;                                                           00187800
187900                  FOR I := N-1 STEP -1 UNTIL 1 DO K[I] := QK[I-1] * T+QP[I];                      00187900
188000                  K[0] := QP[0]                                                                   00188000
188100              END ;                                                                               00188100
188200                  KS := K[0] ;                                                                    00188200
188300                  FOR I := 1 STEP 1 UNTIL N-1 DO KS := KS * SR + K[I] ;                           00188300
188400                  T := IF ABS(KS) LEQ ABS(K[N-1]) * 10 * ETA THEN 0 ELSE                          00188400
188500                  - QP[N]/KS ;                                                                    00188500
188600                  SR := SR + T ;                                                                  00188600
188700                  IF PEVALANDTESTREAL(SR,QP) THEN DEFLATE(1)                                      00188700
188800              END J ;                                                                             00188800
188900  EXIT:       END   REALITERATION ;                                                               00188900
189000  PROCEDURE         FIXEDSHIFTPROCESS(LIMIT) ;                                                    00189000
189100  VALUE             LIMIT ;                                                                       00189100
189200  INTEGER           LIMIT ;                                                                       00189200
189300  COMMENT           CARRIES OUT THE STAGE TWO CALCULATION OF K POLYNOMIALS                        00189300
189400                    WITH TESTING.  LIMIT IS THE MAXIMUM NUMBER OF STEPS ;                         00189400
189500              BEGIN                                                                               00189500
189600  REAL              TR,TI,OTR,OTI,OVI,SVU,SVV ;                                                   00189600
189700  BOOLEAN           TEST,RTEST,PASSED,RPASSED ;                                                   00189700
189800  INTEGER           I,J ;                                                                         00189800
189900                    U := -2 * SR ;                                                                00189900
190000                    V := SR * SR + SI * SI ;                                                      00190000
190100                    TEST := RTEST := TRUE ;                                                       00190100
190200                    PASSED := RPASSED := FALSE ;                                                  00190200
190300                    IF COMPLEXPEVALANDTEST THEN DEFLATE(2) ;                                      00190300
190400                    CALCULATECOEFF ;                                                              00190400
190500                    FOR J := 1 STEP 1 UNTIL LIMIT DO                                              00190500
190600              BEGIN                                                                               00190600
190700                    NEXTKPOLY ;                                                                   00190700
190800                    CALCULATECOEFF ;                                                              00190800
190900                    NEXTESTIMATES ;                                                               00190900
191000                    IF TYPE = 3 OR K[0] = 0 THEN TR := TI := 0 ELSE                               00191000
191100                    CDIVIDE(SR * B - A,-SI * B,C - SR * D, SI * D,TR,TI) ;                        00191100
191200                    IF RTEST AND J GTR 1 THEN                                                     00191200
191300              BEGIN                                                                               00191300
191400                    IF CMOD(TR - OTR,TI - OTI) LSS 0.5 * CMOD(SR + TR,SI + TI)                    00191400
191500                    THEN                                                                          00191500
191600              BEGIN                                                                               00191600
191700                    IF RPASSED THEN                                                               00191700
191800              BEGIN                                                                               00191800
191900                    SVU := SR ;                                                                   00191900
192000                    WRITE(SVK[*],N,K[*]);                                                         00192000
192100                    SR := SR + TR ;                                                               00192100
192200                    REALITERATION ;                                                               00192200
192300  COMMENT           RETURNS ONLY IF THE ITERATION FAILS ;                                         00192300
192400                    RTEST := FALSE ;                                                              00192400
192500                    SR := SVU ;                                                                   00192500
192600                    WRITE(K[*],N,SVK[*]);                                                         00192600
192700                    IF COMPLEXPEVALANDTEST THEN DEFLATE(2) ;                                      00192700
192800              END                                                                                 00192800
192900                    ELSE                                                                          00192900
193000                    RPASSED := TRUE                                                               00193000
193100              END                                                                                 00193100
```

```
193200                  ELSE
193300                  RPASSED := FALSE
193400          END     RTEST ;
193500                  QTR := TR ;
193600                  QTI := TI ;
193700                  IF TEST AND J GTR 1 THEN
193800          BEGIN
193900                  IF ABS(QVI-VI) LSS 0.5 * ABS(VI) THEN
194000          BEGIN
194100                  IF PASSED THEN
194200          BEGIN
194300                  SVU := U ;
194400                  SVV := V ;
194500                  WRITE(SVK[*],N,K[*]);
194600                  U := UI ;
194700                  V := VI ;
194800                  CPLXITERATION ;
194900  COMMENT         RETURNS ONLY IF THE ITERATION FAILS ;
195000                  TEST := FALSE ;
195100                  U := SVU ;
195200                  V := SVV ;
195300                  WRITE(K[*],N,SVK[*]);
195400                  IF COMPLEXPEVALANDTEST THEN DEFLATE(2) ;
195500                  CALCULATECOEFF
195600          END
195700                  ELSE
195800                  PASSED := TRUE
195900          END
196000                  ELSE
196100                  PASSED := FALSE
196200          END     TEST ;
196300                  QVI := VI
196400          END     J ;
196500                  WRITE(SVK[*],N,K[*]);
196600                  CPLXITERATION ;
196700  COMMENT         RETURNS ONLY IF THE ITERATION FAILS ;
196800                  WRITE(K[*],N,SVK[*]);
196900          END     FIXEDSHIFTPROCESS ;
197000  COMMENT         VARIABLES LOCAL TO MAIN ROUTINE ;
197100  REAL            BND,ANGLE ;
197200  INTEGER         LIMIT,CNT1,CNT2 ;
197300  ARRAY           INITIAL[0:DEGREE+1] ;
197400  COMMENT         BODY OF PROCEDURE REALPOLYZEROFINDER STARTS HERE ;
197500   COMMENT        SET INITIAL VALUE FOR PROCEDURE RANDOM ;
197600                  RANDOMSTART := 17;
197700                  N := DEGREE ;
197800                  IF P[0] = 0 THEN GO TO FAILURE ;
197900                  PRESCALE(N,P) ;
198000  NEXTZERO:       IF N = 1 THEN
198100          BEGIN
198200                  ZEROR[DEGREE] := - P[1]/P[0] ;
198300                  ZEROI[DEGREE] := 0
198400          END
198500                  ELSE
198600                  IF N = 2 THEN
198700                  QUADRATIC(P[0],P[1]/2,P[2],ZEROR[DEGREE-1],ZEROI[DEGREE-1]
198800                  ,ZEROR[DEGREE],ZEROI[DEGREE])
198900                  ELSE
199000                  IF N GTR 2 THEN
199100          BEGIN
199200  COMMENT         MAIN ALGORITHM FOR FINDING ONE OR TWO ZEROS ;
```

```
199300                  ND := DEGREE - N + 1 ;                                                  00199300
199400                  IF P[N] = 0 THEN                                                        00199400
199500            BEGIN                                                                         00199500
199600                  ZEROR[ND] := ZEROI[ND] := 0 ;                                           00199600
199700                  N := N-1 ;                                                              00199700
199800                  GO TO NEXTZERO                                                          00199800
199900            END   ;                                                                       00199900
200000                  NOSHIFTPROCESS ;                                                        00200000
200100                  WRITE(INITIAL[*],N,K[*]);                                               00200100
200200                  BND := CAUCHYBOUND(N,P) ;                                               00200200
200300                  FOR CNT1 := 1,2 DO                                                      00200300
200400            BEGIN                                                                         00200400
200500                  LIMIT := 10 ;                                                           00200500
200600                  FOR CNT2 := 1 STEP 1 UNTIL 8 DO                                         00200600
200700            BEGIN                                                                         00200700
200800                  ANGLE := RANDOM(RANDOMSTART) * 3.14159265359 ;                          00200800
200900                  SR := BND * COS(ANGLE) ;                                                00200900
201000                  SI := BND * SIN(ANGLE) ;                                                00201000
201100                  FIXEDSHIFTPROCESS(LIMIT) ;                                              00201100
201200      COMMENT     IF WE RETURN FROM THE FIXED SHIFT STAGE THEN A SUCCESSFUL               00201200
201300                  ITERATION HAS NOT BEEN STARTED.  A NEW SHIFT IS SELECTED                00201300
201400                  AND THE FIXED SHIFT STAGE IS INCREASED ;                                00201400
201500                  LIMIT := LIMIT + 10                                                     00201500
201600            END   CNT2 ;                                                                  00201600
201700      COMMENT     IF 8 SHIFTS DO NOT SUCCEED WE TRY RESTARTING WITH THE                   00201700
201800                  INITIAL K POLYNOMIAL AND USE A DIFFERENT SEQUENCE OF                    00201800
201900                  SHIFTS ;                                                                00201900
202000                  WRITE(K[*],N,INITIAL[*]);                                               00202000
202100            END   CNT1 ;                                                                  00202100
202200                  GO TO FAILURE                                                           00202200
202300            END   OF MAIN ALGORITHM ;                                                     00202300
202400                  GO TO EXIT;                                                             00202400
202500      FAILURE:    WRITE(PRINTER,ERR,DEGREE-N+1);                                          00202500
202600      EXIT:                                                                               00202600
202700            END   REALPOLYZEROFINDER ;                                                    00202700
202800       STARTP;                                                                            00202800
202900       XRANGE(-250,1250);YRANGE(-250,1250);                                               00202900
203000        SUBFRM(0,0,1000,1000);                                                            00203000
203100       DEVICE(3,1,0,0);                                                                   00203100
203200                 BEGIN                                                                    00203200
203300      ARRAY ZR1,ZI1[0:DPOLY1];                                                            00203300
203400      ARRAY ZR2,ZI2[0:DPOLY2];                                                            00203400
203500      ARRAY POLY3,ZR3,ZI3[0:DPOLY2];                                                      00203500
203600      ARRAY POLYK[0:DPOLY1];                                                              00203600
203700      ARRAY K[0:0];                                                                       00203700
203800       WRITE(FILE97,<"FINAL VALUE OF K AND K INCREMENT?">);                               00203800
203900      READ(FILE97,/,K1,KI);                                                               00203900
204000      WRITE(FILE97,<"YRANGE AND INCREMENT?">);                                            00204000
204100      READ(FILE97,/,LY,HY,YI);                                                            00204100
204200      WRITE(FILE97,<"XRANGE AND INCREMENT?">);                                            00204200
204300      READ(FILE97,/,LX,HX,XI);                                                            00204300
204400      FRAME("GRAPH");                                                                     00204400
204500      AT(510,1000);TXTDIR(0);TXTSIZ(0);TXT("IM");                                         00204500
204600      AT(1000,100);TXTDIR(0);TXTSIZ(0);TXT("RE");                                         00204600
204700       AT(20,1000);TXTDIR(0);WRTARY(1,T1);                                                00204700
204800      YRANGE(LY,HY);XRANGE(LX,HX);                                                        00204800
204900      XAXIS(LY+1,XI);YAXIS(0,YI);                                                         00204900
205000      NOLINE;                                                                             00205000
205100      IF DPOLY1>=1 THEN                                                                   00205100
205200                 BEGIN                                                                    00205200
205300      REALPOLYZEROFINDER(DPOLY1,POLY1,ZR1,ZI1);                                           00205300
```

```
205400     FOR J:=1 STEP 1 UNTIL DPOLY1 DO                                                00205400
205500                         BEGIN                                                       00205500
205600     X:=ZR1[J];Y:=ZI1[J];                                                            00205600
205700     AT(X,Y);SYMSEL(2);SYMSIZ(1);PUTSYM;                                             00205700
205800                         END;                                                        00205800
205900                   END;                                                              00205900
206000     NOLINE;                                                                         00206000
206100     REALPOLYZEROFINDER(DPOLY2,POLY2,ZR2,ZI2);                                       00206100
206200     FOR J:=1 STEP 1 UNTIL DPOLY2 DO                                                 00206200
206300                         BEGIN                                                       00206300
206400     X:=ZR2[J];Y:=ZI2[J];                                                            00206400
206500     AT(X,Y);SYMSEL(8);SYMSIZ(1);PUTSYM;                                             00206500
206600                         END;                                                        00206600
206700     NOLINE;                                                                         00206700
206800     FOR K[0]:=0 STEP KI*1.1/20 UNTIL K1 DO                                          00206800
206900       BEGIN                                                                         00206900
207000     POLYMUL(DPOLY1,POLY1,0,K,DPOLY1,POLYK);                                         00207000
207100     POLYADD(DPOLY1,POLYK,DPOLY2,POLY2,DPOLY2,POLY3);                                00207100
207200     REALPOLYZEROFINDER(DPOLY2,POLY3,ZR3,ZI3);                                       00207200
207300       FOR J:=1 STEP 1 UNTIL DPOLY2 DO                                               00207300
207400                         BEGIN                                                       00207400
207500     X:=ZR3[J];Y:=ZI3[J];                                                            00207500
207600     WRITE(E,/,K,X,Y);                                                               00207600
207700     CHARMD;                                                                         00207700
207800     AT(X,Y);LINETO(X,Y);                                                            00207800
207900                         END;                                                        00207900
208000       END;                                                                          00208000
208100       WHEN(10);                                                                     00208100
208200     WRITE(FILE97,<"DO YOU WANT TO CHANGE THE SCALE?">);                             00208200
208300     READ(FILE97,/,SC);                                                              00208300
208400     IF SC="Y" THEN RLOCUS(DPOLY1,DPOLY2,POLY1,POLY2);                               00208400
208500                    END;                                                             00208500
208600      ENDP;                                                                          00208600
208700         END;                                                                        00208700
208800     DEFINE                                                                          00208800
208900                                        READK=                                       00208900
209000                                        WRITE(FILE97,<"VALUE OF K?">);               00209000
209100                                        READ(FILE97,/,K[0])#,                        00209100
209200           PART2=                                                                    00209200
209300     WRITE(FILE97,<"COEFFICIENTS OF C1(S)?">);                                       00209300
209400     READ(FILE97,/,FOR J:=0 STEP 1 UNTIL DC1 DO C1[J]);                              00209400
209500     WRITE(FILE97,<"COEFFICIENTS OF C2(S)?">);                                       00209500
209600     READ(FILE97,/,FOR J:=0 STEP 1 UNTIL DC2 DO C2[J]);                              00209600
209700     WRITE(FILE97,<"COEFFICIENTS OF G1(S)?">);                                       00209700
209800     READ(FILE97,/,FOR J:=0 STEP 1 UNTIL DG1 DO G1[J]);                              00209800
209900     WRITE(FILE97,<"COEFFICIENTS OF G2(S)?">);                                       00209900
210000     READ(FILE97,/,FOR J:=0 STEP 1 UNTIL DG2 DO G2[J]);                              00210000
210100     WRITE(FILE97,<"COEFFICIENTS OF H1(S)?">);                                       00210100
210200     READ(FILE97,/,FOR J:=0 STEP 1 UNTIL DH1 DO H1[J]);                              00210200
210300     WRITE(FILE97,<"COEFFICIENTS OF H2(S)?">);                                       00210300
210400     READ(FILE97,/,FOR J:=0 STEP 1 UNTIL DH2 DO H2[J])#,                             00210400
210500           READCGH1=                                                                 00210500
210600     READK;                                                                          00210600
210700     WRITE(FILE97,<"DEGREE OF C1,C2,G1,G2,H1,H2?">);                                 00210700
210800     READ(FILE97,/,DC1,DC2,DG1,DG2,DH1,DH2);                                         00210800
210900     DCG1:=DC1+DG1;DCG2:=DC2+DG2;DPOLY1:=DCG1+DH1;                                   00210900
211000     DA:=DPOLY1;DPOLY2:=DCG2+DH2;DB:=DPOLY2;                                         00211000
211100     BEGIN                                                                           00211100
211200     LABEL LAB2;                                                                     00211200
211300     ARRAY C1[0:DC1],C2[0:DC2],G1[0:DG1],G2[0:DG2];                                  00211300
211400     ARRAY H1[0:DH1],H2[0:DH2],CG1[0:DCG1],CG2[0:DCG2];                              00211400
```

```
211500     ARRAY A,AA[0:DA],B,BB[0:DB],POLY1[0:DPOLY1],POLY2[0:DPOLY2];                    00211500
211600     PART2;                                                                          00211600
211700     POLYMUL(DC1,C1,DG1,G1,DCG1,CG1);                                                00211700
211800     POLYMUL(DCG1,CG1,DH1,H1,DPOLY1,POLY1);                                          00211800
211900     POLYMUL(DPOLY1,POLY1,0,K,DA,A);                                                 00211900
212000     POLYMUL(DC2,C2,DG2,G2,DCG2,CG2);                                                00212000
212100     POLYMUL(DCG2,CG2,DH2,H2,DPOLY2,POLY2);                                          00212100
212200     POLYMUL(DCG2,CG2,DH2,H2,DB,B);                                                  00212200
212300       U[0]:=1;INB[0]:=1/B[0];                                                       00212300
212400       IF B[0]=1 THEN                                                              00212400
212500      BEGIN                                                                        00212500
212600      POLYMUL(DA,A,0,U,DA,AA);                                                     00212600
212700      POLYMUL(DB,B,0,U,DB,BB);                                                     00212700
212800      END                                                                          00212800
212900        ELSE                                                                       00212900
213000      BEGIN                                                                        00213000
213100       POLYMUL(DA,A,0,INB,DA,AA);                                                  00213100
213200       POLYMUL(DB,B,0,INB,DB,BB);                                                  00213200
213300      END#,                                                                        00213300
213400            READCGH2=                                                             00213400
213500     READK;                                                                       00213500
213600     WRITE(FILE97,<"DEGREE OF C1,C2,G1,G2,H1,H2?">);                              00213600
213700     READ(FILE97,/,DC1,DC2,DG1,DG2,DH1,DH2);                                      00213700
213800       DCG2:=DC2+DG2;DKCG1:=DC1+DG1;                                              00213800
213900     DA:=DC1+DG1+DH2;DPOL1:=DC1+DG1+DH1;DPOL2:=DC2+DG2+DH2;                       00213900
214000     IF DPOL1>DPOL2 THEN DB:=DPOL1 ELSE DB:=DPOL2;                                00214000
214100     BEGIN                                                                        00214100
214200                                                        LABEL LAB2;               00214200
214300     ARRAY C1[0:DC1],C2[0:DC2],G1[0:DG1],G2[0:DG2];                               00214300
214400     ARRAY H1[0:DH1],H2[0:DH2],KC1[0:DC1],A,AA[0:DA];                             00214400
214500     ARRAY B,BB[0:DB],POL1[0:DPOL1],POL2[0:DPOL2];                                00214500
214600     ARRAY CG2[0:DCG2],KCG1[0:DKCG1];                                             00214600
214700     PART2;                                                                       00214700
214800     POLYMUL(0,K,DC1,C1,DC1,KC1);                                                 00214800
214900     POLYMUL(DC1,KC1,DG1,G1,DKCG1,KCG1);                                          00214900
215000     POLYMUL(DKCG1,KCG1,DH2,H2,DA,A);                                             00215000
215100     POLYMUL(DKCG1,KCG1,DH1,H1,DPOL1,POL1);                                       00215100
215200     POLYMUL(DC2,C2,DG2,G2,DCG2,CG2);                                             00215200
215300     POLYMUL(DCG2,CG2,DH2,H2,DPOL2,POL2);                                         00215300
215400     POLYADD(DPOL1,POL1,DPOL2,POL2,DB,B);                                         00215400
215500       U[0]:=1;INB[0]:=1/B[0];                                                    00215500
215600       IF B[0]=1 THEN                                                           00215600
215700      BEGIN                                                                     00215700
215800      POLYMUL(DA,A,0,U,DA,AA);                                                  00215800
215900      POLYMUL(DB,B,0,U,DB,BB);                                                  00215900
216000     WRITE(FILE97,/,FOR J:=0 STEP 1 UNTIL DB DO BB[J]);                         00216000
216100      END                                                                       00216100
216200        ELSE                                                                    00216200
216300      BEGIN                                                                     00216300
216400       POLYMUL(DA,A,0,INB,DA,AA);                                               00216400
216500     WRITE(FILE97,/,FOR J:=0 STEP 1 UNTIL DA DO A[J]);                          00216500
216600       POLYMUL(DB,B,0,INB,DB,BB);                                               00216600
216700      END#,                                                                     00216700
216800            STARTPLOTTER=                                                       00216800
216900      LAB2:WRITE(FILE97,<"WHICH GRAPH?-BODE,NYQUIST,NICHOLS OR RLOCUS?">);      00216900
217000       READ(FILE97,/,GRAPH);                                                    00217000
217100      IF GRAPH="B" THEN BODE(W,DA,DB,A,B);                                      00217100
217200      IF GRAPH="NY" THEN NYQUIST(W,DA,DB,A,B);                                  00217200
217300      IF GRAPH="NI" THEN NICHOLS(W,DA,DB,A,B);                                  00217300
217400      IF GRAPH="R" THEN RLOCUS(DPOLY1,DPOLY2,POLY1,POLY2);                      00217400
217500     WRITE(FILE97,<"DO YOU WANT ANYMORE GRAPH?">);                              00217500
```

```
217600     READ(FILE97,/,MORE);                                                                00217600
217700     IF MORE="Y" THEN GO TO LAB2 ELSE                                                    00217700
217800        BEGIN                                                                            00217800
217900         WRITE(FILE97,<"DO YOU WANT TO CHANGE TO NEW VALUES?">);                         00217900
218000         READ(FILE97,/,NEW);                                                             00218000
218100        END;                                                                             00218100
218200     END;                                                                                00218200
218300     IF NEW="Y" THEN GO TO LAB4  ELSE GO TO LAB5#;                                       00218300
218400      WRITE(FILE97,<"TIME OR FREQUENCY DOMAIN?">);                                       00218400
218500      READ(FILE97,/,DMM);                                                               00218500
218600      IF DMM="T" THEN                                                                   00218600
218700     LAB8: BEGIN                                                                        00218700
218800      WRITE(FILE97,<"O/P OR CLOSED LOOP?">);                                            00218800
218900      READ(FILE97,/,OP);                                                                00218900
219000      IF OP="O/P" THEN                                                                  00219000
219100       BEGIN                                                                            00219100
219200       ARRAY K[0:0];                                                                    00219200
219300       LABEL LAB10;                                                                     00219300
219400       LAB10: READCGH1;                                                                 00219400
219500        TIMEDOMAIN(DA,DB,AA,BB);                                                        00219500
219600       WRITE(FILE97,<"WANT TO CHANGE TO NEW VALUES?">);                                 00219600
219700       READ(FILE97,/,NEW);                                                              00219700
219800       IF NEW="Y" THEN GO TO LAB10;                                                     00219800
219900       END;                                                                             00219900
220000         END                                                                            00220000
220100      ELSE                                                                              00220100
220200       BEGIN                                                                            00220200
220300      ARRAY K[0:0];                                                                     00220300
220400       LABEL LAB10;                                                                     00220400
220500      LAB10: READCGH2;                                                                  00220500
220600        TIMEDOMAIN(DA,DB,AA,BB);                                                        00220600
220700       WRITE(FILE97,<"WANT TO CHANGE TO NEW VALUES?">);                                 00220700
220800       READ(FILE97,/,NEW);                                                              00220800
220900       IF NEW="Y" THEN GO TO LAB10;                                                     00220900
221000       END;                                                                             00221000
221100       END;                                                                             00221100
221200     WRITE(FILE97,<"WANT ANYMORE TIME-DOMAIN PLOT?">);                                  00221200
221300     READ(FILE97,/,MTDP);                                                               00221300
221400     IF MTDP="Y" THEN GO TO LAB8 ELSE GO TO LAB7;                                       00221400
221500      END                                                                               00221500
221600      ELSE                                                                              00221600
221700       IF DMM="F" THEN                                                                  00221700
221800     LAB6: BEGIN                                                                        00221800
221900      WRITE(FILE97,<"WHICH PLOT?-O/P OR CLOSED LOOP?">);                                00221900
222000     READ(FILE97,/,OP);                                                                 00222000
222100      IF OP="O/P" THEN                                                                  00222100
222200        BEGIN                                                                           00222200
222300       LABEL LAB4;                                                                      00222300
222400       ARRAY K[0:0];                                                                    00222400
222500      LAB4: READCGH1;                                                                   00222500
222600        STARTPLOTTER;                                                                   00222600
222700        END                                                                             00222700
222800      ELSE IF OP="CLOSED" THEN                                                          00222800
222900        BEGIN                                                                           00222900
223000       LABEL LAB4;                                                                      00223000
223100       ARRAY K[0:0];                                                                    00223100
223200      LAB4: READCGH2;                                                                   00223200
223300                                                                                        00223300
223400     LAB2: WRITE(FILE97,<"WHICH GRAPH?-BODE,NYQUIST OR NICHOLS ?">);                    00223400
223500       READ(FILE97,7,GRAPH);                                                            00223500
223600      IF GRAPH="B" THEN BODE(W,DA,DB,A,B);                                              00223600
```

```
223700      IF GRAPH="NY" THEN NYQUIST(W,DA,DB,A,B);                                    00223700
223800      IF GRAPH="NI" THEN NICHOLS(W,DA,DB,A,B);                                    00223800
223900     WRITE(FILE97,<"DO YOU WANT ANYMORE GRAPH?">);                                00223900
224000     READ(FILE97,/,MORE);                                                         00224000
224100     IF MORE="Y" THEN GO TO LAB2 ELSE                                             00224100
224200        BEGIN                                                                     00224200
224300         WRITE(FILE97,<"DO YOU WANT TO CHANGE TO NEW VALUES?">);                  00224300
224400         READ(FILE97,/,NEW);                                                      00224400
224500        END;                                                                      00224500
224600     END;                                                                         00224600
224700     IF NEW="Y" THEN GO TO LAB4 ELSE GO TO LAB5;                                  00224700
224800           END;                                                                   00224800
224900           END;                                                                   00224900
225000     LAB5:BEGIN END;                                                              00225000
225100     WRITE(FILE97,<"DO YOU WANT ANYMORE FREQUENCY-DOMAIN PLOT?">);                00225100
225200     READ(FILE97,/,OUI);                                                          00225200
225300     IF OUI="Y" THEN GO TO LAB6 ELSE GO TO LAB7;                                  00225300
225400     LAB7: IF DMM="T" THEN                                                        00225400
225500           BEGIN                                                                  00225500
225600        WRITE(FILE97,<"WHAT ABOUT FREQ. DOMAIN PLOT?">);                          00225600
225700        READ(FILE97,/,OUI);                                                       00225700
225800        IF OUI="Y" THEN GO TO LAB6 ELSE GO TO LAB9;                               00225800
225900           END                                                                    00225900
226000      ELSE IF DMM="F" THEN                                                        00226000
226100           BEGIN                                                                  00226100
226200        WRITE(FILE97,<"WHAT ABOUT TIME-DOMAIN RESPONSE?">);                       00226200
226300        READ(FILE97,/,OUI);                                                       00226300
226400        IF OUI="Y" THEN GO TO LAB8 ELSE GO TO LAB9;                               00226400
226500           END;                                                                   00226500
226600     LAB9:  END.                                                                  00226600
```