MODEL ORDER REDUCTION IN FREQUENCY DOMAIN AND ITS APPLICATIONS

A DISSERTATION

submitted in partial fulfilment of the
requirements for the award of the degree
of
MASTER OF ENGINEERING
in
ELECTRICAL ENGINEERING
(With Specialization in Measurement and Instrumentation)

BY

DINESH CHANDRA AGARWAL

DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITY OF ROORKEE
ROORKEE-247 667 (INDIA)

JULY, 1990

DEDICATED
TO
MY LATE PARENTS

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **"MODEL ORDER REDUCTION IN FREQUENCY DOMAIN AND ITS APPLICATIONS"** in partial fulfilment of the requirement for the award of Degree of MASTER OF ENGINEERING (M & I), submitted in the department of 'ELECTRICAL ENGINEERING' of the University is an authentic record of my own work carried out during a period from Nov. 1989 to June 1990 under the supervision of **Dr.R.N.Mishra,** Professor, Department of Electrical Engineering, University of Roorkee, Roorke.

The matter embodied in this thesis has not been submitted by me for the award of any other degree.

Dinesh chandra Agarwal

(DINESH CHANDRA AGARWAL)

Dated: 6-7-90.

This is certify that the above statement made by the candidate is correct to the best of my knowledge.

R. N. Mishra.

(R.N. MISHRA)
Professor,
Electrical Engg. Deptt.
University of Roorkee,
Roorkee-247667

Dated: 6-7-90

ACKNOWLEDGEMENT

I wish to express my deep and sincere gratitude to **Dr. R.N.Mishra,** Professor, Electrical Engineering Department, Univesity of Roorkee for his valuable guidance during each and every phase of this work.

I wish to acknowledge to Prof. R.B.Saxena, Head of the Department, Electrical Engineering Department, University of Roorkee for providing necessary facilities which have made this work possible.

I extend my gratitude to staff members of Computer Lab., Electrical Engg. Department, for their cooperation.

I am also thankful to Mr. Vijay Pande, a student of M.E. (SEOR), for the help given by him.

Roorkee

Dated : 6-7-90.

Dinesh Chandra Agarwal

(DINESH CHANDRA AGARWAL)

ABSTRACT

The work included in this thesis deals with model reduction techniques in frequency domain i.e. based on a transfer function description of the original system.

The first chapter introduces model reduction problem, its necessity and a broad classification of various model reduction techniques. Reduction by Cauer forms, Routh-Hurwitz array, the Integral square error methods are described in chapter-2. The method to obtain step response of a system is presented in chapter-3. The respective step response of the illustrative examples are shown for comparision purpose. A scheme to design controllers, using reduced order models obtained from Cauer forms method, is given in chapter-4.

The computer programs, in FORTRAN, for model reduction, step response and Nyquist plot, have also been developed and implemented successfully on a PC.

CONTENTS

Page No.

chapter :1

introduction

1.1 MOTIVATION FOR MODEL REDUCTION

Every physical system can be translated into mathematical model. The mathematical models of large systems are very complex and they can not be reduced by hand calculations. Fast digital computers can be used to reduce these complex models.

The mathematical procedure of system modelling often leads to comprehensive description of a process in the form of high order differential equations which are difficult to use either for analysis or controller synthesis. It is hence useful, and sometimes necessary, to find the possibility of finding some equation of the same type but of lower order that may be considered to adequately reflect the dominant characteristics of the system under consideration. Some of the reasons for using reduced order models of higher order linear systems could be:

(a) To have a better understanding of the system:

A system of uncomfortably high order poses difficulties in its analysis, synthesis or identification. An obvious method of dealing with such type of system is to approximate it by a low order system which reflect the characteristics of original system such as time constant, damping ratio, natural frequency etc.

(b) To reduce computational complexity:

The development of state-space methods and optimal

control techniques have made the design of control system for high order multivariable system quite feasible. When the order of systems become high, special numerical techniques are required to permit the calculation to be done at a reasonable cost on fast digital computers. This saves both time and memory required by computer.

(c) To reduce hardware complexity:

A control system design for a high order system is likely to be very complicated and of a high order itself. This is particularly true for controllers based on optimal control theory. Controllers designed on the basis of low-order model will be more reliable, less costly and easy to implement and maintain.

1.2 APPLICATIONS OF REDUCED ORDER MODELS

Reduced order models and reduction techniques have been widely used for the analysis and synthesis of high order systems. Some of the uses to which these have been put are:

(1) Prediction of the transient response sensitivity of high order systems using low order models.

(2) Predicting dynamic errors of high order systems using low-order equivalents.

(3) Control system design.

(4) Adaptive control using low order models.

(5) Designing reduced order estimators.

(6) Suboptimal control derived by simplified models.

1.3' STATEMENT OF MODEL REDUCTION

The reduction of a high order system into its lower order approximants in frequency domain can be stated as:

Given a transfer function description of a higher order single input - single output system:

$$G_o(S) = \frac{a_o + a_1 S + a_2 S^2 + \ldots\ldots + a_n S^{n-1}}{b_o + b_1 S + b_2 S^2 + \ldots\ldots + b_{n+1} S^n} = \frac{N(S)}{D(S)}$$

where n is the order of the system.

A reduced order model is desired, which can adequately describe the significant dynamic behaviour of the original system and can be expressed as:

$$G_r(S) = \frac{c_o + c_1 S + c_2 S^2 + \ldots\ldots + c_r S^{r-1}}{d_o + d_1 S + d_2 S^2 + \ldots\ldots + d_{r+1} S^r} \quad , \forall \quad r<n$$

where r is the order of reduced order system.

In time domain, the systems can be described by the following state space equations.

Original System	Reduced Order System (r<n)
$\dot{\underline{X}}(t) = A\underline{X}(t) + B\underline{u}(t)$	$\dot{\underline{X}}_r(t) = A_r\underline{X}_r(t) + B_r\underline{u}(t)$
$y(t) = C\underline{X}(t) + D\underline{u}(t)$	$y_r(t) = C_r\underline{X}_r(t) + D_r\underline{u}(t)$
where,	where,
$\underline{X}(t)$ = n x 1 state vector	$\underline{X}_r(t)$ = r x 1 state vector
$\underline{u}(t)$ = m x 1 input vector	A_r = r x r system matrix

Contd..

Original System	Reduced Order System (r<n)
A = n x n system matrix	B_r = r x m input matrix
B = n x m input matrix	C_r = 1 x r output matrix
y(t)= 1 x 1 output vector	D_r = 1 x m transmission matrix
C = 1 x n output matrix	$y_r(t)$= 1 x 1 vector of reduced system
D = 1 x m transmission matrix	

(For SISO l = m = 1) and in physical systems, transmission matrix, in general, in zero.

1.4 CLASSIFICATION OF MODEL REDUCTION TECHNIQUES

The order reduction techniques can broadly be classified as:

1.4.1 Time domain Simplification Techniques

In time domain reduction techniques the original and reduced systems are expressed in state space form. The order of matrices A_r, B_r, C_r are less than A, B, C and the output y_r will be a close approximation to y for specified inputs. The time domain techniques belong to either of the following categories:

(1) Modal Analysis:

This category attempts to attain the dominant eigen values of the original system and then obtains the remaining parameters of the low order model in such a way that its response, to a certain specified input should approximate closely to that of high order system. The method proposed

by DAVISON [1], AOKI [2] belong to this category. Davison's method consists of diagonalising of the system matrix and neglecting the large eigen values. In this case, the input is taken as step function and all the eigen values are assumed to be distinct. This restriction, however, was removed by CHIDAMBARA [4] and DAVISION [3]. AOKI [2] took a more general approach based on aggregation. A method to improve the quality of simplified aggregated models of systems without increasing order of the state differential equations has been given by GRUCA et.al. [5]. It consisted of introduction of delay in the output vector of aggregated model to minimize a quality index function of the output vector. However, the numerical difficulties and the absence of guide lines for selecting the weighting matrices in performance index of this method were well observed by the researchers. INOOKA et.al. [6] proposed a method based on combining the method of aggregation and integral square error criterian. An important variation of dominant eigen value concept was proposed by KUPPURAJULU and ELANGOVAN [7] wherein the high order system is replaced by three models, successively representing the initial, intermediate and final stages of the transient response.

The above out-lined approaches, though useful in many applications, suffer from the following disadvantages:

(i) The computation of eigen values, eigen vectors and the aggregation matrix may be quite formidable for a very high order system.

(ii) In cases, where the eigen values of a system are close together or where the eigen values are not easily identified, these methods obviously fail.

(iii) There may be considerable difference between the steady state responses of the high order system and its low order model to certain inputs [1]. However this shortcoming was removed by CHIDAMBARA [4] at the cost of poor matching during transient period. The above mentioned points led to the optimum order reduction approach.

(2) Optimum Model Reduction

This second group is based on obtaining a low order model of a given high order system so that its impulse or step response will match to that of the original system in optimum manner with no restriction on the location of eigen values. Such techniques aim at minimizing a selected performance criterian. Which in general, is a function of error between the response of the original high order system and its reduced order approximant. The parameter of reduced order model (ROM) are then obtained either from the necessary conditions of optimality or by means of a search algorithm. The approximations have been studied for step and impulse responses.

Chidambara (1969) gave two techniques for model order reduction where the integral of the squared error between

the step response of the exact and simplified model is minimized. SINHA and BERZNAI [8] solved the problem by using pattern - search algorithms, BANDLR et.al. [9] used three different gradient techniques for the minimization of performance index in the simplification problem. YAHAGI [10] obtained low order model by using the technique of least square fit, linear, programming and parameter optimization. For state space representation the most important results were obtained by WILSON et.al. [11]. But this also requires the solution of Lyapunov type equations.

But whatever be the approach to the problem, the main objective is that the reduced order approximant should reproduce the significant characteristic of the parent system as closely as possible.

1.4.2 Frequency Domain Simplification Techniques

Most frequency domain simplification techniques start with the transfer function description of the original system. The objective in this case is that the frequency domain properties of the original system match closely with those of its reduced order equivalent. They can mainly be classified as:

(i) Continued fraction expansion and truncation (CFE)

This method was first proposed by CHEN and SHIEH [12]. Since then various improvements and extension of this

approach have been presented by Chen and Shieh [13].

Chen [14] has extended the CFE techniques to model reduction and design of multivariable control systems. In the formulation of reduced order models by using CFE techniques, the CFE and inversion operation is extremely time consuming and laborious. Computer oriented algorithms for expansion into continued fraction and their inversion have been devised for various Cauer forms. Shieh et.al. [15] have demostrated that the first, second and third Cauer form formulations for order reduction give good approximations in the transient, steady-space and overall region of the response curve respectively. Shieh and Goldman has shown that a mixture of first and the second Cauer forms give good approximations for both the transient and the steady-space responses.

One difficulty with the CFE approach is that the stability of the model is not guaranteed, even though the original system is stable but this method can be used for single-input single-output as well as multi-input multi-output systems. Chen et.al. [34] has given a method for finding the denominator polynomial of the reduced model using the Routh stability criterian and the numerator polynomial coefficients by CFE technique, hence, reduced model will be stable if the original system is stable.

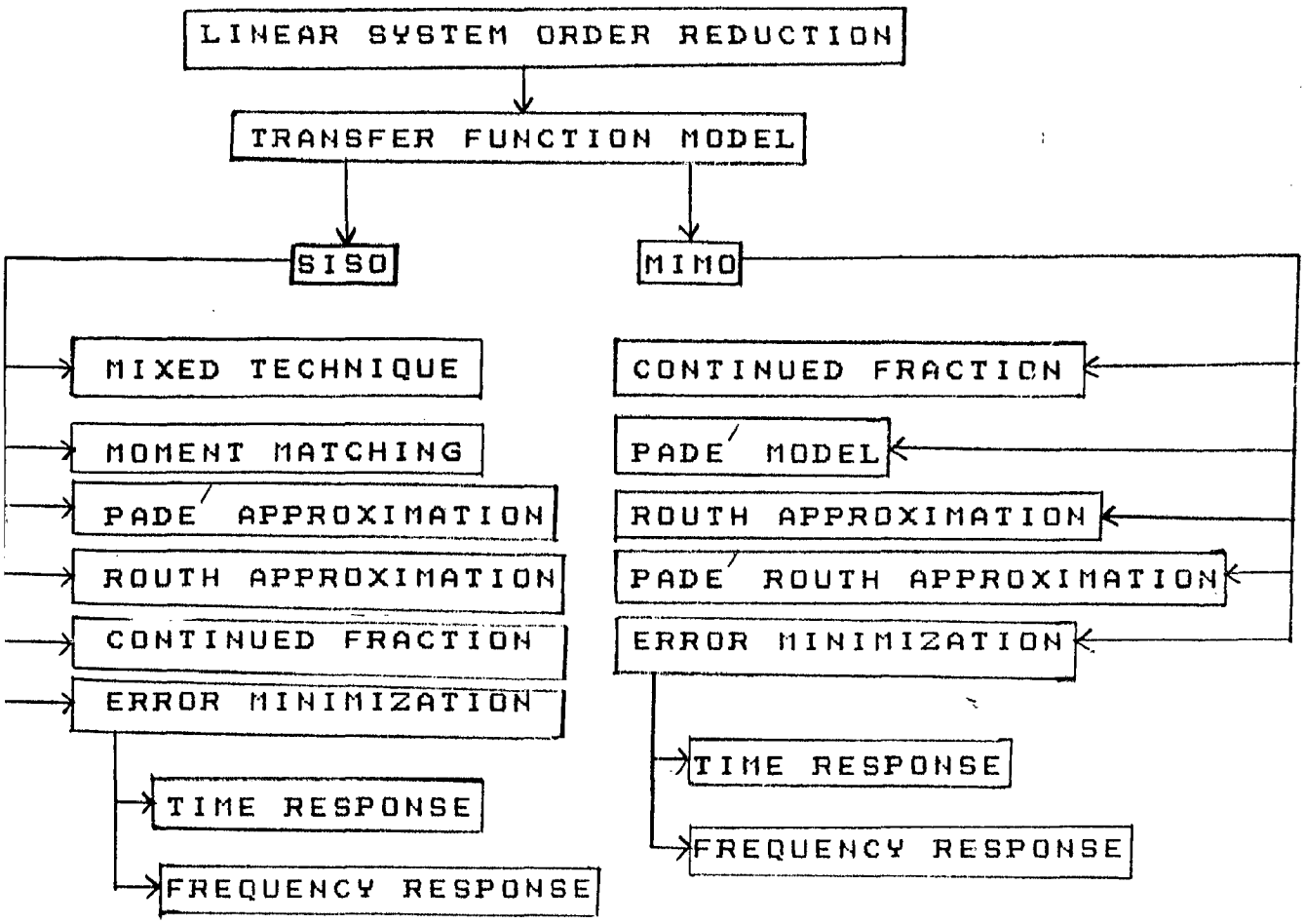


FIG.1.1 CLASSIFICATION OF LINEAR SYSTEM ORDER REDUCTION IN FREQUENCY DOMAIN

(2) Pade'-approximation technique :

Shamash [16] has shown that for the case of rational transfer function, the continued fraction methods are a special case of the time-moments method, which is equivalent to the Pade' approximation method. Pade approximation techniques have a number of very useful advantages, such as, computational simplicity, the fitting of the intial time moments; and the steady-state value of the output of system and model being the same for input of the form $a_i t^i$. The main drawback of this method is that the reduced order model may be unstable (stable) even though the original high order system is stable (unstable). Shamash [18] has given a method for finding the denominator polynomial of the reduced model by using the Routh stability criterion and the numerator polynomial coefficients by matching the initial few time-moments of the system and the model. This ensure that the reduced-order models will always be stable if the high order system is stable.

(3) Moments matching method :

The moments matching technique aims at equating a few lower order moments of the model to those of the original system, and no consideration is given to the remaining moments. This would preserve the low frequency response of

the system G(S) while the transient response of R(S) would be in error. The simplification of large order systems using moments was first suggested by Paynter [19]. A computer oriented algorithm for evaluating moments has been presented by Lal and Mitra [20].

The main drawbacks of methods based on moment matching is that the transient performance of the reduced model may not always be satisfactory and more over there is no guarantee that the reduced order model wil be stable for a stable system.

(4) Matching frequency response :

The main aim of this method is that the magnitude curve of the frequency response of the reduced order model should close enough to the magnitude curve of the frequency response of the original large order system in the bandwidth of interest. In this method due to Levy [21], an error function in the frequency response match over a frequency-range of interest is minimized in a least square sense. Rao et. al. [38] presented a method based on Lavy's curve fitting technique. It is claimed to be useful for classical design of non-linear control system, computational time required is high in this method. The method of Elliott et.al.[35] matches the frequency response of low and high order systems at a prespecified frequencies, though

computational effort is small but there seems to be no prescribed method of selecting the number and frequency points for exact matching. The stability of the reduced system is also not guaranted in this method.

Reddy [40] proposed four error function in which phase error function is minimized between real & imaginary parts of original and reduced order models. The drawback of this method is that reduced model may be unstable though original system is stable. This method can be used for multivariable systems also. Another method for multivariable systems using response matching is reported by Pujara et. al .(37). In this method, error function is minimized at the end frequencies of the band within which matching is required. This method is very simple.Ouyang et.al.[36] reported a mixed method in which the denominator of the reduced order model is constructed from the poles of large disspersion based on the concept of power decomposition i.e. , by neglecting dynamic modes with small dispersion. Numerator parameters are then determined by apply frequency response matching technique. Latest technique in this field is due to Whitfield et.al.[33], they suggested three error criterion method. One is differential equation error criterion, second one is Integral-equation error criterion, and third one is signal error criterion. The first one i.e. differential equation error criterion method is best for

high frequency characteristics matching while integral equation error criterion method is best for low frequency characteristics matching and signal error criterion method is best for middle frequency characteristics matching.

(5) Reduction based on stability criteria :

Hutton and Friedland [22] based their reduction method on α-β expansion that uses the Routh table of the original transfer function. This has a number of useful properties : if the original system is stable, then all approximants will be stable, the sequence of approximants converge monotonically to the original in terms of 'impulse response energy'; the approximants are partial padé approximants in the sense that the first K coefficients of the power series expansions of the K^{th} order approximant and of the original are equal.

This method has the advantage of computational simplicity, because once the R-H array is constructed for the numerator and denominator polynomials; the various reduced order models follow by inspection.

chapter :2

methods for model reduction

2.1 MODEL REDUCTION USING MATRIX CONTINUED FRACTION EXPANSION AND INVERSION

A transfer function matrix is often used to express the relationship between the inputs and outputs of a multi-terminal circuit or a multivariable control system. The expansion of the transfer function matrix into a matrix continued fraction and the inversion of a matrix continued fraction to a transfer function matrix are two fundamentally important operation in multiterminal network and multi-variable system analysis and synthesis. Shieh and Gaudiano [31] developed a generalized Routh algorithm for performing matrix continued fraction expansion and inversion of three matrix cauer forms.

The CFE approach has the major disadvantage that the reduced model may be unstable although the original system is stable. For multivariable systems, this method is restricted to square transfer-function matrices (i.e. the number of inputs and output must be same), but due to Chen et. al. [34]. It becomes possible to get a stable reduced order model by CFE technique.

2.1.1 Method No:1 [31]

Method using second matrix Cauer form CFE

Let the n^{th} order square-transfer function matrix [G(S)] and its r^{th} order reduced equivatent [R(S)]

be representd as

$$[G(s)] = \frac{[A_{2,n}\, S^{n-1} + A_{2,n-1}S^{n-2} + --+ A_{2,3}S^2 + A_{2,2}S + A_{21}]}{[A_{1,n+1}S^n + A_{1,n}S^{n-1} + -- + A_{1,3}S^2 + A_{1,2}S + A_{11}]}$$

..(2.1)

Where A_{iJ} are constant, m by m, matrices, and $A_{1J} = a_J[I]$, J = 1,2,..,n+1. Where each a_J is a coefficient of the common-denominator polynomial or $(S) = \sum_{J=1}^{n+1} a_J S^{J-1}$ and [I] is an identity matrix.

and

$$[R(S)] = \frac{[B_{2,r}S^{r-1} + B_{2,r-1}S^{r-2} + -- + B_{2,3}S^2 + B_{2,2}S + B_{21}]}{[B_{1,r+1}S^r + B_{1,r}S^{r-1} + -- + B_{1,3}S^2 + B_{1,2}S + B_{11}]}$$

..(2.2)

Where $B_{i,J}$ are constant, m by m, matrices, and $B_{1J} = b_J[I]$, J=1,2,..,r+1. Where each b_J is a coefficient of the common-denominator polynomial or

$(S) = \sum_{J=1}^{r+1} b_J\, S^{J-1}$ and [I] is a identity matrix.

Eqn.(2.1) can be expanded in the first matrix Cauer form as

$$[G(S)] = [H_1 + [H_2\frac{1}{S} + [H_3 + [H_4\frac{1}{S} + [--]^{-1}]^{-1}]^{-1}]^{-1}]^{-1}$$

..(2.3)

The reduced models are obtained by truncating the expansion and discarding some partial quotient matrices H_i. The procedure are as follows :

Step 1 : Evaluate H_i, i=1,2,..r; by the generalized matrix Routh algorithm [31]

$$
\begin{array}{lllll}
 & A_{11} & A_{12} & A_{13} & A_{14} \text{ ------} \\
H_1=A_{11}A_{21}^{-1} & A_{21} & A_{22} & A_{23} & \text{---} \\
H_2=A_{21}A_{31}^{-1} & A_{31} & A_{32} & \text{---} & \\
H_3=A_{31}A_{41}^{-1} & A_{41} & \text{---} & & \\
 & \text{-----} & & &
\end{array}
$$

Where,

$$A_{i,J} = A_{i-2,J+1} - H_{i-2}A_{i-1,J+1};\ i=3,4,..,n+1;J=1,2,.. \quad ..(2.4a)$$

$$H_i = A_{i,1}(A_{i+1,1})^{-1};\ i = 1,2,..,2n \quad ..(2.4b)$$

Provided $\det(A_{i+1,1}) \neq 0$ ..(2.4c)

Step 2 : Using H_i coefficients from step 1, evaluate $B_{i,J}$ of eqn.(2.2) by using the following reverse matrix Routh algorithm [31].

$$B_{2r+1,1} = [I] \quad ..(2.5a)$$

$$B_{P,1} = H_P B_{P+1,1};\ P = 2r,2r-1,..,2,1 \quad ..(2.5b)$$

$$B_{J-2,l+1} = B_{J,l} + H_{J-2,l+1};\ J=2r+1,2r,..,3;l=1,2,..,r \quad ..(2.5c)$$

Step 3 : Using eqn.(2.5), the reduced order model as in eqn.(2.2) is formed. If m=1, same method can be used for single input single output systems.

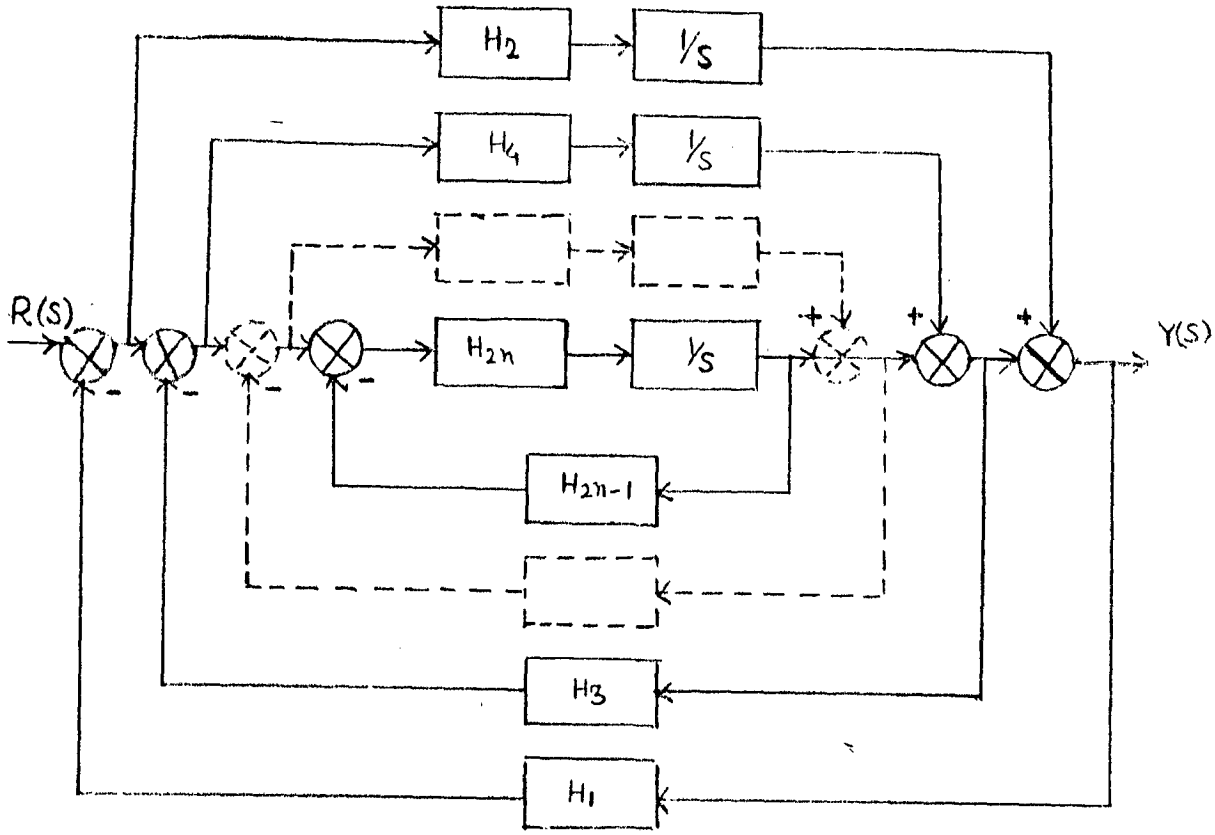


Fig. 2.1 BLOCK DIAGRAM FOR SECOND CAUER FORM

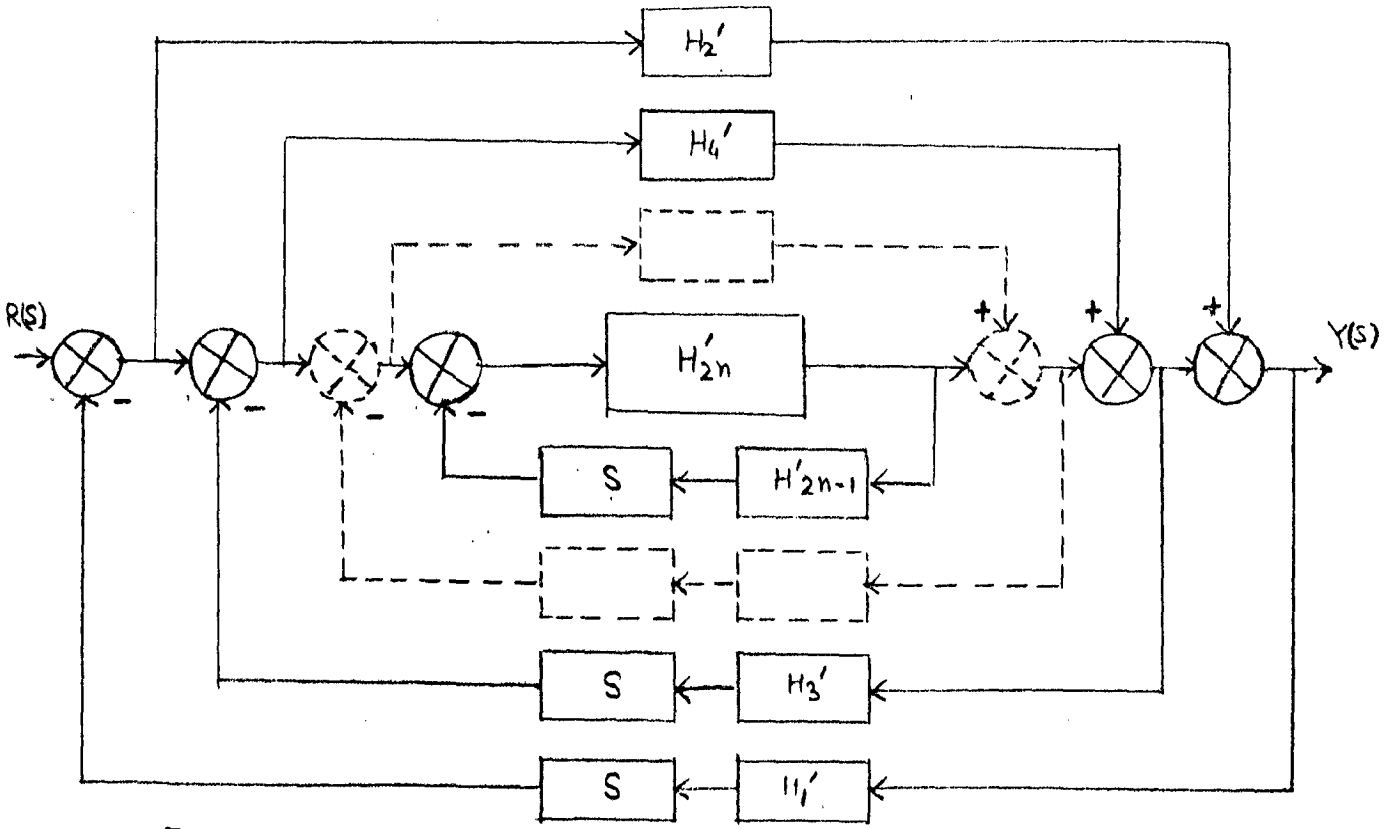


Fig. 2.2 BLOCK DIAGRAM FOR FIRST CAUER FORM

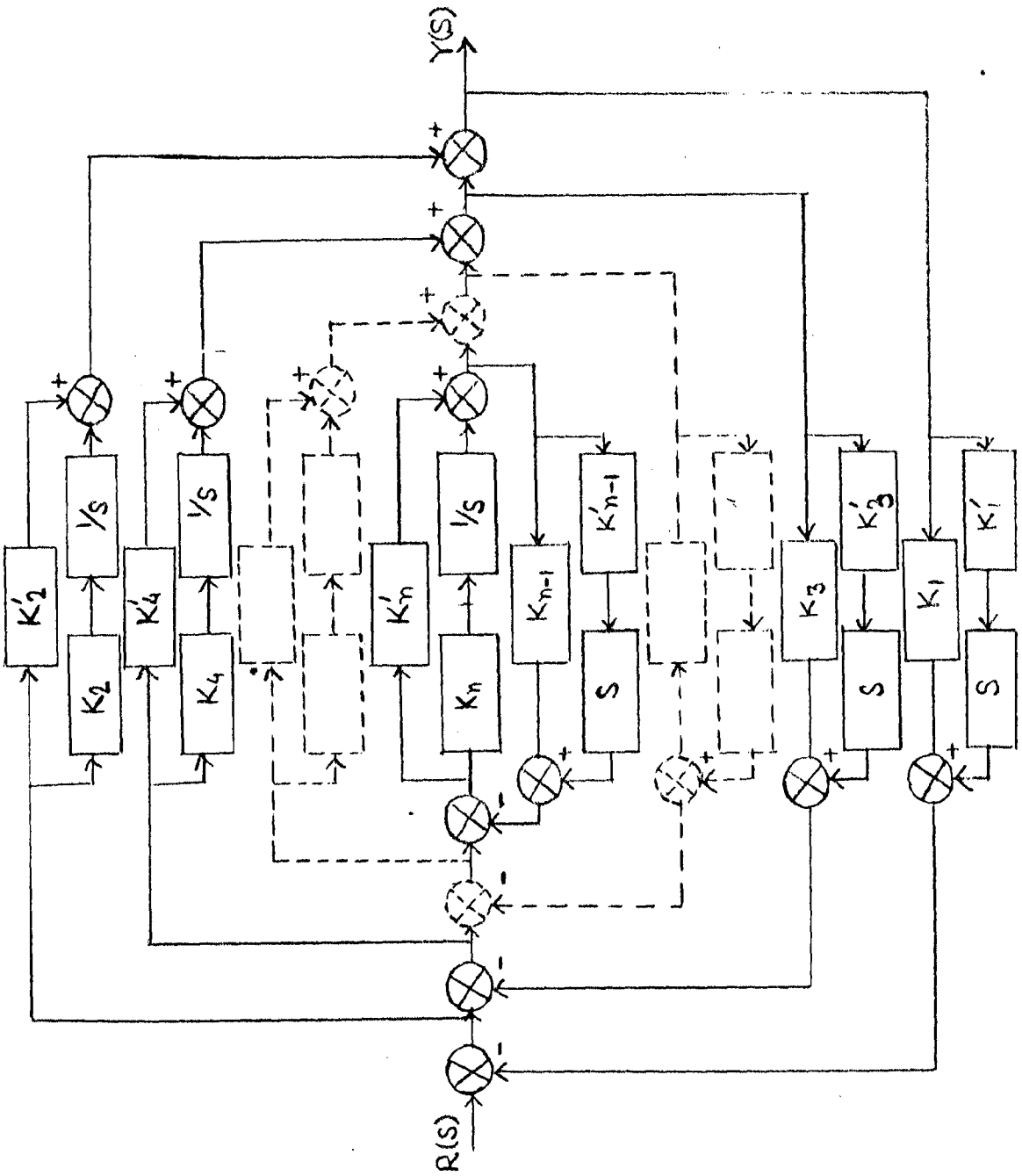


Fig. 2.3 BLOCK DIAGRAM FOR MIXED CAUER FORM

Example 2.1 : This is taken from Shieh [31] and is given by

$$[G(S)] = \left[\begin{pmatrix} -2 & 2 \\ -1 & -1 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} S\right] \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} S + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} S^2\right]^{-1}$$

The generalised matrix Routh array is given by

$$H_1 = \begin{bmatrix} -.25 & -0.5 \\ .25 & -0.5 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H_2 = \begin{bmatrix} -2 & 6 \\ -1 & -1 \end{bmatrix} \qquad \begin{bmatrix} -2 & 2 \\ -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 2 & 0 \\ 1 & 1 \end{bmatrix}$$

$$H_3 = -\begin{bmatrix} .05 & .4 \\ -.05 & .1 \end{bmatrix} \qquad \begin{bmatrix} 1 & .5 \\ 0 & .5 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H_4 = \begin{bmatrix} 4 & -6 \\ 2 & 2 \end{bmatrix} \qquad \begin{bmatrix} -6 & 2 \\ 2 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Solving eqn. (2.5) we get

$$[R_2(S)] = \left[\begin{pmatrix} -2 & 2 \\ -1 & -1 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} S\right]\left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} S + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} S^2\right]^{-1}$$

Example 2.2 : This is taken from (17) and is given by

$$G(S) = \frac{s^3 + 7s^2 + 24s + 24}{s^4 + 10s^3 + 50s + 24 + 35s^2}$$

The generalized Routh array is given by

	24	50	35	10	1
$K_1 = 1$					
	24	24	7	1	
$K_2 = .923$					
	26	28	9	1	
$K_3 = -14.083$					
	-1.846	-1.308	.077		
$K_4 = -.193$					
	9.583	10.083	1		
$K_5 = 15.097$					
	.653	.27			
$K_6 = .106$					
	6.014	1			
$K_7 = 36.667$					
	.164				
$K_8 = .164$					
	1				

Solving eqn.(2.5)d we get

$$R_2(S) = \frac{.730s + 2.504}{s^2 + 3.443s + 2.504}$$

Step responses and Nyquist plots for both original and reduced model are shown in Figures 2.4, 2.5.

2.1.2 Method No.2 [31]

Method using Mixed matrix Cauer form

The n^{th} order system transfer function G(S) given by eqn.(2.1) may be expanded into the third form CFE as

$$[G(S)] = [K_1 + K_1'S + [K_2\frac{1}{s} + K_2' + [K_3 + K_3'S + [K_4\frac{1}{s} + K_4' + [--]^{-1}]^{-1}]^{-1}]^{-1}]^{-1}]^{-1} \quad ..(2.6)$$

For arriving at the r^{th} order model R(S) given by (2.2), the following steps are followed:

Step 1 : Evaluate the matrix quotients of eqn.(2.6) by the generalized matrix Routh algorithm [31].

$$\begin{array}{lll} & A_{11} \quad A_{12}\cdots A_{1,n} \quad A_{1,n+1} & \\ K_1=A_{11}A_{21}^{-1} & & K_1'=A_{1,n+1}\,A_{2,n}^{-1} \\ & A_{21} \quad A_{22}\cdots A_{2,n} & \\ K_2=A_{21}A_{31}^{-1} & & K_2'=A_{2,n}A_{3,n-1}^{-1} \\ & A_{31} \quad A_{32}\cdots A_{3,n-1} & \\ & \vdots & \\ & A_{n,1} \quad A_{n,2} & \\ K_n=A_{n,1} & & K_n'=A_{n,2}(A_{n+1,1})^{-1} \\ (A_{n+1,1})^{-1} & A_{n+1,1} & \end{array}$$

Where,

$$A_{j,1} = A_{J-2,1+1} - K_{J-2}A_{J-1,1+1} - K'_{J-2}A_{J-1,1} \quad ..(2.7a)$$

$$J = 3,4,..,n+1$$

$$1 = 1,2,..$$

and

$$K_p = A_{p,1}(A_{p+1,1})^{-1} \quad ..(2.7b)$$

$$K'_p = A_{p,(n+2-p)}(A_{p+1,n+1-p})^{-1} \quad ..(2.7c)$$

Provided det. $A_{p+1,1} \neq 0$; det. $A_{p+1,n+1-p} \neq 0$

$$P = 1,2,...,n$$

Step 2 : Using K_i and K_i' coefficients from step 1, evaluate $B_{i,J}$ of eqn.(2.2) by using following reverse matrix Routh algorithm,

$$B_{r+1,1} = [I] \qquad ..(2.8a)$$

$$B_{P,1} = K_P B_{P+1,1} \; ; \; P = r, r-1, .., 1 \qquad ..(2.8b)$$

$$B_{P,(r+2-P)} = K'_P \, B_{(P+1),(n+1-P)} \; ; \quad P = r, r-1.., 1 \qquad ..(2.8c)$$

and

$$B_{J-2,1+1} = B_{J,1} + K_{J-2} B_{J-1,1+1} + K'_{J-2} B_{J-1,1} \qquad ..(2.8d)$$

$$J = r+1, r, \ldots, 4, 3$$

$$1 = 1, 2, \ldots, (r+2-J)$$

Step 3 : Using eqn.(2.8), the reduced order model as in eqn.(2.2) is formed. If m=1, same method can be used for single input-single output system.

The generalized matrix Routh array for example 2.1 by this method (method 2) is given by

$$K_1 = \begin{bmatrix} -.25 & -.5 \\ .25 & -.5 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad K'_1 = \begin{bmatrix} .5 & 0 \\ -.5 & 1 \end{bmatrix}$$

$$K_2 = \begin{bmatrix} -1 & .6 \\ -.5 & -.5 \end{bmatrix} \quad \begin{bmatrix} -2 & 2 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 2 & -.5 \\ 0 & 2.5 \end{bmatrix} \qquad K'_2 = \begin{bmatrix} 1 & .2 \\ .5 & .5 \end{bmatrix}$$

Solving eqn.2.8) we get

$$[R_2(S)] = \frac{\begin{bmatrix} -1 & .6 \\ -.5 & -.5 \end{bmatrix} + \begin{bmatrix} 1 & .2 \\ .5 & .5 \end{bmatrix} S}{\begin{bmatrix} .5 & .1 \\ 0 & .4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} s + \begin{bmatrix} .5 & .1 \\ 0 & .4 \end{bmatrix} s^2}$$

and

generalized Routh array for example 2.2 by this method (method 2) is given by

$K_1 = 1$	24 50 35 10 1	$K_1' = 1$
$K_2 = 12$	24 24 7 1	$K_2' = .5$
$K_3 = -.08$	2 4 2	$K_3' = -.105$
$K_4 = 164.931$	-25 -19	$K_4' = 125.347$
	-.152	

Solving eqn. (2.8) we get

$$R_2(S) = \frac{.5\ s + 12}{.5\ s^2 + 13.5s + 12}$$

Step responses and Nyquist plots for both original and reduced model are shown in Figure 2.4 and 2.5.

2.1.3 Method No.3 [31]

Method using First Matrix Cauer Form CFE

The n^{th} order system transfer function G(S) given by eqn.(2.1) may be expanded into the first form CFE as

$$[G(S)] = [H_1'S + [H_2' + [H_3'S + [H_4' + [--]^{-1}]^{-1}\ ]^{-1}\ ]^{-1}\ ]^{-1} \quad ..(2.9)$$

For arriving at the r^{th} order model R(S) given by (2.2), the following steps are followed :

Step 1 : We rewrite the eqn.(2.1) as follows :

$$[G(S)] = \frac{[B_{21}\, S^{n-1} + B_{22}\, S^{n-2} + B_{23}\, S^{n-3} + .. + B_{2n}]}{[B_{11}\, S^{n} + B_{12}\, S^{n-1} + .. + B_{1,n+1}]^{-1}} \qquad ..(2.10a)$$

Where $B_{1,i} = A_{1,(n+2-i)}$; $i = 1,2,..,n+1$ ..(2.10b)

$B_{2,J} = A_{2,(n+1-J)}$; $J = 1,2,..n$ ..(2.10c)

Step 2 : Evaluate H_i' for $i = 1,2,...r$; by the generalized matrix Routh aglrithm [31]

$H_1' = B_{11}B_{21}^{-1}$ $H_2' = B_{21}B_{31}^{-1}$ $H_3' = B_{31}B_{41}^{-1}$

B_{11} B_{12} B_{13} B_{14} - -

B_{21} B_{22} B_{23} --

B_{31} B_{32} --

B_{41} ----

Where, $B_{J,1} = B_{J-2,1+1} - H_{J-2}' B_{J-1,1+1}$; $J = 3,4,...,2r+1$;

$1=1,2,...,r$ ..(2.11a)

and

$$H_p' = B_{p,1}(B_{p+1,1})^{-1}; \quad p = 1,2,... \qquad ..(2.11b)$$

Provided det $B_{p+1,1} \neq 0$

Step 3 : Using H_i' coefficients from step 2, evaluate $A_{i,J}$ by using following reverse matrix Routh algorithm[31].

$$A_{2r+1,1} = [I] \qquad ..(2.12a)$$

$$A_{p,1} = H_p' A_{p+1,1}; \quad p = 2r,2r-1,..,2,1 \qquad ..(2.12b)$$

$A_{J-2,1+1} = A_{J,1} + H_{J-2}' A_{J-1,1+1}$; $J=2r+1,2r,...,3$;

$1 = 1,2,..r$ ..(2.12c)

and calculate $B_{i,J}$ by following equations.

$$B_{1,i} = A_{1,(r+2-i)} \; ; \; i = 1,2,..,r+1 \qquad ..(2.12d)$$

$$B_{2,J} = A_{2,(r+1-J)} ; \; J = 1,2,...,r \qquad ..(2.12e)$$

Step 4 : Using eqn.(2.12) the reduced order model as in the eqn.(2.2) is formed.

If m = 1 same method can be used for single input single output systems.

The generalized matrix Routh array for example 2.1 by the method (method no.3) is given by

$$H_1' = \begin{bmatrix} .5 & 0 \\ -.5 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H_2' = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 2 & 0 \\ 1 & 1 \end{bmatrix} \quad \begin{bmatrix} -2 & 2 \\ -1 & -1 \end{bmatrix}$$

$$H_3' = \begin{bmatrix} -.4 & .3 \\ .4 & -.8 \end{bmatrix} \quad \begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H_4' = \begin{bmatrix} -4 & 1 \\ -2 & -2 \end{bmatrix} \quad \begin{bmatrix} -4 & 1 \\ -2 & -2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Solving eqn.(2.12 we get

$$[R_2(S)] = \frac{\begin{bmatrix} -2 & 2 \\ -1 & -1 \end{bmatrix} + \begin{bmatrix} 2 & 0 \\ 1 & 1 \end{bmatrix} s}{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} s + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} s^2}$$

and

Generalized Routh array for example 2.2 by this method (method no.3) is given by

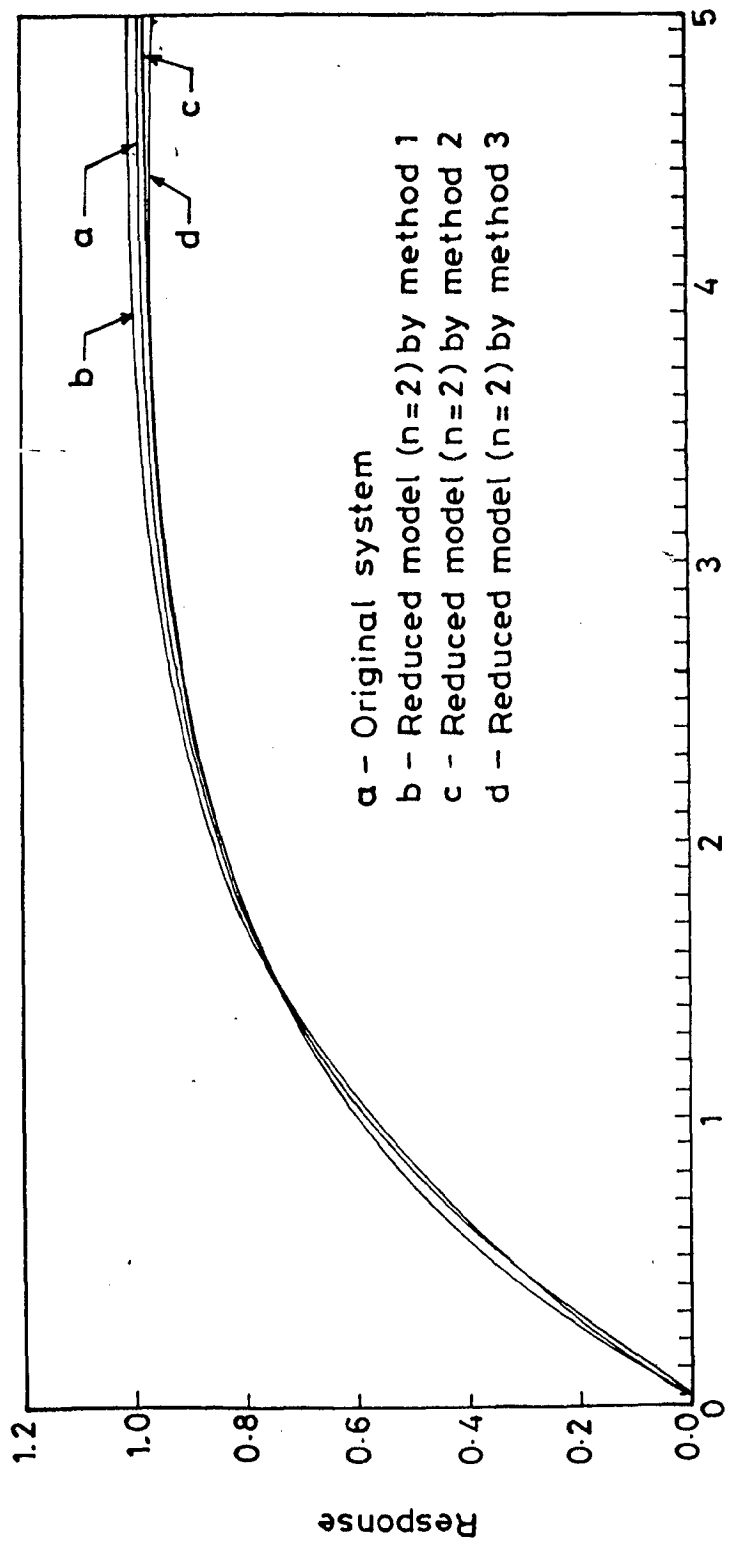


Fig. 2.4 Step responses for example 2.2

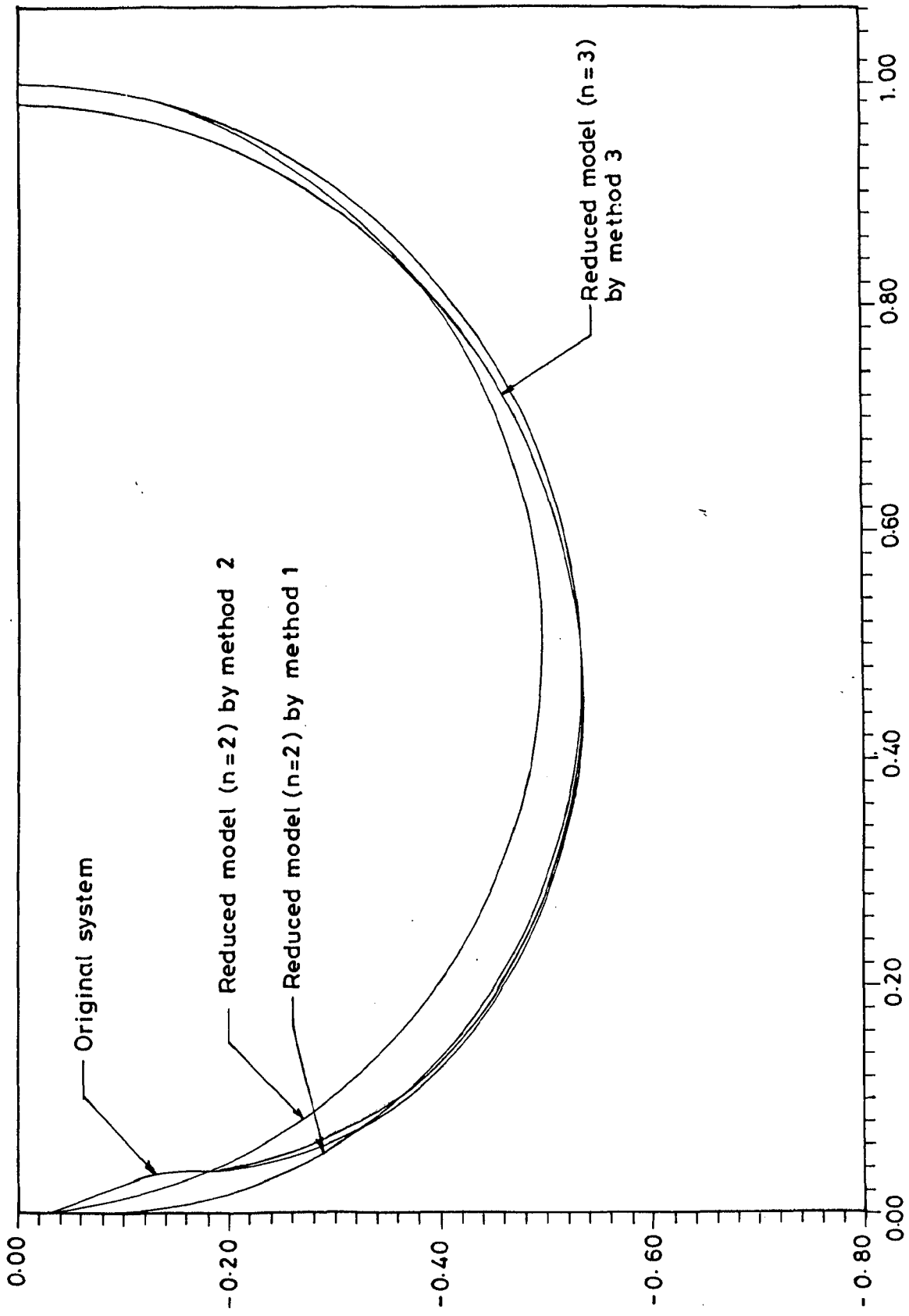


Fig. 2.5 Nyquist plots for example 2.2

	1	10	35	50	24
$H_1'=1$					
	1	7	24	24	
$H_2'= .33$					
	3	11	26	24	
$H_3'= .90$					
	3.33	15.3	16		
$H_4'= -1.19$					
	-2.8	11.6	24		
$H_5'= -.1$					
	29.1	44.6			
$H_6'= 1.83$					
	15.9	24			
$H_7'= 29.78$					
	.53				
$H_8' = .02$					
	24				

Solving eqn. (2.12) we get

$$R_3(S) = \frac{.06s^2+ .34s + .98}{.06s^3+ .53s^2+1.38s+1}$$

Step responses and Nyquist plots for both original and reduced model are shown in Fig.2.4 and 2.5.

2.2 MODEL REDUCTION USING THE ROUTH STABILITY CRITERION

Krishnamurty et. al. [32] have presented an interesting method for the reduction of dimension of a high order transfer function. Their method makes use of the classical Routh-Hurwitz stability array and is applicable to single-input single-output systems. The advantages of this method is that if original system is stable then reduced order model will be stable and computational time is less than CFE method.

Let the transfer function of n^{th} order system be

$$G(S) = \frac{b_{11}S^m + b_{21}S^{m-1} + b_{12}S^{m-2} + b_{22}S^{m-3}+--}{a_{11}S^n + a_{21}S^{n-1} + a_{12}S^{n-2} + a_{22}S^{n-3}+--} \quad ..(2.13)$$

Where $m \leqslant n$ -

To get the reduced order model by this method, the procedure are as follows :

Step 1 : Form Routh stability array for the numerator polynomial. Routh stability array for the numerator polynomial for (2.13) is given by

$$\begin{array}{llll} b_{11} & b_{12} & b_{13} & b_{14} - - - \\ b_{21} & b_{22} & b_{23} & b_{24} - - - \\ b_{31} & b_{32} & b_{33} & -- \\ b_{41} & b_{42} & b_{43} & -- \\ \vdots & & & \\ b_{m,1} & & & \\ b_{m+1,1} & & & \end{array} \quad ..(2.14)$$

Where first row of array consists of odd coefficients (i.e., first, third, fifth, etc.) and the second row consists of even coefficients (i.e., second, forth, sixth, etc.)

The routh array are completed in the conventional way by computing the coefficients of succeeding rows by the algorithm.

$$B_{iJ} = B_{i-2,J+1} - (B_{i-2,1} \cdot B_{i-1,J+1})/B_{i-1,1}$$

; $i \geq 3$ and $1 \leq J \leq (n-i+3)/2$ ..(2.15)

and [.] stands for the integral part of the quantity.

Step 2 Form Routh stability array for the denominator polynomial. The Routh stability array for the denominator polynomial for (2.13) is given by

$$\begin{array}{llll} a_{11} & a_{12} & a_{13} & a_{14} \text{ ---} \\ a_{21} & a_{22} & b_{23} & a_{24} \text{ ---} \\ a_{31} & a_{32} & a_{33} & \text{--} \\ a_{41} & a_{42} & a_{43} & \text{--} \\ \vdots & & & \\ a_{n,1} & & & \\ a_{n+1,1} & & & \end{array} \qquad \text{..(2.16)}$$

Where first row of array consists of odd coefficients and the second row the event coefficients.

The Routh array are completed in the conventional way by computing the coefficient of succeeding rows by the algorithm.

$$a_{iJ} = a_{i-2,J+1} - (a_{i-2,1} \cdot a_{i-1,J+1})/a_{i-1,1};$$

$i \geq 3$ and $1 \leq J \leq (n-i+3)/2$ ..(2.17)

and [.] stands for integral part of the quantity.

Step 3 : r^{th} order numerator polynomial may be easily constructed with the $(n+2-r)^{th}$ and $(n+3-r)^{th}$ rows of Routh stability array of numerator and denominator polynomial with the $(n+1-r)^{th}$ and $(n+2-r)^{th}$ rows of Routh stability array of denominator polynomial of original system.

Example 2.3 : This is taken from Krishnamurthy [32] and is given by

$$G(S) = \frac{35s^7+1086s^6+13285s^5+82402s^4+278376s^3+511812s^2+482964s+194480}{s^8+33s^7+437s^6+3017s^5+11870s^4+27470s^3+37492s^2+28880s+9600}$$

The complete numeraor and denominator Routh array are given below

Numerator table

35	13285	278376	482964
1086	82402	511812	194480
10629.3	261881.1	476696.1	
55645.5	463107.8	194480.0	
173419.1	439546.9		
322069	194480		
334828.5			
194480			

Denominator table

1	437	11870	37492	9600
33	3017	27470	28880	
345.6	11037.6	36616.8	9600	
1963	23973.4	27963.3		
6817.2	31694	9600		
14847.1	25199			
20123.7	9600			
18116.2				
9600				

hence third order model is given by

$$R_3(S) = \frac{322069s^2 + 334828.5s + 194480}{14847.1s^3 + 20123.7s^2 + 25199s + 9600}$$

$$R_3(S) = \frac{32.83\ s^2 + 60.98s + 39.45}{s^3 + 3.14s^2 + 4.03s + 1.95} \quad \text{(BY method 1)}$$

$$R_3(S) = \frac{.05\ s^2 + 3.23s + 23.16}{.002s^3 + .1s^2 + .84s + 1.00} \quad \text{(BY method 3)}$$

Step response and Nyquist plot for all three cases are shown in Fig.(2.6) and (2.7). Different order reduced Tansfer function of example 2.3 by method 1, method 2 and method 3 are shown in tbale 1 and step responses and Nyquist plots for these models are shown in Figures from 2.8 to 2.15.

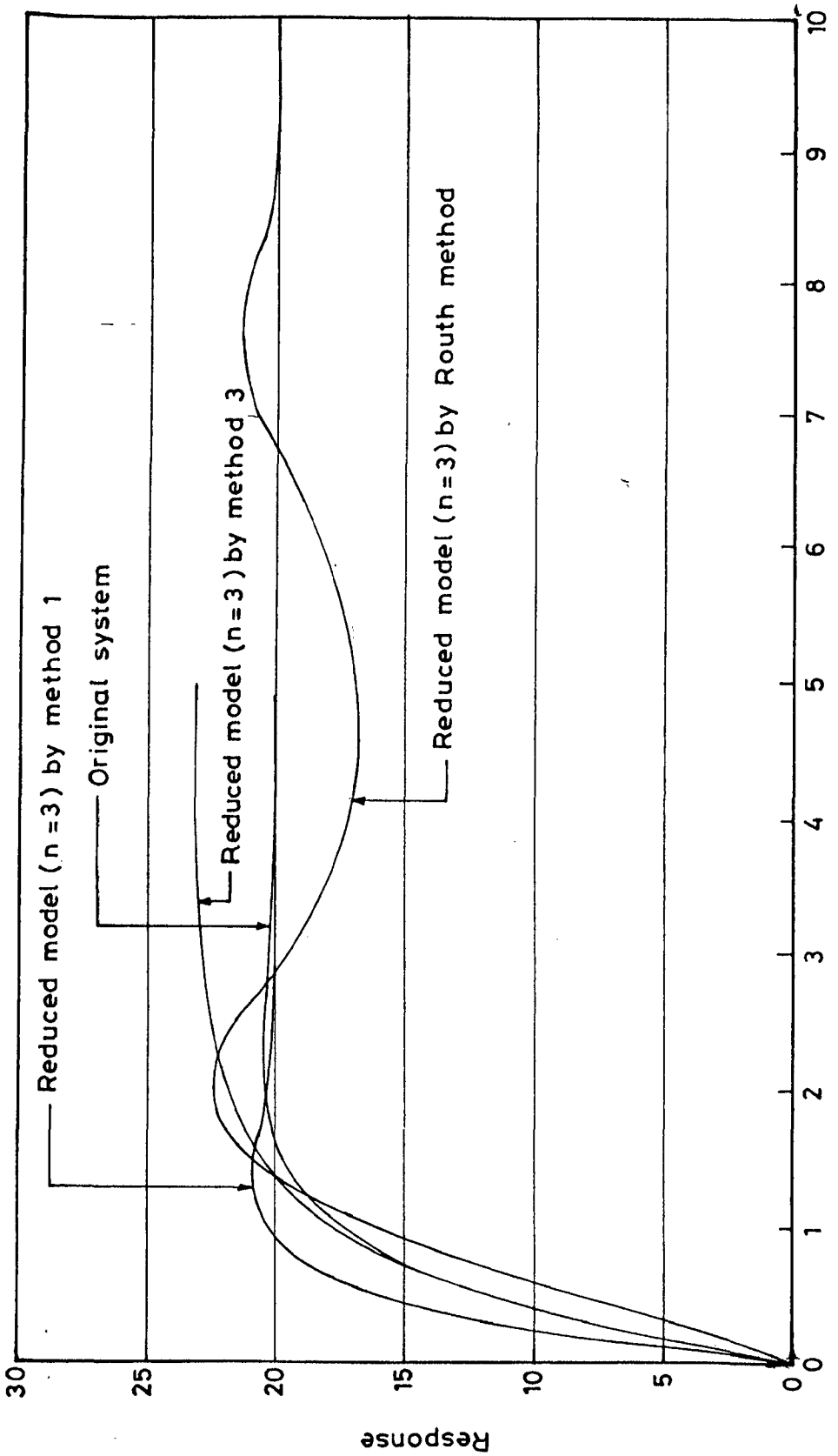


Fig. 2.6 Step responses for example 2.3

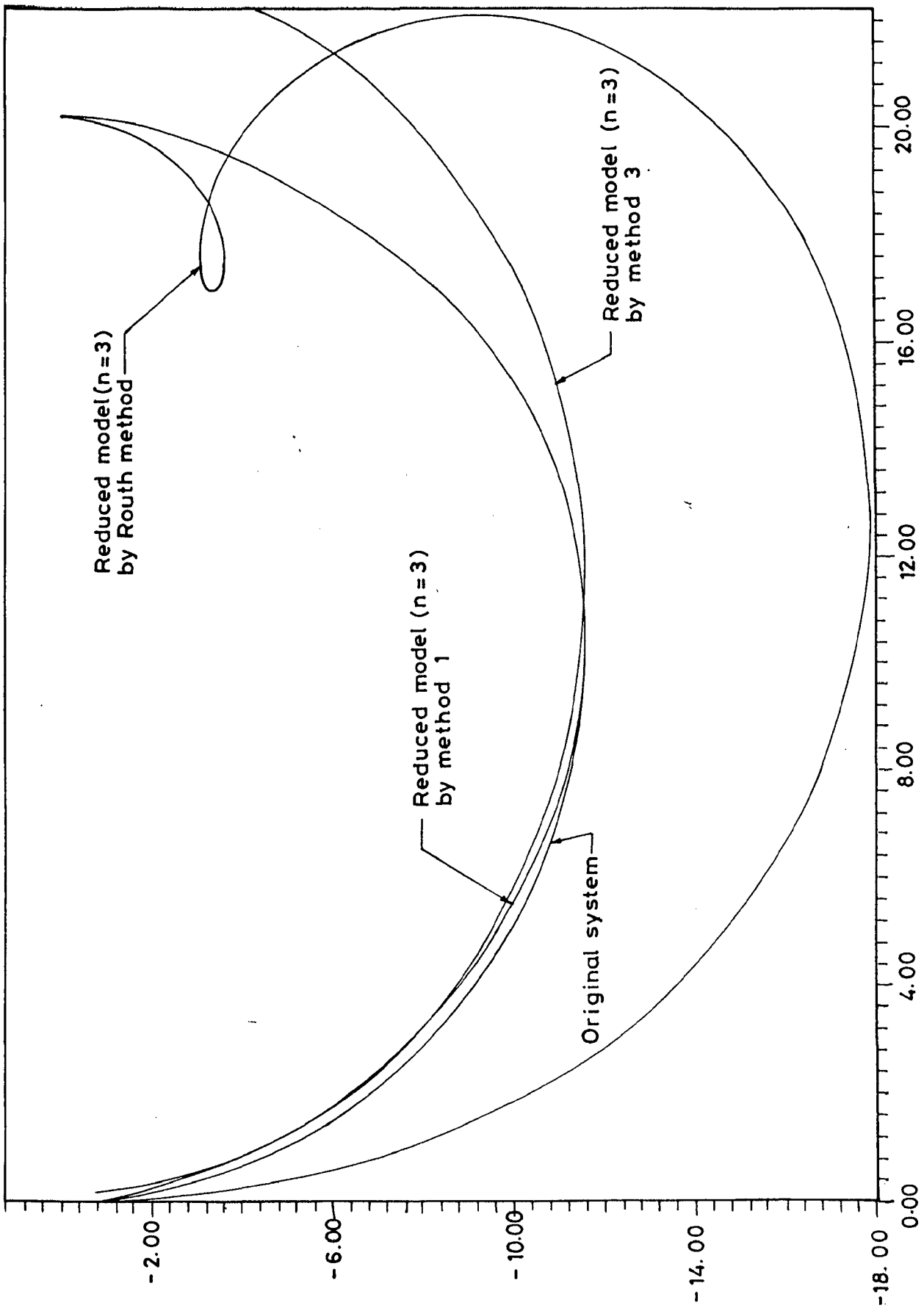


Fig. 2.7 Nyquist plots for example 2.3

Table 1: Different reduced order T.F. for example 2.3.

Sl. No.	Order of the model	Reduced T.F. by Method 1	Method 2	Method 3
1	7	$\frac{35s^6+828.39s^5+7316.61s^4+30716.85s^3+64717.83s^2+66490.28s+28626.66}{s^7+25.64s^6+251.97s^5+1231.1s^4+3268.7s^3+4875.29s^2+4023.95s+1413.08}$	$\frac{1.03s^6+27.12s^5+265.23s^4+1207.35s^3+2691.35s^2+2865.7s+1270.94}{.03s^7+.83s^6+9s^5+47.43s^4+132.81s^3+205.23s^2+174.39s+62.74}$	$\frac{.01s^6+.35s^5+3.52s^4+16.75s^3+39s^2+43.28s+20.24}{.003s^7+.01s^6+.12s^5+.65s^4+1.89s^3+3.02s^2+2.66s+1}$
2	6	$\frac{35s^5+612s^4+3601.2s^3+9134.68s^2+10417s+4857.2}{s^6+19.46s^5+133.65s^4+428.54s^3+713.67s^2+640s+239.76}$	$\frac{5.84s^5+120.52s^4+898.94s^3+2387s^2+2906s+1430.82}{.17s^6+3.77s^5+30.12s^4+107.62s^3+191s^2+180.53s+70.63}$	$\frac{.02s^5+.58s^4+4.97s^3+18.16s^2+29.36s+19.99}{.003s^6+.02s^5+.17s^4+.76s^3+1.67s^2+1.94s+1}$

contd..

3	5	$\frac{34.96s^4+405.22s^3+1384.63s^2+1822.48s+946.51}{s^5+13.52s^4+58.93s^3+114.18s^2+114.49s+46.72}$	$\frac{.75s^4+10.85s^3+43.22s^2+60.4s+32.95}{.02s^5+.35s^4+1.77s^3+3.65s^2+3.84s+1.63}$	$\frac{.06s^4+1.3s^3+9.17s^2+23.74s+19.58}{.003s^5+.04s^4+.33s^3+1.11s^2+1.62s+1}$
4	4	$\frac{34.62s^3+185.91s^2+287.34s+167.57}{s^4+7.12s^3+16.37s^2+18.53s+8.27}$	$\frac{3.17s^3+28.89s^2+37.62s+25.46}{.09s^4+s^3+2.36s^2+2.52s+1.26}$	$\frac{1.08s^3+15.8s^2+54.47s+13.87}{.03s^4+.51s^3+2.33s^2+2.72s+1}$

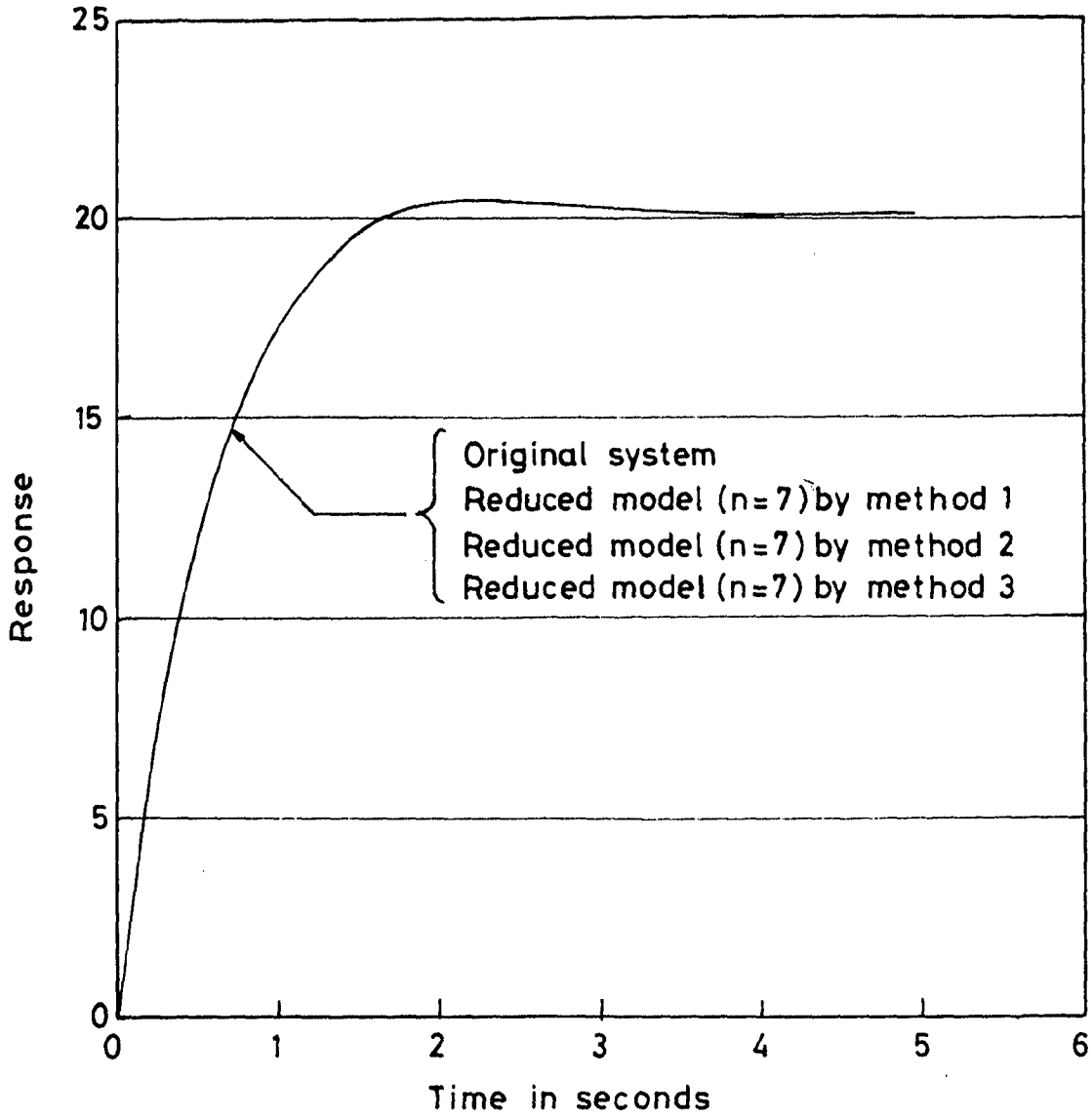


Fig. 2.8 Step responses for example 2.3

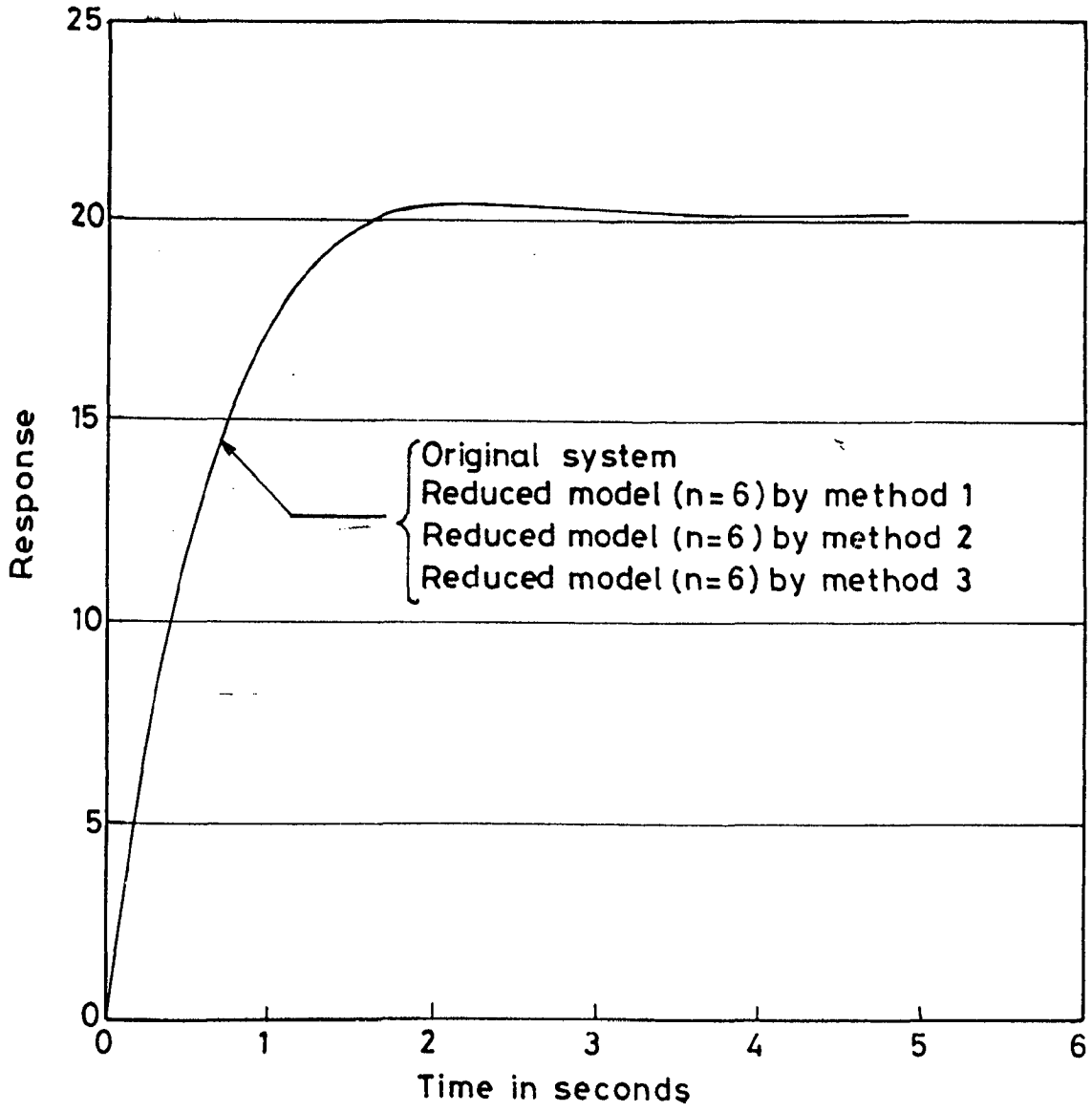


Fig. 2.9 Step responses for example 2.3

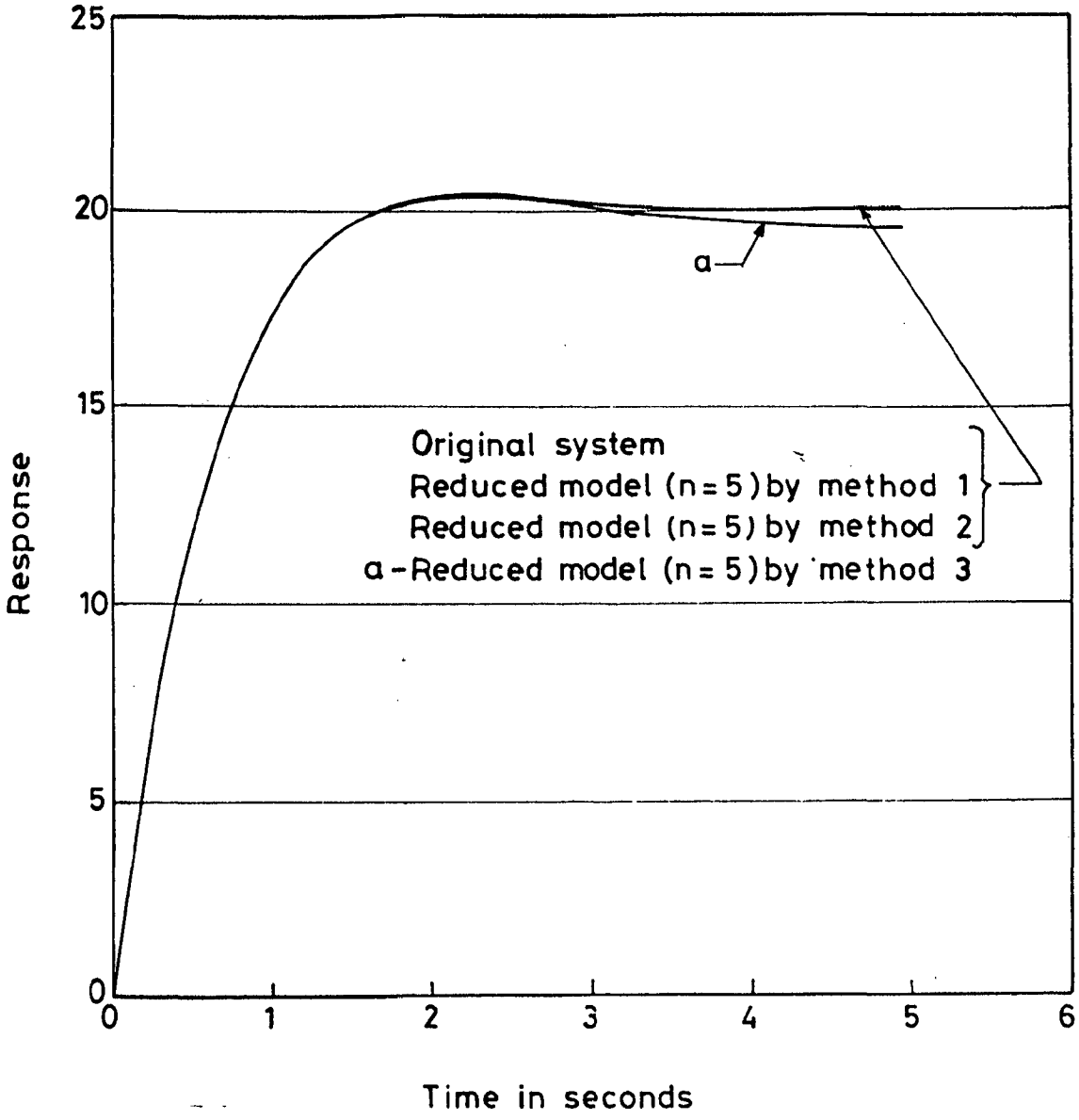


Fig. 2.10 Step responses for example 2.3

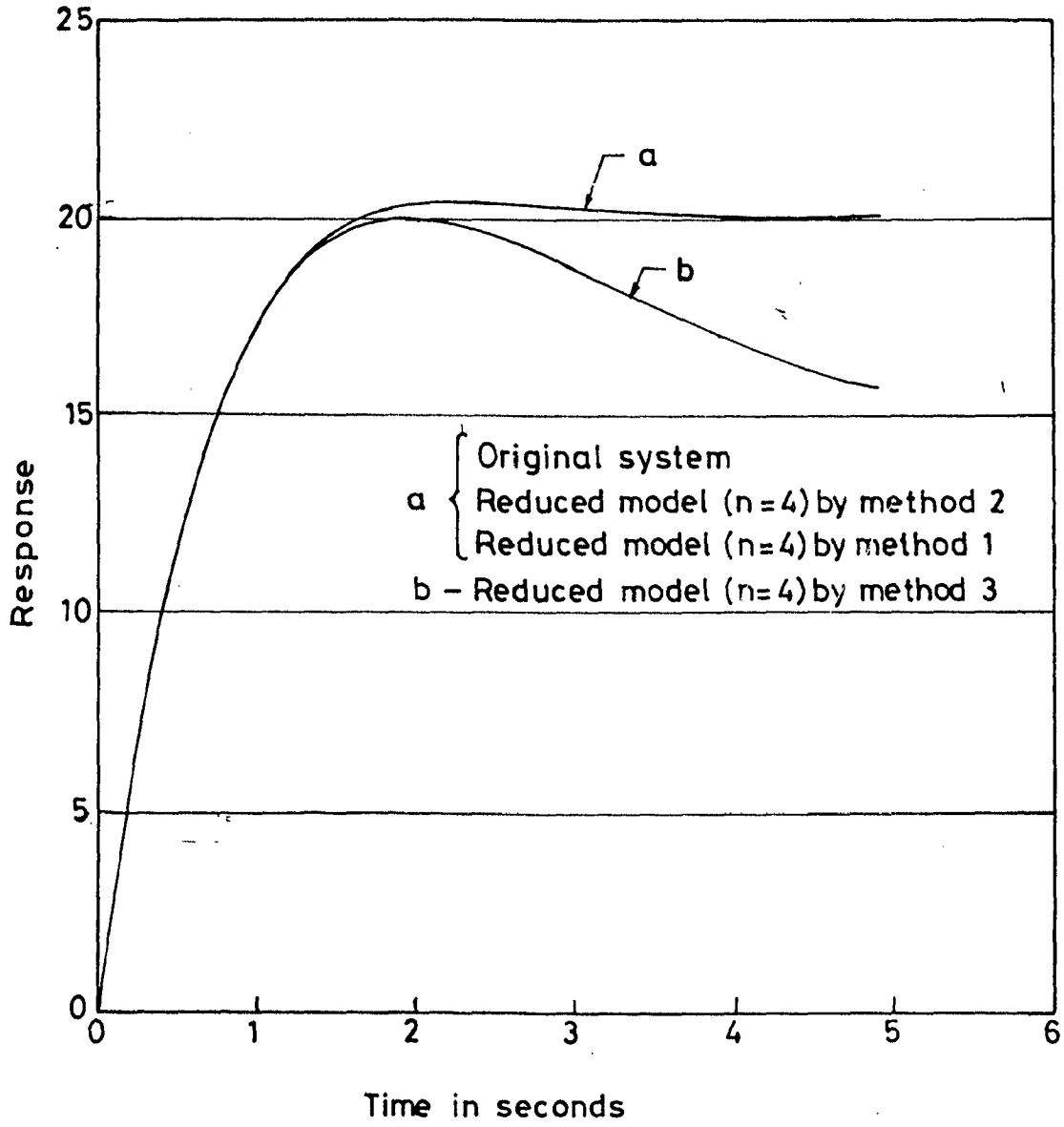


Fig. 2.11 Step responses for example 2.3

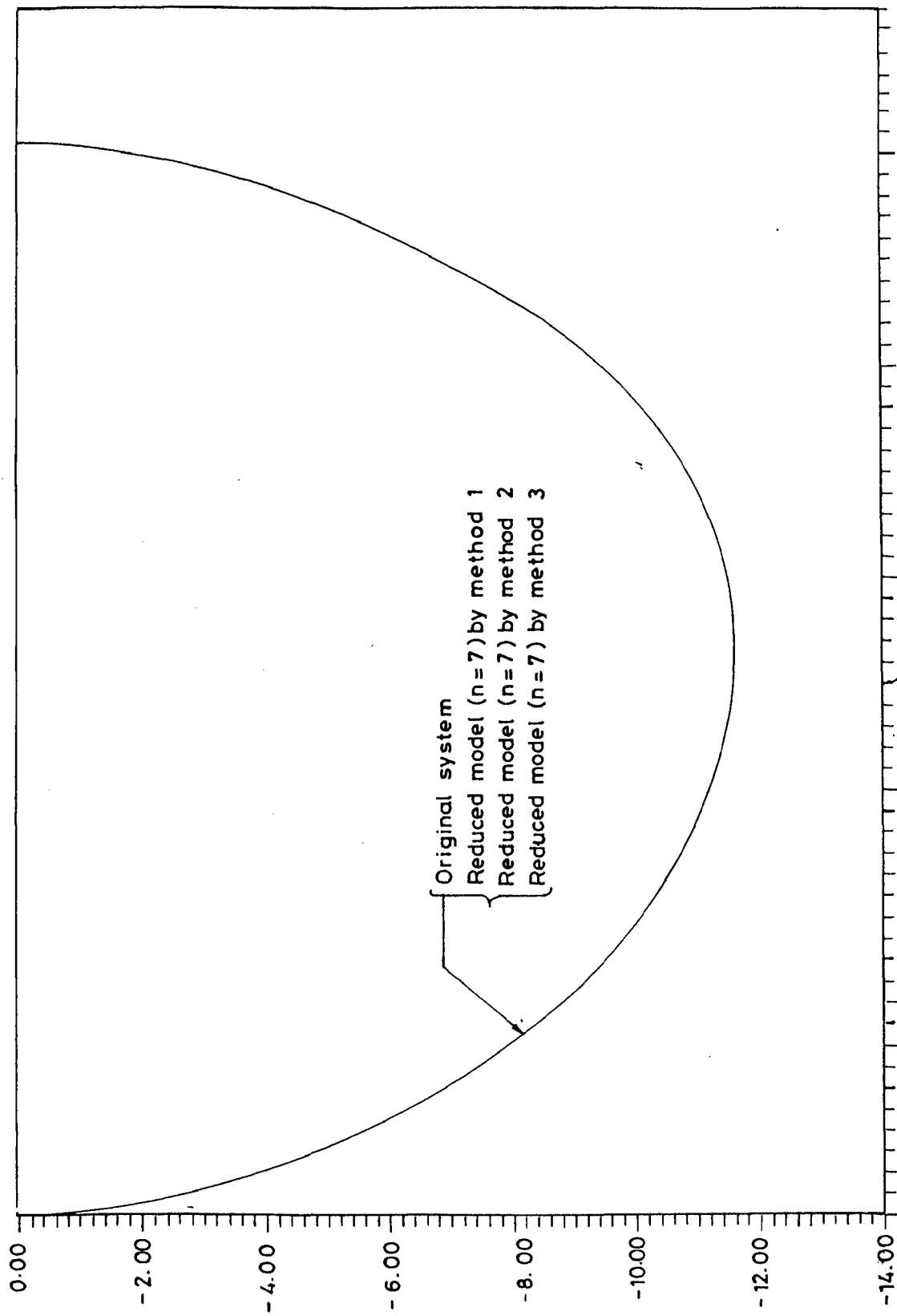


Fig. 2.12 Nyquist plots for example 2.3

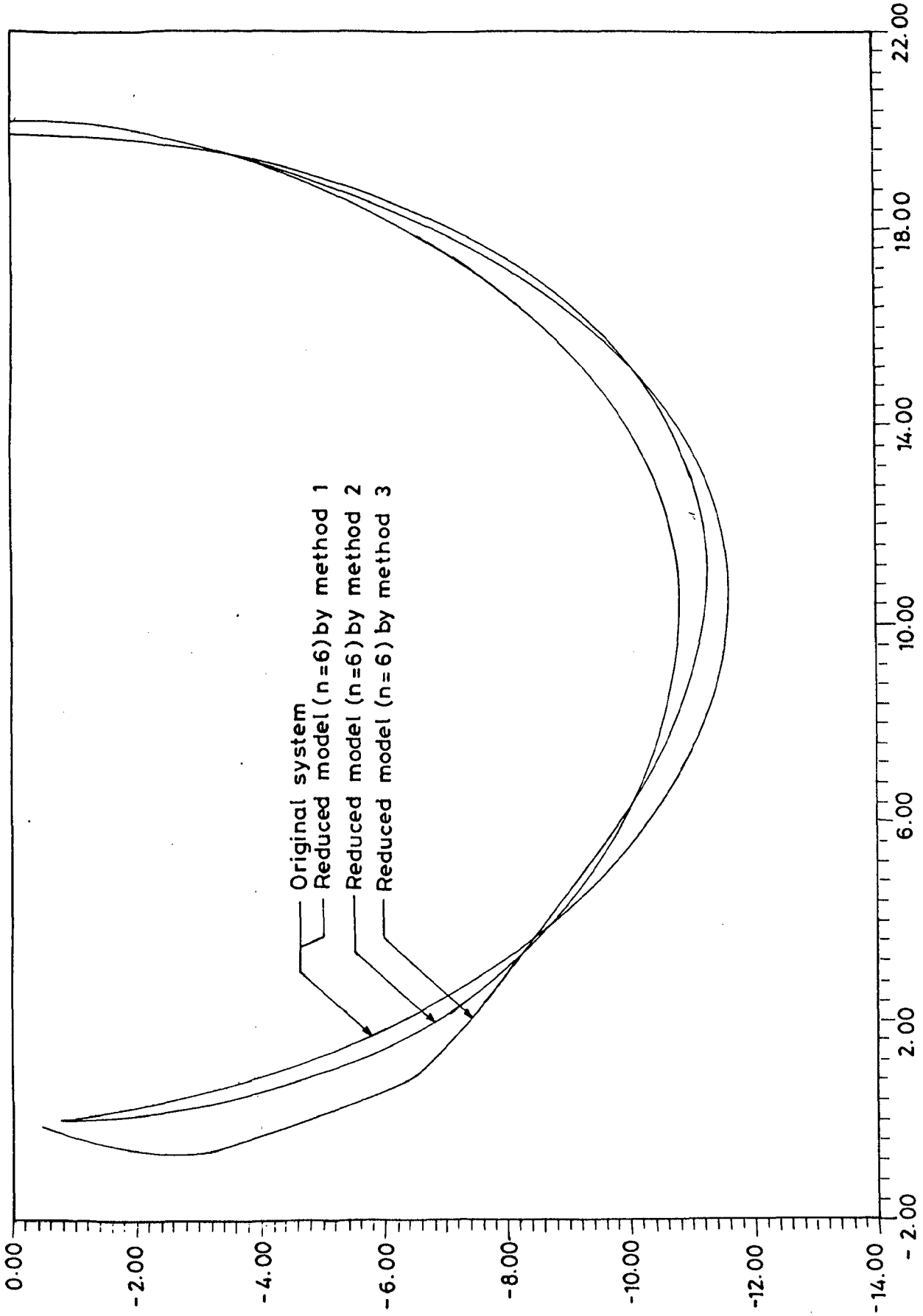


Fig. 2.13 Nyquist plots for example 2.3

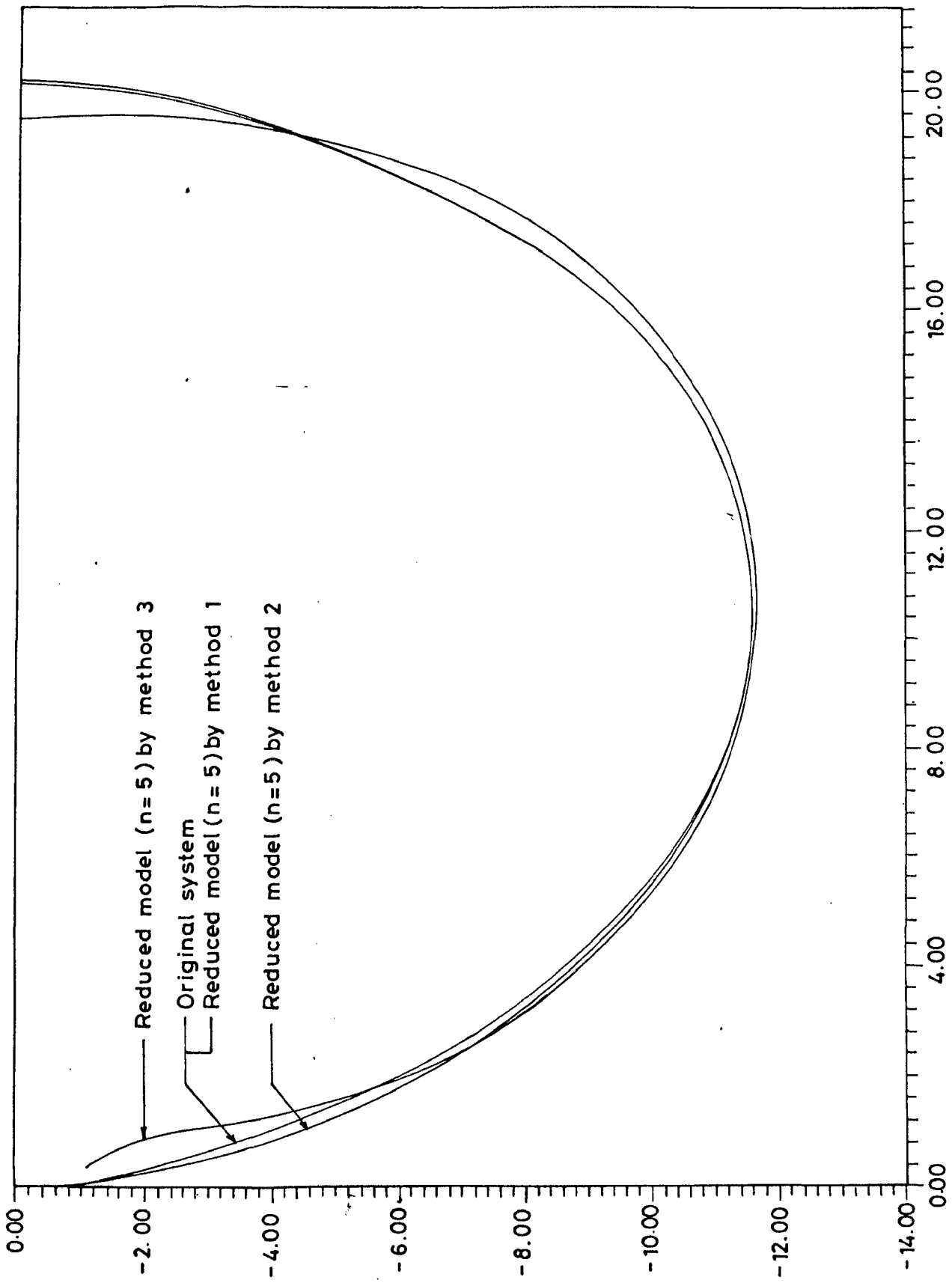


Fig. 2.14 Nyquist plots for example 2.3

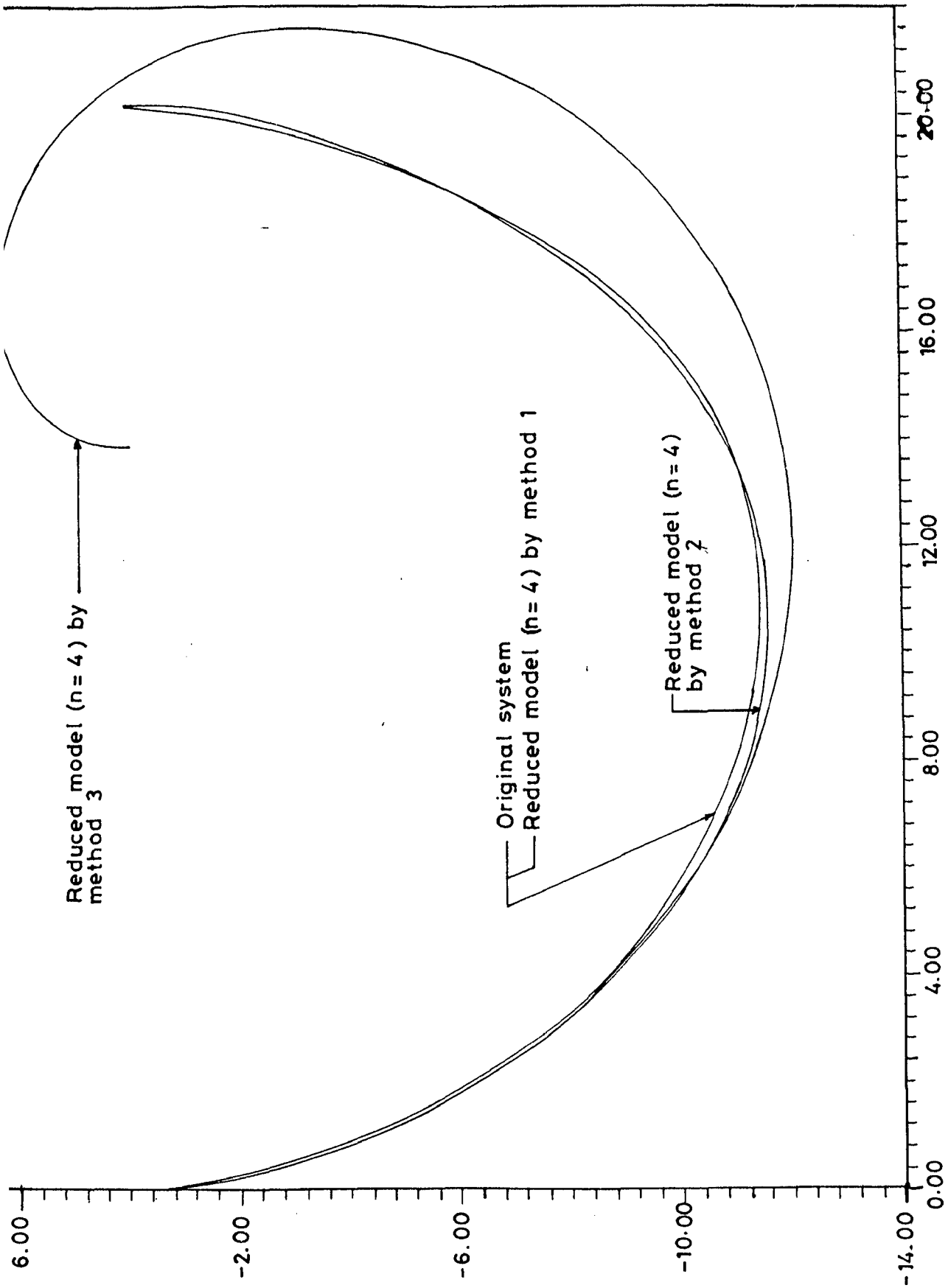


Fig. 2.15 Nyquist plots for example 2.3

2.3 REDUCTION BY INTEGRAL LEAST-SQUARES TECHNIQUES

Several least-squares methods have been proposed as solutions to the problem of fitting a transfer function to a set of frequency response data. The original linear least squares frequency domain curve fitting method was due to Levy in 1959, but is known to give biased parameter estimates. The method was improved by Sanathanan and Koerner in 1963 and in 1979 Lawrence and Rogers introducing an iterative error criterion. In 1988 Whitfield and Williams proposed three different techniques for linear SISO system model reduction [33]. Each method is based on an integral error criterion in the frequency domain which is derived from an equivalent time-domain criterion.

2.3.1 Differential-equation Error Criterion [33]

Let a specified high-order model transfer function G(S) is used to generate frequency response data $\{G(J\omega_k);$ $k = 1,2,..,M\}$ and the required low-order approximate transfer function of specified order n, be represented as

$$G(S) = \frac{A(S)}{B(S)} = \frac{a_o + a_1 S + - - + a_m S^m}{b_o + b_1 S + - - + b_{n-1} S^{n-1} + S^n} \quad ..(2.18)$$

The sought parameters $(a_o,..a_m, b_o,..,b_{n-1})$ will be represent as (a^T, b^T). The system governed by the transfer function (2.18) can also be represented by the associated time-domain differential equation.

$$x^{(n)}(t) + b_{n-1}\, x^{(n-1)}(t) + -- + b_1 x^{(1)}(t) + b_o x(t) =$$

$$a_m\, r^{(m)}(t) + ---+ a_1 r^{(1)}(t) + a_o r(t) \qquad ..(2.19)$$

The error function in time domain is given by [33] as

$$e_d(t) = [y^{(n)}(t) + b_{n-1}\, y^{(n-1)}(t) + -- + b_1 y^{(1)}(t) + b_o y(t)]$$

$$-a_m\, r^{(m)}(t) + - - - + a_1 r^{(1)}(t) + a_o r(t)] \qquad ..(2.20)$$

Where an input r(t) produces output x(t) for lower order system and y(t) denotes the response of a higher order system to the same input i.e. r(t). In frequency domain error function is given by [33]

$$E_{da} = \sum_{k=1}^{M} \omega_k \left| B(J\omega_k)G(J\omega_k) - A(J\omega_k)^2 \right|^2 \left| R\, J\omega_k \right|^2 \qquad ..(2.21)$$

in which $\{\omega_k = 1,2,..\ M\}$ range between ω_{min} and ω_{max}.

Eqn. (2.21) can be solved by linear least-squares solution techniques.

2.3.2 Integral-equation error criterion

For zero initial conditions the transfer function represented by (2.18) can also be represented by the associated time-domain integral eqn.

$$X^o(t) + b_{n-1}\, X^{(1)}(t) + --- + b_1\, X^{(n-1)}(t) + b_o X^{(n)}(t)$$

$$= a_m R^{(n-m)}(t) + ---+ a_1\, R^{(n-1)}(t) + a_o\, R^{(n)}(t) \qquad ..(2.22)$$

Where an input r(t) produces a corresponding output x(t) and the notation is described by the recurrence relationship.

$$X^{o}(t) = X(t) \qquad ..(2.22a)$$

$$X^{(i)}(t) = \int_{o}^{t} X^{(i-1)}(\tau)d\tau \; ; \; i=1,2,.. \qquad ..(2.22b)$$

If y(t) denotes the response of a higher order system to a same input r(t), then, error function in time domain is given by [33] as

$$e_i(t) = [y^{o}(t) + b_{n-1}y^{(1)}(t) + ..+b_1y^{(n-1)}(t) + b_oy^{(n)}(t)] - [a_mR^{(n-m)}(t) + .. + a_1 R^{(n-1)}(t) + a_0 R^{(n)}(t)] \qquad ..(2.23)$$

and corresponding error function is frequency domain is given by [33] as

$$E_{ia} = \sum_{K=1}^{M} \omega_k \left|B(J\omega_k)G(J\omega_k) - A(J\omega_k)\right|^2 \left|R(J\omega_k)\right|^2 / \omega_k^{2n} \qquad ..(2.24)$$

Eqn.(2.24) can be solved by linear least-square solution techniques.

2.3.3 Signal Error Criterion

If a signal r(t) is input to a system with transfer function G(S) subjected to zero initials conditions and the corresponding output signal y(t). If the same signal r(t) is input to the reduced transfer function A(S)/B(S) and the corresponding output signal is x(t), then the error between two output signals is given by

$$e_{(S)} = y(t) - x(t) \qquad ..(2.25)$$

The error function in time domain is given by [33] as

$$E_S = \int_o^\infty e_S^2\,(t)dt \qquad ..(2.26)$$

and In frequency domain error function is given by [33] as

$$E_{sd} = \sum_{k=1}^{M} \omega_k \left| G(j\omega_k) - \frac{A(j\omega_k)}{B(J\omega_k)} \right|^2 |R(J\omega_k)|^2 \qquad ..(2.27)$$

Eqn.(2.27) can be solved by non-linear least-squares solution techniques.

2.3.4 Linear least-squares Solution Techniques

The problem of linear least-squares problems is given by [33]

$$\underset{X}{\text{Min}} \sum_{i=1}^{u} \left[\sum_{J=1}^{v} H_{iJ}\, X_J - C_J \right]^2 \qquad ..(2.28)$$

in which the vector of optimizable parameters is denoted x $= (x_1, x_2, .., x_v)^T$ and the constant $\{H_{iJ};\ i = 1,2,..u;J=1,2, ...,v\}$ and $\{C_i;\ i = 1,2,..u\}$ are prescribed. The optimal solution of (2.28) denoted X* is given by [33].

$$X^* = (H^T H)^{-1}\ H^T C \qquad ..(2.29)$$

in which H is a (uxv) dimentional matrix and C is a u-dimensional vector.

Eqn. (2.29) can also be written as

$$(H^T H)X^* = H^T C \qquad ..(2.30)$$

The preferred approach is to find the Householder solution to the associated set of over determined equations

$$H\,X^* = C \qquad ..(2.31)$$

The differential-equation error criterian E_{da} defined by (2.21) and Integral equation error criterian E_{ia} defined by (2.24) are of the linear least square type.

Both criteria can be expressed in the general form [33] by

$$E_a = \sum_{k=1}^{M} \omega_{ka} \left| B(J\omega_k)\, G(J\omega_k) - A(J\omega_k) \right|^2 \qquad ..(2.32)$$

With $\omega_{ka} = \omega_k \left| R(J\omega_k) \right|^2$ for E_{da} and $\omega_{ka} = \omega_k \left| R(J\omega_k) \right|^2 / \omega_k^{2n}$ for E_{ia}.. Therefore a convenient way of forming the over-determined eqn. set(2.31) for Householder solution is to supply a total of 2M equations, the Odd rows of which are formed by the Coefficients of ω_{ka} $Re[B(J\omega_k)G(J\omega_k) - A(J\omega_k)$ and even rows of which are formed by the coefficients of ω_{ka} $Im\,[B(J\omega_k)\, G(J\omega_k) - A(J\omega_k)]$ [33]

2.3.5 Non-linear least-square Solution Techniques

If $X = [X_1, X_2, \ldots X_v)^T$ is a v-dimentional vector of optimizable parameters, then the most general problem of non-linear least-squares problems can be given by [33]

$$\min_X \sum_{i=1}^{u} \left| f_i(X) \right|^2 \qquad ..(2.33)$$

and the signal error criterion E_{sa} is of this structuresince the modulus squared term can be decomposed into the sum of real and imaginary part squared.

Eqn.(2.33) can be minimized by any method of non-linear optimization technique.

Example 2.4 : This is taken from [33] and is given by

$$G(S) = \frac{5 + 15S}{1+8.1s+7.8s^2+.7s^3}$$

Reduced order model based on integral-equation error criterion E_{ia} with r(t) = 1 is given by

$$G(S) = \frac{5 + 15.47s}{1+8.193+8.265s^2}$$

reduced order model by method 1 i.e. second Cauer form is given by

$$G_2(S) = \frac{.604 + 1.869s}{.121 + .99s + s^2}$$

The step responses and Nyquist plots for above two cases are shown in Figs.2.16 and 2.17.

Example 2.5 : This is also taken from [33] and is given by

$$G(S) = \frac{1}{6 + 11s + 6s^2+s^3}$$

Reduced order model based on integral-equation error criterion E_{ia} with r(t) = 1 is given by

$$G(S) = \frac{.1667}{1+1.833s+s^2}$$

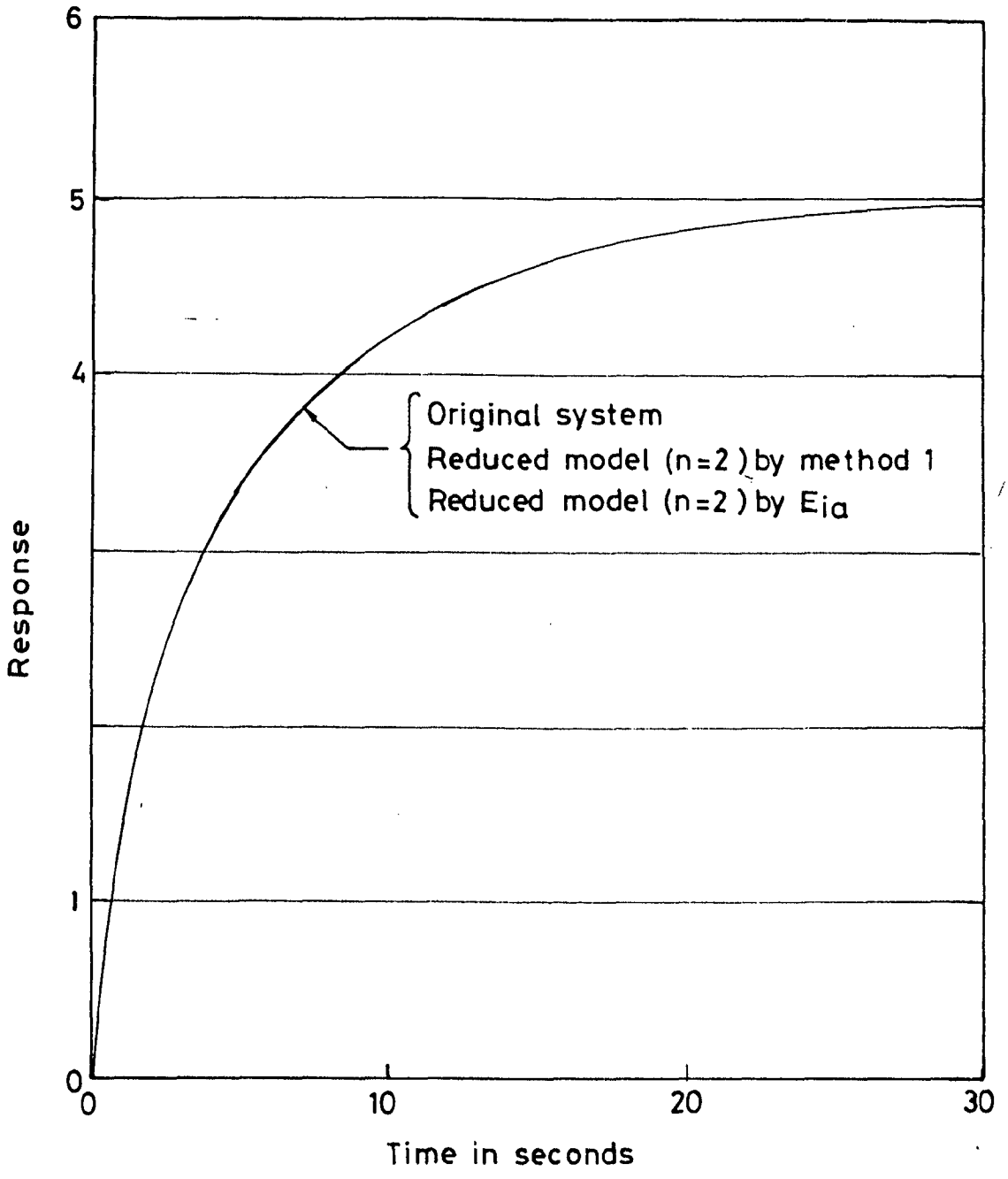


Fig. 2.16 Step responses for example 2.4

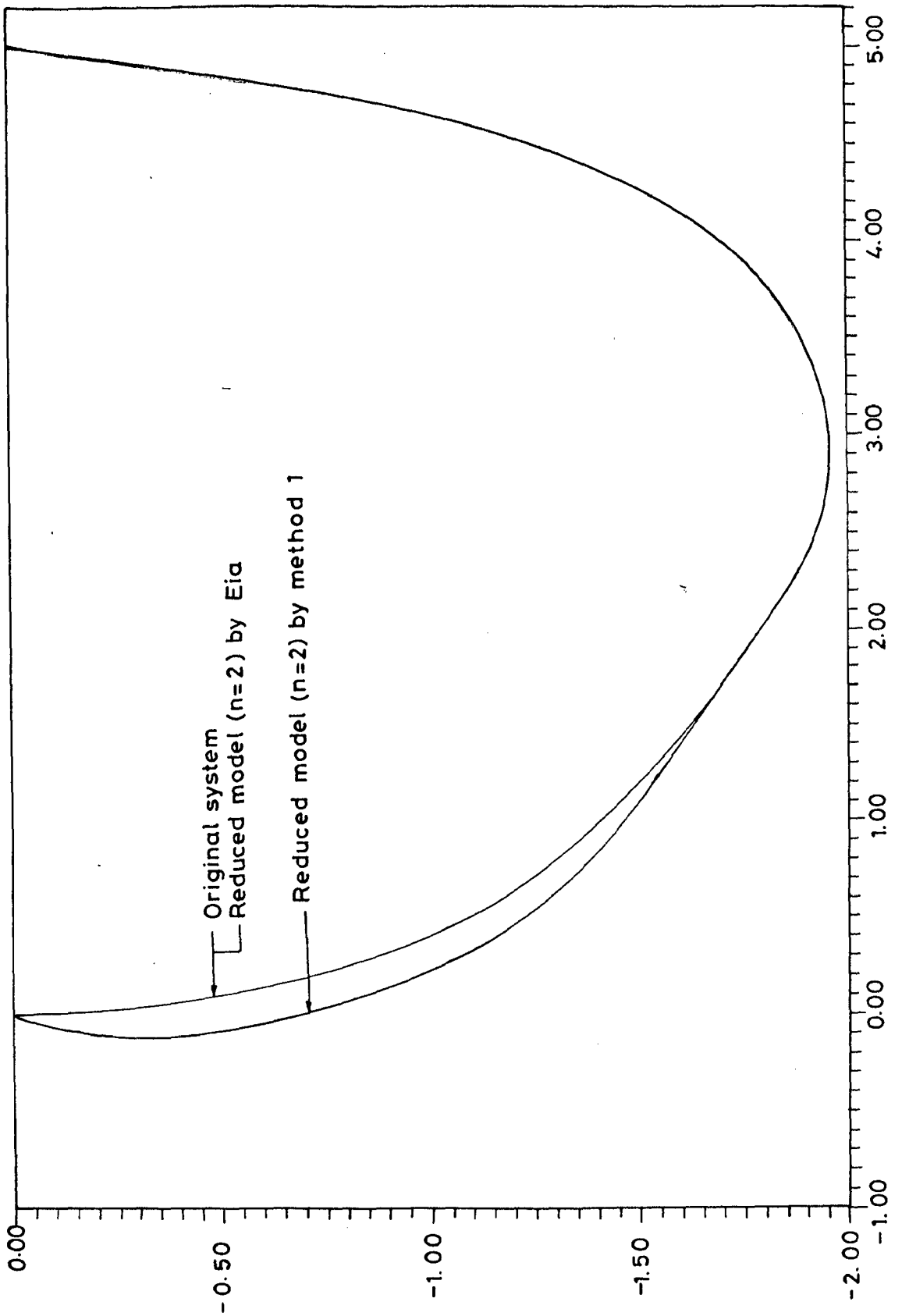


Fig. 2.17 Nyquist plots for example 2.4

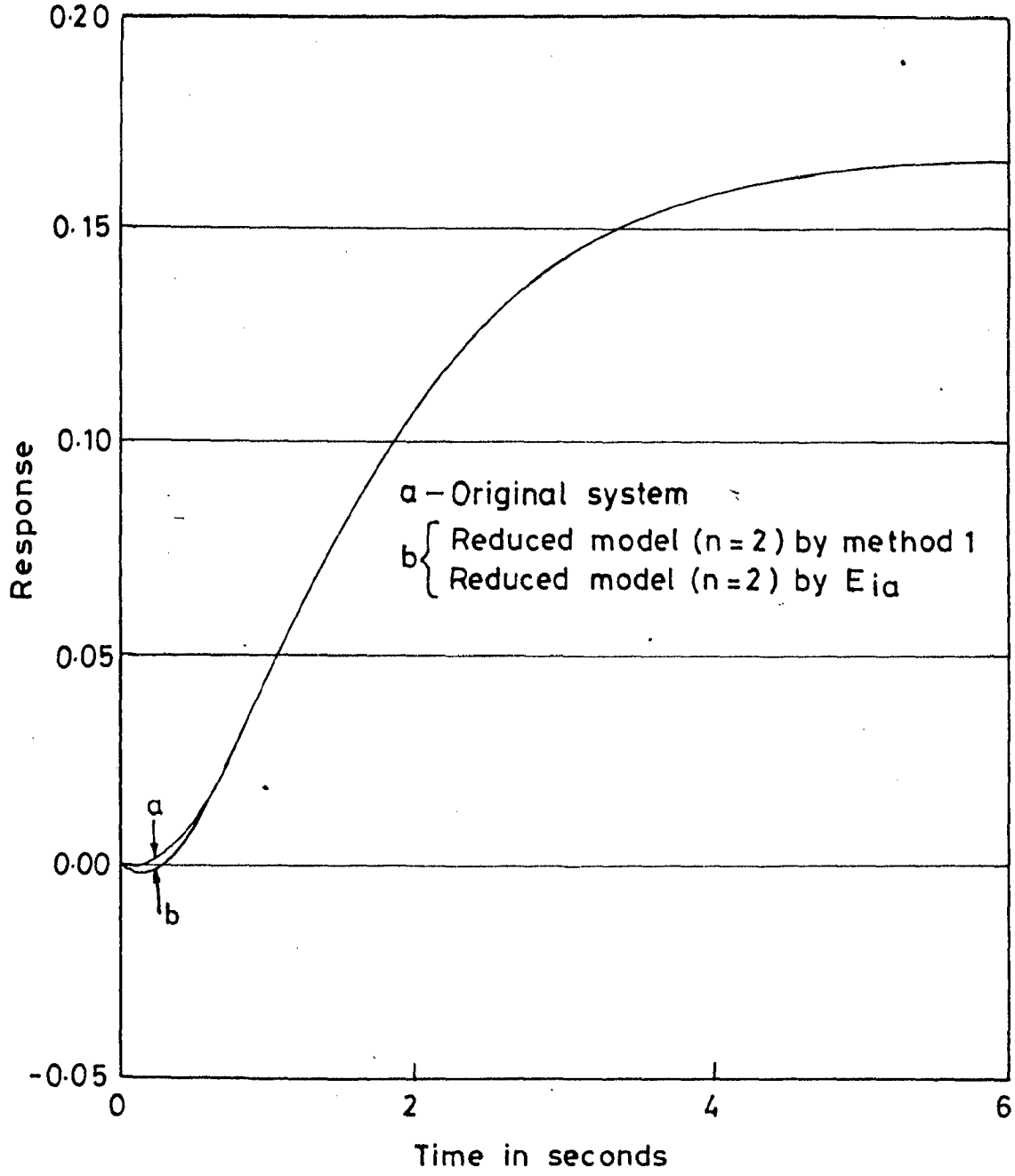


Fig. 2.18 Step responses for example 2.5

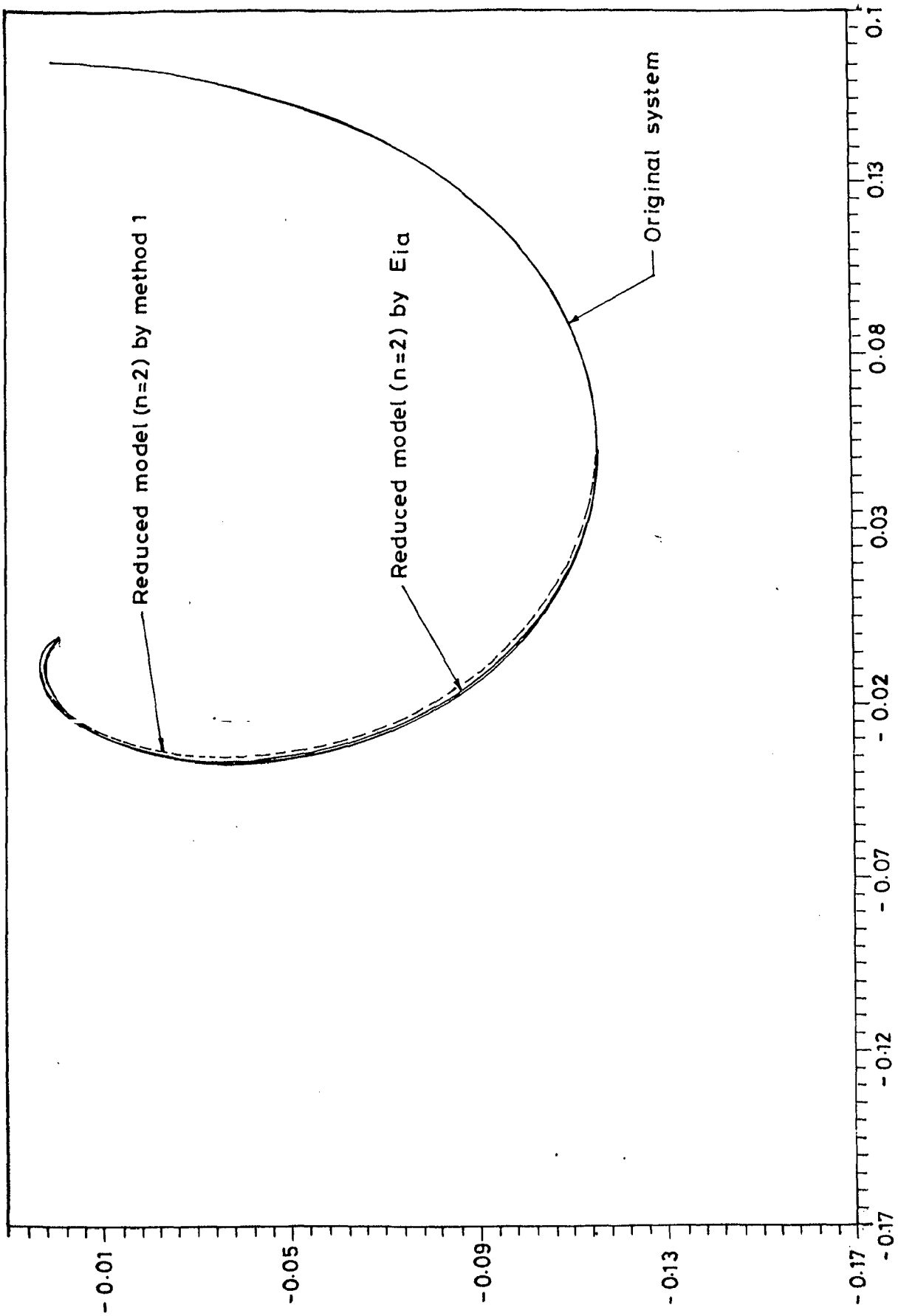


Fig. 2.19 Nyquist plots for example 2.5

reduced order model by method 1 i.e. second Cauer form is given by

$$G_2(S) = \frac{.24 - .04s}{1.44 + 2.4s + s^2}$$

The step responses and Nyquist plot for above two cases are shown in Figs. 2.18 and 2.19.

2.4 MODIFIED CONTINUED FRACTION EXPANSION AND INVERSION

Let the n^{th} order transfer function G(S) and its r^{th} order reduced equivalent R(S) by CFE method be represented as

$$G(S) = \frac{A_n S^{n-1} + A_{n-1}S^{n-2} + -- + A_2S + A_1}{B_{n+1} S^n + B_nS^{n-1} + --- + B_2S + B_1} \quad ..(3.34)$$

and

$$R(S) = \frac{C_r S^{r-1} + C_{r-1} S^{r-2} + -- + C_2S + C_1}{D_{r+1} S^r + D_rS^{r-1} + -- + D_2S + D_1} \quad ..(3.35)$$

Steady state values for original n^{th} order system and reduced model are give by

$$\text{Steady state value of original system} = \lim_{s\to 0} G(S)$$

$$= \frac{A_1}{B_1}$$

$$\text{Steady state value of reduced model} = \lim_{s\to 0} R(S)$$

$$= C_1/D_1$$

We can get the same steady-state value by reduced model only if

$$\frac{A_1}{B_1} = \frac{C_1}{D_1}$$

$$C_1 = D_1 \times \frac{A_1}{B_1} \qquad ..(2.36)$$

Hence reduce order model is given by

$$R(S) = \frac{C_r S^{r-1} + C_{r-1}S^{r-2} + -- + C_2 S + D_1 \frac{A_1}{B_1}}{D_{r+1}S^r + D_r S^{r-1} + -- + D_2 S + D_1} \qquad ..(2.37)$$

The steps are as follows

(1) Get reduced order model by either Cauer first, Cauer second or mixed Cauer method.

(2) If the sign of B_1 and D_1 are opposite, change the sign of D_1 as the sign of B_1.

(3) Get the value of C_1 by equation (2.36)

(4) Put this calculated value C_1 in reduced model which is calculated by CFE method.

The method is illustrated below with two numerical example.

Example 2.6 : Let a system with a transfer function G(S) is given by [17]

$$G(S) = \frac{s^3 + 7s^2 + 24s + 24}{s^4 + 10s^3 + 35s^2 + 50s + 24}$$

reduced order model (n=2; by method-3 is given by

$$G_2(S) = \frac{.357s + .857}{.357s^2 + 1.929s - 1}$$

and reduced order model (n=2) by modified CFE is given by

$$G'_2(S) = \frac{.357s + 1}{.357s^2 + 1.929s + 1}$$

Step responses for both cases are shown in Fig. 2.20.

Example 2.7 : 2nd order model of example 2.3 by method 3 is given by

$$G_2(S) = \frac{23.15 + 2.84s}{1 + .82s + .08s^2}$$

and reduced order model (n=2) by modified CFE is given by

$$G'_2(S) = \frac{20.258333 + 2.84s}{1 + .82s + .08s^2}$$

Step responses for both cases are shown in Fig.2.21

We can get the same steady-state value of both original and reduced model at the cost of transient response and transient response is acceptable but it is not guaranted that at the cost of acceptable transient response we can get the same steady-state value.

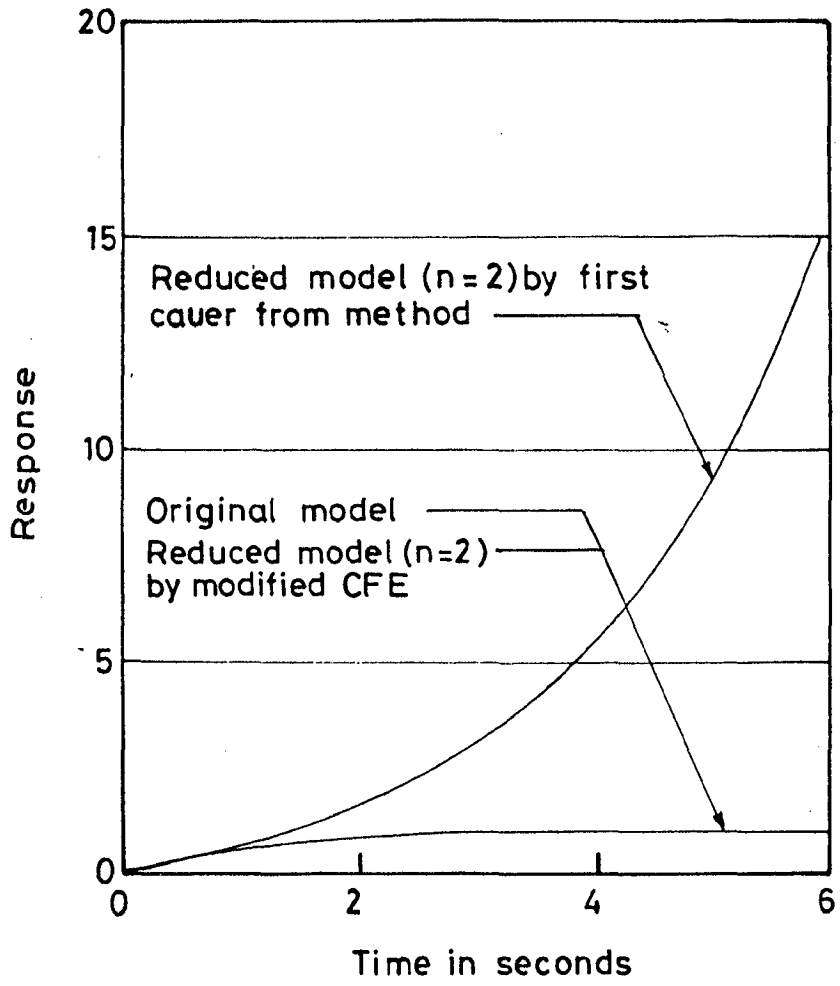


Fig. 2.20 Step responses for example 2.6

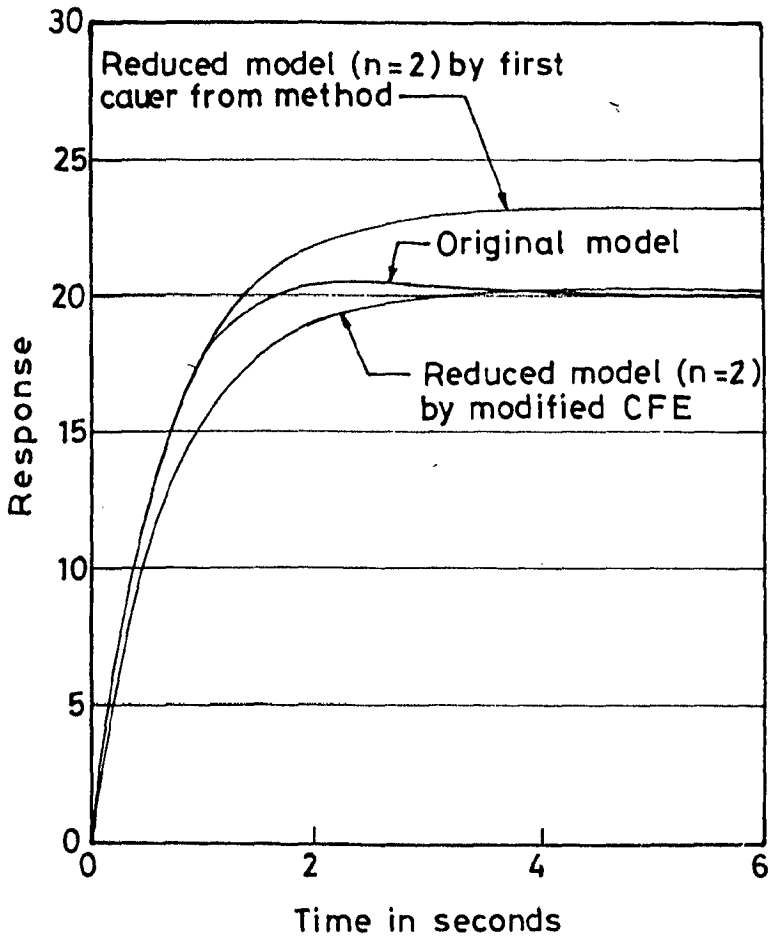


Fig. 2.21 Step responses for example 2.7

chapter :3

method for step response

3.1 INTRODUCTION

For the unit-step input[$R(S) = \frac{1}{s}$], the output response is given by

$$C(s) = \frac{G(s)}{S}$$

i.e.,step response of the system is nothing but the inverse Laplace transform of G(s)/S. Dubner and Abate (23) used fourier series for the numerical inversion of laplace transformation. Durbin (24) improved the same method. Further authors, Simon and Crump [25] used different acceleration methods in order to speed up the convergence of the fourier series. The biggest disadvantage of the above mentioned methods is the dependence on the discretization, truncation error on the free parameter. At the same time method is a bit complex to implement through software. The Laplace transform of a real function f:R → R with f(t) = 0 for t < 0 and its inversion formulae

$$F(s) = L[f(t)] = \int_0^{\infty} e^{-st} f(t)dt \qquad \ldots (3.1)$$

$$f(t) = L^{-1}[f(s)] = \frac{1}{2.\pi.i} \int_{V-i\infty}^{V+i\infty} e^{st} F(s)ds \qquad \ldots (3.2)$$

With $S = V+i$;, ϵ R

$V \in R$ is arbitrary, but greater than the real parts of all the singularities of F(s). The integrals in (3.1) and (3.2) exist for $Re(s) > a \in R$ if.

(a) f is locally integrable

(b) there exist a $t_0 \geq 0$ and K, $a \in R$ such that

$$|f(t)| \leq K e^{at} \text{ for all } t \geq t_o$$

(c) for all $t \in (0,\infty)$ there is a neighbourhood in which f is of bounded variation [26].

$$f(t) = \frac{e^{vt}}{2\pi} \int_{-\infty}^{\infty} [Re(F(V+J\omega)) \cos\omega t - I_m(F(V+J\omega))\sin\omega t]d\omega \quad \ldots (3.3)$$

and on eq.(3.3) fourier transforms are applied in the existing methods.

3.2 Method Implemented

The method implemented for Laplace transform is quite different from the conventional computer methods [23], [24], [25]. The method used in this dissertation is basically based on the conventional way of inverse laplace transform which has been studied in the class room. Only numerical method techniques are applied to it.

The method involves the classical way of finding partial fraction and then applying residue theorem (27). Let us assume that a given rational function F(s) be written

in the form [27].

$$F(s) = \frac{A(S)}{B(S)}$$

$$= \frac{a_o + a_1 s + a_2 s^2 + \ldots + a_m s^m}{b_o + b_1 s + b_2 s^2 + \ldots + b_n s^n} \qquad ..(3.4)$$

Where s is a complex frequency variable, coefficient a_i and b_i are real quantities, and the degree of the numerator polynomial A(S) is less than degree of the denominator polynomial B(S) (i.e. $m < n$). The poles of the function F(s) [the roots of the polynomial B(S)] can be found using Newtons-Horners algorithms. The roots obtained either may be simple root or complex roots. But first considering that we have a simple root located at the value of complex frequency variable P_i then

$$\lim_{S \to P_i} F(S) = \frac{K_i}{(S-P_i)}$$

The coefficient K_i in above equation is reffered to as the residue of the simple pole at P_i. If P_i is real, K_i will be real. In addition, since poles that are complex will always occur in conjugate pairs (this assumes that the b_i are real) it may be shown that the residues of such conjugate pair will also conjugate [26]. The value of the residue K_i may readily be determined directly from the function F(S).

$$K_i = \frac{A(P_i)}{B'(P_i)}$$

Where P_i is a simple pole of F(S), $A(P_i)$ is the numerator polynomial of F(S) evaluated at $S = P_i$ and $B'(P_i)$ is the derivative of the denominator polynomial of F(S) (taken with respect to S), evaluated at $S = P_i$.

If we have multiple roots located at $(S-P_i)^m$, i.e., G(S) is given by $\frac{N(S)}{(S-P_i)^m K(S)}$ where K(S) does not contain $(S-P_i)$ as a factor, i.e., $K(P_i) \neq 0$. Putting $(S-P_i)=y$, we obtain

$$G(S) = \frac{N(y+p_i)}{y^m k(y+p_i)} \qquad ..(3.5)$$

Arrange the terms in $N(y+p_i)$ and $K(y+p_i)$ in ascending power of y and devide the numerator power by the denominator continuing the process until y^m is obtained as a factor of remainder, then we have

$$G(S) = \frac{C_o}{y^m} + \frac{C_1}{y^{m-1}} + \frac{C_2}{y^{m-2}} + \frac{C_{m-1}}{y} + \frac{N(y)}{k(y+p_i)} \qquad ..(3.6)$$

$$G(S) = \frac{C_o}{(S-p_i)^m} + \frac{C_1}{(S-p_i)^{m-1}} + \frac{C_2}{(S-p_i)^{m-2}} \quad \frac{C_{m-1}}{(S-p_i)} + \frac{N(S-p_i)}{K(S)} \qquad ..(3.7)$$

The above relationship can easily be programmed for micro computer. In this way partial fraction expansion could be implemented on micro computer. After getting the partial fraction output, the next aim is to get its inverse Laplace so as to get result in time domain. For this we proceed as follows. Let us assume that such a

function has J simple real pole p_i and h pairs of complex conjugate pole located at p_iC and p_1C, K multiple poles located at P_i.

Where P_iC is the complex conjugate of P_1C. The function may be expanded in the form

$$F(S) = \sum_{i=1}^{J} \frac{K_i}{S-P_i} + \sum_{i=1}^{h} \frac{k_iC}{S-p_iC} + \sum_{i=1}^{h} \frac{k_iC}{S-P_1^C} + \sum_{i=1}^{k} \frac{K_i}{(S-P_i)^n} \quad ..(3.8)$$

Where K_i and K_i^C are residues of which K_i^C has a form $K_i^C = a_i + Jb_i$, where a_i is the real part of the residue and b_i is the imaginary part. If we write $P_i^C = P_i^r \pm J\, p_i^i$ where p_i^r is the real part of the location of p_i^C and p_i^i is the imaginary part then it may be shown that each pair of complex conjugate poles and the residues associated with them will have an inverse Laplace transform of the form

$$2\, e^{p_i^r t} * [a_i \cos(p_i^r t) - b_i \operatorname{Sin}(p_i^i t)] \quad ..(3.9)$$

Thus the complete inverse transform for a function of the type is given by

$$f(t) = \sum_{i=1}^{J} k_i\, e^{p_i t} + 2 \sum_{i=1}^{h} e^{p_i^r t} [a_i \cos(p_i^i t) - b_i \operatorname{Sin}(p_i^i t)] + \sum_{i=1}^{k} \frac{k_i * t^{n-1}}{n-1\,!}\, e^{p_i t} \quad ..(3.10)$$

The complete flow chart of this method is given in App. B.

chapter :4

application of reduction methods for controller design

4.1 INTRODUCTION

The classical techniques of control system design using logarithmic frequency response plots of Bode and Nichols, root locus diagrams of Evans or Nyquist plots are well documented in the literature. These methods are graphical in natural and are normally limited to single input single output systems. With the advent of state space theory, the optimal control approach has been developed to tackle both deterministic and stochastic signals. This requires the solution of high order nonlinear differential equations. With the availability of new computing powers of modern fast digital computers alongwith graphic display facilities, controller design has entered a new era.

The problem of model matching may be stated as -" We have a process (reduced model) whose performance is unsatisfactory and a process (original model) having the desired performance, we have to derive a controller such that the performance of the augmented process matches with that of original model".

In the design of a control system in the frequency domain, the specifications that are usually considered as design goals may be classified as

1. The time domain specifications, e.g. rise time, overshoot etc.

2. The frequency domain specifications, e.g. bandwidth and the phase margin etc.

3. The complex domain specifications, e.g. the damping ratio and the undamped natural angular frequency etc.

To improve the efficiency of any design method, it is advantageous to have the design goals expressed as mathematical functions or transfer function (defined as the standard model).

4.2 THE DESIGN METHOD [28]

The design entails the following steps :

(1) Construction of a specification model that the closedloop system must approximate.

(2) Specification of the structure of the controller.

(3) Determination of the closed-loop transfer function consisting of unknown controller parameters.

(4) Order reduction of the transfer function of step 3 to approximate to that of the model in step 1.

(5) Step 4 yields a set of non-linear algebraic equations that are sequentially solved for the controller parameters.

The model transfer function may be specified as

$$G_M(S) = \frac{g_o + g_1 S + g_2 S^2 + \text{---} + g_u S^u}{h_o + h_1 S + h_2 S^2 + \text{--} + h_V S^V} \qquad ..(4.1)$$

Where $V \geq u$ and in general, $g_o = h_o$

Let the structure of the precompensator be specified as

$$G_c(S) = \frac{K_{00} + K_{01} S + \text{- -} + K_{0i} S^i}{K_{10} + K_{11} S + \text{- -} + K_{1J} S^J} \qquad ..(4.2)$$

With the plant transfer function as

$$G_p(S) = \frac{\alpha_o + \alpha_1 S + \text{- -} + \alpha_m S^m}{\beta_o + \beta_1 S + \text{- -} + \beta_n S^n} \qquad ..(4.3)$$

Where $n \geq m$ and, in general, $\alpha_o = \beta_o$

We have the closed-loop transfer function as

$$G_{C.L.}(S) = \frac{G_c(S)\ G_p(S)}{1 + G_c(S)\ G_p(S)} \qquad ..(4.4)$$

Substitution from equation (4.2) and (4.3) in (4.4) yields an overall transfer function of the form

$$G_{C.L.}(S) = \frac{a_o + a_1 S + \text{- -} + A_q S^q}{b_o + b_1 S + \text{- -} + b_r S^r} \qquad ..(4.5)$$

$$= C_0 + C_1 S + C_2 S^2 + \text{---} \qquad ..(4.6)$$

Where $q = (m+i)$ and $r = (n+J)$

$C_0 = 1$ When $a_0 = b_0$

The coefficients a_o, a_1,...,a_q; b_o, b_1,...,b_r; and C_o, C_1, C_2,... etc. will, in general, contain the controller parametrs K_{00}, K_{01},...,K_{0i}; K_{10}, K_{11},..,K_{1J} and the known constant coefficients α_1, α_2,.. α_m and β_1, β_2,... β_n. Then for $G_M(S)$ to be a Pade' approximant of $G_{C.L.}(S)$, we have :

$$
\begin{aligned}
g_0 &= h_0C_0 \\
g_1 &= h_0C_1 + h_1C_0 \\
g_2 &= h_0C_2 + h_1C_1 + h_2C_0 \\
&\vdots \\
g_u &= h_0C_u + h_1C_{u-1} + \ldots + h_uC_0 \qquad \ldots(4.7) \\
&\vdots \\
0 &= h_o\,C_{u+v} + h_1C_{u+v-1} + \ldots + h_vC_v
\end{aligned}
$$

(i+J+3) equations of the above type can be sequentially solved for (i+J+2) unknown controller parameter of eqn.(4.2). The first equation, i.e. $g_o = h_0C_0$ will be, in general, redundant when $a_0 = b_0$. It should be pointed that the particular triangular form of the non-linear algebraic equation in (4.7) makes their solution possible by simple hand calculations. The design method is illustrated by an example below.

Example 4.1 : Consider the high order plant transfer function from Shamash (29)

$$G_p^4(S) = \frac{S^3 + 12S^2 + 54S + 72}{S^4 + 18S^3 + 97S^2 + 180S + 100}$$

Applying mixed Cauer form method of Chapter-2, we have the following second order reduced model,

$$G_p^2(S) = \frac{2.182\ S + .217}{3.03S^2 + 3.483S + .217}$$

The model transfer function is chosen as [30]

$$G_M(S) = \frac{1 + \alpha(\frac{2\xi}{\omega_n})S}{1+(\frac{2\xi}{\omega_n})S + \frac{S^2}{\omega_n^2}} \qquad ..(4.8)$$

Where ξ is the damping ratio, ω_n is the undamped natural frequency and α is a design variable which has special significance in so far as $\alpha = 0$ in eqn. (4.8) will result in a zero displacement error system, while $\alpha = 1$ will result in a zero velocity error system.

Choosing $\omega_n = 5.0$, $\xi = 0.707$ and $\alpha = .7$; we have

$$G_M(S) = \frac{25.0 + 4.242S}{25.0 + 7.07S + S^2}$$

A closed loop system using a proportional integral type of precompensator, $K_c(1 + \frac{1}{\tau_1 S})$, and unity feadback is designed on the basis of $G_p^2(S)$.

$$G_c(S) = K_c\left(1 + \frac{1}{\tau_1 S}\right)$$

The closed loop transfer function becomes

$$G_{C.L.}(S) = \frac{2.182\ \tau_1 S^2 + (2.182 + .217\ \tau_1)S + .217}{3.03KS^3 + (3.48K + 2.182\ \tau_1)S^2 + (.217K + 2.17\ \tau_1 + 2.182)S + .217}$$

$$= C_0 + C_1 S + C_2 S^2 + C_3 S^3 + - -$$

Where $\tau_1/K_c = K$ and

$$C_0 = 1$$

$$C_1 = -K$$

$$C_2 = \frac{.1443922 - .024578\ \tau_1}{.217} = .6654018 - .1132626\tau_1$$

From eqn. (4.7) we have

$$25 = 25 \qquad ..(4.9a)$$

$$4.242 = 25C_1 + 7.07 \qquad ..(4.9b)$$

$$0 = 25C_2 + 7.07C_1 + 1.0 \qquad ..(4.9c)$$

$$6 = 25C_3 + 7.07C_2 + C_1 \qquad ..(4.9d)$$

Since there are three unknown controller parameters, four equations in (4.9) are formed where (4.9a) is redundant. Though the equation (4.9) are non-linear, the particular triangular form makes them amenable to hand computations by successive substitutions-as explained below. Eqn.(5.9b) yields K = .11312. On substituting this in eqn. (5.9c) we get τ_1 = 5.8114624. Thus we get the following controller parameters.

$$K_c = 51.374314$$

$$\tau_1 = 5.8114624$$

and the closed loop transfer function becomes

$$G_{C.L.}(S) = \frac{36.96S^2 + 10S + .633}{S^3 + 38.15S^2 + 10.12S + .633}$$

From fig. 4.2, it is seen that the closed loop response matching of the original fourth order system with the above controller is exact for the steady-state region and acceptably good for the transient region.

Thus if a high order system can be well-approximated by its reduced order model, a controller design may be based on such a model. From the above example it is found that such controllers when designed by the method of section 4.2 can effectively control the original high order systems.

For matching the steady-state values of original system and reduced model, PI controller is designed, while for matching the transient part, PD controller is designed. If we want to match both transient as well as steady state region, PID controller is designed. We can design all three type of controller by this method. However, this design method should be applied with caution for unstable plants and quite obviously, there is no getting around difficulties of non-minimum phase plants.

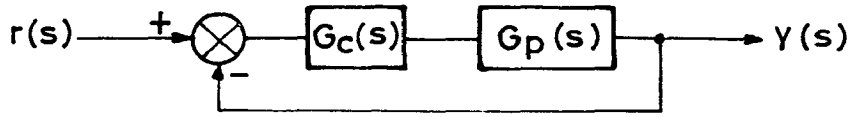


Fig. 4.1 Precompensator arrangement

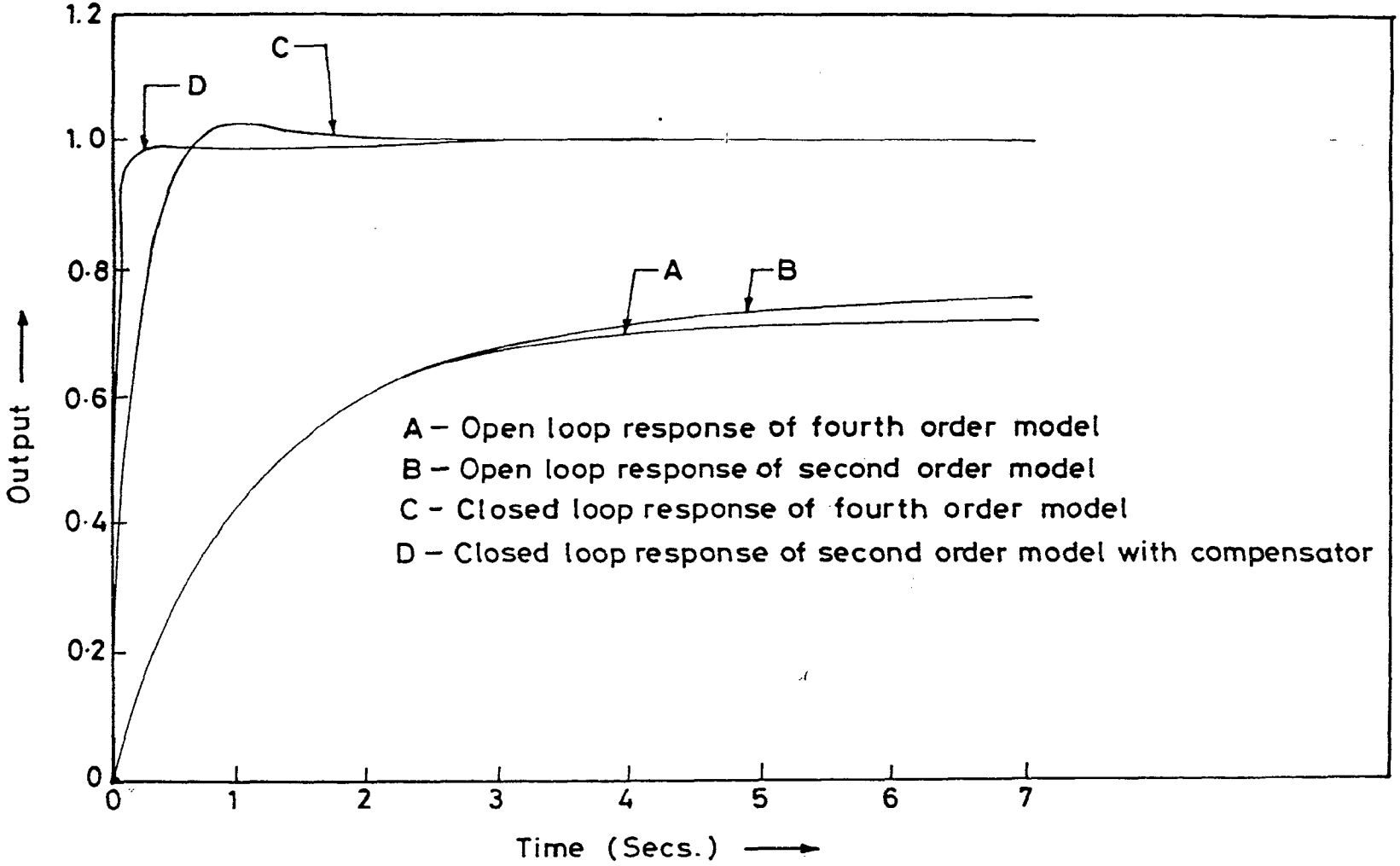


Fig. 4.2 Comparison of step responses

Because, this method is based on approximate model matching and hence, may lead to an unstable overall due to truncation error. This drawback may be overcome by prespecifying some of the pole zero positions in the compensator to exact cancel the effect of right hand side poles and zeros.

chapter :5

conclusion

CONCLUSIONS

The advantages of system order reduction techniques are well known. The main obvious advantages are saving in computational work in the analysis of large scale systems and economy in the design of associated hardware, for optimal and suboptimal controllers.

In this thesis continued fraction expansion technique for reducing the order of large scale systems have tried on typical systems, considered by various researchers. The applicability of continuous time reduction method has been tested for controller design. The softwàre for continued fraction expansion method, step response and Nyquist plot developed in FORTRAN and have been successfully implemented on PC. The main draw back of CFE technique is that, the ROMS may be unstable (stable), even though the original system is stable (unstable).

The first introductory chapter lists the various possible reasons for going in for reduced order models (ROMs) and for the use they have been put to.

In second chapter a detailed proceduce for continued fraction expansion, Routh-Hurwitz array and Integral square methods are presented.

A method has been given for step response of a system in chapter-3.

In chapter-4, the method for controller design has been given. The method is based on Padé approximation and algebraic in nature. The desired performance is converted into a transfer function model which is matched with closed loop system to have identical few time moments. This method does not require any trial and error procedure. However, as this method is based on the principle of approximate model matching. It may lead to poor or unstable control for non-minimum phase or unstable systems.

appendix–A

DETAILS OF THE COMPUTER PROGRAM DEVELOPED

The organisation of the program is as shown in Fig.1. This program consists a main program and 15 subroutines. The purpose of various subroutines used in the program and call statement alongwith their arguments are described below:

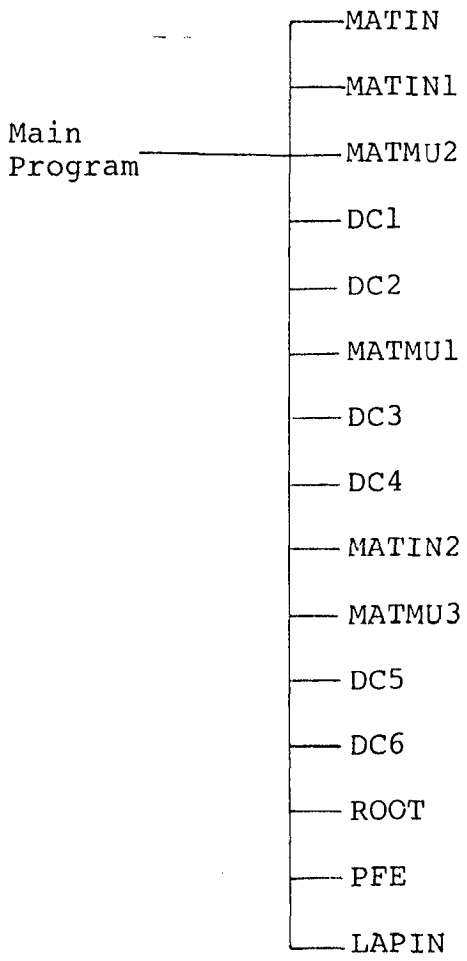


Fig.1 Organisation of the softwave developed

1. Subroutine MATIN : It reads the transfer-function data and calculates the K_i quotients of the mixed Cauer form, H_i quotients of Second Cauerform. It is called by the instruction.

   Call MATIN (A,M,II,B,K1)

2. Subroutine MATIN1 : It reads the transfer-function data and calculates the K'_i quotients of the mixed Cauer form. It is called by the instruction.

   CALL MATIN1 (A,M,II,MM,B,K2)

3. Subroutine MATMU2 : It reads the transfer-function and K_i and K'_i data and calculates the rest elements of Routh array for mixed cauer form. It is called by the instruction.

   CALL MATMU2(A,K1,K2,M,JJ,L,II,D,D1)

4. Subroutine DC1 : It reads K_i and K'_i data and calculates the elements of reduced transfer function by mixed cauer form. It is called by instruction.

   CALL DC1(C, K1, K2, M, II, MM).

5. Subroutine DC2 : It reads K_i and K'_i data and calculates the rest element which are not calculated by subroutine DC1 of reduced transfer function by mixed Cauer form. It is called by instruction.

   CALL DC2(K1,K2,C,J,L,M,K4,K3).

6. Subroutine MATMU1 : It reads transfer function and K_i data and calculates the rest element of Routh array for second Cauer form. It is called by instruction.

7. Subroutine DC3 : It reads K_i data and calculates the elements of reduced trans. function by second cauer form. It is called by instruction.

   CALL DC3(C,K1,M,II,MM).

8. Subroutine DC4 : It reads K_i data and calculates the rest element of the reduced transfer function which are not calculated by subroutine DC3 of second cauer form. It is called by the instruction.

   CALL DC4 (K1,C,J,L,M,K3).

9. Subroutine MATIN2 : It reads T.F. data and calculates H'_i quotients for first Cauer form. It is called by instruction.

   CALL MATIN2 (B1,M,II,B,K2).

10. Subroutine MATMU3 : It reads T.F. data and H'_i data and calculates rest elements of Routh array for first cauer form. It is called by instruction.

    CALL MATMU3(B1,K2,M,JJ,L,II,D1).

11. Subroutine DC5 : It reads H'_i data and calculates the elements of reduced T.F. by first Cauer form. It is called by instruction.

    CALL DC5(C,K2,M,II,MM).

12. Subroutine DC6 : It reads H'_i data and calculates the rest elements of reduced T.F. which are not calculated by subroutine DC5 for first Cauer form. It is called by instruction.

    CALL DC6(K2,C,J,L,M,K4).

13. Subroutine ROOT : This subroutine is used to get the 'roots' of a polynomial. It is called by instruction.

    CALL ROOT (E,N,NCOFS).

14. Subroutine PFE : This subroutine is used for partial fraction expansion. It is called by instruction.

    CALL PFE (NP,NZ,XA,XB).

15. Subroutine LAPIN : This subroutine is used for Laplace inversion. It is called by instruction.

    CALL LAPIN (N,MX,P,Q,F,FIN,STEP,PI,IFIN).

The flow chart of the computer program is as shown in Appendix-B.

appendix–B

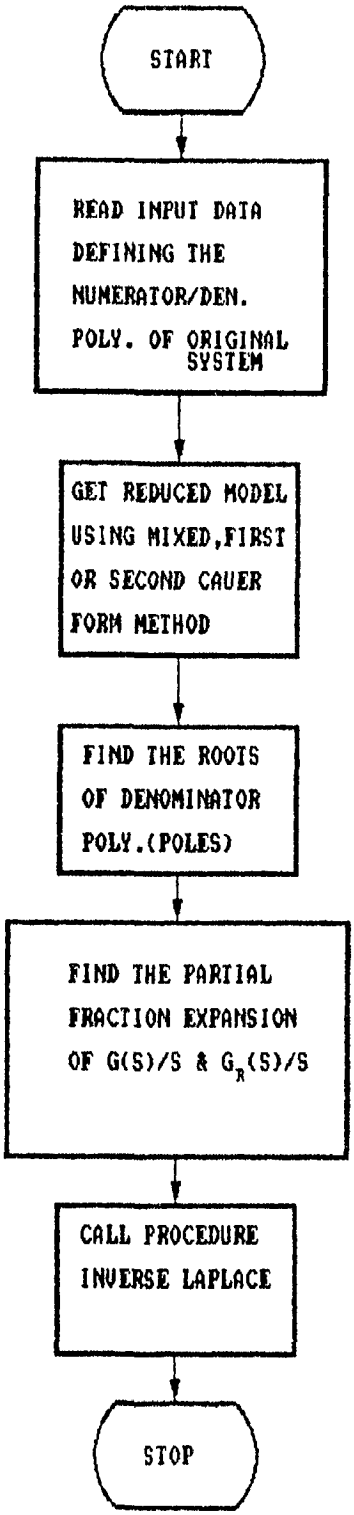
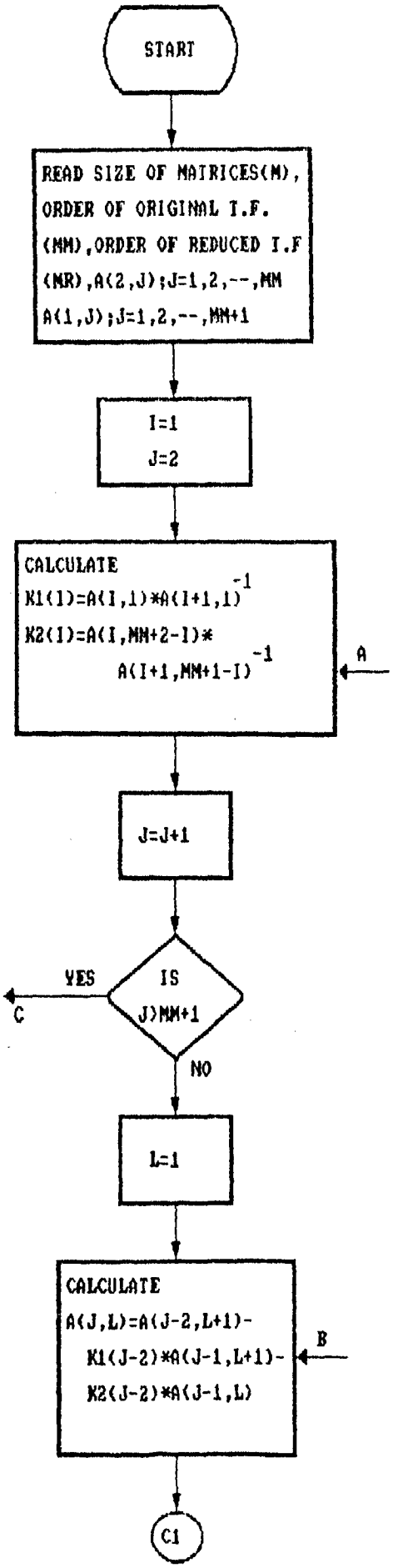
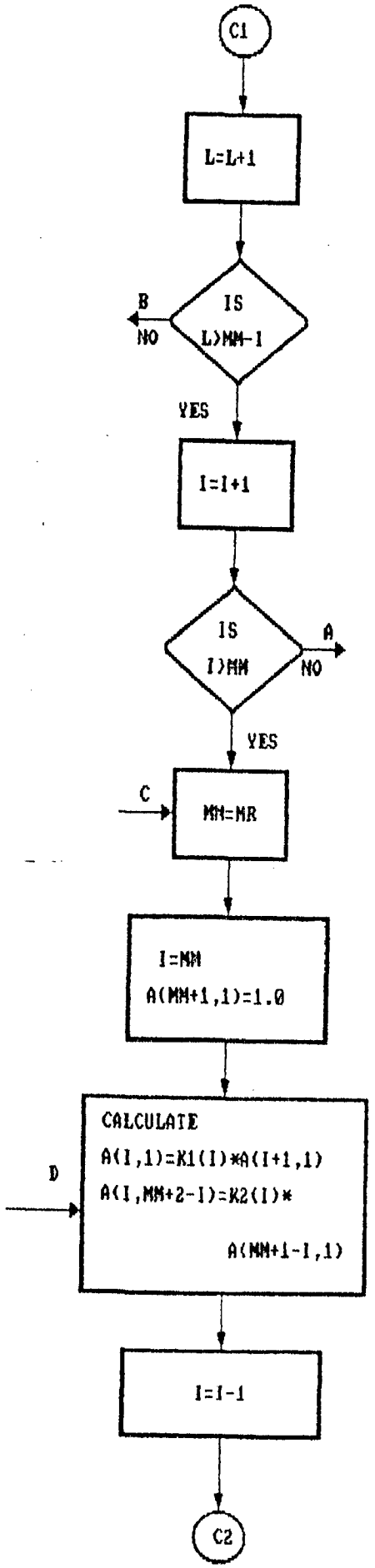


FIG.1 MAIN FLOW CHART


START
READ SIZE OF MATRICES(M), ORDER OF ORIGINAL T.F. (MM),ORDER OF REDUCED T.F (MR),A(2,J);J=1,2,--,MM A(1,J);J=1,2,--,MM+1
I=1
J=2
CALCULATE
K1(I)=A(I,1)*A(I+1,1)^-1
K2(I)=A(I,MM+2-I)*
A(I+1,MM+1-I)^-1
A
J=J+1
IS
J>MM+1
YES
C
NO
L=1
CALCULATE
A(J,L)=A(J-2,L+1)-
K1(J-2)*A(J-1,L+1)-
K2(J-2)*A(J-1,L)
B
C1


C1
L=L+1
IS
L)MM-1
B
NO
YES
I=I+1
IS
I)MM
A
NO
YES
C
MM=MR
I=MM
A(MM+1,1)=1.0
D
CALCULATE
A(I,1)=K1(I)*A(I+1,1)
A(I,MM+2-I)=K2(I)*
A(MM+1-I,1)
I=I-1
C2

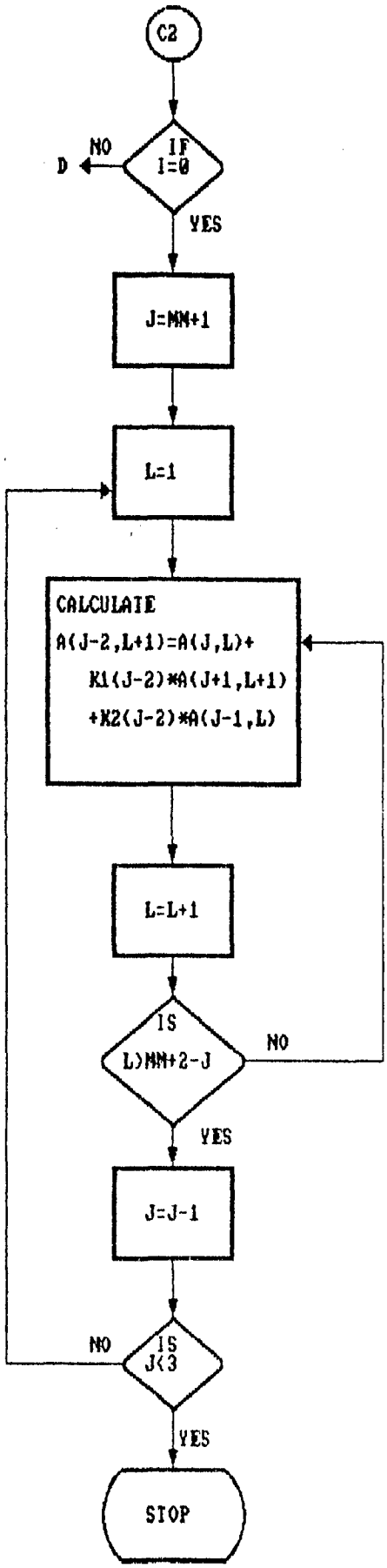
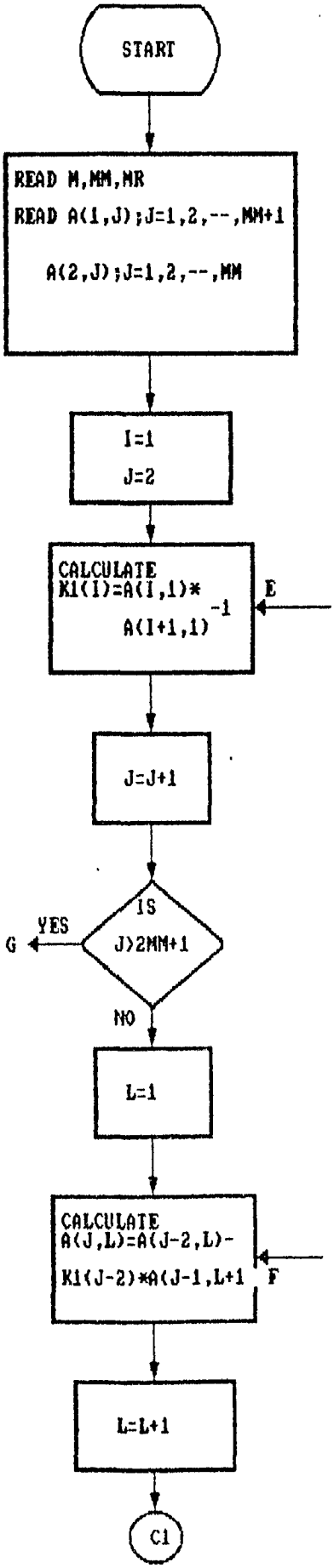
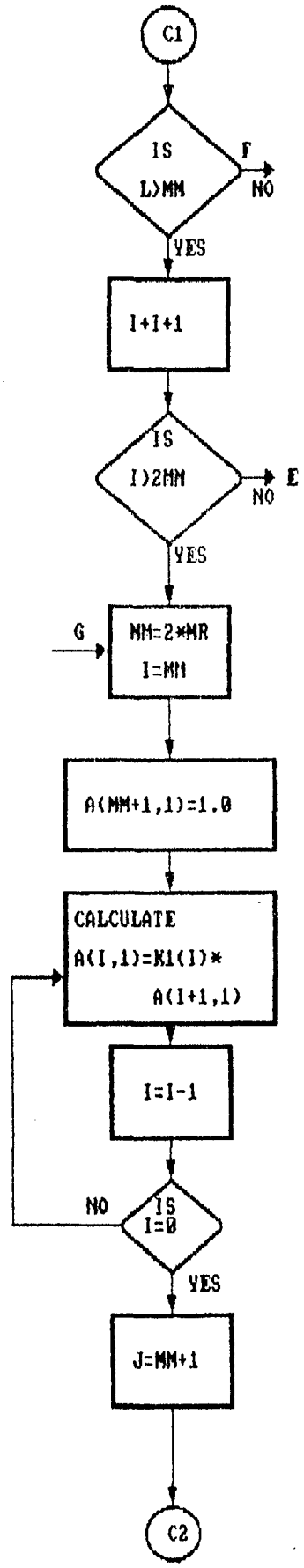


FIG.2 FLOW CHART FOR MIXED CAUER FORM


START
READ M,MM,NR
READ A(1,J);J=1,2,--,MM+1
A(2,J);J=1,2,--,MM
I=1
J=2
CALCULATE
K1(I)=A(I,1)*
A(I+1,1)
-1
E
J=J+1
IS
J>2MM+1
YES
G
NO
L=1
CALCULATE
A(J,L)=A(J-2,L)-
K1(J-2)*A(J-1,L+1
F
L=L+1
C1


C1
IS
L>MM
F
NO
YES
I+I+1
IS
I>2MM
E
NO
YES
G
MM=2*MR
I=MM
A(MM+1,1)=1.0
CALCULATE
A(I,1)=K1(I)*
A(I+1,1)
I=I-1
NO
IS
I=0
YES
J=MM+1
C2

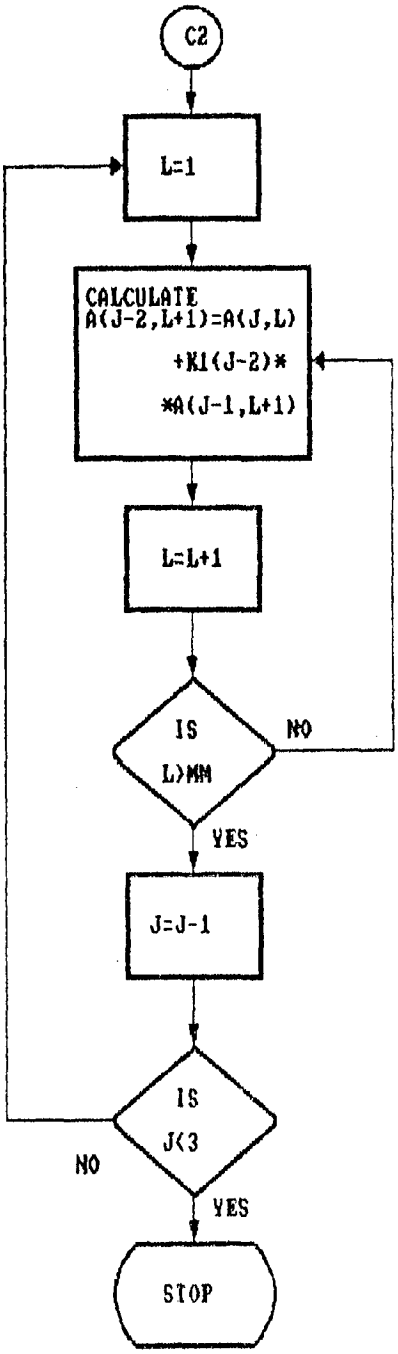
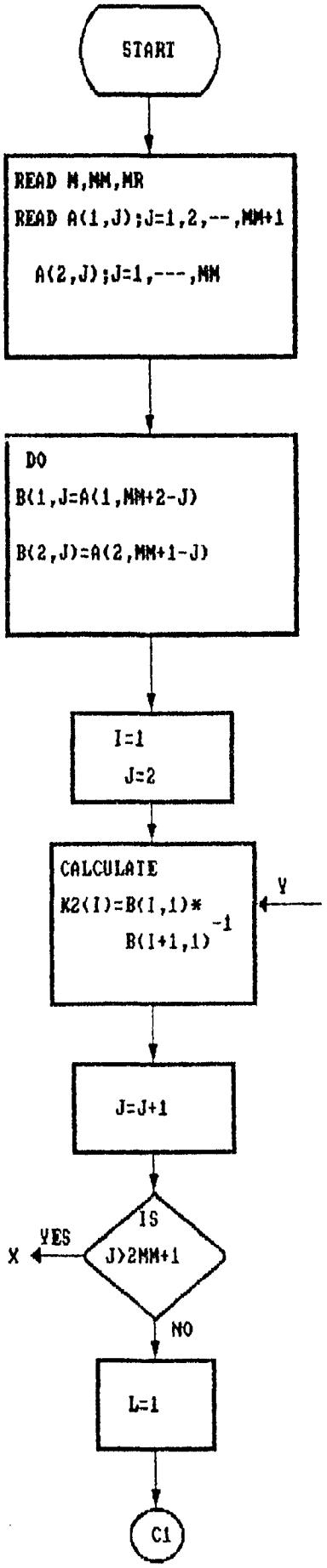
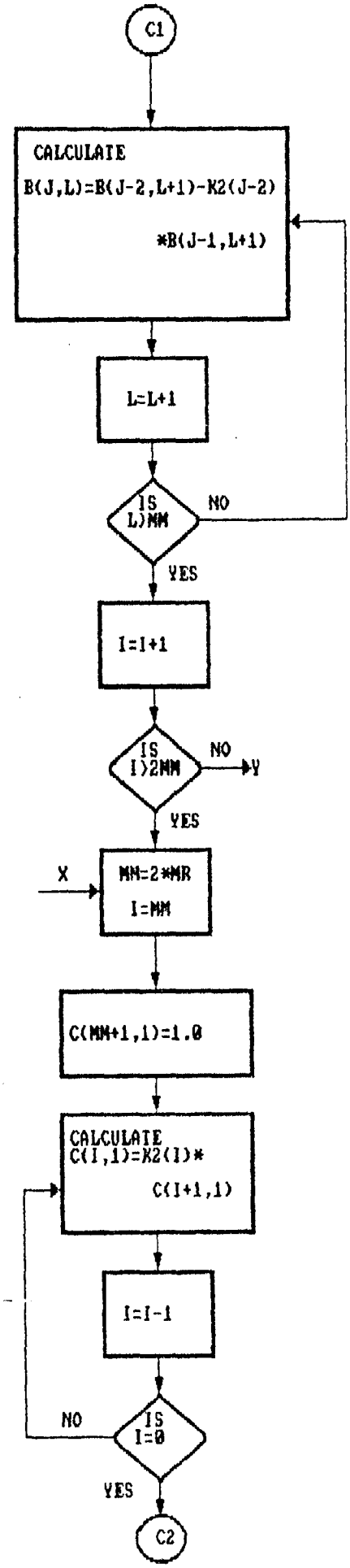


FIG.3 -FLOW CHART FOR 2nd CAUER FORM


START
READ M,MM,MR
READ A(1,J);J=1,2,--,MM+1
A(2,J);J=1,---,MM
DO
B(1,J=A(1,MM+2-J)
B(2,J)=A(2,MM+1-J)
I=1
J=2
CALCULATE
K2(I)=B(I,1)*
B(I+1,1)^-1
V
J=J+1
IS
J>2MM+1
YES
X
NO
L=1
C1


C1
CALCULATE
B(J,L)=B(J-2,L+1)-K2(J-2)
*B(J-1,L+1)
L=L+1
IS
L)MM
NO
YES
I=I+1
IS
I)2MM
NO
Y
YES
X
MM=2*MR
I=MM
C(MM+1,1)=1.0
CALCULATE
C(I,1)=K2(I)*
C(I+1,1)
I=I-1
NO
IS
I=0
YES
C2

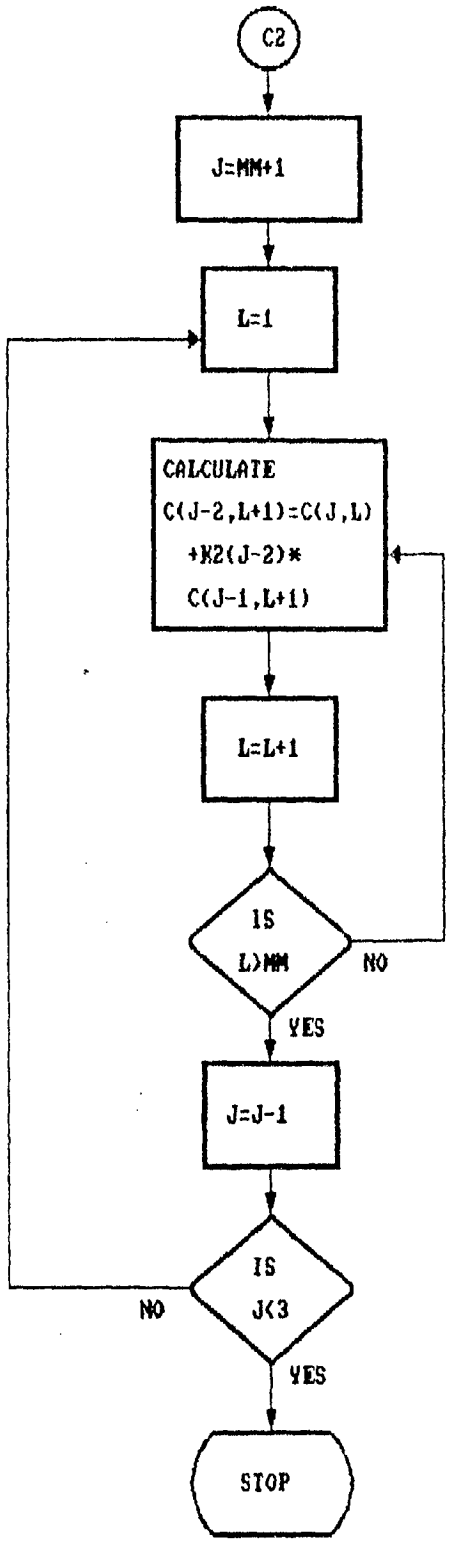


FIG.4 FLOW CHART FOR 1ST CAUER FORM

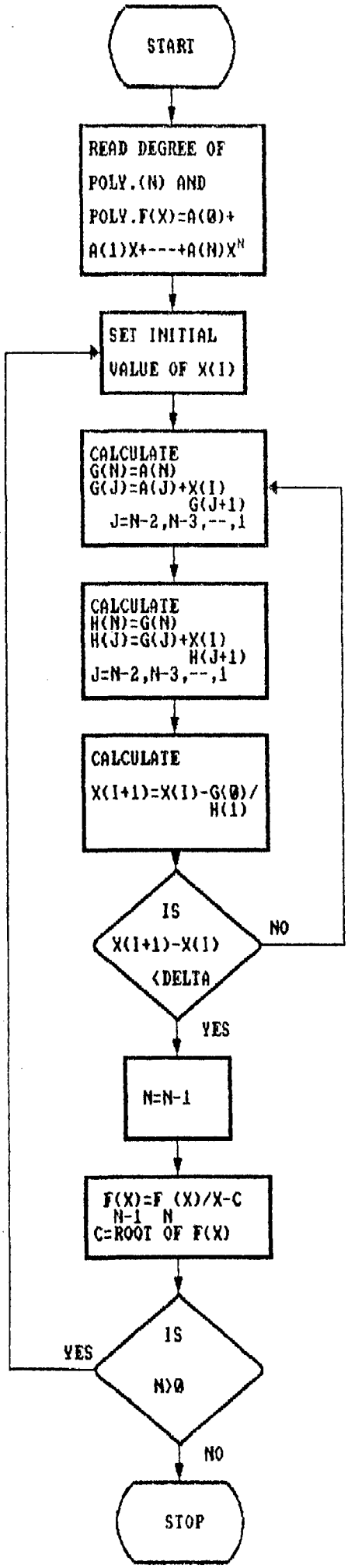


FIG.5 FLOW CHART FOR ROOTS OF A POLY.

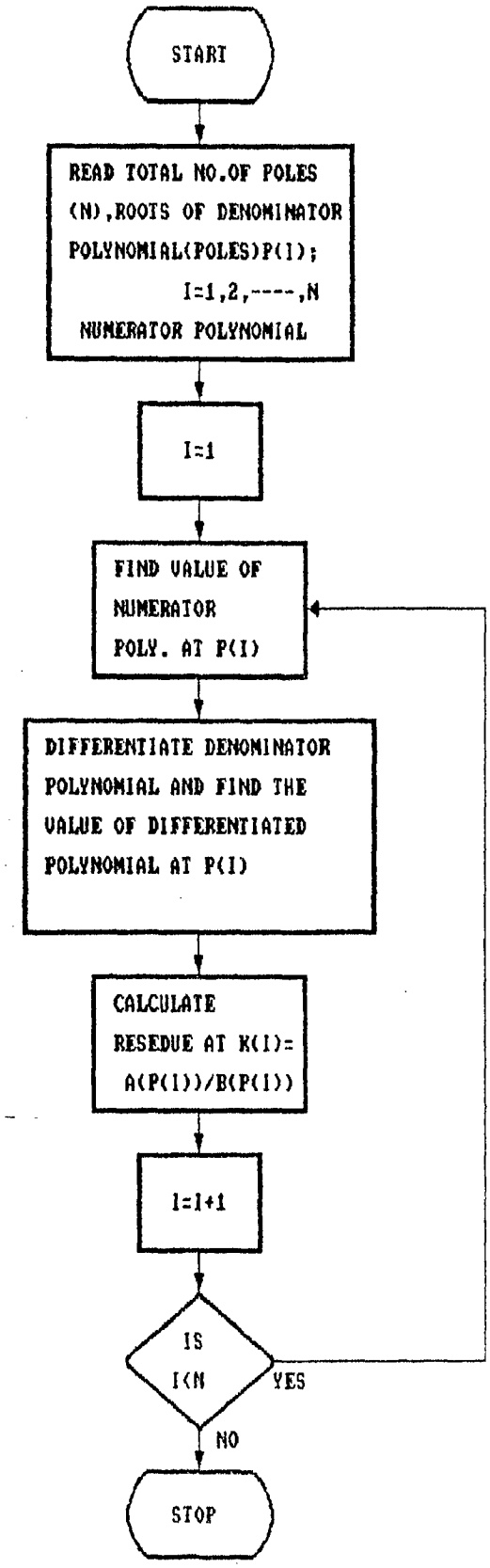
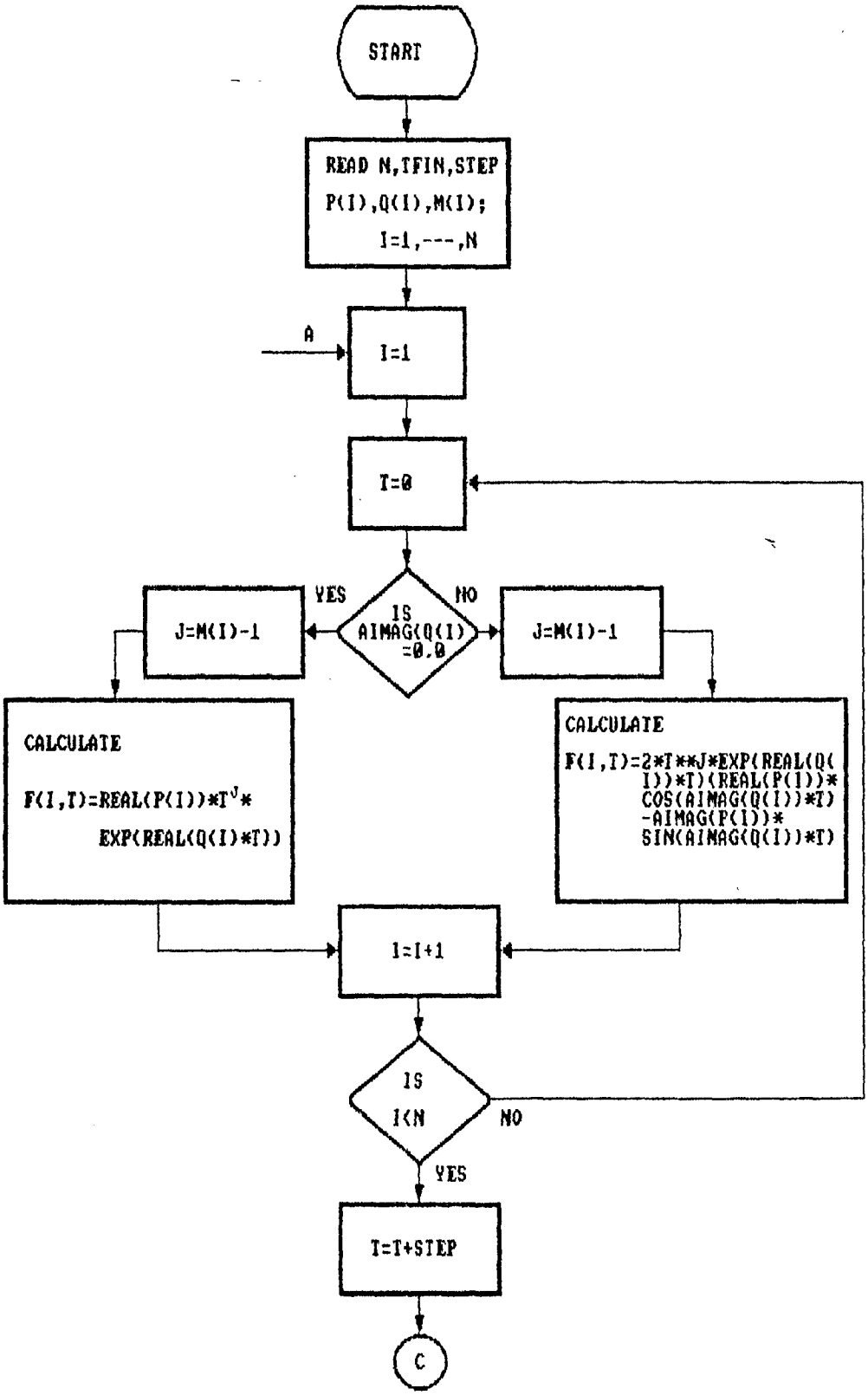


FIG.6 FLOW CHART FOR PARTIAL FRACTION


START
READ N,TFIN,STEP
P(I),Q(I),M(I);
I=1,---,N
A
I=1
T=0
IS
AIMAG(Q(I)
=0.0
YES
NO
J=M(I)-1
J=M(I)-1
CALCULATE
F(I,T)=REAL(P(I))*T^J*
EXP(REAL(Q(I)*T))
CALCULATE
F(I,T)=2*I**J*EXP(REAL(Q(
I))*T)(REAL(P(I))*
COS(AIMAG(Q(I))*T)
-AIMAG(P(I))*
SIN(AIMAG(Q(I))*T)
I=I+1
IS
I<N
NO
YES
T=T+STEP
C

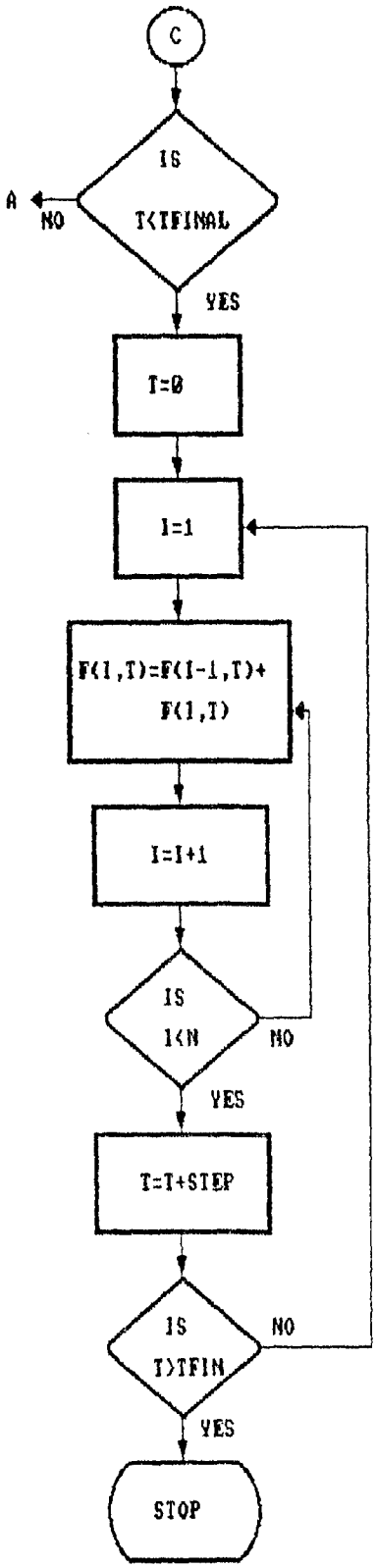


FIG.7 FLOW CHART FOR LAPLACE INVERSE

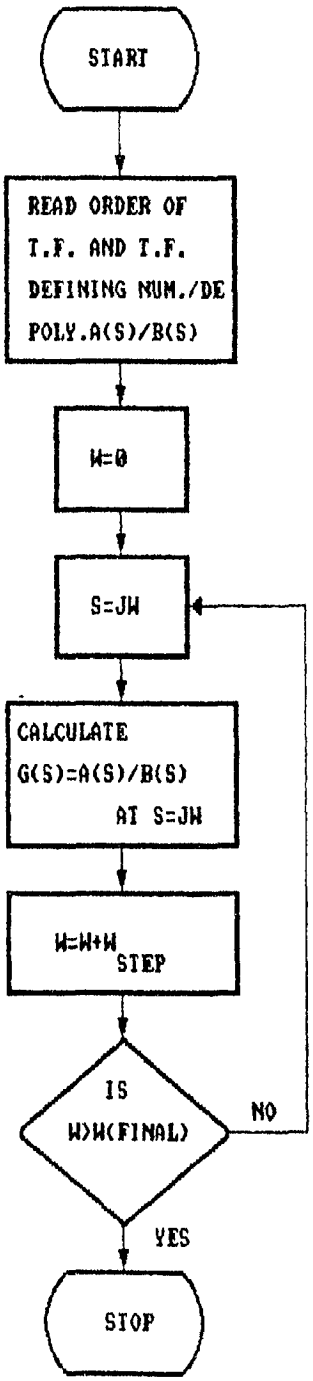


FIG.8 FLOW CHART FOR NYQUEST PLOT

appendix-C

```
**********************************************************************C*********
******

            MODEL ORDER REDUCTION USING FRE. DOMAIN TECH.
                              BY
                      CAUER FORM FOR MIMO SYSTEM
**********************************************************************
      MM=ORDER OF THE ORIGINAL MODEL
      MR=ORDER OF THE MODIFIED MODEL
      M=SIZE OF MATRICES
      K1 and K2 ARE MATRIX QUOTIENTS OF THE MATRIX CAUER FORM
      DIMENSION A(8,8,2,2),B(8,8,2,2),K1(8,8,2,2),D(8,8,2,2),D1(8,8,2,2)
     1  ,K2(8,8,2,2),K3(8,8,2,2),K4(8,8,2,2),C(8,8,2,2),B1(8,8,2,2),E(37)
     1  ,AWORK(37),U(36),V(36),MX(10),F(10,100),P1(3,100),BB(121)
      COMPLEX XA(20),XB(20),XC(20),CC(20),P(10),Q(10)
      INTEGER  XM(20)
ARGE
      REAL A,B,K1,D,D1,K2,K3,K4,C
      OPEN(UNIT=1,FILE='D1.OUT')
      OPEN(UNIT=2,FILE='D2.OUT')
      OPEN(UNIT=3,FILE='D3.OUT')
      OPEN(UNIT=4,FILE='D4.OUT')
      OPEN(UNIT=5,FILE='D5.OUT')
      WRITE(*,34)
      FORMAT(4X,'**ENTER SIZE OF MATRICES **')
      READ(*,*) M
      WRITE(*,36)
      FORMAT(4X,'**ENTER ORDER OF ORIGINAL MODEL **')
      READ(*,*) MM
      WRITE(*,37)
      FORMAT(4X,'**ENTER ORDER OF MODIFIED MODEL **')
      READ(*,*) MR
      WRITE(*,53)
      FORMAT('ENTER 1 FOR FRIST,2 FOR SECOND,3 FOR MIXED CAUE RFORM')
      READ(*,*) CAU
      WRITE(3,202)M,M,MM,MR
      FORMAT(4X,'MATRICES SIZE=',2I2/4X,'ORIGINAL MODEL ORDER=',I2/4X
     1  ,'REDUCED MODEL ORDER=',I2)
      DO 3 I=2,2
      DO 3 J=1,MM
      DO 3 K=1,M
      DO 3 L=1,M
      WRITE(*,39) I,J,K,L
      FORMAT(4X,'ENTER VALUES OF A(',4I2,')')
      READ(*,*) A(I,J,K,L)
      WRITE(3,4) I,J,K,L,A(I,J,K,L)
      FORMAT(4X,'A(',4I2,1X,') =',F10.2)
      CONTINUE
      JS=MM
      GO TO 5102
      IF(A(2,JS,1,1).NE.0.0)GO TO 5101
      JS=JS-1
      GO TO 5102
      AC1=A(2,JS,1,1)
      JS=JS-1
      WRITE(2,*)JS
      JS=JS+1
      IF(JS.EQ.1) GO TO 4003
      DO 4002 I=2,2
      DO 4002 J=1,JS
      DO 4002 K=1,M
      DO 4002 L=1,M
      WRITE(2,*) A(I,J,K,L)/AC1
```

```
2     CONTINUE
      GO TO 4003
3     MM=MM+1
      WRITE(2,*) MM
      DO 6 I=1,1
      DO 6 J=1,MM
      DO 6 K=1,M
      DO 6 L=1,M
      WRITE(*,46) I,J,K,L
      FORMAT(4X,'ENTER VALUES OF A(',4I2,')')
      READ(*,*) A(I,J,K,L)
      WRITE(3,7) I,J,K,L,A(I,J,K,L)
      FORMAT(4X,'A(',4I2,1X,') =',F10.2)
      CONTINUE
      WRITE(*,1350)
0     FORMAT(2X,'ENTER TOTAL TIME OF RESPONSE,STEP **')
      READ(*,*)FIN,STEP
      DO 1399 K=1,M
      DO 1399 L=1,M
      A(1,0,K,L)=0.0
      C(1,0,K,L)=0.0
9     CONTINUE
      AC2=A(1,MM,1,1)
      DO 1398 I=1,1
      DO 1398 J=0,MM
      DO 1398 K=1,M
      DO 1398 L=1,M
      WRITE(2,*) A(I,J,K,L)/AC2
8     CONTINUE
      AC3=AC1/AC2
      MM=MM-1
      DO 10 I=1,5
      DO 10 J=1,5
      DO 10 K=1,5
      DO 10 L=1,5
      K1(I,J,K,L)=0.0
      K2(I,J,K,L)=0.0
      CONTINUE
      IF(CAU.EQ.1) GO TO 1
      IF(CAU.EQ.2) GO TO 5
      WRITE(3,201)
      FORMAT(4X,'MODEL ORDER REDUCTION USING MIXED CAUER FORM')
      JJ=2
      DO 30 II=1,MM
      CALL MATIN(A,M,II,B,K1)
      CALL MATIN1(A,M,II,MM,B,K2)
      JJ=JJ+1
      IF(JJ.GT.(MM+1)) GO TO 100
      LL=MM-II
      DO 40 L=1,LL
      CALL MATMU2(A,K1,K2,M,JJ,L,II,D,D1)
      CONTINUE
      CONTINUE
      DO 92 II=1,MM
      DO 92 I=1,M
      DO 92 J=1,M
      WRITE(3,93) II,II,I,J,K1(II,II,I,J)
      FORMAT(4X,'K1(',4I2,') =' F10.2)
      CONTINUE
      DO 94 II=1,MM
      DO 94 I=1,M
      DO 94 J=1,M
```

```
      WRITE(3,96) II,II,I,J,K2(II,II,I,J)
      FORMAT(4X,'K2(',4I2,') =' F10.2)
      CONTINUE
      MM=MR
      DO 20 I=1,M
      DO 20 J=1,M
      C(MM+1,1,I,J)=0.0
      IF(I.EQ.J) C(MM+1,1,I,J)=1.0
      CONTINUE
      II=MM
      GO TO 50
      CALL DC1(C,K1,K2,M,II,MM)
      II=II-1
      IF(II.EQ.0) GO TO 60
      GO TO 50
      J=MM+1
      GO TO 80
      III=MM+2-J
      DO 70 L=1,III
      CALL DC2(K1,K2,C,J,L,M,K4,K3)
      CONTINUE
      J=J-1
      IF(J.LT.3) GO TO 19
      GO TO 80
      WRITE(3,95)
      FORMAT(4X,'MODEL ORDER REDUCTION USING SECOND CAUER FORM')
      JJ=2
      MK=2*MM
      DO 35 II=1,MK
      CALL MATIN(A,M,II,B,K1)
      JJ=JJ+1
      IF(JJ.GT.(2*MM+1)) GO TO 105
      DO 45 L=1,MM
      CALL MATMU1(A,K1,M,JJ,L,II,D)
      CONTINUE
      CONTINUE
5     MM=2*MR
      DO 192 II=1,MK
      DO 192 I=1,M
      DO 192 J=1,M
      WRITE(3,193) II,II,I,J,K1(II,II,I,J)
3     FORMAT(4X,'K1(',4I2,') =' F8.3)
2     CONTINUE
      DO 25 I=1,M
      DO 25 J=1,M
      C(MM+1,1,I,J)=0.0
      IF(I.EQ.J) C(MM+1,1,I,J)=1.0
      CONTINUE
      II=MM
      GO TO 55
      CALL DC3(C,K1,M,II,MM)
      II=II-1
      IF(II.EQ.0) GO TO 65
      GO TO 55
      J=MM+1
      GO TO 85
      III=MM/2
      DO 75 L=1,III
      CALL DC4(K1,C,J,L,M,K3)
      CONTINUE
      J=J-1
      IF(J.LT.3) GO TO 19
```

```
      GO TO 85
      JJ=2
      DO 13 J=1,MM
      DO 13 K=1,M
      DO 13 L=1,M
      B1(2,J,K,L)=A(2,MM+1-J,K,L)
 3    CONTINUE
      MJ=MM+1
      DO 14 J=1,MJ
      DO 14 K=1,M
      DO 14 L=1,M
      B1(1,J,K,L)=A(1,MM+2-J,K,L)
 4    CONTINUE
      WRITE(3,295)
295   FORMAT(4X,'MODEL ORDER REDUCTION USING FIRST CAUER FORM')
      MK=2*MM
      DO 31 II=1,MK
      CALL MATIN2(B1,M,II,B,K2)
      JJ=JJ+1
      IF(JJ.GT.(2*MM+1)) GO TO 101
      DO 41 L=1,MM
      CALL MATMU3(B1,K2,M,JJ,L,II,D1)
41    CONTINUE
31    CONTINUE
101   MM=2*MR
      DO 392 II=1,MK
      DO 392 I=1,M
      DO 392 J=1,M
      WRITE(3,393) II,II,I,J,K2(II,II,I,J)
393   FORMAT(4X,'K2(',4I2,') =' F10.2)
392   CONTINUE
      DO 21 I=1,M
      DO 21 J=1,M
      C(MM+1,1,I,J)=0.0
      IF(I.EQ.J) C(MM+1,1,I,J)=1.0
21    CONTINUE
      II=MM
      GO TO 51
51    CALL DC5(C,K2,M,II,MM)
      II=II-1
      IF(II.EQ.0) GO TO 61
      GO TO 51
61    J=MM+1
      GO TO 81
81    III=MM/2
      DO 71 L=1,III
      CALL DC6(K2,C,J,L,M,K4)
71    CONTINUE
      J=J-1
      IF(J.LT.3) GO TO 1119
      GO TO 81
1119  I=1
      DO 1013 J=1,MR
      DO 1013 K=1,M
      DO 1013 L=1,M
      B1(2,J,K,L)=0.0
      B1(2,J,K,L)=C(2,MR+1-J,K,L)
1013  CONTINUE
      IM=MR+1
      DO 1014 J=1,IM
      DO 1014 K=1,M
      DO 1014 L=1,M
```

```
            B1(1,J,K,L)=0.0
            B1(1,J,K,L)=C(1,MR+2-J,K,L)
 1014       CONTINUE
            DO 1015 J=1,MR
            DO 1015 K=1,M
            DO 1015 L=1,M
            C(2,J,K,L)=0.0
            C(2,J,K,L)=B1(2,J,K,L)
 1015       CONTINUE

      DO 1016 J=1,IM
      DO 1016 K=1,M
      DO 1016 L=1,M
      C(1,J,K,L)=0.0
      C(1,J,K,L)=B1(1,J,K,L)
5     CONTINUE
      GO TO 19
      JH=MR
      GO TO 5011
1     IF(C(2,JH,1,1).NE.0.0)GO TO 5012
      JH=JH-1
      GO TO 5011
2     CA1=C(2,JH,1,1)
      JH=JH-1
      WRITE(2,*)JH
      JH=JH+1
      DO 308 I=2,2
      DO 306 J=1,MR
      DO 308 K=1,M
      DO 308 L=1,M
      WRITE(3,309) I,J,K,L,C(I,J,K,L)
      FORMAT(4X,'C(',4I2,1X,') =',F10.2)
      CONTINUE
      IF(JH.EQ.1) GO TO 4010
      DO 1308 I=2,2
      DO 1308 J=1,JH
      DO 1308 K=1,M
      DO 1308 L=1,M
      WRITE(2,*) C(I,J,K,L)/CA1
3     CONTINUE
      GO TO 4010
8     MR=MR+1
      WRITE(2,*) MR
      DO 416 I=1,1
      DO 416 J=1,MR
      DO 416 K=1,M
      DO 416 L=1,M
      WRITE(3,417) I,J,K,L,C(I,J,K,L)
      FORMAT(4X,'C(',4I2,1X,') =',F10.2)
      CONTINUE
      CA2=C(1,MR,1,1)
      DO 1397 I=1,1
      DO 1397 J=0,MR
      DO 1397 K=1,M
      DO 1397 L=1,M
      WRITE(2,*)C(I,J,K,L)/CA2
7     CONTINUE
      CA3=CA1/CA2
      IF(M.GT.1) STOP
      REWIND 2
      WRITE(3,24)
      FORMAT(2X,'ROOTS OF ORIGINAL T.F. NUMERATOR POLYNOMIAL')
      READ(2,*) N
```

```
      IF(N.EQ.0)GO TO 4004
      NCOFS=N+1
      DO 22 I=1,NCOFS
      READ(2,*) E(I)
      CONTINUE
      GO TO 4004
4     READ(2,*)NA
      WRITE(4,*) N,NA
      IF(N.EQ.0) GO TO 4005
      CALL ROOT(E,N,NCOFS)
      GO TO 4005
5     WRITE(3,26)
      FORMAT(2X,'ROOTS OF ORIGINAL T.F. DENOMIRATOR POLYNOMIAL')
      N=NA
      NCOFS=N+1
      DO 27 I=1,NCOFS
      READ(2,*) E(I)
      CONTINUE
      CALL ROOT(E,N,NCOFS)
      WRITE(3,28)
      FORMAT(2X,'ROOTS OF MODIFIED T.F. NUMERATOR POLYNOMIAL')
      READ(2,*) N
      IF(N.EQ.0) GO TO 4006
      NCOFS=N+1
      DO 29 I=1,NCOFS
      READ(2,*) E(I)
      CONTINUE
      GO TO 4006
6     READ(2,*)NA
      WRITE(4,*) N,NA
      IF (N.EQ.0) GO TO 4007
      CALL ROOT(E,N,NCOFS)
      GO TO 4007
7     WRITE(3,42)
      FORMAT(2X,'ROOTS OF MODIFIED T.F. DENOMINATOR POLYNOMIAL')
      N=NA
      NCOFS=N+1
      DO 44 I=1,NCOFS
      READ(2,*) E(I)
      CONTINUE
      CALL ROOT(E,N,NCOFS)
      REWIND 4
      DO 49 K=1,2,1
      READ(4,*)NZ,NP
      DO 47 I=1,NZ
      READ(4,1001) ZPE,ZIM
1     FORMAT(2X,F20.6,2X,F20.6)
      XA(I)=CMPLX(ZPE,ZIM)
      CONTINUE
      DO 48 J=1,NP
      READ(4,1001) ZPE,ZIM
      XB(J)=CMPLX(ZPE,ZIM)
      CONTINUE
      CALL PFE(NP,NZ,XA,XB)
      CONTINUE
      REWIND 5
      READ(5,*)N
      READ(5,*)(MX(I),I=1,N)
      DO 302 J=1,N,1
      READ(5,*) ZRE,ZIM
      IF(ABS(ZIM).LT..00001) ZIM=0.0
      Q(J)=CMPLX(ZRE,ZIM)
```

```
302     CONTINUE
        DO 1392 J=1,N
        READ(5,*) PRE,PIM
        IF(ABS(PIM).LT..00001) PIM=0.0
        P(J)=CMPLX(PRE,PIM)
1392    CONTINUE
        CALL LAPIN(N,MX,P,0,F,FIN,STEP,P1,IFIN)
        DO 2 K=0,IFIN,1
        P1(2,K)=F(N,K)*AC3
2       CONTINUE
        READ(5,*)N
        READ(5,*)(MX(I),I=1,N)
        DO 1393 J=1,N,1
        READ(5,*) ZRE,ZIM
        IF(ABS(ZIM).LT..00001) ZIM=0.0
        Q(J)=CMPLX(ZRE,ZIM)
1393    CONTINUE
        DO 1394 J=1,N
        READ(5,*) PRE,PIM
        IF(ABS(PIM).LT..00001) PIM=0.0
        P(J)=CMPLX(PRE,PIM)
1394    CONTINUE
        CALL LAPIN(N,MX,P,Q,F,FIN,STEP,P1,IFIN)
        DO 1395 K=0,IFIN,1
        P1(3,K)=F(N,K)*CA3
1395    CONTINUE
        DO 1396 K=0,IFIN,1
        WRITE(1,1440)P1(1,K),P1(2,K),P1(3,K)
1440    FORMAT(F8.2,F8.4,F8.4)
1396    CONTINUE
        WRITE(3,1400)
1400    FORMAT(6X,'TIME',5X,'ORIGINAL',5X,'REDUCED')
        DO 1402 K=0,IFIN,1
        WRITE(3,1430) P1(1,K),P1(2,K),P1(3,K)
1430    FORMAT(6X,F4.2,5X,F8.4,7X,F8.4)
1402    CONTINUE
        STOP
        END
        SUBROUTINE MATIN(A,M,II,B,K1)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES THE K1 QUOTIENTS OF THE MIXED
C                    AND FIRST MATRIX CAUER FORM
C
C*************************************************************************
        DIMENSION A(8,8,2,2),B(8,8,2,2),K1(8,8,2,2)
        REAL A,B,K1
        DO 5 I=1,M
        DO 5 J=1,M
        B(II+1,1,I,J)=A(II+1,1,I,J)
5       CONTINUE
        DO 35 LM=1,1
        DO 35 N=1,M
        DO 40 I=1,M
        DO 40 J=1,M
        IF((I.NE.N).AND.(J.NE.N))B(II+1,LM,I,J)=B(II+1,LM,I,J)-B(II+1,LM,I
     1  ,N)*B(II+1,LM,N,J)/B(II+1,LM,N,N)
40      CONTINUE
        B(II+1,LM,N,N)=-1.0/B(II+1,LM,N,N)
        DO 35 NN=1,M
        IF (NN.EQ.N) GO TO 35
        B(II+1,LM,NN,N)=B(II+1,LM,N,N)*B(II+1,LM,NN,N)
```

```
      B(II+1,LM,N,NN)=B(II+1,LM,N,N)*B(II+1,LM,N,NN)
35    CONTINUE
      DO 50 I=1,M
      DO 50 J=1,M
      B(II+1,1,I,J)=-B(II+1,1,I,J)
50    CONTINUE
      DO 30 I=1,M
      DO 30 K=1,M
      DO 30 J=1,M
      K1(II,II,I,K)=K1(II,II,I,K)+A(II,1,I,J)*B(II+1,1,J,K)
30    CONTINUE
      RETURN
      END
      SUBROUTINE MATIN1(A,M,II,MM,B,K2)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES THE K2 QUOTIENTS OF THE MIXED
C                       MATRIX CAUER FORM
C
C*************************************************************************
      DIMENSION A(8,8,2,2),B(8,8,2,2),K2(8,8,2,2)
      REAL A,B,K2
      DO 5 I=1,M
      DO 5 J=1,M
      B(II+1,MM-II+1,I,J)=A(II+1,MM-II+1,I,J)
5     CONTINUE
      DO 35 N=1,M
      DO 40 I=1,M
      DO 40 J=1,M
      IF((I.NE.N).AND.(J.NE.N)) B(II+1,MM-II+1,I,J)=B(II+1,MM-II+1,I,J)-
     1  B(II+1,MM-II+1,I,N)*B(II+1,MM-II+1,N,J)/B(II+1,MM-II+1,N,N)
40    CONTINUE
      B(II+1,MM-II+1,N,N)=-1.0/B(II+1,MM-II+1,N,N)
      DO 35 NN=1,M
      IF(NN.EQ.N) GO TO 35
      B(II+1,MM-II+1,NN,N)=B(II+1,MM-II+1,N,N)*B(II+1,MM-II+1,NN,N)
      B(II+1,MM-II+1,N,NN)=B(II+1,MM-II+1,N,N)*B(II+1,MM-II+1,N,NN)
35    CONTINUE
      DO 50 I=1,M
      DO 50 J=1,M
      B(II+1,MM-II+1,I,J)=-B(II+1,MM-II+1,I,J)
50    CONTINUE
      DO 30 I=1,M
      DO 30 K=1,M
      DO 30 J=1,M
      K2(II,II,I,K)=K2(II,II,I,K)+A(II,MM-II+2,I,J)*B(II+1,MM-II
     1  +1,J,K)
30    CONTINUE
      RETURN
      END
      SUBROUTINE MATMU2(A,K1,K2,M,JJ,L,II,D,D1)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES THE REST ELEMENT OF
C                       ROUTH ARRAY
C
C*************************************************************************
      DIMENSION A(8,8,2,2),K1(8,8,2,2),D(8,8,2,2),K2(8,8,2,2),D1(8,8,2,2
     1  )
      REAL A,K1,D,D1,K2
      DO 30 I=1,M
      DO 30 K=1,M
```

```
      D(II,II,I,K)=0.0
      DO 30 J=1,M
      D(II,II,I,K)=D(II,II,I,K)+K1(II,II,I,J)*A(JJ-1,L+1,J,K)
30    CONTINUE
      DO 40 I=1,M
      DO 40 K=1,M
      D1(II,II,I,K)=0.0
      DO 40 J=1,M
      D1(II,II,I,K)=D1(II,II,I,K)+K2(II,II,I,J)*A(JJ-1,L,J,K)
40    CONTINUE
      DO 60 I=1,M
      DO 60 J=1,M
      A(JJ,L,I,J)=A(JJ-2,L+1,I,J)-D(II,II,I,J)-D1(II,II,I,J)
      WRITE(3,4) JJ,L,I,J,A(JJ,L,I,J)
4     FORMAT(4X,'A(',4I2,') =',F10.2)
60    CONTINUE
      RETURN
      END
      SUBROUTINE DC1(C,K1,K2,M,II,MM)
C*************************************************************************
C
C     THIS SUBROUTINE CALCULATES SOME ELEMENTS OF MODIFIED
C                          T.F
C
C*************************************************************************
      DIMENSION C(8,8,2,2),K1(8,8,2,2),K2(8,8,2,2)
      REAL K1,K2,C
      DO 10 I=1,M
      DO 10 J=1,M
      C(II,1,I,J)=0.0
      C(II,MM+2-II,I,J)=0.0
10    CONTINUE
      DO 30 I=1,M
      DO 30 K=1,M
      DO 30 J=1,M
      C(II,1,I,K)=C(II,1,I,K)+K1(II,II,I,J)*C(II+1,1,J,K)
      C(II,MM+2-II,I,K)=C(II,MM+2-II,I,K)+K2(II,II,I,J)*C(II+1,MM+1-II,
     1 J,K)
30    CONTINUE
      RETURN
      END
      SUBROUTINE DC2(K1,K2,C,J,L,M,K4,K3)
C*************************************************************************
C
C     THIS SUBROUTINE CALCULATES SOME ELEMENTS OF MODIFIED
C                          T.F
C
C*************************************************************************
      DIMENSION C(8,8,2,2),K1(8,8,2,2),K2(8,8,2,2),K3(8,8,2,2),K4(8,8,2
     1 ,2)
      REAL K1,K2,K3,K4,C
      DO 20 I=1,M
      DO 20 K=1,M
      K4(J-1,L+1,I,K)=0.0
      K3(J-1,L+1,I,K)=0.0
      DO 20 JD=1,M
      K4(J-1,L+1,I,K)=K4(J-1,L+1,I,K)+K2(J-2,J-2,I,JD)*C(J-1,L,JD,K)
      K3(J-1,L+1,I,K)=K3(J-1,L+1,I,K)+K1(J-2,J-2,I,JD)*C(J-1,L+1,JD,K)
20    CONTINUE
      DO 30 I=1,M
      DO 30 K=1,M
      C(J-2,L+1,I,K)=C(J,L,I,K)+K3(J-1,L+1,I,K)+K4(J-1,L+1,I,K)
```

```
      DO 30 I=1,M
      DO 30 K=1,M
      C(J-2,L+1,I,K)=C(J,L,I,K)+K3(J-1,L+1,I,K)
30    CONTINUE
      RETURN
      END
      SUBROUTINE MATIN2(B1,M,II,B,K2)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES THE K2 QUOTIENTS OF THE SECOND
C                       MATRIX CAUER FORM
C
C*************************************************************************
      DIMENSION B1(8,8,2,2),B(8,8,2,2),K2(8,8,2,2)
      REAL B1,B,K2
      DO 5 I=1,M
      DO 5 J=1,M
      B(II+1,1,I,J)=B1(II+1,1,I,J)
5     CONTINUE
      DO 35 LM=1,1
      DO 35 N=1,M
      DO 40 I=1,M
      DO 40 J=1,M
      IF((I.NE.N).AND.(J.NE.N))B(II+1,LM,I,J)=B(II+1,LM,I,J)-B(II+1,LM,I
     1  ,N)*B(II+1,LM,N,J)/B(II+1,LM,N,N)
40    CONTINUE
      B(II+1,LM,N,N)=-1.0/B(II+1,LM,N,N)
      DO 35 NN=1,M
      IF (NN.EQ.N) GO TO 35
      B(II+1,LM,NN,N)=B(II+1,LM,N,N)*B(II+1,LM,NN,N)
      B(II+1,LM,N,NN)=B(II+1,LM,N,N)*B(II+1,LM,N,NN)
35    CONTINUE
      DO 50 I=1,M
      DO 50 J=1,M
      B(II+1,1,I,J)=-B(II+1,1,I,J)
50    CONTINUE
      DO 30 I=1,M
      DO 30 K=1,M
      DO 30 J=1,M
      K2(II,II,I,K)=K2(II,II,I,K)+B1(II,1,I,J)*B(II+1,1,J,K)
30    CONTINUE
      RETURN
      END
      SUBROUTINE MATMU3(B1,K2,M,JJ,L,II,D1)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES THE REST ELEMENT OF
C                       ROUTH ARRAY
C
C*************************************************************************
      DIMENSION B1(8,8,2,2),K2(8,8,2,2),D1(8,8,2,2)
      REAL B1,K2,D1
      DO 30 I=1,M
      DO 30 K=1,M
      D1(II,II,I,K)=0.0
      DO 30 J=1,M
      D1(II,II,I,K)=D1(II,II,I,K)+K2(II,II,I,J)*B1(JJ-1,L+1,J,K)
30    CONTINUE
      DO 50 I=1,M
      DO 50 J=1,M
      B1(JJ,L,I,J)=0.0
50    CONTINUE
```

```
30      CONTINUE
        RETURN
        END
        SUBROUTINE MATMU1(A,K1,M,JJ,L,II,D)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES THE REST ELEMENT OF
C                    ROUTH ARRAY OF FIRST
C                      CAUER FORM
C
C*************************************************************************
        DIMENSION A(8,8,2,2),K1(8,8,2,2),D(8,8,2,2)
        REAL A,K1,D
        DO 30 I=1,M
        DO 30 K=1,M
        D(II,II,I,K)=0.0
        DO 30 J=1,M
        D(II,II,I,K)=D(II,II,I,K)+K1(II,II,I,J)*A(JJ-1,L+1,J,K)
30      CONTINUE
        DO 60 I=1,M
        DO 60 J=1,M
        A(JJ,L,I,J)=A(JJ-2,L+1,I,J)-D(II,II,I,J)
        WRITE(3,1) JJ,L,I,J,A(JJ,L,I,J)
1       FORMAT(4X,'A(',4I2,') =',F10.2)
60      CONTINUE
        RETURN
        END
        SUBROUTINE DC3(C,K1,M,II,MM)
C*************************************************************************
C
C       THIS SUBROUTINE CALCULATES SOME ELEMENTS OF MODIFIED
C                            T.F
C
C*************************************************************************
        DIMENSION C(8,8,2,2),K1(8,8,2,2)
        REAL K1,C
        DO 10 I=1,M
        DO 10 J=1,M
        C(II,1,I,J)=0.0
10      CONTINUE
        DO 30 I=1,M
        DO 30 K=1,M
        DO 30 J=1,M
        C(II,1,I,K)=C(II,1,I,K)+K1(II,II,I,J)*C(II+1,1,J,K)
30      CONTINUE
        RETURN
        END
        SUBROUTINE DC4(K1,C,J,L,M,K3)
*************************************************************************
C
C       THIS SUBROUTINE CALCULATES SOME ELEMENTS OF MODIFIED
C                            T.F
C
C*************************************************************************
        DIMENSION C(8,8,2,2),K1(8,8,2,2),K3(8,8,2,2)
        REAL K1,K3,C
        DO 20 I=1,M
        DO 20 K=1,M
        K3(J-1,L+1,I,K)=0.0
        DO 20 JD=1,M
        K3(J-1,L+1,I,K)=K3(J-1,L+1,I,K)+K1(J-2,J-2,I,JD)*C(J-1,L+1,JD,K)
20      CONTINUE
```

```
            DO 60 I=1,M

          DO 60 J=1,M
          B1(JJ,L,I,J)=B1(JJ,L,I,J)+B1(JJ-2,L+1,I,J)-D1(II,II,I,J)
          WRITE(3,4) JJ,L,I,J,B1(JJ,L,I,J)
4         FORMAT(4X,'B1(',4I2,') ='F10.2)
60        CONTINUE
          RETURN
          END
          SUBROUTINE DC5(C,K2,M,II,MM)
C*************************************************************************
C
C         THIS SUBROUTINE CALCULATES SOME ELEMENTS OF MODIFIED
C                              T.F
C
C*************************************************************************
          DIMENSION C(8,8,2,2),K2(8,8,2,2)
          REAL K2,C
          DO 10 I=1,M
          DO 10 J=1,M
          C(II,1,I,J)=0.0
10        CONTINUE
          DO 30 I=1,M
          DO 30 K=1,M
          DO 30 J=1,M
          C(II,1,I,K)=C(II,1,I,K)+K2(II,II,I,J)*C(II+1,1,J,K)
30        CONTINUE
          RETURN
          END
          SUBROUTINE DC6(K2,C,J,L,M,K4)
C*************************************************************************
C
C         THIS SUBROUTINE CALCULATES SOME ELEMENTS OF MODIFIED
C                              T.F
C
C*************************************************************************
          DIMENSION C(8,8,2,2),K2(8,8,2,2),K4(8,8,2,2)
          REAL K2,K4,C
          DO 20 I=1,M
          DO 20 K=1,M
          K4(J-1,L+1,I,K)=0.0
          DO 20 JD=1,M
          K4(J-1,L+1,I,K)=K4(J-1,L+1,I,K)+K2(J-2,J-2,I,JD)*C(J-1,L+1,JD,K)
20        CONTINUE
          DO 30 I=1,M
          DO 30 K=1,M
          C(J-2,L+1,I,K)=C(J,L,I,K)+K4(J-1,L+1,I,K)
30        CONTINUE
          RETURN
          END
          SUBROUTINE ROOT(E,N,NCOFS)
C*************************************************************************
C
C                THIS SUBROUTINE CALCULATES THE ROOTS OF A
C                           POLYNOMIALS
C
C*************************************************************************
          DIMENSION E(37),AWORK(37),U(36),V(36)
          DATA U,V/36*0.,36*0./
1         WRITE(3,5) N
5         FORMAT(2X,' DEGREE OF POLYNOMIAL',2X,I2)
          IF(N.LE.0) GO TO 200
          IF(N.GT.36) GO TO 210
```

```
      IR=1
      DO 100 I=1,NCOFS
      IM1=I-1
      WRITE(3,20) IM1
20    FORMAT(2X,' COEFFS OF X**',I2)
      WRITE(3,*) E(I)
30    FORMAT(F15.5)
100   CONTINUE
      CALL POLRT(E,AWORK,N,U,V,IER)
      GO TO (200,210,220,230) IER
      WRITE(3,105)
105   FORMAT(3X,'ROOTS ARE')
      WRITE(3,110)
110   FORMAT(2X,'0',9X,'REAL PART',13X,'IMAGINARY PART')
      WRITE(3,4) (U(I),V(I),I=1,N)
      WRITE(4,4) (U(I),V(I),I=1,N)
4     FORMAT(2X,F20.6,2X,F20.6)
999   RETURN
200   WRITE(3,205)
205   FORMAT(2X,'0**ERROR**DEGREE MUST EXCEED 0')
      GO TO 1
210   WRITE(3,215)
215   FORMAT(3X,'**ERROR**DEGREE MUST BE LESS THAN 37')
      GO TO 1
220   WRITE(3,225)
225   FORMAT(3X,'0**WARNING**ALGORITHM DID NOT CONVERSE TO SPECIFIED
     1 ACCURACY')
      GO TO 999
230   WRITE(3,235)
235   FORMAT(3X,'0**ERROR**COEFFICIENT OF HIGHEST POWER CANNOT BE = 0')
      GO TO 1
      RETURN
      END
      SUBROUTINE POLRT(XCOF,COF,M,ROOTR,ROOT1,IER)
      DIMENSION XCOF(1),COF(1),ROOTR(1),ROOT1(1)
      DOUBLE PRECISION XO,YO,X,Y,XPR,YPR,UX,UY,V,YT,XT,U,XT2,YT2,SUMSQ
     1 ,DX,DY,TEMP,ALPHA,DABS
C     XCOF -VECTOR OF M+1 COEFFICIENTS OF THE POLYNOMIAL
C            ORDERED FROM SMALLEST TO LARGEST POWER
C     COF  -WORKING VECTOR OF LENGTH M+1
C     M    -ORDER OF POLYNOMIAL
C     ROOTR-RESULTANT VECTOR OF LENGTH M CONTAINING REAL ROOTS
C            OF THE POLYNOMIAL
C     ROOT1-RESULTANT VECTOR OF LENGTH M CONTAINING THE
C            CORRESSPONDING IMAGINARY ROOTS OF THE POLYNOMIAL
C     IER  -ERROR CODE: WHERE IER=0 MEANS NORMAL RETURN,IER=1
C            MEANS DEGREE < 1, IER=2 MEANS DEGREE > 36, IER=3
C            MEANS ALGORITHM FAILED TO CONVERSE, AND IER=4
C            MEANS COEFF OF HIGHEST POWER=0
      IFIT=0
      N=M
      IER=0
      IF(XCOF(N+1))10,25,10
10    IF(N) 15,15,32
15    IER=1
20    RETURN
25    IER=4
      GO TO 20
30    IER=2
      GO TO 20
32    IF(N-36) 35,35,30
35    NX=N
```

```
      NXX=N+1
      N2=1
      KJ1=N+1
      DO 40 L=1,KJ1
      MT=KJ1-L+1
40    COF(MT)=XCOF(L)
45    XO=.0500101
      YO=0.01000101
      IN=0
50    X=XO
      XO=-10.0*YO
      YO=-10.0*X
      X=XO
      Y=YO
      IN=IN+1
      GO TO 59
55    IFIT=1
      XPR=X
      YPR=Y
59    ICT=0
60    UX=0.0
      UY=0.0
      V=0.0
      YT=0.0
      XT=1.0
      U=COF(N+1)
      IF(U) 65,130,65
65    DO 70 I=1,N
      L=N-I+1
      TEMP=COF(L)
      XT2=X*XT-Y*YT
      YT2=X*YT+Y*XT
      U=U+TEMP*XT2
      V=V+TEMP*YT2
      FI=I
      UX=UX+FI*XT*TEMP
      UY=UY-FI*YT*TEMP
      XT=XT2
70    YT=YT2
      SUMSQ=UX*UX+UY*UY
      IF(SUMSQ) 75,110,75
75    DX=(V*UY-U*UX)/SUMSQ
      X=X+DX
      DY=-(U*UY+V*UX)/SUMSQ
      Y=Y+DY
78    IF(DABS(DY)+DABS(DX)-1.0D-05) 100,80,80
C     STEP ITERATION COUNTER
80    ICT=ICT+1
      IF(ICT-500)  60,85,85
85    IF(IFIT) 100,90,100
90    IF(IN-5)50,95,95
C     SET ERROR CODE TO 3
95    IER=3
      GO TO 20
100   DO 105 L=1,NXX
      MT=KJ1-L+1
      TEMP=XCOF(MT)
      XCOF(MT)=COF(L)
105   COF(L)=TEMP
      ITEMP=N
      N=NX
      NX=ITEMP
```

```
        IF(IFIT) 120,55,120
110     IF(IFIT)115,50,115
115     X=XPR
        Y=YPR
120     IFIT=0
122     IF(DABS(Y)-1.D-4*DABS(X))135,125,125
125     ALPHA=X+X
        SUMSQ=X*X+Y*Y
        N=N-2
        GO TO 140
130     X=0.0
        NX=NX-1
        NXX=NXX-1
135     Y=0.0
        SUMSQ=0.0
        ALPHA=X
        N=N-1
140     COF(2)=COF(2)+ALPHA*COF(1)
        IF(N.GT.2)GO TO 145
        COF(3)=COF(3)+ALPHA*COF(2)-SUMSQ*COF(1)
        GO TO 155
145     DO 150 L=2,N
150     COF(L+1)=COF(L+1)+ALPHA*COF(L)-SUMSQ*COF(L-1)
155     ROOT1(N2)=Y
        ROOTR(N2)=X
        N2=N2+1
        IF(N2.GT.M) GO TO 20
        IF(SUMSQ) 160,165,160
160     Y=-Y
        SUMSQ=0.0
        GO TO 155
165     IF(N) 20,20,45
        RETURN
        END
        SUBROUTINE PFE(NP,NZ,XA,XB)
C*************************************************************************
C
C          THE PROGRAM DOES THE PARTIAL FRACTION EXPANSION OF
C    P(S)/Q(S)=(S-XA(1))(S-XA(2))..(S-XA(N))/(S-XB(1))..(S-XB(N+1))
C            HAVING NUMERATOR'S POWER< DENOMINATOR'POWER.
C
C*************************************************************************
        COMPLEX XA(20),XB(20),XC(20),CC(20)
        INTEGER  XM(20)
        WRITE(3,1)
1       FORMAT('## DEGREE OF NUMERATOR < DEGREE OF DENOMINATOR ##')
        WRITE(3,2)
2       FORMAT('** DEGREE OF NUMERATOR & DENOMINATOR ARE ** ')
        WRITE(3,51)NZ,NP
        WRITE(5,*) NP
        IF(NZ.EQ.0)GO TO 4
        WRITE(3,3)
3       FORMAT(' REAL & IMAGINARY VALUES OF Nr ROOTS  ARE')
        WRITE(3,5)
5       FORMAT(' REAL PART(Nr)        IMAG. PART (Nr) ')
        WRITE(3,52)(REAL(XA(I)),AIMAG(XA(I)),I=1,NZ)
    4   WRITE(3,6)
6       FORMAT(' REAL & IMAGINARY VALUES OF Dr ROOTS  ARE')
        DO 30 I=1,NP
        XM(I)=1
30      CONTINUE
        DO 15 I=1,NP
```

```
      IF(XM(I).EQ.0)GO TO 15
      I1=I+1
      DO 25 J=I1,NP
      IF(XB(J).NE.XB(I))GO TO 15
      XM(I)=XM(I)+1
      XM(J)=0
25    CONTINUE
15    CONTINUE
      WRITE(3,8)
8     FORMAT(' MULTIPLICITY ')
      WRITE(3,53)(I,XM(I),I=1,NP)
      WRITE(5,*)(XM(I),I=1,NP)
      WRITE(3,9)
9     FORMAT(' REAL PART(Dr)        IMAG. PART (Dr) ')
      WRITE(3,52)(REAL(XB(I)),AIMAG(XB(I)),I=1,NP)
      DO 91 I=1,NP
      WRITE(5,*)REAL(XB(I)),AIMAG(XB(I))
91    CONTINUE
      DO 10 KK=1,NP
      IF(XM(KK).EQ.0)GO TO 10
      IF(XM(KK).GT.1)GO TO 14
      WRITE(3,12)
12    FORMAT('  THE COEFFS OF P.F.E FOR POLES OF MULTIPLICITY=1 ARE
     1 :')
      WRITE(3,54)XM(KK),KK
      WRITE(3,11)
11    FORMAT( '   REAL PART          IMAG PART '     )
      CALL PARF1(XA,XB,NZ,NP,KK,XC)
      WRITE(3,52)REAL(XC(KK)),AIMAG(XC(KK))
      WRITE(5,*)REAL(XC(KK)),AIMAG(XC(KK))
      GO TO 10
14    WRITE(3,16)
16    FORMAT('  THE COEFFS OF P.F.E FOR POLES OF MULTIPLICITY>1 ARE
     1 :')
      WRITE(3,54)XM(KK),KK
      WRITE(3,17)
17    FORMAT( '  REAL PART            IMAG PART  '     )
      CALL PARFM(XA,XB,XM(KK),NZ,NP,KK,CC)
      DO 18 I=1,XM(KK)
      WRITE(3,52)REAL(CC(I)),AIMAG(CC(I))
      WRITE(5,*)REAL(CC(I)),AIMAG(CC(I))
18    CONTINUE
10    CONTINUE
51    FORMAT(1X,'ORDER Nr=',I3,/,1X,'ORDER Dr=',I3)
52    FORMAT(1X,F10.6,8X,F10.6)
53    FORMAT(1X,'POLE NO.=',I2,3X,'MULTIPLICITY=',I2)
54    FORMAT(1X,'FOR MULTIPLICITY=',I2,2X,'POLE NO.=',I2)
      RETURN
      END
      SUBROUTINE PARF1(XA,XB,NZ,NP,KK,XC)
      COMPLEX XA(1),XB(1),XC(1),PNR,DR,Q1,Q2
      J=1
      PNR=CMPLX(1.,0.)
      IF(NZ.EQ.0)GO TO 33
      DO 20 JJ=1,NZ
      PNR=PNR*(XB(KK)-XA(JJ))
20    CONTINUE
33    IF(KK.LE.J)GO TO 40
      J=KK
      Q1=CMPLX(1.,0.)
      DO 30 JJ=1,J-1
      Q1=Q1*(XB(KK)-XB(JJ))
```

```
   30    CONTINUE
         IF(J.EQ.NP)GO TO 60
         GO TO 80
   60    Q2=CMPLX(1.,0.)
         GO TO 70
   40    Q1=CMPLX(1.,0.)
   80    Q2=CMPLX(1.,0.)
         DO 50 K1=J+1,NP
         Q2=Q2*(XB(KK)-XB(K1))
   50    CONTINUE
   70    DR=Q1*Q2
         XC(KK)=PNR/DR
   10    CONTINUE
         RETURN
         END
         SUBROUTINE PARFM(XA,XB,MM,NZ,NP,KK,CC)
         COMPLEX XA(1),XB(1),XC(1),NUM(20),XD(20),CC(20),
        1 CB(20),BB(20),AA(20),B1
         B1=XB(KK)
         J=1
         IF (NZ.EQ.0)GO TO 11
         DO 10 K=1,NZ
         AA(K)=XA(K)-B1
   10    CONTINUE
   11    JJ=1
         IF(KK.LE.J)GO TO 20
         J=KK
         DO 30 JJ=1,J-1
         CB(JJ)=XB(JJ)-B1
   30    CONTINUE
         IF(J.EQ.NP)GO TO 40
   20    II=JJ
         DO 50 K1=J+MM,NP
         BB(K1)=XB(K1)-B1
         CB(II)=BB(K1)
         II=II+1
   50    CONTINUE
   40    NN=0
         NUM(1)=CMPLX(1.,0.)
         IF(NZ.EQ.0)GO TO 41
         DO 1  I=1,NZ
         CALL PROD(NUM,AA(I),NN)
    1    CONTINUE
   41    ND =0
         XD(1)=CMPLX(1.,0.)
         DO 100 I=1,NP-MM
         CALL PROD(XD,CB(I),ND)
  100    CONTINUE
         CALL DIV(NUM,XD,MM,CC)
         RETURN
         END
         SUBROUTINE DIV(XA,XB,MM,CC)
         COMPLEX XA(1),XB(1),CC(1),S
         DO 10 I=1,MM
         S=CMPLX(0.,0.)
         DO 20 J=1,I-1
         S=S+CC(J)*XB(I-J+1)
   20    CONTINUE
         CC(I)=(XA(I)-S)/XB(1)
   10    CONTINUE
         RETURN
         END
```

```
      SUBROUTINE PROD(XA,P,NA)
      COMPLEX XA(21),P
      XA(NA+2)=XA(NA+1)
      NA=NA+1
      IF(NA.EQ.1)GO TO 2
      DO 1 I=2,NA
      J=NA+2-I
      XA(J)=XA(J-1)-P*XA(J)
    1 CONTINUE
    2 XA(1)=-P*XA(1)
      RETURN
      END
      SUBROUTINE LAPIN(N,MX,P,Q,F,FIN,STEP,P1,IFIN)
C*************************************************************************
C
C     THIS SUBROUTINE IS USED FOR LAPLACE INVERSE
C
C*************************************************************************
      COMPLEX P(10),Q(10)
      DIMENSION F(10,100),MX(10),P1(3,100)
      IFIN=FIN/STEP
      DO 10 K=0,IFIN,1
      P1(1,K)=K*STEP
10    CONTINUE
      DO 30 I=1,N,1
      IF(AIMAG(Q(I)).NE.0.0)GO TO 60
      J=MX(I)-1
      LL=1
      DO 90 L=1,J,1
      LL=LL*L
90    CONTINUE
      IF(J.EQ.0) LL=1
      DO 20 K=0,IFIN,1
      F(I,K)=(REAL(P(I))*(P1(1,K)**J)*EXP(REAL(Q(I))*P1(1,K)))/LL
20    CONTINUE
      GO TO 30
60    J=MX(I)-1
      LL=1
      DO 190 L=1,J,1
      LL=LL*L
190   CONTINUE
      IF(J.EQ.0) LL=1
      DO 110 KM=1,N,1
      IF(Q(I).NE.CONJG(Q(KM))) GO TO 110
      Q(KM)=0
      P(KM)=0
110   CONTINUE
      DO 40 K=0,IFIN,1
      F(I,K)=(P1(1,K)**J*2*EXP(REAL(Q(I))*P1(1,K))*((REAL(P(I))*
     1 COS(AIMAG(Q(I))*P1(1,K)))-(AIMAG(P(I))*SIN(AIMAG(Q(I))*P1(1,K))))
     1 )/LL
40    CONTINUE
30    CONTINUE
      DO 80 K=0,IFIN,1
      F(0,K)=0.0
      DO 50 I=1,N
      F(I,K)=F(I-1,K)+F(I,K)
      IF (I.NE.N)GO TO 50
50    CONTINUE
80    CONTINUE
      RETURN
      END

C*************************************************************************
```

```
C*************************************************************************
C       THIS PROGRAM IS FOR NYQUIST PLOT
C*************************************************************************
        DIMENSION A(2,10),C(2,10)
        COMPLEX A1,A2,A3(2000),S
        OPEN(UNIT=1,FILE='D7.OUT')
        OPEN(UNIT=2,FILE='D8.OUT')
        WRITE(*,2)
2       FORMAT('**ENTER ORDER OF ORIGINAL  T.F.** ')
        READ(*,*) MM
        MH=MM+1
        I=2
        DO 3 J=1,MM
        WRITE(*,4)I,J
4       FORMAT('ENTER VALUE OF A(',2I2,')')
        READ(*,*) A(I,J)
3       CONTINUE
        I=1
        DO 5 J=1,MH
        WRITE(*,6)I,J
6       FORMAT('ENTER VALUE OF A(',2I2,')')
        READ(*,*) A(I,J)
5       CONTINUE
        DO 10 K=0,1000,1
        AK=.05*K
        S=CMPLX(0.,AK)
        A1=CMPLX(0.,0.)
        A2=CMPLX(0.,0.)
        A3(K)=CMPLX(0.,0.)
        IF(K.EQ.0) GO TO 90
        DO 20 I=1,MM,1
        A1=A1+A(2,I)*(S**(I-1))
20      CONTINUE
        DO 30 J=1,MH,1
        A2=A2+A(1,J)*(S**(J-1))
30      CONTINUE
        GO TO 100
90      A1=A1+A(2,1)
        A2=A2+A(1,1)
100     A3(K)=A1/A2
        X1=REAL(A3(K))
        Y1=AIMAG(A3(K))
        WRITE(2,23)X1,Y1
23      FORMAT(F8.3,F8.3)
        WRITE(1,25)AK,X1,Y1
25      FORMAT(F8.3,F8.3,F8.3)
10      CONTINUE
        STOP
        END
C*************************************************************************
```

references

REFERENCES

1. Davison, E.J., 'A method for simplifying linear dynamic systems', IEEE Trans. Aut.Contr. Vol.AC-11, pp 93-101, Jan. 1966.

2. Aoki, M., 'Control of large dynamic systems by aggregation', IEEE trans, Automat. Control, Vol.AC-13, pp.246-256, 1968.

3. Davison, E.J., 'A New method for simplifying large linear dynamic systems, IEEE trans, Automat. Control, Vol. AC-13, pp 214-215, April, 1968.

4. Chidambara, M.R., 'On a method for simplifying linear dynamic systems', IEEE Trans, Automat. Control, Vol. AC-12;, pp 119-121, Feb.1967.

5. Gruca, A. and Bertrand, P., 'Approximation of high order systems by low order models with delays', Int. J. Control, Vol.28, No.6, pp 953-965, 1978.

6. Inooka, H. and Obinata, G., 'Mixed method of aggregation and ISE approach for system reduction', Electronics Lett., Vol.13, No.3, pp.88-90, Feb.1977.

7. Kuppurajulu, A. and Elangovan, S., 'System analysis by simplified models', IEEE trans. Autom. Control, Vol.AC-15, pp 234-237, April 1970.

8. Bereznia, G.T., and Sinha, N.K., 'Optimum approximation of high order systems by low order models', Int. J. Contr. Vol. 14, No.5, pp.957-959,1971.

9. Bondler, J.W., Markettos, N.D., and Sinha, N.K., 'Optimum systems modelling using recent gradient methods', Int. J. systems Sci., Vol.4, No.I,pp 33-44, 1973.

10. Yahagi, T., 'Optimal approximation of Transfer functions of linear dynamical systems', IEEE trans. circuits and systems', Vol. CAS-24, No.10, pp 560-568, Oct. 1977.

11. Wilson, D.A. and Mishra, R.N., 'Optimal reduction of multivariable systems', Int. J. Control, Vol.29, No.2, pp 267-278, 1979.

12. Chen, C.F. and Shieh, L.S., 'A novel approach to linear model simplification', Int. J. Control, Vol.22, No.2, pp.231-238, 1972.

13. Calfe, M.R., and Healey, M., 'Continued-fraction model reduction techniques for multivariable systems', Proc. IEE, Vol.121, pp.393-395, 1974.

14. Chen, C.F., 'Model reduction of multivariable control systems by means of matrix continued fractions', Int. J. Control, Vol.20, No.2, pp.225-238, 1974.

15. Shieh, L.S. and Gaudiano, F.F., 'Some properties and applications of matrix continued fraction', IEEE trans. on circuits and systems, Vol.CAS-22, No.9, pp.721-728, Sept. 1975.

16. Shamash, Y., 'Order reduction of linear systems by pade' approximation methods', IEEE Trans. on Automat. Control., Vol.AC-19, PP.615-616, 1974.

17. Large scale systems modelling by Magdi S Mahmomd, Madan G. Singh, Vol.3.

18. Shamash, Y., 'Model reduction using the Routh stability criterion and the Pade' approximation technique, Int. J. Control., Vol.21, No.3, pp.475-484, March 1975.

19. Paynter, H.M., 'Regelungstecknik moderne theorien undihre verbendbarkeit', pp.243-250, 1956.

20. Lal, M. and Mitra, R., 'Simplification of large system dynamics using a moment evaluation algorithm' IEEE trans, Autom. Control, Vol.AC-19, No.5, PP.602-603, Oct.1974.

21. Levy, E.C., 'Complex-curve fitting' IRE Trans. Automat. Control, Vol. AC-4, pp. 37-43, May 1959.

22 Hutton, M.F. and Friedland, B., 'Routh approximations for reducing order of linear, time invariant systems', IEEE trans. Autom, Control, Vol.AC-20, no.3, pp.329-337, June 1975.

23. H.Dubner and D.J.Abate : 'Numerical Inversion of Laplace Transforms by Relation them to finite Fourier Transform', J.ACM 15(1), 115-123, 1967.

24. F.Durbin, 'Numerical inversion of laplace transforms: An effective improvement of Dubner and Abate's Method', ACM, 17(4), 1973, 371-376.

25. K.S. Crump's 'Numerical Inversion of Laplace transforms using a fourier series Approximations, J.AC, 23(1), 1976, 89-96.

26. G. Honig and U. Hirdes : 'A method for numerical Inversion of Laplace transform Journal of computational and applied mathematics', 10(1984), 113-132.

27. E. Krezig, 'A text on Engineering Mathematics' (Lib).

28. Ph.D. thesis by J.Pal, 1980.

29. Shamash, Y., 'Closed-loop feedback systems using reduced order models', Electron Lett., Vol.12, no.24, pp.638-639, Nov.1976.

30. Towill, D.R., 'Transfer function techniques for control engineers, Ilifle Books, Ltd., London, 1970.

31. Shieh, L.S. and Gaudiano, F.F., 'Matrix continued fraction expansion and inversion by the generalized matrix Routh algorithm', Int. J. Control, 1974, Vol.20, No.5, 727-737.

32. Krishnamurty, V. and Seshadri, V., 'Model reduction using the Routh stability criterion', IEEE trans. Autom. Control. Vol.AC-23, no.4, pp.729-731, Aug.1978.

33. Whitfield, A.H. and Williams, N.G., 'Integral least-squares techniques for frequency-domain model reduction' Int. J. systems, 1988, Vol.19, No.8, pp.1355-1373.

34. Chen, T.C., Change, C.Y. and Han, K.W., 'Model reduction using the stability equation method and the continued fraction method', Int. J. Control, Vol.32. No.1, 1980, pp 81-94.

35. Elliott, H. and Wolovich, W.A., 'A frequency domain model reduction procedure', Automatica Vol.16, 1980, 261-270.

36. Ouyang, M., Liaw, C.M. and Pan, C.T., 'Model reduction by power decomposition and frequency response matching', IEEE trans, Automation control, Vol.AC-32, No.1, 1987, 60-62.

37. Pujara, L.R. and Rattan, K.S., 'Model reduction of linear multivariable control systems via frequency response matching', IEEE proceedings, Vol.131, Pt.D, No.6, 1984, 243-247.

38. Rao, S.V. and Lamba, S.S., 'A new frequency domain technique for the simplification of linear dynamic system', Int.J. Control, Vol.20, No.1, 1974, 71-79.

39. Reddy, A.S.S.R., 'A new frequency domain simplification of transfer function', Int. J. Control, Vol.23, No.3, 1976, 403-408.

40. Rao, A.S., Lamba, S.S. and Bandopadhyay, B., 'Choice of model order for multivariable systems' Int.J.Syst. Sci Vol.14, No.10, 1983, 1171-1183.