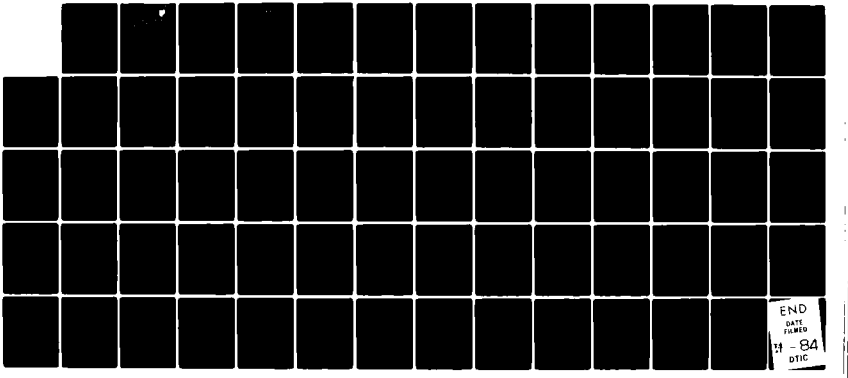
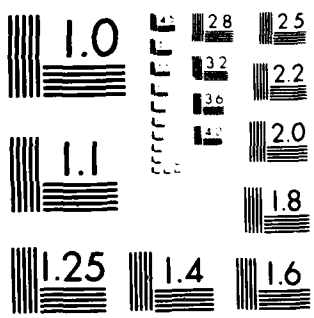
AD-A135 703 VOLTERRA TRANSFER FUNCTIONS FROM PULSE TESTS FOR MILDLY 1/1
NONLINEAR CHANNEL..(U) UNIVERSITY OF SOUTH FLORIDA
TAMPA DEPT OF ELECTRICAL ENGINEER.. V K JAIN ET AL.
UNCLASSIFIED JUL 83 RADC-TR-83-157 F30602-82-C-0135 F/G 9/4 NL





RADC-TR-83-157
Interim Report
July 1983

VOLTERRA TRANSFER FUNCTIONS FROM PULSE TESTS FOR MILDLY NONLINEAR CHANNELS

University of South Florida

Vijay K. Jain, Aubrey M. Bush and Daniel J. Kenneally

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-83-157 has been reviewed and is approved for publication.

APPROVED: Daniel J. Kenneally

DANIEL J. KENNEALLY
Project Engineer

APPROVED: W. S. Tuthill

W. S. TUTHILL, Colonel, USAF
Chief, Reliability & Compatibility Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (RBCT) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-83-157	2. GOVT ACCESSION NO. A135703 / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) VOLTERRA TRANSFER FUNCTIONS FROM PULSE TESTS FOR MIDLY NONLINEAR CHANNELS	5. TYPE OF REPORT & PERIOD COVERED Interim Report October 82 - March 83 6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) Vijay K. Jain (USF) Aubrey M. Bush (GIT) Daniel J. Kenneally (RADC)	8. CONTRACT OR GRANT NUMBER(s) F30602-82-C-0135
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of South Florida Department of Electrical Engineering Tampa FL 33620	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62702F 23380342
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (RBCT) Griffiss Air Force Base NY 13441	12. REPORT DATE July 1983 13. NUMBER OF PAGES 70
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same	15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

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

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Daniel J. Kenneally (RBCT)

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

Nonlinear Communications Channels	Poles	Network Identification
Volterra Series	Residues	
Volterra Transfer Functions	Quadratic Functions	
Pulse Testing	Cubic Functions	
Computer Analysis	Pencil of Functions	

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

Multichannel communications systems are often mildly nonlinear, hence they are characterizable by the Volterra series. The methodology described herein represents a numerical implementation of an RADC in-house concept formulation for pulse testing linear and quadratic Volterra systems. This analytic formulation, in terms of the appropriate convolutions, expressed the linear and quadratic responses to square pulse input waveforms. These responses contain, in canonic form, the system poles and residues which are

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

then determined by suitable identification methods and algorithms to provide the Volterra Transfer functions. In this paper we describe a method for determining the Volterra transfer-functions H_1 (s_1) and H_2 (s_1,s_2) from pulse tests. The method involves two transient tests in the laboratory, followed by analysis by the computer. The latter consists of (a) pole determination using the pencil-of-functions method, and (b) computation of the residues by a least-squares technique. Advantages of the method include the rapidity of the laboratory tests, as contrasted with traditional frequency-scan approaches, and the explicit determination of the transfer functions. Furthermore, the method is readily extendible to H_3 (s_1,s_2,s_3) and even to higher order transfer functions, although the computations grow very rapidly for these cases.

ACKNOWLEDGEMENT

The authors wish to express their gratitude to Mr. J. F. Spina of the Rome Air Development Center, and Dr. D. D. Weiner of Syracuse University for their helpful comments.

TABLE OF CONTENTS

LIST OF FIGURES

VOLTERRA TRANSFER FUNCTIONS FROM PULSE TESTS FOR MILDLY NONLINEAR CHANNELS

SUMMARY

The objective of this effort is the development of the analytical and experimental procedures necessary for equipment EMC characterization in terms of nonlinear (Volterra) transfer functions. The primary emphasis is on the development and exploitation of efficient network identification methods for determining the nonlinear circuit transfer function models of the RF emitter ports, the RF coupling paths, and the RF receptor ports of a system level model used for intrasystem interference analysis and prediction. A secondary emphasis is on the specification, synthesis, and design of suitable interference compensation transfer function which when appropriately combined with the identified nonlinear port transfer functions, can effectively suppress selective, undesired nonlinear responses to acceptable levels, consistent with some overall system performance measure. This report describes a method for determining linear and quadrative Volterra transfer functions from pulse tests. The method involves two transient tests in the laboratory, followed by data analysis by the computer, and is readily extendable to cubic and even higher order transfer functions where the computations grow very rapidly.

I. INTRODUCTION

One of the main problems encountered in the design and testing of frequency-division-multiplexed systems, e.g., analog coaxial cable, is the determination of intermodulation distortion arising from the repeater amplifiers. In these multichannel systems, many intermodulation products generated in an amplifier often have the same frequency and result in particularly high interference levels at certain frequencies. Furthermore, when the number of amplifiers used in the system is large, as is usually the case, certain distortion products (or nonlinear mixes) are generated and propagate in phase along the line with little or no losses. In addition other distortion products may also arise in the ancillary networks used in conjunction with the repeater amplifiers; for example, directional filters, power filters, compandors, and couplers may combine and generate intermodulation products. These products may increase to an exceedingly high level due to the accumulation of these distortion waveforms along the line. Therefore it is important

in testing such systems to make certain that the distortion lies within acceptable limits. The Volterra series expansion [1]-[3] permits description of the nonlinear system in a compact form and, in turn, enables expression of the distortion in terms of the multivariable transfer-functions [4]-[6].

In this paper we describe a method for determining the Volterra transfer-functions $H_1(s_1)$ and $H_2(s_1,s_2)$ from pulse tests. The method involves two transient tests in the laboratory, followed by analysis of the computer. The latter consists of (a) pole determination using the pencil-of-functions method [7]-[9], and (b) computation of the residues by a least-squares technique. Advantages of the method include the rapidity of the laboratory tests, as contrasted with traditional frequency-scan approaches, and the explicit determination of the transfer functions. Furthermore, the method is readily extendible to $H_3(s_1,s_2,s_3)$ and even to higher order transfer functions, although the computations grow very rapidly for these cases.

The structure of the paper is as follows. In section II we give a brief description of the pencil-of-functions method for the linear transfer function case. Section III discusses the nature of the Volterra response of a 2-variable nonlinear system. The identification of such nonlinear systems is discussed in Section IV and some examples are given in Section V. User guides to associated computer programs are given in Sections VI and VII.

II. POLE-ZERO MODELING BY PENCIL-OF-FUNCTIONS METHOD

Recorded input, output responses of a network can be integrated, or first-order filtered, to yield a family of signals, called information signals. Application of the pencil-of-functions theorem [9] to this family yields, in a closed form, the identified parameters of the network function. The procedure for this pole-zero modeling method is described below, and the program listing (IGRAM) for the identification routine is given in Appendix A.

Linear Identification Problem

Given the input-output observations

$$\{u(k)\},\ \{y(k)\}, \qquad k = 0, 1, \ldots, K-1 \tag{3}$$

arising from a physical system (see Fig. 1) believed to be linear, and of finite order, it is desired to find a system model

$$H(z) = \frac{b_1 z^{-1} + \ldots + b_n z^{-n}}{1 + a_1 z^{-1} + \ldots + a_n z^{-n}} \tag{4}$$

$$= \sum_{i=1}^{n} \frac{R_i z^{-1}}{1 - z_i z^{-1}} \tag{5}$$

which best fits the observations (in some sense). A solution can be obtained by use of the pencil-of-functions method [7], [8]. The final formula is

$$H(z) = \frac{z^{-1}\left[\sum_{i=1}^{n} \sqrt{D}_{n+1+i}\,(1-qz^{-1})^{n-i}\right]/D}{\left[\sum_{i=1}^{n} \sqrt{D}_i\,(1-qz^{-1})^{n-i}\right]/D} \tag{6}$$

where q is the pole of each of the first-order filters in a filter cascade employed for generating a set of information signals $y_0, \ldots, y_n, u_0, \ldots, u_n$ (see Fig. 2). The numbers D_i are the diagonal cofactors of the covariance matrix of the information signals (omitting u_0) and $D = \sqrt{D_1} + \ldots + \sqrt{D_{2n+1}}$ A tutorial example of the application of Equation (6) is given in Appendix B.

Equivalent s and z Domain Functions

Conversion between s and z domain functions can be carried out on the basis of impulse-invariant or step-invariant transformations [10]:

$$\sum_{i=1}^{n} \frac{K_i}{s-s_i} \quad \text{Imp. Inv.} \quad \leftrightarrow \quad \sum_{i=1}^{n} \frac{K'_i}{(1-z_i z^{-1})} \tag{7a}$$

$$\text{Step Inv.} \quad \leftrightarrow \quad \sum_{i=1}^{n} K''_i \frac{z^{-1}}{(1-z_i z^{-1})} \tag{7b}$$

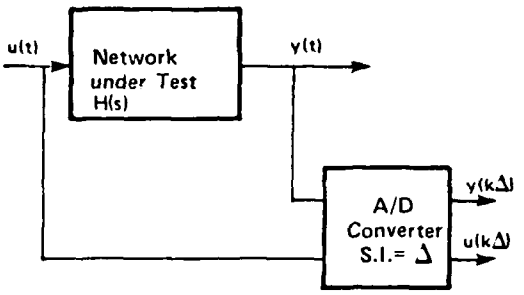


Fig. 1 Response-pair from system under test

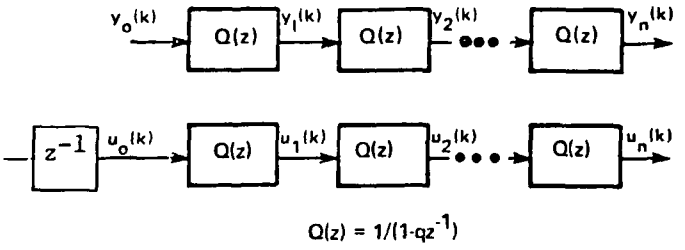


Fig. 2 Generation of information signals

where $z_i = e^{-s_i\Delta}$ $K_i' = K_i\Delta$ and $K_i'' = K_i(1-z_i)/s_i$; Δ is the sampling interval. We shall use the step-invariant transformation in this paper. The program listing for s to z (STOZ) is given in Appendix C.

III. VOLTERRA RESPONSE TO SQUARE-PULSE INPUTS

A. Volterra Series Representation of Nonlinear Systems

In the analysis of wide-band amplifiers, it is often assumed that the output signal depends only on the input signal at the same instant of time. The input/output relation can thus be expressed with a power series expansion as follows:

$$y(t) = a_1x(t) + a_2x^2(t) + a_3x^3(t) + \ldots \tag{8}$$

where x(t) and y(t) denote the input and output signals, respectively, and the coefficients a_n are time-independent constants. In general, however, the output y(t) is also dependent on the past input signal. A generalization of (8) in this case is a series of convolution integrals

$$y(t) = y_1(t) + y_2(t) + y_3(t) + \ldots \tag{9a}$$

where

$$y_n(t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\tau_1,\tau_2, \cdots,\tau_n)x(t - \tau_1)x(t - \tau_2) \cdots x(t - \tau_n)\, d\tau_1 d\tau_2 \cdots d\tau_n \tag{9b}$$

and h_n is a real-valued symmetric function of n real variables. Expression (9) is usually referred to as the Volterra series. This representation shows that a nonlinear system may be regarded as the combination of a linear and a number of higher order nonlinear subsystems. Each of these subsystems is characterized by an impulse response $h_n(\tau_1,\tau_2,\cdots\tau_n)$, which is also called a Volterra kernel. For a physically realizable system, h_n will have the value zero for all n whenever any of its argument is negative. Also, these

kernels are absolutely summable for stability. The nth order transfer function is defined as the n-fold Laplace transform [11] of h_n, i.e.,

$$H_n(s_1,..,s_n) = \int_{-\infty}^{\infty}\int .. \int h_n(\tau_1,\tau_2,..\tau_n)e^{-(s_1\tau_1+..+ s_n\tau_n)}d\tau_1..d\tau_n \quad (10)$$

In particular, we shall call $H_1(s_1)$ the linear transfer function and $H_2(s_1,s_2)$ the quadratic transfer function, and the corresponding contributions to the response as the linear and quadratic responses, respectively.

B. Two-Variable Volterra System

The two variable Volterra system is characterized by equation (8) when it is terminated at n = 2, i.e.,

$$y(t) = y_1(t) + y_2(t)$$

In the rest of the paper we shall assume that the quadratic subsystem $H_2(s_1,s_2)$ has the form shown in Fig. 3.

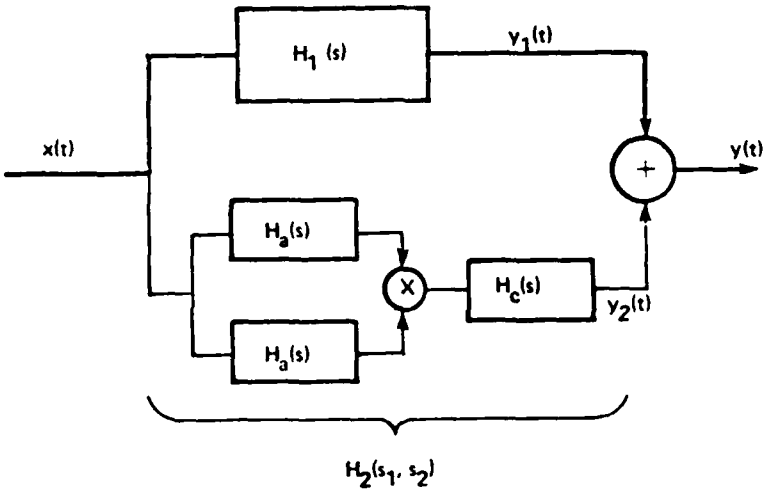
In testing a two-variable Volterra system, the following remark is very useful. It is possible to separate the linear and quadratic response by performing two measurements. Specifically, let the response of the system to the inputs $x^+(t) = x(t)$ and $x^-(t) = -x(t)$ be denoted as $y^+(t)$ and $y^-(t)$ respectively. From (9) it follows that $y^+(t) = y_1(t) + y_2(t)$ and $y^-(t) = -y_1(t) + y_2(t)$, so that

$$y_1(t) = \frac{1}{2}[y^+(t) - y^-(t)] \quad (11a)$$

$$y_2(t) = \frac{1}{2}[y^+(t) + y^-(t)] \quad (11b)$$

Clearly, $H_1(s_1)$ can be identified from the pair $x(t)$ and $y_1(t)$ by using the technique of Section II. Examples of successful identification of linear kernels are given in [7] and [12]; the former includes the case of a wideband RF amplifier. Fig. 4 is reproduced from reference [7].

We shall therefore concentrate on the identification of $H_2(s_1,s_2)$ from



$$H_a(s) = \sum_{i=1}^{n} \frac{A_i}{(s + a_i)}$$

$$H_c(s) = \sum_{k=1}^{n} \frac{C_k}{(s + c_k)}$$

Fig. 3 2-Variable system: Linear subsystem $H_1(s)$, and quadratic subsystem $H_2(s_1,s_2)$

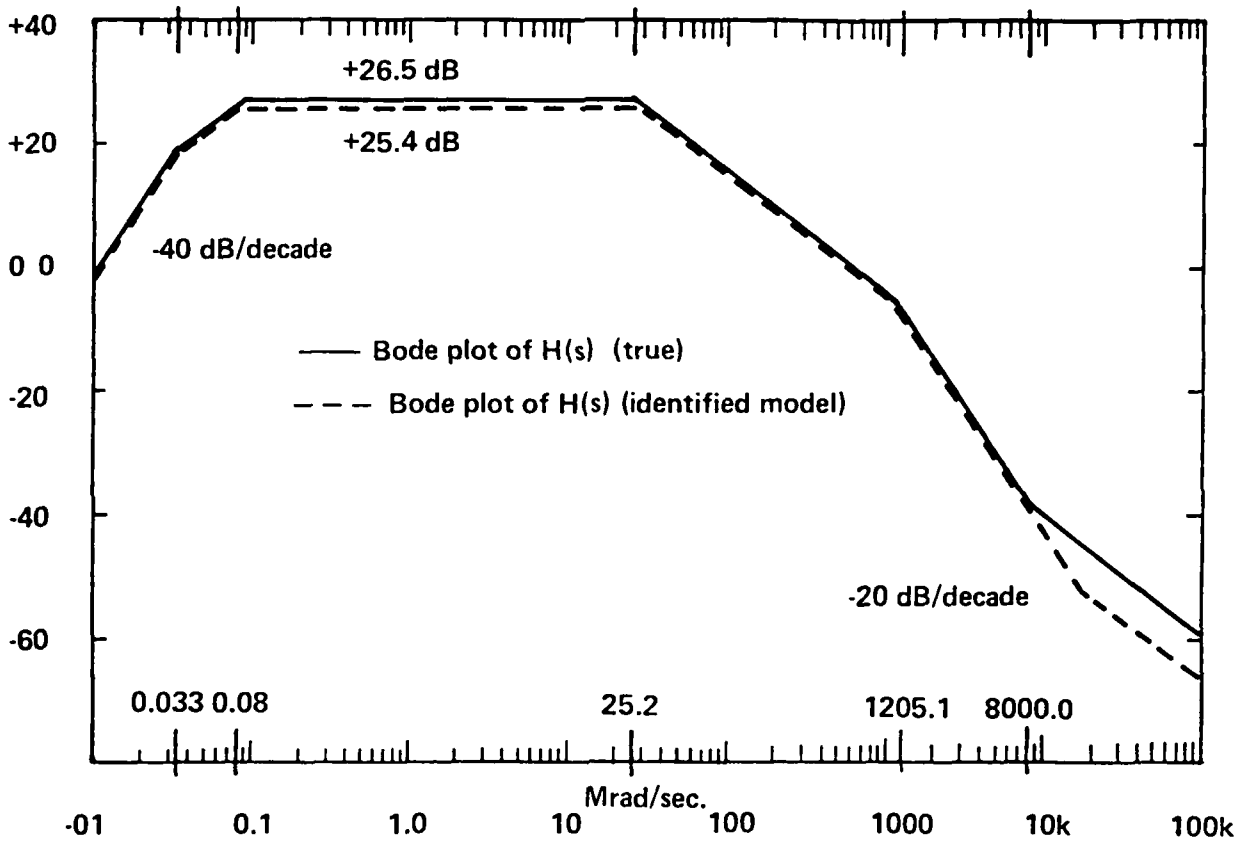


Fig. 4 Comparison of the magnitude (Bode) plots of the identified transfer function and the network function of an RF amplifier

the pair $x(t)$ and $y_2(t)$.

C. Square-Pulse Response of Quadratic TF

From the block-diagram of Fig. 3, it can be readily shown that

$$H_2(s_1,s_2) = H_a(s_1)H_a(s_2)H_c(s_1+s_2) \tag{12}$$

so that the associated two-dimensional response is[1]

$$Y_{(2)}(s_1,s_2) = H_a(s_1)H_a(s_2)H_c(s_1+s_2)X(s_1)X(s_2) \tag{13a}$$

For a square pulse input $p(t) = u(t) - u(t-T)$, the associated response for the specific transfer functions of Fig. 3 becomes[2]

$$Y_{(2)}(s_1,s_2) = \sum_{i,j,k=1}^{n} \frac{A_iA_jC_k(1-e^{-s_1T})(1-e^{-s_2T})}{s_1s_2(s_1+a_i)(s_2+a_j)(s_1+s_2+c_k)} \tag{13b}$$

From this the response $Y_2(s)$ can be found at once by use of the theorem of Appendix D. The inverse transform yields[2]

$y_2(t)$ =

$$\sum_{i,j,k=1}^{n} \frac{A_iA_jC_k}{a_ia_j}\left[d_o^k - d_1^{ik}e^{-a_it} - d_1^{jk}e^{-a_jt} + d_3^{ijk}e^{-(a_i+a_j)t} + (d_4^{jk}-d_5^{ijk})e^{-c_kt}\right]u(t)$$

for $0 \le t \le T$

$$\sum_{i,j,k=1}^{n} \frac{A_iA_jC_k}{a_ia_j}\left[d_1^{ik}(L^i-1)(L^j-1)e^{-(a_i+a_j)t'} + \right.$$

$$\left.((d_4^{jk}-d_5^{ijk})N^k + d_5L^i + d_5^{jik}L^j - d_5^{ijk} - d_4^{ik})e^{-c_kt'}\right] u(t')$$

for $T \le t$ (14)

where

$$t' = t - T$$

$$L^i = e^{-a_iT}$$

$$N^k = e^{-c_kT}$$

---

[1] Parantheses around 2 are used to emphasize that $Y_{(2)}$ is the associated two-dimensional response.

[2] Note that if H_c has $m \neq n$ poles, then the index k runs from 1 to m in (13) and (14)

$$d_o^k = 1/c_k$$

$$d_1^{ik} = 1/(c_k - a_i) \qquad (15)$$

$$d_3^{ijk} = 1/(c_k - a_i - a_j)$$

$$d_4^{jk} = a_j/c_k(c_k - a_j)$$

$$d_5^{ijk} = a_j/(c_k - a_i)(c_k - a_i - a_j)$$

IV. IDENTIFICATION OF $H_2(s_1,s_2)$ USING PULSE INPUTS

In this section we deal with the central problem of the paper i.e., identification of $H_2(s_1,s_2)$ from the Volterra response $y_2(t)$. The problem is considered in two parts. First the estimation of poles of H_a and H_c is considered. Next, the residues are estimated.

A. Identification of Poles

A.1 Poles from Response over $0 \leq t \leq T$ (Segment 1)

For the symmetric quadratic subsystem the response to the square-pulse $p(t) = u(t) - u(t-T)$ was shown to be given by (14). Over the time interval $0 < t < T$ this response can be visualized as the unit-step (at t=0) response of a linear system shown in Fig. 5(a) where $a_{ij} \triangleq a_i + a_j$ and the residues P_i, Q_{ij} and R_i are defined according to the first part of (14).

Clearly, the pencil-of-functions method [7],[9] can be used to determine

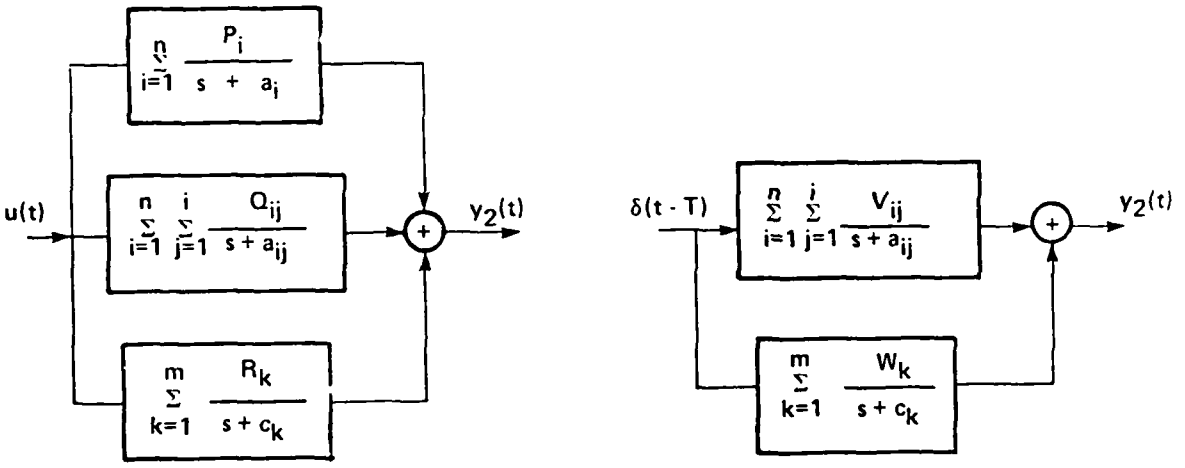
a_i $\qquad$ $i = 1, \ldots, n$

a_{ij} $\qquad$ $i=1,\ldots,n;$ $\qquad$ $j=1,\ldots,i$

c_k $\qquad$ $k = 1,\ldots,m$

Note that in general the total number of poles is

$$N = n + \frac{n(n+1)}{2} + m \qquad (16)$$



(a) Valid for $0 \leq t \leq T$

(b) Valid for $T \leq t$

Fig. 5 Equivalent linear system for quadratic response of $H_2(s_1,s_2)$. Note that $a_{ij} = a_i + a_j$

If the quadratic is completely symmetric, i.e., if $H_a = H_c$, then the total number of poles is again given by (16); however, the poles a_i occur with a multiplicity of 2.

A.2 Poles from Response over $T \leq t$ (Segment 2)

Over the time interval $T \leq t$ the quadratic response is given by the second part of (14), and can be visualized as the unit impulse (at t=T) response of a linear system as shown in Fig. 5(b)

Again the pencil-of-functions method can be applied to determine

$$a_{ij} \qquad i = 1,\ldots,n \qquad j = 1,\ldots,i$$

$$c_k \qquad k = 1,\ldots,m$$

Note that now the total number of poles is

$$N = \frac{n(n+1)}{2} + m \tag{17}$$

A.3 Remarks

1. The test engineer has a choice here between the use of the two segments of $y_2(t)$. At this time it appears that the use of Segment 2 is preferable, since the dimensionality of identification is lower in this case (as evidenced from a comparison of (17) with (16)), although quite extensive experimentation is necessary to make a definitive statement. For brevity of the paper we will assume that Segment 2 is used for identification, and will give details only for this case.

2. Since the identified values $\hat{a}_{ij}$ and $\hat{c}_k$ will generally not coincide with the true values, it is necessary to isolate the poles by visual inspection or by a suitable computer routine. For example, if these numbers for the case (n,m) = (2,1) are

2.1, 4.15, 9.8, 5.95

then

$$\hat{a}_{11} = 2.1$$

$$\hat{a}_{12} = 4.15$$

$$\hat{a}_{22} = 5.95$$

$$\hat{c}_1 = 9.8$$

and we may take

$$\hat{a}_1 = 1.05$$

$$\hat{a}_2 = 3.1$$

$$\hat{c}_1 = 9.8$$

B. Identification of Residues

After the poles have been determined, we can write

$$y_2(t) = \sum_{i=1}^{n} \sum_{j=1}^{i} \sum_{k=1}^{m} A_i A_j C_k \, f_{ijk}(t) \qquad (18a)$$

where $f_{ijk}(t)$ are defined in accordance with (14)(and are now known since they are functions of the poles and the pulse width T). Further, by defining

$$e_\nu = A_i A_j C_k \qquad \nu = i + j + k - 2 \qquad (18b)$$

$$g_\nu(\mu) = f_{ijk}(\mu\Delta) \qquad \mu = 0,\ldots,M-1 \qquad (18c)$$

we obtain the following set of simultaneous equations[3]

$$\underline{Y} = G\,\underline{E} \qquad (19)$$

where

$$\underline{Y} = [y_2(0)\ y_2(\Delta). \ldots .\ y_2((M-1)\Delta)]^T$$

$$G = [g_\nu(\mu)] \qquad M \times I \text{ matrix}$$

$$\underline{E} = [e_1 \ldots e_I]$$

Note that M denotes the number of time samples used in (18) and (19), and I denotes the number of unknown residue-products. If M is chosen equal

[3] If some of the poles are complex (occuring, in conjugate pairs), the associated residues are also complex. In such cases, it is possible to equate real (and imaginary) parts on both sides of the equation (19) to obtain real coefficient equations

to I then, we obtain the solution

$$\underline{E} = G^{-1} \underline{Y} \tag{20}$$

More generally, if M > I, then we obtain the least-squares solution [13]

$$\underline{E} = (G^T G)^{-1} G^T \underline{Y} \tag{21}$$

Finally, the equations (18b) can be solved straightforwardly to yield the residues A_i and C_k. Because of the homogeneity of the relations any one of the real residues (or the magnitude of a complex residue) can be taken as 1.

V. EXAMPLES

Two computer-generated examples will be presented. The first is a simple case with n=m=1 and is intended to clearly present the details of the procedure. The second is a more complicated case, representing a somewhat realistic channel. It has a linear transfer function characterized by a 6th order butterworth filter, and a quadratic transfer function with n=3 and m=1.

Example 1

Consider that the response $y_2(t)$ of a quadratic subsystem, with $H_a(s)=1/(s+1)$ and $H_c(s)=1/(s+0.5)$, to a square pulse of one second duration has been measured.

The second segment of the response ($y_2(t)$, $1 \leq t \leq 2$) is sampled with Δ=0.02 sec. and used for identification. To first determine the parameters of the equivalent linear system, a unit sample pulse input is assumed and q=0.8 is taken for the generation of the information signals. The Gram matrix of the correlation signals turns out to be

$$\begin{bmatrix} 0.88303D+01 & & & & \\ 0.41058D+02 & 0.19507D+03 & & & \\ 0.1884\,D+03 & 0.90828D+03 & 0.42871D+04 & & \\ -0.1\,\,\,77D+01 & -0.46824D+01 & -0.12867D+02 & 0.27778D+01 & \\ -0.9138\,D+01 & -0.33616D+02 & -0.11623D+03 & 0.77160D+01 & 0.35151D+02 \end{bmatrix}$$

The normalized cofactor square-roots ($\sqrt{D}_i / \sqrt{D}_1$) are

1.0000 0.35933 0.032129 0.22357 0.026502

By use of (6) and (7b) the negatives of the poles are computed to be[4]

[4] The reason to associate 1.9997 with a_{11} is that when segment 1 is analyzed, a pole at 0.9996 also shows up.

$$a_{11} = 1.9997$$

$$c_1 = 0.5002$$

so that

$$a_1 = 0.9998$$

$$c_1 = 0.5002$$

Note that the waveform-fit error in modeling the poles of the equivalent linear system was almost negligible (normalized mean-square error = 10^{-7}) pointing to the success of pulse testing approach.

Now, using these poles we find

$$y_2(t') = A_1A_1C_1\ (-0.2664375\ e^{-1.9997t'} + 0.4119195e^{-0.5002t'})$$

Using a single point for numerator computation, $y_2(t'=0) = 0.292924$, we obtain $A_1A_1C_1 = 2.013$, so that[5]

$$A_1 = 1$$

$$C_1 = 2.013$$

Example 2

Linear TF: Sixth order lowpass butterworth filter with cutoff $f_c = 10$ MHz

$$H_1(s) = \frac{6.1528908(10^{46})}{[s^6 + 2.4276(10^8)s^5 + 2.9467(10^{16})s^4 + 2.2676(10^{24})s^3 + 1.1633(10^{32})s^2 + 3.78358(10^{39})s + 6.1528908(10^{46})]}$$

$$= \frac{6.1528908(10^{10})}{[\lambda^6 + 242.76\lambda^5 + 29467\lambda^4 + 2267581\lambda^3 + 1.6133(10^8)\lambda^2 + 3.78358(10^9)\lambda + 6.1528908(10^{10})]}$$

[5] A better value $A_1A_1C_1 = 2.0007$ is obtained using 50 points and formula (21).

Where $\lambda = (10^{-6})s$ (units of Mrad./s).

Quadratic TF $$H_a(s)=H_c(s)= \frac{50.5(10^{13})}{s^2 + (10^7)s + 25.25\ (10^{14})}$$

$$= \frac{505}{\lambda^2 + 10\lambda + 2525}$$

The above 2-variable system was excited by a square pulse p(t) of duration T = 1 sec and the response $y^+(t)$ recorded. The system was next excited by the opposite polarity pulse -p(t) and the corresponding response $y^-(t)$ also recorded. From these responses we obtained the linear response $y_1(t)$ and the quadratic response $y_2(t)$ by use of (11).

The results of identification are given below:

Estimated Linear Transfer Function -

$$\hat{H}(\lambda) = \frac{15.2\ \lambda\ + 6.1523(10^{10})}{[\lambda^6 + 242.76\lambda^5 + 29467\lambda^4 + 2267593\lambda^3 + 1.6133(10^8)\lambda^2 + 3.784(10^9)\lambda + 6.15237(10^{10})\ ]}$$

Estimated Quadratic Transfer Function

$$\hat{H}_a(\lambda)= \hat{H}_c(\lambda)= \frac{0.003\ \lambda\ +\ 505.07}{\lambda^2 + 9.9998\lambda + 2524.8}$$

VI. USER GUIDE FOR THE COMPUTER PROGRAM 'IGRAM'

The computer program IGRAM has two functional parts. The first deals with the reading, or internal generation, of the response pair of the system. The second pertains to identification of the system. This second part also contains the noise correction facility which is useful when the data are corrupted by noise. On an optional basis the identified z-domain model can also be converted to the s-domain. Output from the program is in two forms: a) the printed output consisting of the identified model and the error of fit, and b) certain plot files which may be used for plotting purposes.

The program uses the following subroutines

BUILDA — Constructs the transformation matrix A to help implement (6).(see reference [7])

BUILDZ — Constructs the covariance matrix of the noise signals (assuming unit variance at the input of the processing filters)

CORUPT — Adds noise to the system response if the variable 'VAR' is non-zero. Note that this is normally used in simulation mode, i.e., when a known system function H(z) is used and a response pair is generated within the program.

ERROR — Calculates the percent mean square error and percent RMS error; the latter is defined as

$$100\sqrt{\frac{\sum_{k=0}^{K-1}(\hat{y}(k)-y(k))^2}{\sum_{k=0}^{K-1}y^2(k)}}$$

FILLV Generates the input waveform according to the input option selected. These waveforms range from a simple step to a rectangular pulse , and from a continuous sine wave to a sinusoidal burst. Also included are certain special waveforms such as a chirp and triangular pulse. (See page 22 for details.)

FIX Performs noise correction on the Gram matrix G to enable estimation of a more reliable system model. Estimated noise variance is *

$$\sigma^2 = \frac{1}{G^{-1} \odot W}$$

where W is the covariance matrix of noise vector. The estimated gram matrix is

$$F = G - \sigma^2 W$$

GKRDCT This routine finds the cofactors and/or the inverse of a square matrix. It also calculates the the denominator parameters through pencil-of-functions method.

GRAMI Performs the Pencil-Of-Functions technique for identification. It calls several subroutines, notably GKRDCT and BUILDA.

IZTOS Separates the numerator and denominator parameters and collects them into two vectors. Also it calls ZTOS for z domain to s domain conversion

POLCON Constructs the polynomial corresponding to a known set of roots.

*The matrix DOT product AΘB means $\Sigma \Sigma a_{ij} b_{ij}$

POLRT	Computes the real and complex roots of a real polynomial. Limited to a 36th (or lower) order polynomial.
PRCVEC	Prints a double precision complex vector in the form of a+jb, where a is the real part and b is the imaginary part.
PRMAT	Prints a double precision two dimensional array.
PRVEC	Prints a double precision one dimensional array.
RESPON	determines the response of a discrete time transfer function H(z) to an input sequence v(k).The coefficients of H(z) are in the vector GAMMA.
ZTOS	Converts the z-domain model H(z) to an equivalent s-domain model H(s).

INPUT CARDS

Card 1 Title card

Card 2 Subtitle card (containing list of variables on the next card)

Card 3 (12I5)
N= System order (for simulation purposes)
NPT= Number of points (generated by simulation or read)
IPLT= 0 no plots
= 1 plot on printer
= 2 fill-in plot files
ISIM Selects type of mode desired
= 0 lab data mode
= 1 simulation mode; i.e., response-pair is generated internally based on a given H(z)

= 2 mixed mode; output data is lab data. input is generated in the program

INPT Selects type of input desired for simulation (see page 22 for further details)

NPUL, NCY Parameters of specific inputs. used only if ISIM equals 0 or 2 (see page 22 for further explanation)

If ISIM = 0 use the following cards

Card 4 to Card 4+NPT/7	NPT output data points (7F10.0)
Next NPT/7 Cards	NPT input data points (7F10.0)

If ISIM = 1 use these cards

Card 4	Denominator of H(z) transfer function
Card 5	Numerator of H(z) transfer function

Both cards are (7F10.0)

If ISIM =2 use this arrangement

Card 4 to Card 4+NPT/7	NPT output data points (7F10.0)

Next card — Subtitle card

Next card — Subtitle card (containing name of variables on the next card)

Next card (6I5,F10.0, 6F5.0)

N= model order

NPT = Number of points (generated by simulation or read)

ISKIP = Skip interval in printer plots. For example, if ISKIP=5 every 5th point of the data array is plotted (thus reducing the size of the plot)

IREM and IDLY adjust for the type of numerator desired in the model, i.e.,

$$H(z) = \frac{b_0 + b_1 z^{-1} + \ldots\ldots + b_n z^{-n}}{1 + a_1 z^{-1} + \ldots\ldots + a_n z^{-n}}$$

IREM = 0 no effect

= 1 $b_n = 0$ is imposed

= m $b_{n-m+1} = \ldots b_m = 0$

IDLY = 0 no effect

= 1 numerator is multiplied by z^{-1}

= m numerator is multiplied by z^{-m}

An example:

If the model is desired to be of the form

$$H(z) = \frac{b_1 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

then N=2 and IREM and IDLY should be set to 1.

DELTA sampling interval

IBIAS = 1 enables bias extraction option

Q value of the processing filter pole

VAR variance of the noise added to the simulated response

DFAC A threshold used in the noise correction subroutine FIX

Next card (12I5)

IPR level of output printing
- = 0 minimal printing
- = 1 maximum printing

IFIX noise correction option
- = 0 no correction
- = 1 do noise correction

IPL1 and IPL2 plot options
- = 0 no plot files
- = 1 create plot data files

Further explanation of INPT. NTH, NCY. NPUL

INPT

Specifies desired input waveform

- = 0 zero input
- = 1 impulse located at k = 1
- = 2 step function (amplitude = 0.1 * NPUL)
- = 3 square pulse (width = NPUL)
- = 4 doublet (total width = NPUL)
- = 5 positive triangular pulse (width = NPUL)
- = 6 triangular pulse, positive and negative (width = NPUL)
- = 7 square wave (period = NPUL)
- = 8 square wave burst. followed by an exponentially decaying tail (width = NPUL, frequency=NCY/NPUL, time constant = NPT/10)
- = 9 exponential decay (time constant = NPUL)
- = 10 impulse train (period = NPUL)
- = 11 exponentially decaying sine oscillation (time constant = NPT/5, period = NPUL)
- = 12 exponentially decaying cosine oscillation (time constant = NPT/5, period = NPUL)
- = 13 random signal, sigma = 0.1 * NPUL (variance = sigma **2)

= 14 cosine burst (frequency = NCY/NPUL, width = NPUL)

= 15 sine burst (frequency = NCY/NPUL, width = NPUL)

= 16 cosine chirp (initial frequency = NCY/NPUL, width = NPUL)

= 17 sine chirp (initial frequency = NCY/NPUL, width = NPUL)

NTH

This causes the program to subsample the waveforms. That is, every nth point of the data (measured or simulated data) is used, the rest are discarded. If NTH=3, then every third point of the input and output data is used in the identification procedure.
This option is sometimes useful in nonlinear systems studies. More will be said about it elsewhere.

NCY

When an input waveform is a burst NCY defines the number of oscillations desired in the time length NPUL. When an input is a chirp it defines the initial number of cycles per time duration of NPUL.

NPUL

Any input that has a specific time duration (burst, chirp, pulse, impulse, square wave) will have NPUL equal to the duration of that waveform.

VII. USER GUIDE FOR THE COMPUTER PROGRAM 'STOZ'

STOZ is a s-domain to z-domain conversion program. Given an s-domain transfer function H(s), it finds an equivalent z-domain transfer function H(z) by one of several methods, e.g. impulse-invariant or step invariant transformation. Thus, given the continuous-time description of a linear system, it computes the equivalent discrete-time description.
The input arrays A and B are filled according to the differential equation

$$a(0)*y + a(1)*D(1,y) + ... + a(n)*D(n,y) = b(0)*u + b(1)*D(1,u) + ... + b(n)*D(n,u)$$

(where D(m,f) = the mth time Derivative of function f(t)), or the transfer function

$$H(s) = \frac{b_0 + b_1 s + ... + b_n s^n}{a_0 + a_1 s + ... + a_n s^n}$$

$$a_n = 1 \quad \text{(mandatory)}$$

STOZ returns arrays A and B containing the equivalent discrete-time description stored according to the difference equation

$$y(k) + a(1)*y(k-1) + ... + a(n)*y(k-n) = b(0)*u(k) - b(1)*u(k-1) - .... - b(n)*u(k-n)$$

or, the transfer function

$$H(z) = \frac{b_0 + b_1 z^{-1} + ... + b_n z^{-n}}{1 + a_1 z^{-1} + ... + a_n z^{-n}}$$

The poles of the continuous domain must be distinct and non-zero for the transformation to be valid.

The program uses the following subroutines

PCSTZ	Constructs the polynomial corresponding to a known set of roots; the kth root is omitted if k is not equal to zero.
POLRT	Computes the real and complex roots of a real polynomial. Limited to a 36th (or lower) order polynomial.
PRCVEC	Prints a double precision complex vector in the form of a+jb, where a is the real part and b is the imaginary part.
PRVEC	Prints a double precision one dimensional array.

Note that POLRT, PRCVEC and PRVEC are the same as in IGRAM, therefore their listing will be omitted in Appendix B.

DATA CARD SET PREPARATION

Card 1 Title card

Card 2 Subtitle card (containing list of variables on the next card)

Card 3 Contains the following variables (3I5,F10.0,I5)

N = order of system

N (maximum) = one less than the dimension subscript

IMTHD = 0 for the impulse invariant conversion
= 1 for the pulse invariant conversion
= 2 for the trapezoidal invariant conversion
= 3 for the logrithmic transform conversion

IPOLZ = 1 poles and zeroes are given as complex numbers (real and imaginary); negatives of poles and zeros are read; ie, for a pole of (s+2), input +2.0 +0.0 (2F10.0 per pole, 4 poles per card)

= 0 denominator and numerator are read in polynomial form. Coefficients ordered from low to high degree, denominator information always read first. Highest order denominator coeff must be 1.0

Note: When pole-zero data is entered a gain card must follow the last pole card. If there are no zeros, use blank cards in the normal zeros positions.

DELTA = Sampling interval in seconds

IZERO = 1 zeros of z-domain function are printed out

If IPOLZ = 0

Card 4 (8F10.0) Denominator polynomial coefficients

Card 5 (8F10.0) Numerator polynomial coefficients

If IPOLZ = 1

Card 4 (8F10.0) Denominator poles (complex values)

Card 5 (8F10.0) Numerator zeros (complex values)

Card 6 (F10.0) Numerator gain value

VIII CONCLUSIONS

Communication channels, both microwave and satellite, generally exhibit nonlinear characteristics. When frequency division multiplexing is employed, it is important to evaluate the intermodulation distortion using Volterra models. We have presented a modeling procedure which employs pulse testing and performs pencil-of-functions analysis. The advantages of the method are: a) the use of a commonly available signal source; b) the rapidity of the test (only two transient tests are required), and c) the resulting model is in the compact transfer-function(s) form. The success of the procedure was demonstrated by computer generated examples. In actual application, however, several points of caution must be exercised. For example, in the reverse polarity test, the amplitude and width of thepulse must be closely duplicated. The development in thepaper was carried out under the assumption of a baseband channel. However, with suitable modifications, such as the use of a carrier frequency burst instead of a square pulse, it should apply to bandpass channels also. Mathematical details of this important extension remain to be examined.

APPENDIX A

```
1
C           PROGRAM IGRAM       REV 10/82 V K JAIN
C
C
C     ****************************************************************
C     *                                                              *
C     *                    PROGRAM NAME 'IGRAM'                      *
C     *                       REVISED 10/82                          *
C     *                  ____________________                        *
C     *                                                              *
C     * NETWORK TRANSFER-FUNCTION IDENTIFICATION PROGRAM UTILIZING   *
C     *                PENCIL-OF-FUNCTIONS METHOD                    *
C     *                ___________________________                   *
C     *                                                              *
C     *                    DEVELOPED  AT THE                         *
C     *                  UNIVERSITY OF SOUTH FLORIDA                 *
C     *                                                              *
C     *                                                              *
C     *                                                              *
C     *                  ___________________________                 *
C     *                                                              *
C     *                       FOR ASSISTANCE                         *
C     *                          CONTACT                             *
C     *                       DR. V. K. JAIN                         *
C     *                        813/974-2581                          *
C     ****************************************************************
C
C
C
C
      DIMENSION X(1024),V(1024),XORG(1024),VORG(1024),XREC(1024)
      DIMENSION DATA(1024,2),DATA2(1024,2),BUFF(3072)
      DIMENSION G(20,20),Z(20,20),GAMMA(20),XLAMDA(20),COEFF(20)
      DIMENSION GDUM(20,20),GEST(20,20),GGG(20,20)
      REAL*8 GDUM,GEST,GGG
      DIMENSION HPULSE(64)
      DIMENSION TITLE(80)
      INTEGER LABEL(20),CCHK
      INTEGER ENDFIL / '//   ' /
      REAL*8 G,Z,GAMMA,XLAMDA,COEFF
      REAL*8 DELTA,Q,QSAV,DELSAV,AVGQ,AVGW,SUMV2,XSAV
      EQUIVALENCE (Z(1,1),BUFF(1)),(G(1,1),BUFF(1501))
      EQUIVALENCE (DATA(1,1),X(1)),(DATA(1,2),VORG(1))
      EQUIVALENCE (XORG(1),DATA2(1,1)),(XREC(1),DATA2(1,2))
      COMMON /GKRD/IGKR
      COMMON /DA1/NN,IFIX
      COMMON /DA2/IPR
      COMMON /DA3/DFAC
      COMMON /BIAS/IBIAS
      COMMON /DELAY/ IDLY
      DATA CCHK/4H   _/
      COMMON /WAVE/NCY
C
C
C
      REWIND 9
      IGKR=1
      RDEL=0.002
      MAX=20
      MAXPL=1024
      NPT=MAXPL
C
C     WRITE(6,2)
C     WRITE(6,1022)
```

```
      READ(5,1021)TITLE
      WRITE(6,1021)(TITLE(I),I=1,70)
      READ(5,1021)TITLE
      IF(IPR.GE.2)WRITE(6,1023)
C     ..............................................................
C
C     READ LABORATORY RECORDED RESPONSE-PAIR,
C     OR GENERATE IT BY SIMULATION
C
C
4320  READ(5,1001)N,MP1,IPLT,ISIM,INPT,NPUL,NTH,NCY
      ISIMIN=0
      IF(ISIM.EQ.2)ISIMIN=1
      IF(ISIM.EQ.2)ISIM=0
      IF(NCY.EQ.0)NCY=2
      IF(NTH.EQ.0)NTH=1
      IF(ISIM.NE.0)GO TO 3306
      READ(5,6995)(XORG(K),K=1,MP1)
C     WRITE(6,6996)(XORG(K),K=1,MP1)
      IF(ISIMIN.EQ.0)READ(5,6995)(VORG(K),K=1,MP1)
      IF(ISIMIN.EQ.1)CALL FILLV(VORG,MP1,INPT,NPUL)
C     WRITE(6,3)
C     WRITE(6,6996)(VORG(K),K=1,MP1)
      GO TO 3308
3306  CONTINUE
      NPNP2=N+N+2
      MP1=N+1
      NP2=N+2
      CALL FILLV(VORG,MP1,INPT,NPUL)
      READ(5,701)(COEFF(I),I=1,MP1)
      READ(5,701)(COEFF(I),I=NP2,NPNP2)
      CALL PRVEC(COEFF,NPNP2)
701   FORMAT(7F10.0)
      CALL RESPON(XORG,VORG,N,COEFF,XLAMDA,MP1)
      KK=0
      DO 703 K=1,MP1,NTH
      KK=KK+1
      VORG(KK)=VORG(K)
703   XORG(KK)=XORG(K)
      MP1=(MP1+NTH-1)/NTH
      MP2=MP1+1
      DO 705 K=MP2,MAXPL
      VORG(K)=0.0
705   XORG(K)=0.0
3308  CONTINUE
C
4321  READ(5,1033,END=1234)(LABEL(I),I=1,20)
      IF (LABEL(1).EQ.ENDFIL) GO TO 1234
      READ(5,1021)TITLE
      READ(5,1)N,MP1,ISKIP,IREM,IBIAS,IDLY,DELTA,
     +QSAV,VAR,DFAC
      READ(5,1021)TITLE
      READ(5,1001)IPR,IFIX,IPL1,IPL2
      IF(N.EQ.10000)GO TO 4320
      IF(IPR.GE.1)WRITE(6,1000)N,MP1,DELTA,IREM,IBIAS,IDLY
C
      IGKR=1
      IZTS=2
      IF(QSAV.EQ.0.0)QSAV=1.0D00
      Q=QSAV
      NM1=N-1
      NP1=N+1
      NP2=N+2
      NPNP1=N+N+1
      NPNP2=N+N+2
      NN=N-IREM
```

```
C
C     ADD NOISE (OF GIVEN VARIANCE) TO OUTPUT, IF DESIRED
C
C
      VMAX=0.0
      XMAX=0.0
      XXSUM=0.0
23    DO24I=1,MP1
      IF(ABS(VORG(I)).GT.VMAX)VMAX=ABS(VORG(I))
      IF(ABS(XORG(I)).GT.XMAX)XMAX=ABS(XORG(I))
      V(I)=VORG(I)
      X(I)=XORG(I)
24    XXSUM=XXSUM+XORG(I)*XORG(I)
      INOISE=0
      IF(VAR.GE.1.E-10)INOISE=1
      IF(INOISE.EQ.1)SNR=XXSUM/(VAR*NPT)
      IF(INOISE.EQ.1)SNRDB=10.0*ALOG10(SNR)
      IF(INOISE.EQ.1)WRITE(6,1044)SNR,SNRDB
      IF(INOISE.EQ.0)WRITE(6,1045)
      XV=0.5*XMAX/VMAX
      DO 22 K=1,MP1
C     VORG(K)=XV*VORG(K)
22    CONTINUE
      IF(VAR.GE.1.0E-9)CALL CORUPT(XORG,X,VAR,MP1,MAXPL)
C
C     ADD BIAS TO DATA IF IBIAS.NE.0
C     (SET BIAS1 AND BIAS2 CONSTANTS TO ARTIFICIALLY ADD BIAS)
      IF(IBIAS.EQ.0)GO TO 6611
      BIAS1=0.0
      BIAS2=0.0
      DO 10 I=1,MP1
      V(I)=V(I)+BIAS1
10    X(I)=X(I)+BIAS2
6611  CONTINUE
C     IF(IPR.GE.1.AND.IPLT.EQ.0)WRITE(6,6993)
C     IF(IPR.GE.1.AND.IPLT.EQ.0)WRITE(6,6994)(X(K),K=1,MP1)
C
      IF(IPLT.EQ.0)GO TO 6633
C     SCALE INPUT WAVEFORM FOR REASONS OF PLOTTING CONVENIENCE
      VSCAL=1.0
      DO 807 K=1,MP1
807   VORG(K)=VORG(K)*VSCAL
      IF(ABS(VSCAL-1.0).GE.1.E-6)WRITE(6,1047)VSCAL
      IF(IPR.GE.1)WRITE(6,1003)
      IF(IPLT.EQ.1.OR.IPLT.EQ.3)
     +CALL PLOTIT(DATA ,2,MP1,1,MP1,ISKIP,MAXPL,1,1.0)
      IF(IPLT.LE.1.OR.IPL1.EQ.0)GO TO 6633
C     LABEL(18)=CCHK
6633  CONTINUE
C     ..............................................
C
C     PERFORM IDENTIFICATION FROM INPUT-OUTPUT PAIR
C     THEN PREPARE FILE FOR PLOTTING. FILE WILL CONTAIN
C     (NOISY RESPONSE, INPUT, ORIGINAL RESPONSE, MODEL RESPONSE)
C
      CALL GRAMI(X,V,MP1,N,DELTA,Q,IZTS,GAMMA,XLAMDA,G,Z,MAX,IREM,IDLY)
      CALL ERROR(XREC,VORG,GAMMA,MP1,N,XLAMDA,XORG)
      IF(IPLT.EQ.0) GO TO 6544
      IF(IPR.GE.1)WRITE(6,66)
      IF(IPR.GE.1)WRITE(6,1004)
      IF(IPLT.EQ.1.OR.IPLT.EQ.3)
     +CALL PLOTIT(DATA2,2,MP1,1,MP1,ISKIP,MAXPL,1,1.0)
      IF(IPLT.EQ.1.OR.IPL2.EQ.0)GO TO 6544
C     LABEL(18)=CCHK
6998  FORMAT(2X,'HERE I AM IPLT=',I4)
      WRITE(9,6997)((DATA(K,1),DATA(K,2),
```

```
     +DATA2(K,1),DATA2(K,2)),K=1,MP1)
6544  CONTINUE
C
C     WRITE(6,2)
C     WRITE(6,1022)
100   GO TO 4321
1234  CONTINUE
C     ..................................................
C
C
1     FORMAT(6I5,F10.0,6F5.0)
2        FORMAT('1')
3     FORMAT(/)
66    FORMAT(//,1X,'TRUE RESPONSE versus RECONSTRUCTED RESPONSE',//)
1000  FORMAT(1H1,50X,'STARTING SIMULATION',/20X,'SYSTEM ORDER = ',I5,
     1/,20X,'M + 1 = ',I5,///,20X,'SAMPLING INTERVAL = ',F10.6,//,20X,
     2'IREM = ',I5,/,20X,'IBIAS = ',I5,/,20X,'IDLY= ',I5,/)
1001  FORMAT(12I5)
1003  FORMAT(//,5X,'INPUT (+) AND OUTPUT (*) OF THE PLANT',/)
1004  FORMAT(//,5X,' OUTPUT (*) AND RECONSTRUCTION (+) ',/)
1021  FORMAT(80A1)
1022  FORMAT(/////////////////////////////////////////////////////////)
1023  FORMAT(/,1X,'**************************************************',
     1 '**********************************')
1033  FORMAT(20A4)
1044  FORMAT(2X,'SNR  =',F12.5,'       SNRDB=',F10.2,/)
1045  FORMAT(2X,'SNR  =INFINITY (NO NOISE)')
1047  FORMAT(2X,'INPUT WAS SCALED,THEREFORE DIVIDE NUM BY',/,
     +5X,F10.4)
6994  FORMAT(2X,10F10.4)
6993  FORMAT(2X,'RESPONSE DATA')
6995  FORMAT(7F10.0)
6996  FORMAT(2X,10F8.0)
6997  FORMAT(6(G11.4,1X))
C     CALL PICSIZ(0.0,0.0)
998   CONTINUE
      END FILE 9
      STOP
C
C        DEFINITION OF PARAMETERS USED IN THE SIMULATION OF A
C        LINEAR DYNAMIC SYSTEM
C
C        X IS THE 3ORRUPTED OUTPUT SEQUENCE
C        V IS THE CORRUPTED INPUT SEQUENCE
C        GAMMA IS THE COEFFICIENT VECTOR
C
C        MAX = ACTUAL DIMENSION SIZE OF 2-DIM ARRAYS IN THE DIMENSION
C        STATEMENT
C        N = ORDER OF SYSTEM
C        THE MAXIMUM VALUE OF N IS MAX/2-1
C        MP1 = M+1, THE TOTAL NUMBER OF SAMPLED POINTS IN EACH SEQUENCE
C
C        RHO = EXPECTATION( W(K)*Q(K) )
C
C        DELTA IS THE SAMPLING INTERVAL
C
C        IGRM = 1 GRAM IS PERFORMED
C
C        IPLT=0  NO  PLOTS
C        IPLT=1  PLOTS ONLY WITH PRINTER
C
C        IBIAS =0 NO BIAS IS ASSUMED PRESENT ON INPUT-OUTPUT DATA
C        IBIAS =1 SMALL VALUES OF INPUT-OUTPUT BIAS ARE ASSUMED PRESENT
C                 ON THE DATA.
C
      END
```

```
      SUBROUTINE BUILDA(A,Q,DEL,N,MAX)
      REAL*8 A(MAX,1),Q(1),DEL(1),PROD
      NP1=N+1
      NPNP2=N+N+2
      A(1,NP1)=1.0D00
      PROD=1.0D00
      DO312K=1,N
      I=NP1-K
      PROD=PROD/DEL(I)
      A(1,I)=PROD
  312 A(K+1,NP1)=0.0D00
      DO313I=2,NP1
      DO313K=1,N
      J=NP1-K
  313 A(I,J)=(A(I,J+1)-Q(J)*A(I-1,J+1))/DEL(J)
      DO314I=1,NP1
      DO314J=1,NP1
      A(I,J+NP1)=0.0D00
      A(I+NP1,J)=0.0D00
  314 A(I+NP1,J+NP1)=A(I,J)
C     WRITE(6,1005)
 1005 FORMAT(1X,'A-MATRIX')
C     CALL PRMAT(A,NPNP2,NPNP2,MAX)
      RETURN
      END
      SUBROUTINE BUILDZ(Z,R,QI,NP1,NPT,NDIM)
C     ---------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Z(NDIM,1),R(1)
      COMMON /DA2/IPR
      Q=QI
      N=NP1-1
      NPNP2=NP1+NP1
      R(1)=1.0
      DO 1 I=1,NPNP2
      IF(I.GE.2)R(I)=R(I-1)
      DO 1 J=1,NPNP2
    1 Z(I,J)=0.0
      DO 2 K=1,NPT
      NPTK=NPT+1-K
      DO 3 J=1,NP1
      DO 3 I=J,NP1
    3 Z(I,J)=Z(I,J)+R(I)*R(J)*NPTK
      R(1)=0.0
      DO 4 I=1,N
    4 R(I+1)=Q*R(I+1)+R(I)
    2 CONTINUE
      DO 174 I=1,NPNP2
      DO 168 J=I,NPNP2
  168 Z(I,J)=Z(J,I)
      IF(IPR.GE.2)WRITE(6,220)(Z(I,J),J=1,NPNP2)
  174 CONTINUE
  220 FORMAT(2X,5(2X,G13.6))
      RETURN
      END
      SUBROUTINE CORUPT(F,X,VAR,NPT,NDIM)
C
C     CALLS GGNML (VIA IMSL)
C     ADDS NOISE
C     ----------------------------------------------------
      DIMENSION F(1),X(1),R(1024)
      COMMON /DA2/IPR
      DOUBLE PRECISION DT,DSEED
      FBAR=0.
      EBAR=0.
      FESUM=0.
```

```
      EESUM=0.
      IF(IPR.GE.1)WRITE(6,489)VAR
C
C     GENERATE STANDARD NORMAL RV, THEN CORRECT STD DEV
C
      SIGMA=SQRT(VAR)
      DSEED=123457.D0
      CALL GGNML(DSEED,NPT,R)
      DO 26 K=1,NPT
      R(K)=SIGMA*R(K)
26    X(K)=F(K)+R(K)
C
      DO 211 K=1,NPT
      FBAR=FBAR+F(K)
      EBAR=EBAR+R(K)
      EESUM=EESUM+R(K)*R(K)
      FESUM=FESUM+2.0*F(K)*R(K)
211   CONTINUE
      FBAR=FBAR/NPT
      EBAR=EBAR/NPT
      IF(IPR.GE.1)WRITE(6,482)FBAR,EBAR,FESUM,EESUM
      IF(IPR.LE.2) GO TO 411
      WRITE(6,8)
      WRITE(6,110)(X(K),K=1,NPT)
      WRITE(6,18)
      WRITE(6,115)(R(K),K=1,NPT)
      WRITE(6,1)
411   CONTINUE
999   CONTINUE
C     FORMAT  STATEMENTS
C
8     FORMAT(10X,' CORRUPTED RESPONSE')
18    FORMAT(10X,' ADDITIVE NOISE')
210   FORMAT(2X,5(2X,G14.7))
110   FORMAT(20(1X,F5.2))
115   FORMAT(1X,20(1X,F5.3))
482   FORMAT(2X,5HFBAR=,E11.4,6H EBAR=,E11.4,5H FE2=,E11.4,4H EE=,E11.4)
489   FORMAT(2X,'VARIANCE OF NOISE=',E12.4)
1     FORMAT(/)
      RETURN
      END
      SUBROUTINE ERROR(XREC,V,GAMMA,MP1,N,XLAMDA,XORG,FDBACK,IDLY)
      DIMENSION XREC(1),V(1),XORG(1)
      DIMENSION  VVV(20)
      REAL*8 GAMMA(1),XLAMDA(1),AVGW,SUMV2,AVGQ
      INTEGER FDBACK
      NPNP2=N+N+2
      CALL RESPON(XREC,V,N,GAMMA,XLAMDA,MP1,FDBACK)
      AVGW=0.0D00
      SUMV2=0.0D0
      DO26I=1,MP1
      SUMV2=SUMV2+XORG(I)*XORG(I)
      AVGQ=XORG(I)-XREC(I)
26    AVGW=AVGW+AVGQ*AVGQ
      AVGW=AVGW/SUMV2
      AVGQ=DSQRT(AVGW)
      AVGW=100.0*AVGW
      AVGQ=100.0*AVGQ
      WRITE(6,27)SUMV2,AVGW,AVGQ
C27    FORMAT(1X,'PER CENT MEAN POWER ERROR OF RECONSTRUCTION',F8.3,///,
C     11X,'PER CENT OF SQUARE ROOT OF POWER ERROR OF RECONSTRUCT.',F8.3)
27    FORMAT(1X,'SS RESPONSE=',E11.4,'   % POWER ERROR=',E11.4,/,
     +'                          %  RMS  ERROR=',E11.4)
      RETURN
      END
      SUBROUTINE FILLV(V,NPT,INPUT,NPUL)
```

```
C     FILLS THE ARRAY FOR INPUT ACCORDING TO INPUT OPTION DESIGNATED
C     INPUT = 0: ZERO INPUT
C             1: IMPULSE AT K=1
C             2: STEP FN.
C             3: SQPULSE;   4: DOUBLET,    PUL WIDTH=NPUL
C             5: TR+( );  6: TR+-( ),     PUL WIDTH=NPUL
C             7: SQUARE WAVE,    PERIOD=NPUL
C             8: SQ.W BURST(F=NCY/NPUL,WIDTH=NPUL),EXP(TC=NPT/10)
C             9: EXP (T.C.=NPUL)
C            10: IMP. TRAIN (P=NPUL)
C            11: COS OSC( )  12: SIN OSC T.C.=NPT/5, P=NPUL
C            13: RANDOM. SIGMA=.1*NPUL
C            14: COS BURST;  15: SIN B.  F=NCY/NPUL,  WIDTH=NPUL
C            16: COS CHIRP   17: SIN C.  FINIT=NCY/NPUL, WIDTH=NPUL
      DIMENSION V(1)
      COMMON /WAVE/NCY
      DOUBLE PRECISION DSEED
      PI2=6.28318530
      DO 90 I=1,NPT
90    V(I)=0.0
      IF(INPUT.EQ.0) GO TO 999
      GO TO (1,2,3,3,5,5,7,8,9,10,11,12,13,14,15,16,17),INPUT
1     V(1)=1.0
      DO 101 I=2,NPT
101   V(I)=0.0
      GO TO 999
2     CONTINUE
      IF(NPUL.EQ.0)NPUL=10
      AMP=0.1*NPUL
      DO 102 I=1,NPT
102   V(I)=AMP
      GO TO 999
3     CONTINUE
      N1=NPUL/2+1
      DO 103 I=1,NPUL
      V(I)=1.0
103   IF(INPUT.EQ.4.AND.I.GE.N1)V(I)=-1.0
      GO TO 999
5     CONTINUE
      IF(INPUT.EQ.6)NPUL=NPUL/2
      N1=NPUL/2+1
      N2=NPUL+1
      IF(INPUT.EQ.6)N2=NPUL+NPUL/2+1
      N3=NPUL*2+1
      NPUL2=NPUL/2
      DO 150 I=1,N1
150   V(I)=FLOAT(I-1)/NPUL2
      DO 151 I=N1,N2
151   V(I)= 2.0-FLOAT(I-1)/NPUL2
      IF(INPUT.EQ.5)GO TO 999
      DO 152 I=N2,N3
152   V(I)=-4.0+FLOAT(I-1)/NPUL2
      GO TO 999
7     DO 107 I=1,NPT
      V(I)=1.0
      TPUL=I/NPUL
      IF(I-TPUL*NPUL.GE.NPUL/2) V(I)=-1.0
107   CONTINUE
      GO TO 999
8     NP=NPUL/NCY
      NPD2=NP/2
      DO 55 I=1,NPUL
      V(I)=1.0
      IL=MOD(I,NP)
      IF(IL.GT.NPD2)V(I)=-1.0
55    CONTINUE
```

```
      SN=SIGN(1.0,V(NPUL))
      TC=0.2*(NPT-NPUL)
      DO108I=NPUL,NPT
      ARG1=FLOAT(NPUL-I)/TC
108   V(I)=SN*EXP(ARG1)
      GO TO 999
9     DO 109 I=1,NPT
      ARG2=-FLOAT(I)/FLOAT(NPUL)
109   V(I)=EXP(ARG2)
      GO TO 999
10    DO 110 I=1,NPT,NPUL
110   V(I)=1.0
      GO TO 999
11    DO 111 I=1,NPT
      ARG4=-5.0*FLOAT(I)/FLOAT(NPT)
      ARG5=6.2832*FLOAT(I)/FLOAT(NPUL)
      V(I)=EXP(ARG4)*COS(ARG5)
111   CONTINUE
      GO TO 999
12    DO 112 I=1,NPT
C     ARG3=-FLOAT(I)/FLOAT(NPUL)
      ARG4=-5.0*FLOAT(I)/FLOAT(NPT)
      ARG5=6.2832*FLOAT(I)/FLOAT(NPUL)
      V(I)=EXP(ARG4)*SIN(ARG5)
C     V(I)=EXP(ARG3)+EXP(ARG4)*SIN(ARG5)
112   CONTINUE
      GO TO 999
13    DSEED=789457.D0
      CALL GGNML(DSEED,NPT,V)
      IF(NPUL.EQ.0)NPUL=10
      DO 113 I=1,NPT
113   V(I)=0.1*V(I)*NPUL
      GO TO 999
14    CONTINUE
      NP=NPUL/NCY
      DO 114 I=1,NPUL
114   V(I)=COS(FLOAT(I-1)*PI2/NP)
      GO TO 999
15    CONTINUE
      NP=NPUL/NCY
      DO 115 I=1,NPUL
115   V(I)=SIN(FLOAT(I-1)*PI2/NP)
      GO TO 999
16    CONTINUE
      WO=PI2*NCY/NPUL
      DO 116 I=1,NPUL
      TI2=2.0*(I-1)/NPUL
      WW=(1.0+TI2**2)*WO
116   V(I)=SIN(FLOAT(I-1)*WW)
      GO TO 999
17    CONTINUE
999   CONTINUE
      RETURN
      END
      SUBROUTINE FIX(G,P,C,D,X,N,NC,SIG,NDIM,IFIX)
C     ------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
C
C     ESTIMATE NOISE INTENSITY SIG (ASSUME WHITE NOISE)
C     CORRECT NOISY MATRIX= C
C     P DENOTES NOISE MATRIX FOR UNIT NOISE
C     NC IS THE NONZERO SUBMATRIX OF P =COV OF NOISE
C
      DIMENSION G(NDIM,1),P(NDIM,1.,C(NDIM,1),D(NDIM,1),X(1)
      DOUBLE PRECISION SUMDET,DT
      COMMON /DA2/IPR
```

```
      COMMON /DA3/DFAC
      SI=SIG
      IF(IFIX.EQ.0)GO TO 51
      GDP=G(1,1)/P(1,1)
      JCT=0
      SIG=0.0
210   FORMAT(1X,5G12.5)
      DO 211 I=1,N
211   IF(IPR.GE.2)WRITE(6,210)(G(I,J),J=1,N)
3     JCT=JCT+1
      SUMDET=0.0
      CALL GKRDCT(G,D,GDET,X,N,NDIM,0)
      DO 212 I=1,N
212   IF(IPR.GE.2)WRITE(6,210)(D(I,J),J=1,N)
      IF(JCT.EQ.1)DETG=GDET
      DO 7 I=1,NC
      DO 7 J=1,NC
7     SUMDET=SUMDET+D(I,J)*P(I,J)
      IF(SUMDET.LT.0.0.AND.IFIX.NE.2)GO TO 11
      SI=1.0D0/SUMDET
      WRITE(6,32)JCT,ICT,GDET,SUMDET,SI
      IF(SI/GDP.GT.0.1)WRITE(6,31)
      ICT=0
51    CONTINUE
      DO 9 I=1,N
      DO 9 J=1,N
      C(I,J)=G(I,J)-SI*P(I,J)
9     CONTINUE
      IF(IFIX.EQ.0)GO TO 11
      CALL GKRDCT(C,D,CDET,X,N,NDIM,0)
      IF(CDET.LT.0.0.OR.CDET.GT.GDET)ICT=ICT+1
      IF(ICT.GT.5)GO TO 10
      IF(ICT.GT.0)SI=SI/2.0
      IF(ICT.GT.0)GO TO 51
      IF(JCT.GE.5)GO TO 11
10    THR=DETG*DFAC
      IF(JCT.EQ.1)WRITE(6,33)DETG,DFAC,THR
      SIG=SIG+SI
      IF(CDET.LE.THR)GO TO 22
      DO 23 II=1,N
      DO 23 JJ=1,N
23    G(II,JJ)=C(II,JJ)
22    CONTINUE
      WRITE(6,34)JCT,ICT,GDET,SI,CDET
      IF(CDET.GT.THR)GO TO 3
31    FORMAT(2X,'NOISE VAR EXCESSIVE, SIG/GDP.GT.0.1')
32    FORMAT(1X,'J,I GDET,SUMDET,SI:',2I2,4E11.2)
33    FORMAT(1X,'GDET,DFAC,THR:',6E11.2)
34    FORMAT(1X,'J,I  SI,CDET',2I2,4E11.2)
11    CONTINUE
      DO 25 I=1,N
      IF(IPR.GE.2)WRITE(6,210)(C(I,J),J=1,N)
      DO 25 J=1,N
25    G(I,J)=C(I,J)
      RETURN
      END
      SUBROUTINE GKRDCT(X,Y,DET,XLAMDA,N,MAX,IOPT)
      REAL*8 X(MAX,1),Y(MAX,1),A,B,C,D,E,DET,XLAMDA(1)
      REAL*8 SCAL(20),RSC(20),DT,SC,AD,BD
      INTEGER NUM(2,20)
      COMMON /DA2/IPR
      COMMON /GKRD/IGKR
      COMMON /GKRD2/DT
C     IGKR=0  USE IS MADE OF THE FIRST ROW OF ADJOINT
C          1   DIAGONAL(NEGATIVE ENTRIES SET TO ZERO)
C          2   ABSOLUTE VALUE OF DIAGONAL
```

```
C
C         IOPT = 0: RETURN X INVERSE IN Y
C                1: Calculate PARAMETER VECTOR
C
      IF(N.NE.1)GO TO 3
      Y(1,1)=1.0/X(1,1)
      DET=X(1,1)
      GO TO 61
3     CONTINUE
C
C     SCALE
C
      ISCL=1
      IF(ISCL.EQ.0)WRITE(6,804)
804   FORMAT(2X,'***WARNING!! SCALING IN GKRDCT DISABLED**')
      DO 11 I=1,N
      SCAL(I)=1.0D0
      IF(ISCL.GE.1.AND.X(I,I).GT.0.1E-20)SCAL(I)=DSQRT(X(I,I))
11     RSC(I)=1.0/SCAL(I)
      DO 6 I=1,N
      DO 6 J=1,N
6     Y(J,I)=X(J,I)*RSC(I)*RSC(J)
C     CALL PRMAT(Y,N,N,MAX)
      A=1.0D0
      DO 43 I=1,N
      B=0.0D0
      L=I
      M=I
C
C        FIND LARGEST ENTRY A(L,M) IN LOWER DIAGONAL SUBMATRIX
C
      DO 18 J=I,N
      DO 18 K=I,N
      IF(DABS(Y(K,J)).LE.B)GO TO 18
      B=DABS(Y(K,J))
      L=K
      M=J
18    CONTINUE
C
C        INTERCHANGE ROWS
C
      IF(L.EQ.I)GO TO 24
      DO 23 J=1,N
      C=Y(L,J)
      Y(L,J)=Y(I,J)
23    Y(I,J)=C
C
C        INTERCHANGE COLUMNS
C
24    IF(M.EQ.I)GO TO 29
      DO 28 J=1,N
      C=Y(J,M)
      Y(J,M)=Y(J,I)
28    Y(J,I)=C
C
C        BEGIN SWEEP COLUMNS TO THE RIGHT
C        ARRAYS NUM(1,.) ,NUM(2,.) KEEP RECORD
C        OF ROW AND COLUMN INTERCHANGES
C
29    NUM(1,I)=L
      NUM(2,I)=M
      B=Y(I,I)
      Y(I,I)=A
      DO 42 J=1,N
      IF(J.EQ.I)GO TO 42
      C=-Y(I,J)
```

```
      Y(I,J)=0.0D0
      DO 41 K=1,N
      D=Y(K,I)*C
      E=Y(K,J)*B+D
C     IF(DABS(E).LT.1.0D-10*DABS(D))E=0.0D0
41    Y(K,J)=E/A
42    CONTINUE
43    A=B
C
C        RESTORE COLUMNS
C
      DO 58 I=2,N
      J=N+1-I
      K=NUM(2,J)
      IF(K.EQ.J)GO TO 52
      DO 51 L=1,N
      C=Y(K,L)
      Y(K,L)=Y(J,L)
51    Y(J,L)=C
52    K=NUM(1,J)
C
C        RESTORE ROWS
C
      IF(K.EQ.J)GO TO 58
      DO 57 L=1,N
      C=Y(L,K)
      Y(L,K)=Y(L,J)
57    Y(L,J)=C
58    CONTINUE
      DET=A
      DO 59 I=1,N
59    DET=DET*SCAL(I)*SCAL(I)
      IF(IPR.EQ.3)WRITE(6,337)DET,A,(RSC(I),I=1,N)
      IF(IOPT.EQ.1)GO TO 61
      DO 60 I=1,N
      DO 60 J=1,N
60    Y(I,J)=Y(I,J)*RSC(I)*RSC(J)/A
61    CONTINUE
C     WRITE(6,703)
703   FORMAT(2X,'GKRDCT:  PROCESSED MATRIX Y')
C     IF(IPR.EQ.1)CALL PRMAT(Y,N,N,MAX)
      IF(IOPT.NE.1)GO TO 1000
      IF(Y(1,1).LT.0.0D0) GO TO 1000
C
      DO 806 I=1,N
806   IF(IPR.GE.3)WRITE(6,803)(Y(I,J),J=1,N)
      SC=1.0D0
      DO 200 I=2,N
      SC=SC*DT
      RSC(I)=RSC(I)/RSC(1)
      A=Y(I,I)
803   FORMAT(2X,6G12.5)
      XLAMDA(I)=RSC(I)*DSQRT(A/Y(1,1))*DSIGN(1.0D0,Y(1,I))
200   CONTINUE
      XLAMDA(1)=1.0D0
      IF(IPR.EQ.1)WRITE(6,106)(XLAMDA(IP),IP=1,N)
106   FORMAT(5X,'SYNTHETIC PARAMETER VECTOR',10G12.5)
1000  CONTINUE
337   FORMAT(1X,'DET,A,RSC(I): ',7E11.2)
339   FORMAT(1X,'ERROR.  RATIO OF II=',I1,' JJ=',I1,'  COFAC NEG')
      RETURN
      END
      SUBROUTINE GRAMI(X,V,MP1,N,DELTA,QSAV,KOPT,GAMMA,XLAMDA,G,Z,MAX,
     1IREM,IDLY)
C     THIS SUBROUTINE PERFORMS THE GRAMI TECHNIQUE
C
```

```
      DIMENSION X(1),V(1),G(MAX,1),Z(MAX,1),GAMMA(1),XLAMDA(1),Q(20),
     1DEL(07)
      DIMENSION GAM(25)
      DOUBLE PRECISION GAM
      DOUBLE PRECISION G,Z,GAMMA,XLAMDA,DELTA,DEL,PROD,Q,QSAV
      DIMENSION GDUM(20,20),GEST(20,20),GGG(20,20)
      REAL*8 GDUM,GEST,GGG
      REAL*8 VARQ,VARW,FAC
      REAL*8 SCALE(20),SCAL
      COMMON /GKRD/IGKR
      COMMON /GKRD2/DT
      COMMON /DA1/NN,IFIX
      COMMON /DA2/IPR
      COMMON /DA3/DFAC
      COMMON /BIAS/IBIAS
C
      DT=DELTA
      IF(IPR.GE.1)WRITE(6,1000)
1000  FORMAT(1H1,20X,'THE GRAM I TECHNIQUE')
C        JOPT = 0 IF DIRECT TRANSMISSION IS ASSUMMED
      JOPT=0
      IF(IREM.NE.0)JOPT=1
C
C        DEL IS THE NUMERATOR OF THE KNOWN FIRST ORDER DIGITAL FILTERS
      DO19I=1,N
      DEL(I)=1.0D00
19    Q(I)=QSAV
      IF(IPR.GE.0)WRITE(6,2020)QSAV
2020  FORMAT(30X,'Q PARAMETERS:  Q=',F8.3)
C     CALL PRVEC(Q,N)
      NP1=N+1
      NP2=N+2
      NPNP1=N+N+1
      NPNP2=N+N+2
      NR=NP1-IREM
      NP1PIR=NP1+IREM
      DO 12 I=1,MAX
      DO 12 J=1,MAX
12    Z(I,J)=0.0D00
      VARQ=0.0
      VARW=0.0
      DO 300 I=1,NP1
      VARW=VARW+V(I)*V(I)
300   VARQ=VARQ+X(I)*X(I)
      VARQ=DSQRT(VARQ/NP1)
      VARW=DSQRT(VARW/NP1)
C
C
C        CALCULATING THE G MATRIX
      IF(IBIAS.EQ.0)GO TO 11
      NPNP2=N+N+2+1
      NR=NP1-IREM+1
11    CONTINUE
C
      DO10I=1,NPNP2
      GAM(I)=0.0
      GAMMA(I)=0.0D00
      DO10J=I,NPNP2
10    G(I,J)=0.0D00
      GAM(1)=1.0
      DO50K=1,NP1
      IF(K-IDLY)25,25,24
25    GAMMA(NP2)=0.0D00
      GO TO 26
24    FAC=1.0
      GAMMA(NP2)=V(K-IDLY)/FAC
```

```
      FAC=1.0
      GAMMA(I)=X(K)/FAC
26    CONTINUE
      DO30I=1,N
      GAM(I+1)=GAM(I+1)*Q(I)+GAM(I)*DEL(I)
      GAMMA(I+1)=GAMMA(I)*DEL(I)+GAMMA(I+1)*Q(I)
30    GAMMA(I+NP2)=GAMMA(I+NP1)*DEL(I)+GAMMA(I+NP2)*Q(I)
      IF(IBIAS.EQ.1)GAMMA(NPNP2)=GAM(I+1)
      DO 40 I=1,NPNP2
      DO 40 J=I,NPNP2
40    G(I,J)=G(I,J)+GAMMA(I)*GAMMA(J)
C     WRITE(6,1025)K,(GAMMA(I),I=1,NPNP2)
1025  FORMAT(2X,10F8.3)
50    CONTINUE
C
C     PERFORM SCALING ON G-MATRIX
      DO 735 I=1,NPNP2
      SCALE(I)=DSQRT(G(I,I))
      SCALE(I)=1.0
735   CONTINUE
C      WRITE(6,1008)
1008   FORMAT(10X,'SCALE VECTOR')
C     CALL PRVEC(SCALE,NPNP2)
      DO 736 I=1,NPNP2
      DO 736 J=1,NPNP2
736   G(I,J)=G(I,J)/(SCALE(I)*SCALE(J))
1023  CONTINUE
C     WRITE(6,1009)
1009  FORMAT(10X,'---G MATRIX---')
C     DO 56 I=1,NPNP2
C56   IF(IPR.GE.2)WRITE(6,3)(G(J,I),J=1,I)
3     FORMAT(1X,10D13.5)
42       CONTINUE
      DO60I=2,NPNP2
      K=I-1
      DO60J=1,K
60    G(I,J)=G(J,I)
C
C
C     NOISE ESTIMATION
      IF(IFIX.LE.0)GO TO 878
      CALL BUILDZ(Z,XLAMDA,QSAV,NP1,MP1,MAX)
      CALL FIX(G,Z,GEST,GDUM,XLAMDA,NPNP2,NP1,SIG2,MAX,1)
878   CONTINUE
C
C
      IF(JOPT)70,90,70
70    DO80J=1,NPNP2
      DO80I=1,NR
      G(NP1+I,J)=G(NP1PIR+I,J)
80    CONTINUE
      NPNP2=NPNP2-IREM
      DO85J=1,NPNP2
      SCALE(NP1+J)=SCALE(NP1PIR+J)
      DO85I=1,NR
      G(J,NP1+I)=G(J,NP1PIR+I)
85    CONTINUE
90    CONTINUE
      DO 55 I=1,NPNP2
55    IF(IPR.GE.2)WRITE(6,3)(G(J,I),J=1,I)
      CALL GKRDCT(G,Z,DET,XLAMDA,NPNP2,MAX,1)
C     DE-SCALE SYNTHETIC COEFFICIENT VECTOR, XLAMDA
      DO 741 I=1,NPNP2
741   XLAMDA(I)=XLAMDA(I)/SCALE(I)
      IF(IBIAS.EQ.0)GO TO 43
      NPNP2=NPNP2-1
```

```
      NR=NR-1
43    CONTINUE
      XMEAN=XLAMDA(NPNP2+1)
      IF(JOPT)120,130,120
120   NPNP2=NPNP2+IREM
      IF(IPR.GE.2)WRITE(6,210)(XLAMDA(I),I=1,NPNP2)
210   FORMAT(1X,5G12.5)
      DO122I=1,NR
122   XLAMDA(NPNP2-I+1)=XLAMDA(NP2+NR-I)
      DO123I=1,IREM
123   XLAMDA(NP1+I)=0.0D00
130   CONTINUE
      FAC=1.0
      DO301I=NP2,NPNP2
301   XLAMDA(I)=XLAMDA(I)*FAC
      IF(IPR.GE.3)WRITE(6,1001)
1001  FORMAT(10X,'THE SYNTHETIC COEFFICIENT VECTOR, XLAMDA, IS')
      IF(IPR.GE.3)CALL PRVEC(XLAMDA,NPNP2)
      DO 150 I=1,NPNP2
150   GAMMA(I)=XLAMDA(I)
C
C        GENERATING GAMMA FROM XLAMDA
      CALL BUILDA(G,Q,DEL,N,MAX)
      DO160I=1,NPNP2
      GAMMA(I)=0.0D00
      DO160J=1,NPNP2
160   GAMMA(I)=GAMMA(I)+G(I,J)*XLAMDA(J)
165   CONTINUE
      DO200I=2,NPNP2
200   GAMMA(I)=GAMMA(I)/GAMMA(1)
      GAMMA(1)=1.0D00
      IF(IDLY.EQ.0)GO TO 172
      IDLY1=IDLY+1
      DO 170 II=IDLY1,NP1
      I=NPNP2+1-II
170   GAMMA(I+IDLY)=GAMMA(I)
      DO 172 I=1,IDLY
      GAMMA(I+NP1)=0.0D00
172   CONTINUE
C
C        CALCULATING THE EQUIVALENT CONTINUOUS DESCRIPTION
C     CALL PRVEC(GAMMA,NPNP2)
      CALL IZTOS(GAMMA,N,DELTA,KOPT)
      IF(IPR.GE.1)WRITE(6,1003)
1003  FORMAT(///,1X,100(1H-),/,1X,100(1H-))
      RETURN
      END
      SUBROUTINE IZTOS(GAMMA,N,DELTA,IZTS)
C
C        IZTOS SEPARATES THE NUMERATOR FROM THE DENOMINATOR PARAMETERS
C        IN GAMMA
C
      DIMENSION GAMMA(1),X1(20),X2(20)
      DOUBLE PRECISION GAMMA,X1,X2,DELTA
      COMMON /DA1/NN,IFIX
      COMMON /DA2/IPR
      NP1=N+1
200   DO3I=1,NP1
      X1(I)=GAMMA(I)
3     X2(I)=-GAMMA(NP1+I)
      CALL ZTOS(X1,X2,N,DELTA,IZTS)
      IZTS=IZTS+1
      IF(IZTS.EQ.4) GO TO 200
      RETURN
      END
      SUBROUTINE POLCON(C,R2,K,N)
```

```
C
C     A POLYNOMIAL CONSTRUCTION PROGRAM NEEDED FOR ZTOS
C
      DIMENSION C(1),R2(1)
      COMPLEX*16 C,R2,COMP
      REAL*8 DC(2)
      EQUIVALENCE (COMP,DC)
      NP1=N+1
      DO10I=2,NP1
10    R2(I)=0.0D00
      R2(1)=1.0D00
      DO4I=1,N
      COMP=C(I)
      IF(I.EQ.K.OR.(DC(1).EQ.0.0D0.AND.DC(2).EQ.0.0D0))GO TO 4
      DO2JJ=1,I
      J=I-JJ+1
2     R2(J+1)=R2(J+1)*C(I)+R2(J)
      R2(1)=R2(1)*C(I)
4     CONTINUE
      RETURN
      END
      SUBROUTINE POLRT(XCOF,COF,M,ROOTR,ROOTI,IER)
C
C           COMPUTES THE REAL AND COMPLEX ROOTS OF A REAL POLYNOMIAL
C
C        DESCRIPTION OF PARAMETERS
C           XCOF -VECTOR OF M+1 COEFFICIENTS OF THE POLYNOMIAL
C                 ORDERED FROM SMALLEST TO LARGEST POWER
C           COF  -WORKING VECTOR OF LENGTH M+1
C           M    -ORDER OF POLYNOMIAL
C           ROOTR-RESULTANT VECTOR OF LENGTH M CONTAINING REAL ROOTS
C                 OF THE POLYNOMIAL
C           ROOTI-RESULTANT VECTOR OF LENGTH M CONTAINING THE
C                 CORRESPONDING IMAGINARY ROOTS OF THE POLYNOMIAL
C           IER  -ERROR CODE WHERE
C                 IER=0  NO ERROR
C                 IER=1  M LESS THAN ONE
C                 IER=2  M GREATER THAN 36
C                 IER=3  UNABLE TO DETERMINE ROOT WITH 500 INTERATIONS
C                        ON 5 STARTING VALUES
C                 IER=4  HIGH ORDER COEFFICIENT IS ZERO
C
      DIMENSION XCOF(1),COF(1),ROOTR(1),ROOTI(1)
      DOUBLE PRECISION XO,YO,X,Y,XPR,YPR,UX,UY,V,YT,XT,U,XT2,YT2,SUMSQ,
     1 DX,DY,TEMP,ALPHA,XCOF,COF,ROOTR,ROOTI
C
C           LIMITED TO 36TH ORDER POLYNOMIAL OR LESS.
C           FLOATING POINT OVERFLOW MAY OCCUR FOR HIGH ORDER
C           POLYNOMIALS BUT WILL NOT AFFECT THE ACCURACY OF THE RESULTS.
C
C        METHOD
C           NEWTON-RAPHSON ITERATIVE TECHNIQUE.  THE FINAL ITERATIONS
C           ON EACH ROOT ARE PERFORMED USING THE ORIGINAL POLYNOMIAL
C           RATHER THAN THE REDUCED POLYNOMIAL TO AVOID ACCUMULATED
C           ERRORS IN THE REDUCED POLYNOMIAL.
C
      IFIT=0
      N=M
      IER=0
      IF(XCOF(N+1))10,25,10
   10 IF(N) 15,15,32
C
C        SET ERROR CODE TO 1
C
   15 IER=1
   20 IF(IER)200,201,200
```

```
200   WRITE(6,203)IER
203   FORMAT(1X,'ERROR CALLED FROM POLRT, IER = ',I3)
201   RETURN
C
C        SET ERROR CODE TO 4
C
   25 IER=4
      GO TO 20
C
C        SET ERROR CODE TO 2
C
   30 IER=2
      GO TO 20
   32 IF(N-36) 35,35,30
   35 NX=N
      NXX=N+1
      N2=1
      KJ1 = N+1
      DO 40 L=1,KJ1
      MT=KJ1-L+1
   40 COF(MT)=XCOF(L)
C
C        SET INITIAL VALUES
C
   45 XO=.00500101
      YO=0.01000101
C
C        ZERO INITIAL VALUE COUNTER
C
      IN=0
   50 X=XO
C
C        INCREMENT INITIAL VALUES AND COUNTER
C
      XO=-10.0*YO
      YO=-10.0*X
C
C        SET X AND Y TO CURRENT VALUE
C
      X=XO
      Y=YO
      IN=IN+1
      GO TO 59
   55 IFIT=1
      XPR=X
      YPR=Y
C
C        EVALUATE POLYNOMIAL AND DERIVATIVES
C
   59 ICT=0
   60 UX=0.0
      UY=0.0
      V =0.0
      YT=0.0
      XT=1.0
      U=COF(N+1)
      IF(U) 65,130,65
   65 DO 70 I=1,N
      L =N-I+1
      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-10)100,80,80
C
C        STEP ITERATION COUNTER
C
   80 ICT=ICT+1
      IF(ICT-500) 60,85,85
   85 IF(IFIT)100,90,100
   90 IF(IN-5) 50,95,95
C
C        SET ERROR CODE TO 3
C
   95 IER=3
      GO TO 20
  100 DO 105 L=1,NXX
      MT=KJ1-L+1
      TEMP=XCOF(MT)
      XCOF(MT)=COF(L)
  105 COF(L)=TEMP
      ITEMP=N
      N=NX
      NX=ITEMP
      IF(IFIT) 120,55,120
  110 IF(IFIT) 115,50,115
  115 X=XPR
      Y=YPR
  120 IFIT=0
122   IF(DABS(Y)-1.0D-8*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)
  145 DO 150 L=2,N
  150 COF(L+1)=COF(L+1)+ALPHA*COF(L)-SUMSQ*COF(L-1)
  155 ROOTI(N2)=Y
      ROOTR(N2)=X
      N2=N2+1
      IF(SUMSQ) 160,165,160
  160 Y=-Y
      SUMSQ=0.0
      GO TO 155
  165 IF(N) 20,20,45
      END
      SUBROUTINE PRCVEC(A,N)
C
C     PRCVEC PRINTS A DOUBLE PRECISION COMPLEX VECTOR
C        C      A COMPLEX NUMBER OF THE FORM A+JB IS PRINTED ( A, B J)
C
      DIMENSION A(1)
      COMPLEX*16 A
```

```
      IF(N.EQ.0) GO TO 100
C     WRITE(6,920)
      WRITE(6,910)(A(I),I=1,N)
910   FORMAT(1X,1H(,D22.15,1H,,D22.15,3H J))
C     WRITE(6,920)
100   CONTINUE
920   FORMAT(2(/))
      RETURN
      END
      SUBROUTINE PRMAT(A,N,M,NMAX)
      DOUBLE PRECISION A
C
C     THIS SUBROUTINE OUTPUTS DOUBLE PRECISION DOUBLE DIMENSIONED ARRAY
      DIMENSION A(NMAX,1)
      WRITE(6,1)
      DO2I=1,N
2     WRITE(6,3)(A(I,J),J=1,M)
3     FORMAT(1X,10D13.5)
      WRITE(6,1)
      WRITE(6,1)
1     FORMAT(/)
      RETURN
      END
      SUBROUTINE PRVEC(A,N)
C
C     THIS SUBROUTINE OUTPUTS DOUBLE PRECISION SINGLE DIMENSIONED ARRAY
      DIMENSION A(1)
      DOUBLE PRECISION A
      WRITE(6,1)(A(I),I=1,N)
1     FORMAT(1X,10D13.5)
31    FORMAT(/)
      RETURN
      END
      SUBROUTINE RESPON(X,V,N,GAMMA,XLAMDA,MP1)
      DIMENSION X(1),V(1),GAMMA(1),XLAMDA(1)
      REAL*8 XSAV,GAMMA,XLAMDA
      NM1=N-1
      NP1=N+1
      NPNP1=N+N+1
      NPNP2=N+N+2
      DO 19 I=1,NPNP1
19    XLAMDA(I)=0.0D00
      XSAV=0.0D00
C
C
      DO 20 K=1,MP1
      IF(N.EQ.1)GO TO 25
      DO 21 I=1,NM1
      J=NP1-I
21    XLAMDA(J)=XLAMDA(J-1)
25    CONTINUE
      DO 22 I=1,N
      J=NPNP2-I
22    XLAMDA(J)=XLAMDA(J-1)
      XLAMDA(1)=XSAV
      XLAMDA(NP1)=V(K)
      XSAV=0.0D00
      DO 23 I=1,NPNP1
23    XSAV=XSAV-GAMMA(I+1)*XLAMDA(I)
      IF(DABS(XSAV).GE.1.0D10)XSAV=0.0D00
20    X(K)=XSAV
 77    FORMAT(2X,10F7.2)
78    FORMAT(1X,/)
      RETURN
      END
      SUBROUTINE ZTOS(B,A,N,DELTA,IZTS)
```

```
      COMMON /DA2/IPR
      COMMON /DELAY/IDLY
C
C
C     CONVERSION OF A DISCRETE TIME SYSTEM H(Z) TO A CONTINUOUS TIME SYS
C
C
C     H(Z)=(A(1) + A(2)*ZETA +....)/(1 + B(2)*ZETA +....)
C                        ZETA = 1/Z
C
C     H(S)=(A(1) + A(2)*S + ..... + A(N+1)*S**N)/DENOM
C
C               DENOM=B(1) + B(2)*S + ..... + B(N+1)*S**N
C
C                    B(1) = 1 ALWAYS
C
C
      DIMENSION B(20),A(20),TEMP(20),RR(20),RI(20),CR(20),CA(20),
     +CAA(20),CA1(20),CB(20),CF(20),CF1(20),CG(20)
      COMPLEX*16 CA,CAA,CA1,CB,CR,CON1,CON2,CONT,FAC,A1,A2,B1,B2,AA1,BB1
     1CG,CF1,CF
      REAL*8 B,A,TEMP,RR,RI,DELTA
      CONT=0.0D00
      IORP=IZTS
      NP1=N+1
      NNP1=NN+1
      IF(IPR.GE.1)WRITE(6,989) NN
989   FORMAT(10X,'NN = ',I5)
999   FORMAT(/)
      IF(IPR.GE.1)WRITE(6,999)
      WRITE(6,1000)
1000  FORMAT(' Z-DOMAIN DENOMINATOR')
      CALL PRVEC(B,NP1)
      WRITE(6,1001)
1001  FORMAT(' Z-DOMAIN NUMERATOR')
      CALL PRVEC(A,NP1)
      IF(IZTS.EQ.0) GO TO 919
      IF(IZTS.EQ.1) GO TO 200
      IF(IZTS.EQ.2) GO TO 250
      IF(IZTS.EQ.3) GO TO 200
      IF(IZTS.EQ.4) GO TO 250
200   CONTINUE
C
C
C     LOGARITHMIC TRANSFORMATION
C
C
C     WORK ON NUMERATOR
C
C
      NMD=NP1-IDLY
      DO 14 I=1,NMD
14    A(I)=A(I+IDLY)
      A1=0.0D0
      B1=0.0D0
      DO 30I=1,NP1
      IF(I.LE.NNP1)A1=A1+A(I)
30    B1=B1+B(I)
      IF(NN.EQ.0)GO TO 469
      CALL POLRT(A,TEMP,NN,RR,RI,IER)
      DO15 I=1,NN
15    CA(I)=DCMPLX(RR(I),RI(I))
      DO 7 I=1,NN
7     CA(I)=(+1.0/DELTA)*CDLOG(CA(I))
      IF(NN.EQ.N) GOTO471
469   CONTINUE
      DO 470 I=NNP1,NP1
```

```
      CAA(I)=0.0D0
470   CA(I)=0.0D0
471   CONTINUE
      IF(NN.EQ.0)CAA(1)=1.0D0
C
C
C     NOW THE FIRST NN ENTRIES OF CA CONTAIN THE S-DOMAIN ZEROES OF NUME
C     AND THE REMAINING ENTRIES ARE ZEROED OUT.
C
      IF(NN.NE.0)CALL POLCON(CA,CAA,0,N)
C
C
C     WORK ON DENOMINATOR
C
C
919   CALL POLRT(B,TEMP,N,RR,RI,IER)
      DO16 I=1,N
      CR(I)=DCMPLX(RR(I),RI(I))
16    CF(I)=1.0D00/CR(I)
909   IF(IPR.GE.2)WRITE(6,1002)
      IF(IPR.GE.2)CALL PRCVEC(CF,N)
      IF(IZTS.EQ.0) GO TO 900
235   DO6 I=1,N
6     CR(I)=(-1.0/DELTA)*CDLOG(CR(I))
      IF(IPR.GE.2)WRITE(6,240)
240   FORMAT(' LOGARITHMIC TRANSFORMATION')
      IF(IPR.GE.1)WRITE(6,999)
      WRITE(6,2000)
2000  FORMAT(' POLES IN S DOMAIN')
      IF(IPR.GE.1)CALL PRCVEC(CR,N)
      DO3000I=1,N
3000  CR(I)=-CR(I)
      CALL POLCON(CR,CB,0,N)
C
C
C     ADJUST DC GAIN CONSTANT
C
C
      A2=CAA(1)
      B2=CB(1)
      FAC=(A1/B1)*(B2/A2)
      DO 603 I=1,NNP1
603   CAA(I)=CAA(I)*FAC
      GO TO 2010
C
C
C     DELAYED PULSE INVARIANT TRANSFORMATION
C
C
C     SHIFTS NUMERATOR COEFFICIENTS FOR DELAY
C
C
250   CONT=A(1)
      DO 300 I=1,N
300   A(I)=A(I+1)-CONT*B(I+1)
      A(NP1)=0.0
400   CALL POLRT(B,TEMP,N,RR,RI,IER)
      DO61I=1,N
      CR(I)=DCMPLX(RR(I),RI(I))
61    CF(I)=1.0D00/CR(I)
      IF(IPR.GE.2)WRITE(6,1002)
1002  FORMAT(1X,'THE POLES OF THE Z-DOMAIN')
      IF(IPR.GE.2)CALL PRCVEC(CF,N)
C
C
C     PARTIAL FRACTION EXPANSION
```

```
C
C
      DO3I=1,N
      CON1=1.0D00
      CON2=0.0D00
      DO4J=1,N
      CON2=CON2*CR(I)+A(N-J+1)
      IF(I-J)5,4,5
5     CON1=CON1*(1.0D00-CR(I)*CF(J))
4     CONTINUE
3     CA(I)=CON2/CON1
C
C
C     TRANSFORMATION OF DENOMINATOR AND NUMERATOR
C
C
224   DO2I=1,N
      CR(I)=CDLOG(CR(I))/DELTA
      CA(I)=CA(I)*CR(I)/(1.0D00-CF(I))
2     CONTINUE
      CA(NP1)=0.0D00
      IF(IPR.GE.2)WRITE(6,241)
241   FORMAT(' DELAYED PULSE TRANSFORMATION')
      IF(IPR.GE.0)WRITE(6,999)
226   WRITE(6,1004)
1004  FORMAT(' NEGATIVE OF THE POLES IN THE S-DOMAIN',I4)
      IF(IPR.GE.0)CALL PRCVEC(CR,N)
      IF(IPR.GE.2)WRITE(6,1003)
1003  FORMAT(1X,'NUMERATOR CONSTANTS OF FACTORIZED H(S)')
      IF(IPR.GE.2)CALL PRCVEC(CA,N)
      CALL POLCON(CR,CB,0,N)
      DO71I=1,NP1
71    CAA(I)=0.0D00
      DO9K=1,N
      CALL POLCON(CR,CF1,K,N)
      DO9J=1,N
9     CAA(J)=CAA(J)+CF1(J)*CA(K)
      CAA(NP1)=0.0D00
2010  CONTINUE
      DO 450 I=1,NP1
450   CAA(I)=CAA(I)+CONT*CB(I)
C
C
C
403   IF(IPR.GE.1)WRITE(6,1005)
1005  FORMAT(' S-DOMAIN DENOMINATOR')
      IF(IPR.GE.1)CALL PRCVEC(CB,NP1)
      IF(IPR.GE.1)WRITE(6,1006)
1006  FORMAT(' S-DOMAIN NUMERATOR')
      IF(IPR.GE.1)CALL PRCVEC(CAA,NP1)
      DO20I=1,NP1
      B(I)=CB(I)
20    A(I)=CAA(I)
900   RETURN
      END
C
END OF DATA
```

APPENDIX B

A Tutorial Example of the Application of Eq. (6)

Consider the setup of Fig. 6 where u (k) denotes the input signal and y (k) the output. The network is known to be of first order. The measurements are made every 1 ms for 5 samples, k = 0,1,...,4.

Suppose the following signals are generated as a result of a unit pulse input:

$y_0(k)$	0	0.5	0.4	0.32	0.256
$y_1(k)$	0	0.5	0.9	1.22	1.476
$u_0(k)$	0	1	0	0	0
$u_1(k)$	0	1	1	1	1

The Gram matrix of the signals y_0, y_1 and u_1 is

$$F = \begin{bmatrix} 0.57794 & & \\ 1.37831 & 4.7270 & \\ -1.47602 & -4.0960 & 4.0 \end{bmatrix}$$

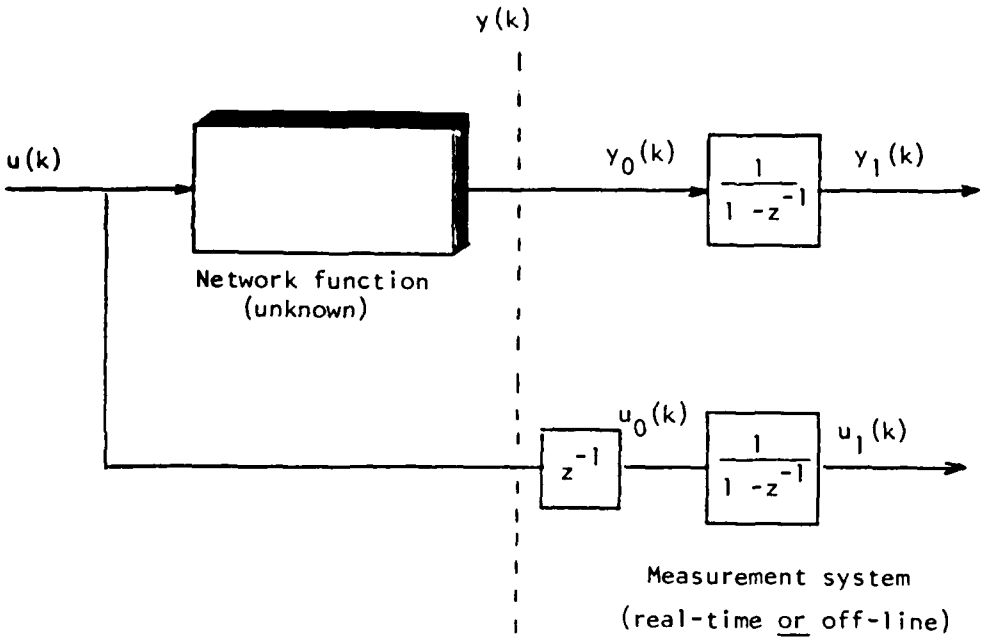
which yields the following square-roots of the diagonal cofactors.

$$\sqrt{D_1} = 1.4597 \qquad \sqrt{D_2} = 0.36492 \qquad \sqrt{D_3} = 0.9123$$

Note that the signs of these square-roots are chosen in direct correspondence with the signs of the cofactors of the first row of F [9]. Now, substitution into (6) yields

$$(1 - 0.8\ z^{-1})\ Y(z) = 0.5\ z^{-1}\ U(z)$$

Clearly, the true parameters have been recovered.



$$\text{Unknown network function} = \frac{0.5z^{-1}}{1 - 0.8z^{-1}}$$

Fig. 6. A first order test system

APPENDIX C

```
C
C
C
C
C
C     ****************************************************************
C     *                                                              *
C     *                    PROGRAM NAME 'STOZ'                       *
C     *                                                              *
C     *     ____________________________________________             *
C     *                                                              *
C     *      S-DOMAIN  TO Z-DOMAIN    CONVERSION    PROGRAM          *
C     *                ____________________________                  *
C     *                                                              *
C     *                   DEVELOPED  AT THE                          *
C     *                 UNIVERSITY OF SOUTH FLORIDA                  *
C     *                                                              *
C     *                                                              *
C     *                                                              *
C     *                ____________________________                  *
C     *                                                              *
C     *                     FOR ASSISTANCE                           *
C     *                        CONTACT                               *
C     *                     DR. V. K. JAIN                           *
C     *                      813/974-2581                            *
C     ****************************************************************
C
C
C
C           S-DOMAIN  TO Z-DOMAIN    CONVERSION    PROGRAM
C       (FROM A GIVEN RATIONAL TRANSFER FUNCTION H (S), IT FINDS
C        AN EQUIVALENT Z-DOMAIN TRANSFER FUNCTION H (Z) BY ONE
C          OF SEVERAL METHODS, E.G. IMPULSE-INV. OR STEP-INV.)
C
C
C
C
C         GIVEN THE CONTINUOUS DESCRIPTION, SUBROUTINE COMPUTES THE
C         EQUIVALENT DISCRETE DOMAIN DESCRIPTION OF A LINEAR
C         DYNAMIC SYSTEM
C
C         STOZ GENERATES H(Z) AND THE CORRESPONDING DIFFERENCE
C         EQUATION FROM THE TRANSFER FUNCTION H(S)
C
C         THE INPUT ARRAYS A AND B ARE FILLED ACCORDING TO THE
C         DIFFERENTIAL EQUATION
C              B(1)*Y(T)+B(2)*D(1,Y(T))+...+B(N+1)*D(N,Y(T))
C                   -A(1)*U(T)-A(2)*D(1,U(T))-...-A(N+1)*D(N,U(T)) = 0
C         WHERE D(M,F(T)) = THE MTH TIME DERIVATIVE OF FUNCTION, F
C
C         B(N+1) MUST EQUAL 1
C
```

```
C       RETURNS ARRAYS A AND B CONTAINING THE EQUIVALENT DISCRETE
C       DESCRIPTION STORED ACCORDING TO THE DIFFERENCE EQUATION
C            B(1)*Y(K)+B(2)*Y(K-1)+...+B(N+1)*Y(K-N)
C                 -A(1)*U(K)-A(2)*U(K-1)-...-A(N+1)*U(K-N) = 0
C
C       B(1) ALWAYS EQUALS 1
C
C       THE POLES OF THE CONTINUOUS DOMAIN MUST BE DISTINCT AND
C       NON-ZERO FOR THE TRANSFORMATION TO BE VALID
C       ------------------------------------------
C
C       DATA CARD SET PREPARATION
C
C       N = ORDER OF SYSTEM
C       N (MAXIMUM) = ONE LESS THAN THE DIMENSION SUBSCRIPT
C
C       IMTHD = 0 FOR THE IMPULSE INVARIANT DESCRIPTION .
C             = 1 FOR THE PULSE INVARIANT DESCRIPTION .
C             = 2 FOR THE TRAPEZOIDAL INVARIANT DESCRIPTION .
C             = 3 FOR THE LOGRITHMIC TRANSFORM DESCRIPTION .
C
C       IPOLZ = 1 IF POLES AND ZEROES ARE READ
C                 MUST BE COMPLEX (REAL,IMAGINARY)
C                 NEGATIVES OF POLES AND ZEROS READ; IE,
C                 FOR A POLE OF (S+2), INPUT +2.0    +0.0
C                 (2F10.0 PER POLE, 4 POLES PER CARD)
C
C             = 0 IF DENOMINATOR AND NUMERATOR ARE READ
C                 IN POLYNOMIAL FORM.
C                 COEFFICIENTS ORDERED FROM LOW TO HIGH DEGREE,
C                 DENOMINATOR INFORMATION ALWAYS READ FIRST.
C                 HIGHEST ORDER DENOMINATOR COEFF MUST BE 1.0
C
C             NOTE:  WHEN POLE-ZERO DATA IS ENTERED A GAIN CARD
C                    MUST FOLLOW THE LAST POLE CARD.
C                    IF THERE ARE NO ZEROS, USE BLANK CARDS IN
C                    THE NORMAL ZEROS POSITIONS.
C
C       IZERO = 1 ZEROES OF Z-DOMAIN FN ARE PRINTED OUT
C
C
C
C       START EACH DATA CARD SET WITH A DESCRIPTION CARD,
C         CONTAINING UP TO 51 CHARACTERS COLS 2-52
C       FIRST DATA CARD CONTAINS:
C         N,IMTHD,IPOLZ,IN 3F5 FORMAT, PLUS DELTA IN F10.0 FORMAT
C       SECOND GROUP OF DATA CARDS IS N POLES OR N+1 DENOMINATOR
C         COEFFICIENTS.  AFTER LAST POLE CARD, USE A GAIN CARD.
C       LAST GROUP OF DATA CARDS IS N ZEROS (OR BLANKS),
C         OR N+1 NUMERATOR COEFFICIENTS (BLANKS FOR ZERO COEFFS)
C       THE DATA FORMAT FOR EACH OF THE SYSTEM PARAMETER CARDS
C         IS 8F10.0
C
C       AS MANY SETS OF DATA CARDS MAY BE RUN AS DESIRED
C       ------------------------------------------------
```

```
C
C     STOZ MAIN PROGRAM
C
      REAL*8 B(20),A(20),RR(20),RI(20),DELTA,TEMP(20)
      COMPLEX*16 CR(20),CA(20),CB(20),CAA(20),CA1(20),
     1 TEM(20),CON1,CON2,CONT
      DIMENSION TITLE(52)
C
C     READ TITLE AND FIRST DATA CARD
C
100   READ(5,920,END=5999)TITLE
      WRITE(6,910)
910   FORMAT(6(/))
      WRITE(6,920)TITLE
920   FORMAT(52A1)
      READ(5,920)TITLE
      WRITE(6,930)
930   FORMAT(/,1X,71('*'))
      READ(5,940)N,IMTHD,IPOLZ,DELTA,IZZERO
940   FORMAT(3I5,F10.0,3I5)
      WRITE(6,950)N,IMTHD,IPOLZ,DELTA
950   FORMAT(//3X,'N =',I4,5X,'IMTHD =',I4,5X,'IPOLZ =',I4,5X,
     1 'DELTA =',G17.10,//)
      NP1=N+1
      NP2=N+2
      NPNP1=N+N+1
      NPNP2=N+N+2
      IF(IPOLZ.NE.1) GO TO 300
C
C     READ POLES AND ZEROS
C
      READ(5,960)(CR(I),I=1,N)
960   FORMAT(8F10.0)
      CALL POLCON(CR,TEM,0,N)
      DO109 I=1,NP1
109   B(I)=TEM(I)
C
C     READ GAIN CARD
      READ(5,960) RK
C
C     READ ZEROS
      READ(5,960) (CA(I),I=1,N)
      CALL POLCON(CA,TEM,0,N)
      DO 209 I=1,NP1
209   A(I)=TEM(I)*RK
      GO TO 310
C
C     READ DENOMINATOR AND NUMERATOR COEFFICIENTS
C
300   READ(5,960) (B(I),I=1,NP1)
      READ(5,960) (A(I),I=1,NP1)
310   CONTINUE
C
C     PRINT DENOMINATOR AND NUMERATOR COEFFICIENTS
C
```

```
      WRITE(6,970)
970   FORMAT(' S-DOMAIN DENOMINATOR')
      CALL PRVEC(B,NP1)
      WRITE(6,980)
980   FORMAT(' S-DOMAIN NUMERATOR')
      CALL PRVEC(A,NP1)
C
C     DETERMINE ORDER OF NUMERATOR
C
      NN=N
      DO 309 I=1,NP1
      II=NP1+1-I
      IF(A(II).NE.0.0) GO TO 400
309   NN=NN-1
400   CONTINUE
      WRITE(6,990) NN
990   FORMAT(' ORDER OF S-DOMAIN NUMERATOR =',I5,//)
      IF(NN.LT.0) GO TO 5029
      NNP1=NN+1
C
C     FACTOR DENOMINATOR TO FIND POLES
C
      IF(IPOLZ.NE.0) GO TO 500
      CALL POLRT(B,TEMP,N,RR,RI,IER)
      DO 409 I=1,N
409   CR(I)=DCMPLX(RR(I),RI(I))
500   WRITE(6,991)
991   FORMAT(' POLES OF THE S-DOMAIN')
      CALL PRCVEC(CR,N)
C
      IF(IMTHD.NE.3) GO TO 1500
C
C     LOGRITHMIC TRANSFORM
C
1000  CONTINUE
      WRITE(6,9100)
9100  FORMAT(/,' LOGRITHMIC TRANSFORM')
C
C     WORK ON NUMERATOR
C
      IF(NN.EQ.0) GO TO 1030
      IF(IPOLZ.NE.0) GO TO 1010
      CALL POLRT(A,TEMP,NN,RR,RI,IER)
      DO 1009 I=1,NN
1009  CA(I)=DCMPLX(RR(I),RI(I))
1010  WRITE(6,9200)
9200  FORMAT(' ZEROS IN  S DOMAIN')
      CALL PRCVEC(CA,NN)
      DO 1029 I=1,NN
1029  CA(I)=CDEXP(CA(I)*DELTA)
      IF(NN.EQ.N) GO TO 1100
1030  DO 1039 I=NNP1,NP1
      CAA(I)=0.0D0
1039  CA(I)=0.0D0
1100  CONTINUE
C
C     NOW THE FIRST NN ENTRIES OF CA CONTAIN THE
C     Z-DOMAIN ZEROS OF THE TRANSFER FUNCTION, WHILE THE
C     REMAINING ENTRIES ARE ZEROED OUT.
```

```
C     WORK ON DENOMINATOR
C
      DO 1129 I=1,N
1129  CR(I)=CDEXP(CR(I)*DELTA)
C
C     NOW CR CONTAINS THE N Z-DOMAIN POLES
C
C     FORM NUMERATOR AND DENOMINATOR
C     Z-DOMAIN POLYNOMIALS
C
      IF(NN.EQ.0) CAA(1)=1.0D0
      IF(NN.NE.0) CALL PCSTZ(CA,CAA,0,NN)
      CALL PCSTZ(CR,CB,0,N)
C
C     NOW CB CONTAINS THE N+1 Z-DOMAIN DENOMINATOR COEFFICIENTS,
C     AND CAA CONTAINS THE NN+1 NUMERATOR COEFFICIENTS.
C
C     ADJUST DC GAIN CONSTANT
C
      A1=A(1)/B(1)
      A2=1.0D0
      DO 1209 I=1,NN
1209  A2=A2+CAA(I+1)
      B2=1.0D0
      DO 1219 I=1,N
1219  B2=B2+CB(I+1)
      FAC=A1*B2/A2
      DO 1229 I=1,NNP1
1229  CAA(I)=CAA(I)*FAC
C
C      NOW CAA CONTAINS THE ADJUSTED Z-DOMAIN NUMERATOR COEFFICIENTS
C      AND FAC CONTAINS THE GAIN FACTOR USED FOR THE ADJUSTMENT
C
      GO TO 5000
C
1500  CONTINUE
C
C     NON-LOGRITHMIC TRANSFORMATIONS
C
C     ADJUST FOR DIRECT TRANSMISSION
C     THIS ROUTINE REQUIRES THAT B(NP1) = 1.0
C
      IF(NN.LT.N) GO TO 1510
      CONT=A(NP1)
      DO1509 I=1,N
1509  A(I)=A(I)-CONT*B(I)
C
C     FIND NUMERATOR CONSTANTS FOR PARTIAL FRACTION EXPANSION
C
1510  DO1529 I=1,N
      CON1=1.0D00
      CON2=0.0D00
      DO1519 J=1,N
      CON2=CON2*CR(I)+A(N-J+1)
      IF(I-J)1512,1519,1512
1512  CON1=CON1*(CR(I)-CR(J))
1519  CONTINUE
1529  CA(I)=CON2/CON1
      WRITE(6,9300)
9300  FORMAT(' NUMERATOR CONSTANTS OF THE FACTORIZED H(S)')
      CALL PRCVEC(CA,N)
```

```
C     CONVERT THE FIRST ORDER PARTIAL FRACTIONS TO Z DOMAIN
C
      IMTHD=IMTHD+1
      GO TO (2000,3000,4000), IMTHD
C
C     IMPULSE INVARIANT
C
2000  DO2009I=1,N
      CA(I)=CA(I)*DELTA
2009  CR(I)=CDEXP(CR(I)*DELTA)
      WRITE(6,3351)
3351  FORMAT(2X,'Z-DOMAIN RESIDUES')
      CALL PRCVEC(CA,N)
      GO TO 4500
C
C     PULSE INVARIANT
C
3000  DO3009 I=1,N
      CON1=CDEXP(CR(I)*DELTA)
      CA(I)=CA(I)*(CON1-1.0D00)/CR(I)
3009  CR(I)=CON1
      WRITE(6,3351)
      CALL PRCVEC(CA,N)
      GO TO 4500
C
C     TRAPEZOIDAL INVARIANT
C
4000  ICHECK=2
      DO4009 I=1,N
      CON1=CDEXP(CR(I)*DELTA)
      CON2=CA(I)/(CR(I)*CR(I)*DELTA*CON1)
      CONT=CONT+CON2*((1.0D00-CR(I)*DELTA)*CON1-1.0D00)
      CA(I)=CON2*(1.0D00-CON1)*(1.0D00-CON1)
4009  CR(I)=CON1
      GO TO 4500
C
C     CONSTRUCT THE Z DOMAIN DENOMINATOR
C     AND NUMERATOR POLYNOMIALS
C
4500  CONTINUE
      CALL PCSTZ(CR,CB,0,N)
      DO 4509 I=1,N
4509  CAA(I)=0.0D00
      DO 4519 K=1,N
      CALL PCSTZ(CR,CA1,K,N)
      DO 4519 J=1,N
4519  CAA(J)=CAA(J)+CA1(J)*CA(K)
      CAA(NP1)=0.0D00
C
C           ADJUST FOR DIRECT TRANSMISSION
C
      DTXC=(0.0,0.0)
      IF(NN.NE.N) CONT=DTXC
      CAA(NP1)=CONT*CB(NP1)
      DO 4529 I=1,N
4529  CAA(I)=CAA(I)+CONT*CB(I)
C
C           SHIFT NUMERATOR TO COMPLETE PULSE INVARIANT TRANSFORM
C           WHEN NUMERATOR HAS LOWER ORDER THAN DENOMINATOR
```

```
      CONTINUE
C
C     ------------------------------------------------------------
C
C
C     PRINT THE TRANSFORMED COEFFICIENTS
C
5000  CONTINUE
C
      WRITE(6,9510)
9510  FORMAT(' POLES IN THE Z DOMAIN')
      CALL PRCVEC(CR,N)
      WRITE(6,9520) FAC
9520  FORMAT(' GAIN FACTOR USED =',E14.7,//)
      IF(IZZERO.EQ.0)GO TO 3323
      NN=N-1
      DO 3321 I=1,N
3321  B(I)=CAA(NP1-I)
      CALL POLRT(B,TEMP,NN,RR,RI,IER)
      DO 3322 I=1,NN
3322  CA(I)=DCMPLX(RR(I),RI(I))
      WRITE(6,9530)NN
9530  FORMAT(' ZEROS IN THE Z DOMAIN',I4)
      CALL PRCVEC(CA,NN)
3323  CONTINUE
      WRITE(6,9540)
9540  FORMAT(' Z-DOMAIN DENOMINATOR')
      CALL PRCVEC(CB,NP1)
      WRITE(6,9550)
9550  FORMAT(' Z-DOMAIN NUMERATOR')
      CALL PRCVEC(CAA,NP1)
      DO 5019 I=1,NP1
      B(I)=CB(I)
5019  A(I)=CAA(I)
1239  FORMAT(7F10.7)
      WRITE(6,1239)(B(I),I=1,NP1)
      WRITE(6,1239)(A(I),I=1,NP1)
      GO TO 100
C
5029  WRITE(6,9560)
9560  FORMAT(/,1X,'NUMERATOR ORDER LESS THAN ZERO',//)
5999  STOP
      END
```

APPENDIX D

GEORGE'S THEOREM

This theorem helps evaluation of the quadratic volterra response (or the bilinear response) by inspection from the associated response.

Theorem If

$$Y_{(2)}(s_1,s_2) = \frac{1}{(s_1+a)(s_2+b)}\ C(s_1+s_2) \tag{B1}$$

then

$$Y_2(s) = \frac{C(s)}{s+a+b} \tag{B2}$$

Proof: It is readily shown that

$$Y_2(s) = \frac{1}{2\pi j}\int_{-j\infty}^{j\infty} \frac{C(s)}{(s_1+a)(s-s_1+b)}\ ds_1 \tag{B3}$$

(see Bush [11]). Then the desired result follows immediately by use of the residue theorem.

Corollary Let

$$Y_{(2)}(s_1,s_2) = \frac{e^{-\alpha s_1}\ e^{-\beta s_2}}{(s_1+a)(s_2+b)}\ C(s) \tag{B4}$$

If $\beta > \alpha$, then

$$Y_2(s) = e^{-(\beta-\alpha)a}\ \frac{e^{-\beta s}C(s)}{(s+a+b)} \tag{B5}$$

If $\alpha > \beta$ then

$$Y_2(s) = e^{-b(\alpha-\beta)}\ \frac{e^{-\alpha s}C(s)}{(s+a+b)} \tag{B6}$$

REFERENCES

[1] V. Volterra, Theory of Functionals and of Integral and Integro-Differential Equations. Dover Publications, New York, 1959.

[2] N. Wiener, Nonlinear Problems in Random Theory. The Technology Press, M.T.T., and John Wiley, New York, 1958.

[3] Y. H. Ku and A. A. Wolfe, "Volterra-Wiener functionals for the analysis of nonlinear systems, " J. Franklin Institute, Vol. 281, pp. 9-26, January 1966.

[4] S. Narayanan, "Application of Volterra series to intermodulation distortion analysis of transistor feedback amplifiers," IEEE Trans. Circuit Theory, Vol. CT-17, pp. 518-523, November 1970.

[5] J. Goldman, "A Volterra series description of crosstalk interference in communication systems," Bell System Technical Journal, Vol. 52, pp. 649-688, May 1973.

[6] K. Y. Chang, "Intermodulation noise and products due to frequency dependent nonlinearities in CATV systems," IEEE Trans. Comm., Vol. COM-23, Janauary 1975.

[7] V. K. Jain, "Advanced technique for blackbox modeling,"Rome Air Development Center Technical Report, RADC-TR-80-343, 1980.

[8] V. K. Jain, "Representation of sequences," IEEE Trans. Audio and Electroacoustics," Vol. AU-19, pp. 515-523, September 1971.

[9] V. K. Jain, "Filter analysis by use of pencil-of-functions," IEEE Trans. Circuits and Systems, Vol. CAS-21, pp. 574-583, September 1974.

[10] W. D. Stanley, Digital Signal Processing. Reston (Prentice-Hall), Reston, 1975.

[11] A. M. Bush, "Some Techniques for the Synthesis of Nonlinear Systems," Sc.D. Thesis, Dept. of Electrical Engineering, M.I.T., 1965.

[12] E. J. Ewen, "Black-box identification of nonlinear Volterra Systems," Ph.D. Dissertation, Syracuse University, December 1975.

[13] C. R. Rao and S. K. Mitra, Generalized Inverse of Matrices, and Applications. New York, Wiley, 1971.

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.