COMPUTATION OF SENSITIVITY MEASURES

By

JACK HAMMOND PRIDGEN

Bachelor of Science
Southern Methodist University
Dallas, Texas
1963

Master of Science
Oklahoma State University
Stillwater, Oklahoma
1965

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
DOCTOR OF PHILOSOPHY
May, 1970

COMPUTATION OF SENSITIVITY MEASURES

Thesis Approved:

Charles M. Bacon
Thesis Adviser

Arthur M. Breipohl

Harold Tristan

Richard L. Cummins

Jeanne Agnew

D. Dunham
Dean of the Graduate College

ACKNOWLEDGMENTS

First, I wish to take this opportunity to thank Dr. A. M. Breipohl and Dr. C. M. Bacon who served as thesis advisers for their interest, assistance, criticism, and encouragement during the course of this study.

The time and effort expended by the other members of my graduate committee, Professors H. T. Fristoe, R. L. Cummins, and Jeanne L. Agnew, is also appreciated. Dr. W. A. Blackwell and Dr. L. L. Grigsby assisted early in the study before leaving Oklahoma State University. Dr. J. M. Walden also graciously supplied his computer program listing and trial problems.

I am very grateful for the financial assistance which I have received during my graduate studies. Sandia Corporation has provided partial support for the research performed here, and the Department of Health, Education, and Welfare provided a three-year graduate fellowship.

In addition, I offer my thanks to Miss Velda Davis (with the capable assistance of her colleague, Mrs. Margaret Estes) for the final typing of this manuscript.

Finally, I would like to express my deep appreciation to my wife, Mary, and my parents whose understanding, encouragement, and sacrifice were invaluable during my graduate studies.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Motivation

As society has become more complex, it has become more difficult to fulfill its needs and desires with simple, unsophisticated machines and devices. Engineering systems have become so complex that in many cases engineers are unable to predict their performance by "pencil and paper" methods. Indeed, in these cases the process of converting a physical system description to a mathematical model for analysis can easily occupy the total available time and capability of the practicing engineer. This modeling process usually consists of:

(a) selecting the major functions to be modeled;

(b) isolating individual constituent parts or components if possible;

(c) constructing a mathematical model for each component which adequately describes it in terms of its system performance; and

(d) formulating the system of equations comprising the mathematical model for the complete physical system from the component models and their interconnections.

After the model has been formulated, much of the remaining analysis and design effort is delegated to automatic computing machinery. At present the potential of analog and digital computers for freeing the

creative talents of scientists and engineers from tedious analysis procedures appears to be virtually unlimited. The advent of computers and their widespread availability has already provided engineers with more time for conceiving and developing new systems of interconnected components. However, more efficient, accurate algorithms are needed to accomplish a "total system analysis" including the formulation of the system mathematical model, its solution and direct evaluation of performance data for use in design.

The celebrated Heisenberg "uncertainty principle" suggests a theoretical limit on the accuracies with which measurements may be made. Even though unlimited resources and time are made available for construction of physical systems, it is impossible to realize and verify exact parameter values. It thus becomes important for engineers to consider the effects which imprecise parameter values manifest on the performance of physical systems. Large amounts of engineering effort are expended in predicting these effects from mathematical models of the systems under consideration with the object of selecting systems which exhibit acceptably low levels of dependency on the actual parameter values.

Quite often the designer must set tolerances during design procedures and he is thus placed in the untenable position of specifying tolerance limits without the required skills or time to analyze the system for parameter variation. Often he resorts to the extreme of over-specifying parameter tolerances, i.e, he requires a one percent value where a ten percent value would have been quite acceptable. This may be disastrous since the cost of components tends to increase exponentially with the decrease in tolerance limits.

Extensive literature exists on the problems of analyzing parameter variation effects and synthesizing systems less sensitive to parameter variations. The common theme throughout this literature is the notion of partial derivatives or derivatives of system variables with respect to the parameters that vary. In spite of the extensive literature available, sensitivity analysis remains a difficult, time consuming task for the practicing engineer. Algebraic difficulties are encountered in all but the simplest problems, and in order to save time the engineer normally must choose a single criteria of sensitivity ignoring the multitude of other suggested criteria. Thus, it seems highly desirable to attempt to provide new methodology in sensitivity analysis to improve the designer's capability in tolerance specification. This dissertation reports on research undertaken to provide this new methodology in the form of a computer-aided design tool.

1.2 Scope of Study

The primary objectives of this study are:

(a) to survey the important concepts and techniques of sensitivity theory and to identify those common characteristics of significance in the development of a truly general design tool;

(b) to investigate the application of some well known numerical techniques to sensitivity studies; and

(c) to develop a design tool, capable of supplying most sensitivity measures with a minimum of engineering effort.

Chapter II introduces the most widely used definitions of sensitivity and interrelates them. Sensitivity models useful in the

classical transform domain and the time domain are presented. Probabilistic and deterministic sensitivities are considered as are single parameter and multi-parameter sensitivities. Important relations between certain of these sensitivities are cited which provide a basis for the computational algorithms to be implemented in the design tool.

The state-space model was selected as the basic system description for the following reasons:

(a) the state variable method is extremely general and is capable of dealing effectively with time varying systems and nonlinear systems;

(b) the problem of formulating the system of first order differential equations in vector form is mainly topological in nature and may be easily accomplished on the computer. The formulation rules may be stated very concisely and methodically by use of linear graph theory and are applicable to a broad class of electrical and non-electrical lumped-parameter systems;

(c) solution techniques for both linear and nonlinear models may be implemented efficiently on the computer; and

(d) as will be shown, the state sensitivity model may be formulated directly from the state-space model and its solutions provide the derivatives with respect to parameters which are so crucial to sensitivity analysis.

In order to accomplish the objective of providing a useful day-to-day design tool, it is necessary to generate a computer program capable of providing the desired information for decision-making in the design process. Kuo (1) states that general analysis programs should be

capable of steady-state dc and ac analysis as well as transient analysis. Such programs should possess the following desired features:

(a) The program should have a convenient, simple input language to describe the system element types, their associated parameters and their interconnected pattern in the system.

(b) The ideal program should be able to handle a wide class of models of physical devices with a capability of changing not only parameters of a device but also its topological model.

(c) It is desirable to be able to deal with nonlinearities in either piecewise linear form or by means of a specified mathematical function.

(d) There should be a large number of output options in a compact format.

(e) There should be some feature of automatic parameter modification for sensitivity, tolerance and worst-case studies.

(f) The program should contain error checks on the reliability of the output.

The research and development activity presented in this dissertation has produced two programs capable of handling linear and nonlinear systems which include the above features. These programs are implemented in FORTRAN to facilitate their implementation on different computers with "batch" processing mode of execution allowed. The linear program, described in Chapter III, will evaluate single and multi-parameter sensitivities as well as a probabilistic measure in the time domain. Pole-zero sensitivities are also readily available as outputs. Chapter IV describes the nonlinear program and the various

types of nonlinearities that can be implemented. Chapter V presents several practical examples of both linear and nonlinear systems to illustrate the broad capabilities of the design tool. A summary of the dissertation is given in Chapter VI together with some conclusions and suggestions for further study.

The appendices provide developments of a detailed nature not suitable for inclusion in the main body of the dissertation. Appendix A describes a new technique for obtaining the simultaneous impulse solutions of the linear time-invariant state model and its sensitivity model. A program for furnishing pole-zero sensitivity information for a multi-variable system described by a linear time-invariant state model is presented in Appendix B. Appendix C states and provides proof for a theorem that is used in the development of the design tool programs in order to decrease the required computer storage. These programs are documented by Appendices D and E for the linear program and the nonlinear program, respectively.

CHAPTER II

SENSITIVITY MEASURES AND THEIR EVALUATION

2.1 Introduction

The purposes of this chapter are twofold:

(a) to present the fundamental concepts of sensitivity theory and to identify those characteristics common to the many sensitivity measures; and

(b) to introduce the notion of the sensitivity operator and to demonstrate its usefulness in the evaluation of a wide variety of sensitivity measures in both the time domain and the frequency domain.

This chapter provides the analytical foundation for the development of the computer programs discussed in Chapter III and Chapter IV.

In the process of performing a sensitivity analysis, the first problem to be resolved is the choice of a sensitivity measure or definition. This chapter describes many of the measures that have been suggested in the literature. The author's intent is not to advocate any particular sensitivity measure or definition as being superior to the others. Instead a useful design tool has been developed that is capable of providing many of the definitions discussed. Thus, the practicing engineer can use the definition best suiting his particular problem.

Section 2.2 presents definitions of sensitivity that are based on transform models. Classical deterministic frequency domain definitions are presented as well as some probabilistic measures. In Section 2.3 the sensitivity operators are introduced, their form derived, and methods of obtaining time solutions for general and impulse drivers are discussed. An example is included to illustrate the meaning of these operators. The general form of useful sensitivity measures that make use of these operators is presented in Section 2.4. Deterministic measures and a measure based on the mean square error due to component variation are presented. This measure may be applied in either the frequency or time domain. In Section 2.5, the fundamental relationships between the time domain and frequency domain linear models are discussed. A method of finding the pole-zero sensitivities is presented and extensions are made which are useful in obtaining other standard sensitivities.

2.2 Frequency Domain Sensitivity Analysis

In the past many measures of sensitivity have been suggested. A large number of these measures are defined by the manner in which a system transfer function changes as one or more parameters vary.

The basic idea of single element sensitivity was first formulated by Bode (2). He defined the complex number

$$S_{p(\text{Bode})}^{T(j\omega)} = \left[\frac{d(\ell n\ T(j\omega))}{d(\ell n p)}\right]^{-1} \qquad (2.2.1)$$

to be the sensitivity of the transfer function $T(j\omega)$ with respect to the varying parameter p where ω is the radian frequency. Mason (3)

took the reciprocal of Equation (2.2.1) to define sensitivity in the study of feedback theory as

$$S_p^{T(j\omega)} = \frac{d(\ell n\ T(j\omega))}{d(\ell n\, p)} \quad . \tag{2.2.2a}$$

This definition of sensitivity is by far the most commonly used one for linear systems. By noting that for differential changes in p the sensitivity of Equation (2.2.2a) may be given as

$$S_p^{T(j\omega)} = \frac{p \cdot d\,[\ell n\ T(j\omega)]}{dp}$$

then

$$S_p^{T(j\omega)} = p\,\frac{dA(\omega)}{dp} + jp\,\frac{d\ Ph(\omega)}{dp} \tag{2.2.2b}$$

where

$$A(\omega) = \ell n\ |T(j\omega)|$$

$$Ph(\omega) = \text{Arg}\ T(j\omega) \quad .$$

The Mason sensitivity of the attenuation characteristic is

$$S_p^{A(\omega)} = \frac{p \cdot d\ A(\omega)}{A(\omega) \cdot dp} = \frac{\text{Re}\left[S_p^{T(j\omega)}\right]}{A(\omega)} \quad . \tag{2.2.2c}$$

Similarly the Mason sensitivity of the phase characteristic is

$$S_p^{Ph(\omega)} = \frac{\text{Im}\left[S_p^{T(j\omega)}\right]}{Ph(\omega)} \quad .$$

A large class of system functions are bilinear functions of the parameter p. These system functions can be expressed as

$$T(s) = \frac{N_1(s) + p\ N_2(s)}{D_1(s) + p\ D_2(s)} = \frac{N(s)}{D(s)} \tag{2.2.3}$$

where $N_1(s)$, $N_2(s)$, $D_1(s)$, and $D_2(s)$ are functions of the complex

variable s and are independent of the parameter p. For these functions Equation (2.2.2a) can be written as

$$S_p^{T(j\omega)} = \frac{D_1(j\omega)}{D(j\omega)} - \frac{N_1(j\omega)}{N(j\omega)}$$

and

$$\frac{dA(\omega)}{dp} = \mathrm{Re}\left[\frac{N_2(j\omega)}{N(j\omega)} - \frac{D_2(j\omega)}{D(j\omega)}\right]$$

$$\frac{d\,Ph(\omega)}{dp} = \mathrm{Im}\left[\frac{N_2(j\omega)}{N(j\omega)} - \frac{D_2(j\omega)}{D(j\omega)}\right] .$$

These formulas can be used to facilitate calculation of the sensitivities.

As an alternate means of conducting frequency domain sensitivity analysis, the notion of pole-zero sensitivity has been developed. This sensitivity is very useful when dealing with poles and zeros near the $j\omega$ axis. Work in this area has been performed by Papoulis, Truxal, Howowitz, Ur, Calahan, and Huelsman, to mention only a few (4, 5, 6, 7, 8, 9).

Consider the zero sensitivity first and let the system transfer function have the form

$$T(s) = \frac{K_o \prod_{i=1}^{n} (s - z_i)}{\prod_{j=1}^{m} (s - \lambda_j)}$$

where K_o is a term depending only on p. Then by definition, the zero sensitivity with respect to p is

$$S_p^{z_i} = \frac{dz_i}{d(\ell n\, p)} = p\frac{dz_i}{dp} \quad . \qquad (2.2.4)$$

The pole sensitivity is similarly defined. These definitions of pole and zero sensitivities are used rather than a percentage change in the location of the pole or zero as implied by Equation (2.2.2a) because the poles and zeros may occur at the origin. For bilinear system functions Equation (2.2.4) takes the well-known form

$$S_p^{z_i} = \left. \frac{-p\, N_2(s)}{N(s)/(s-z_i)} \right|_{s=z_i} .$$

The relation between the classical sensitivity of Equation (2.2.2) and that of Equation (2.2.4) was shown by Ur (7) to be

$$S_p^{T(s)} = \sum_{\ell=1}^{m} \frac{S_p^{\lambda_\ell}}{s - \lambda_\ell} - \sum_{j=1}^{n} \frac{S_p^{z_j}}{s - z_j} + \frac{1}{K_o} S_p^{K_o} \qquad (2.2.5)$$

where

$$S_p^{K_o} = p \cdot \frac{dK_o}{dp} .$$

The results cited above are applicable only when single parameter variation is assumed. However, the sensitivity analysis of practical systems must allow consideration of multiparameter variation. For example, it is important for the designer to know whether the separate influences of several parameter changes on the system function tend to add or cancel. To take this phenomenom into account several multiparameter sensitivity measures have been introduced.

Goldstein and Kuo (10) extended Mason's definition of single element parameter sensitivity to the multiparameter case. Let $T(s, x_1, x_2, \ldots, x_n)$ be a function of n parameters $x_1, x_2, \ldots, x_n$, where s, the complex frequency, is considered as a fixed variable. Then the multiparameter sensitivity $\underline{S}^T$ is given by

$$\underline{S}^T = \nabla_{\ell n\, \underline{x}}\, \ell n\, T(s, x_1, x_2, \ldots, x_n)$$

$$= \left[\frac{d(\ell n\, T)}{d(\ell n\, x_1)} \quad \frac{d(\ell n T)}{d(\ell n x_2)} \quad \cdots \quad \frac{d(\ell n\, T)}{d(\ell n\, x_n)}\right]$$

$$= \left[S^T_{x_1} \quad S^T_{x_2} \quad \cdots \quad S^T_{x_n}\right] \quad .$$

For this definition of multiparameter sensitivity, the norm of the $\underline{S}^T$ vector is given by

$$||\underline{S}^T|| = (\underline{S}^T \cdot \overline{\underline{S}^T})^{1/2} \qquad (2.2.6)$$

where $\overline{\underline{S}^T}$ means the complex conjugate transpose of $\underline{S}^T$. $||\underline{S}^T||$ defines the maximum rate of change of $(\ell n\ T)$ with respect to the $(\ell n\ x_i)$. This measure of sensitivity combines the phase and attenuation information into one number for the designer's consideration. If Equation (2.2.6) is evaluated for $s = j\omega$, Goldstein and Kuo's definition of multiparameter sensitivity reduces to that suggested by Calahan (11). However, Calahan points out that by applying Equation (2.2.2b) the measure of the sensitivity of the magnitude characteristic is the vector norm

$$||\mathrm{Re}(\underline{S}^{T(j\omega)})|| = \sqrt{\sum_{i=1}^{n} \left(\frac{\partial A(\omega)}{\partial x_i}\right)^2 x_i^2} \qquad (2.2.7a)$$

and a measure of the sensitivity of the phase characteristic is

$$||\mathrm{Im}(\underline{S}^{T(j\omega)})|| = \sqrt{\sum_{i=1}^{n} \left(\frac{\partial \mathrm{Ph}(\omega)}{\partial x_i}\right)^2 x_i^2} \quad . \qquad (2.2.7b)$$

The measures of Equations (2.2.6), (2.2.7a), and (2.2.7b) are useful only when the parameter variations are small. This definition also assumes the n parameters of T vary independently.

In order to study dependent parameter variation, the concept of the "sensitivity group" has been developed by Lee (12). The basic idea is to study the system first in order to identify any parameters that vary dependently with respect to the system function. If any parameters are found that satisfy this and certain other criteria, these parameters are members of a sensitivity group. Lee's method is restricted to passive RLC networks and requires differentially small parameter variation. The notation of this method is rather cumbersom and for this reason is not included here. The interested reader should refer to the original thesis.

Hakimi and Cruz (13) allow non-differential parameter variations and circumvent the independent variation problem by defining the multi-parameter sensitivity to be given by

$$S^{H(s)} = \max_{\substack{0 \leq |\xi_i| \leq \delta_i \\ 1 \leq i \leq n}} \frac{\left[\dfrac{\Delta H(s)}{H_n(s)}\right]}{\delta_1 \delta_2 \cdots \delta_n}$$

where

δ_i is the per unit tolerance on the parameter x_i,

ξ_i is the per unit variation of x_i from its nominal value,

$H_n(s)$ is the system function evaluated at the nominal values of the parameters x_i, $i = 1, 2, \ldots, n$, and

$\Delta H(s) = \max [H(s)] - \min [H(s)]$ for all possible values of x_i, $i = 1, 2, \ldots, n$.

H(s) may be either the magnitude or phase characteristic of T(s). Note that for a given frequency ω_o, H_n and the tolerances are fixed so that the only variable as the ξ_i's vary is ΔH. However, the determination of this maximum may not be easy since, in general, this maximum does not occur at the extremes of the ξ_i's. As an alternate to this determination Hakimi and Cruz point out that it is possible to find upper and lower bounds on the system function for each frequency ω_o. This is a highly conservative type of worst case analysis.

An interesting approximation to the pole-zero sensitivity for multiparameter variation has been proposed by Huelsman (14). Huelsman's method essentially consists of three steps:

(1) developing an approximate relation between the changes in the pole-zero locations of the system function and the corresponding changes in the coefficients of the denominator and numerator polynomials;

(2) developing a relation between the changes in the values of the coefficients of the polynomials and the changes in the values of the system parameters; and

(3) writing a matrix expression for these sensitivity relations and normalizing it.

This method yields good agreement with the actual changes in the poles and zeros and seems to require less effort than straightforward evaluation of Equation (2.2.4).

Each of the preceding methods implies a deterministic approach to component variation analysis. Sensitivities of a probabilistic nature are less numerous in the literature but three methods have been shown to be useful when computer aided methods are of primary interest.

Hakimi and Cruz (13) suggest that for a transfer function $T(s, x_1, x_2, \ldots, x_n)$, the parameters x_i should be considered to be random variables. Hence, the transfer function itself is a random variable and the following sensitivity measure can be defined

$$S = \frac{\sqrt{\langle \{\Delta |T|\}^2 \rangle}}{T_{nominal}} \Big/ \sigma_1 \sigma_2 \cdots \sigma_n$$

where

$$\sqrt{\langle \{\Delta |T|\}^2 \rangle}$$

is the root-mean-square value of $\Delta |T|$ and $\sigma_1, \sigma_2, \ldots, \sigma_n$ are the standard deviations of the variables $x_1, x_2, \ldots, x_n$. This measure may be evaluated for simple systems by classical probabilistic means. A more reasonable approach for complex systems is to utilize Monte-Carlo techniques and the computer.

A second approach has been suggested by Clunies-Ross and Husson (15). With topology and the variances on the components held fixed, the designer strives to choose the mean value for the component in such a way that the maximum probability of failure is minimized. Initially a Monte-Carlo routine is carried out to obtain data points. The heart of the procedure is to use these data points to form linear approximations by least-mean-squares methods for each circuit output. By assuming the outputs to be gaussian the designer can solve for the required means with the aid of linear programming. This method also allows circuit constraints to be included in the minimization process. The major difficulty in this type analysis is simply the complexity of the approach.

A third probabilistic method, due to Breipohl and Campbell (16),

is described in Sections 2.4 and 2.5. This method makes use of a sensitivity measure based on the mean square error in the frequency domain between a desired transfer function G_o and the actual transfer function G. Computation of the mean square error may be accomplished through the use of an approximation introduced in Section 2.4. This approach seems very reasonable, but algebraic difficulties are encountered in large networks with many varying parameters. Although the frequency domain sensitivities of Breipohl and Campbell are not completely implemented in the program of Chapter III, a computational algorithm is suggested. An extension of this method to the time domain has been made by the author through the use of the "sensitivity operator" concept discussed in the next section.

2.3 Time Domain Sensitivity Operators

In this section sensitivity operators for time domain models are defined and methods for their derivation and solution are presented. An example is also included to illustrate the properties of these operators. The notion of a sensitivity operator is similar to sensitivity "coefficients" introduced by Tomovic (17).

The deterministic performance characteristics of a large class of engineering systems can be described by the state model composed of a set of n first order differential equations

$$\frac{dx_j}{dt} = f_i(\underline{x}, \underline{u}, p, t) \qquad i = 1, 2, \ldots, n \qquad (2.3.1a)$$

plus a set of ℓ algebraic equations

$$y_j = g_j(\underline{x}, \underline{u}, p, t) \qquad j = 1, 2, \ldots \ell \qquad (2.3.1b)$$

where

$\underline{x}$ denotes an n dimensional column vector with components x_j, j = 1, 2, ..., n, which are the state variables;

$\underline{u}$ denotes an m dimensional column vector with components u_j, j = 1, 2, ..., m, which are input forcing functions of time; and

y_j denotes the j-th component of a column vector $\underline{y}$ which is the vector of output functions.

The linear form of Equations (2.3.1a) and (2.3.1b) is

$$\dot{\underline{x}} = \underline{A}\,\underline{x} + \underline{B}\,\underline{u} \qquad (2.3.2a)$$

$$\underline{y} = \underline{C}\,\underline{x} + \underline{D}\,\underline{u} \quad . \qquad (2.3.2b)$$

The matrices $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ may be time-varying, in general, but will be considered constant throughout this thesis. They depend, of course, on the system parameters.

Definition 2.3.1

The state sensitivity operator with respect to the parameter p for the system state equation, Equation (2.3.1a), is the n dimensional vector time function $\underline{v}^p(t)$ with components

$$v_i^p(t) = \frac{dx_i(t)}{dp} \qquad i = 1, 2, \ldots, n \qquad (2.3.3)$$

where x_i is the i-th state variable.

Definition 2.3.2

The output sensitivity operator with respect to the parameter p for the system output equation, Equation (2.3.1b), is the ℓ dimensional vector time function $\underline{w}^p(t)$ with components

$$w_i^p(t) = \frac{dy_i(t)}{dp} \qquad i = 1, 2, \ldots, \ell \qquad (2.3.4)$$

where y_i is the i-th output variable.

2.3.1 Linear Analysis

Consider the linear model of Equation (2.3.2). It is desired to calculate the state and output sensitivity operators with respect to the parameter p. Consider the matrix function

$$\underline{M}(p) = \underline{N}(p)\ \underline{L}(p)$$

then

$$\underline{M}'(p) = \underline{N}(p)\ \underline{L}'(p) + \underline{N}'(p)\ \underline{L}(p)$$

where

$$\underline{M}'(p) = \frac{d\underline{M}(p)}{dp} \quad .$$

This notation will be used throughout this thesis. Hence, from Equation (2.3.2a)

$$\frac{d}{dp}\ \dot{\underline{x}} = \underline{A}\ \frac{d\underline{x}}{dp} + \underline{A}'\underline{x} + \underline{B}\ \underline{u}' + \underline{B}'\underline{u} \quad .$$

It is assumed that all derivatives exist over the domain of the parameter p.

Then

$$\frac{d}{dt}\left[\frac{d\underline{x}}{dp}\right] = \underline{A}\ \frac{d\underline{x}}{dp} + \underline{A}'\underline{x} + \underline{B}\ \underline{u}' + \underline{B}'\underline{u}$$

or

$$\dot{\underline{v}}^p = \underline{A}\ \underline{v}^p + \underline{Z} \qquad (2.3.5a)$$

where

$$\underline{Z} = \underline{A}'\ \underline{x} + \underline{B}'\ \underline{u} + \underline{B}\ \underline{u}' \quad . \qquad (2.3.5b)$$

Note $\underline{Z}$ depends upon $\underline{x}(t)$, the solution of the original state equation, Equation (2.3.2a). The output sensitivity operator may be similarly written as

$$\underline{w}^p = \underline{C}\,\underline{v}^p + \underline{C}'\,\underline{x} + \underline{D}\,\underline{u}' + \underline{D}'\,\underline{u} \quad . \tag{2.3.5c}$$

Equations (2.3.5a), (2.3.5b), and (2.3.5c) then are the relations whose solutions yield the state and output sensitivity operators.

The state and output sensitivity operators may be used to approximate the solution to the state model when a parameter is perturbed from its nominal value. Let $\underline{x}\,(t, p_o)$ be the true solution when parameter p takes on value p_o and $\underline{x}(t)$ be the solution when the parameter takes on its nominal value p_{nom}. Then

$$\underline{x}(t,p_o) \approx \underline{x}(t) + \underline{v}^p(t)\,\Delta p$$

where

$$\Delta p = p_o - p_{nom} \quad .$$

This type of approximation is more nearly accurate for differentially small variation in the value of the parameter and little may be said, in general, about the accuracy of this approximation for large parameter variations. An example of this type of approximation is presented in Section 2.3.3.

Let us now consider two means of obtaining the solution to the linear state model and its associated sensitivity model. The sensitivity operator equations contain terms dependent on the nominal solution $\underline{x}(t)$. Thus, the solution to the state model must be available before the solution to the sensitivity operator differential equation can be found.

One method is to write Equations (2.3.2a) and (2.3.5a) in the form

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\underline{v}}^p \end{bmatrix} = \begin{bmatrix} \underline{A} & \underline{0} \\ \underline{A}' & \underline{A} \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{v}^p \end{bmatrix} + \begin{bmatrix} \underline{B} & \underline{0} \\ \underline{B}' & \underline{B} \end{bmatrix} \begin{bmatrix} \underline{u} \\ \underline{u}' \end{bmatrix} \tag{2.3.6}$$

and Equations (2.3.2b) and (2.3.5c) as

$$\begin{bmatrix} \underline{y} \\ \underline{w}^p \end{bmatrix} = \begin{bmatrix} \underline{C} & \underline{0} \\ \underline{C}' & \underline{C} \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{v}^p \end{bmatrix} + \begin{bmatrix} \underline{D} & \underline{0} \\ \underline{D}' & \underline{D} \end{bmatrix} \begin{bmatrix} \underline{u} \\ \underline{u}' \end{bmatrix} . \tag{2.3.7}$$

This allows the two models to be solved simultaneously. The solution of the matrix system, Equation (2.3.6), may be then symbolized as

$$\begin{bmatrix} \underline{x}(t) \\ \underline{v}^p(t) \end{bmatrix} = \underline{\theta}(t-a) \begin{bmatrix} \underline{x}(a) \\ \underline{v}^p(a) \end{bmatrix} + \int_a^t \underline{\theta}(t-\lambda) \begin{bmatrix} \underline{B} & \underline{0} \\ \underline{B}' & \underline{B} \end{bmatrix} \begin{bmatrix} \underline{u}(\lambda) \\ \underline{u}'(\lambda) \end{bmatrix} d\lambda \tag{2.3.8}$$

where $\underline{x}(a)$ and $\underline{v}^p(a)$ are the initial values of the state and sensitivity operator vectors at $t = a$ and $\underline{\theta}(t)$ is the state transition matrix. For the linear fixed system considered here, $\underline{\theta}(t)$ is the matrix exponential

$$\underline{\theta}(t) = e^{\underline{F}t} = \sum_{k=0}^{\infty} \frac{t^k}{k!} \underline{F}^k \tag{2.3.9}$$

where

$$\underline{F} = \begin{bmatrix} \underline{A} & \underline{0} \\ \underline{A}' & \underline{A} \end{bmatrix} .$$

The augmentation process of Equations (2.3.6) and (2.3.7) is illustrated

in Figure 1. Equation (2.3.7) can be used to find the output sensitivity operator after Equation (2.3.8) has been evaluated.

Another method for obtaining the time solution to the state sensitivity operator is a sequential one. In this method the state equation, Equation (2.3.2a), is solved over the time interval of interest to yield $\underline{x}(t)$. This term may be used in Equation (2.3.5) for calculation of the state sensitivity operator and, hence, the output sensitivity operator. This method has the advantage of fewer equations in the integration process. However, a corresponding disadvantage is the necessity of storing the solution for $\underline{x}(t)$.

The general solution of the augmented linear state model system, Equation (2.3.6), is given in Equation (2.3.8). This solution may be simplified for the case of an impulse function input at driver u_d to yield the algebraic vector $\underline{y}$ and its change $\underline{w}^p(t)$ for zero initial conditions. This solution is

$$\begin{bmatrix} \underline{y}(t) \\ \underline{w}^p(t) \end{bmatrix}_d = \left\{ \begin{bmatrix} \underline{C} & \underline{0} \\ \underline{C}' & \underline{C} \end{bmatrix} \underline{\theta}(t) \begin{bmatrix} \underline{B} \\ \underline{B}' \end{bmatrix} + \begin{bmatrix} \underline{D} \\ \underline{D}' \end{bmatrix} \delta(t) \right\} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_n \end{bmatrix} \tag{2.3.10}$$

where

$$\gamma_i = \begin{cases} 0 & \text{if } i \neq d \\ 1 & \text{if } i = d \end{cases} \qquad i = 1, 2, \ldots, m$$

and $\theta(t)$ is given in Equation (2.3.9). The equation follows from the development presented in Appendix A.

2.3.2 Nonlinear Analysis

Consider now the nonlinear state model given by Equation (2.3.1).

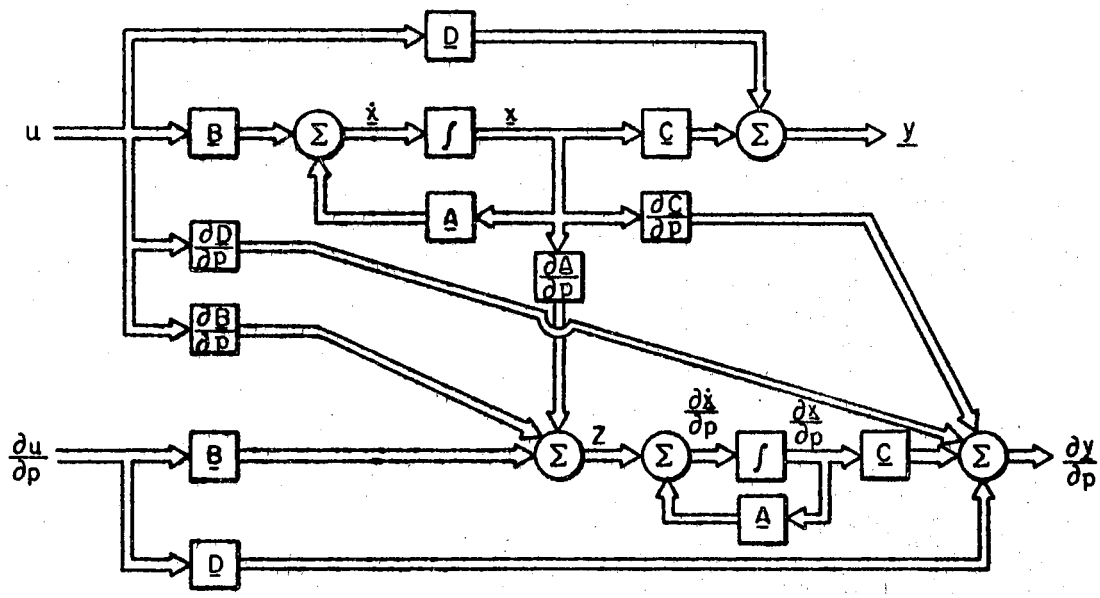


Figure 1. Diagram of Operations for Calculation of Sensitivity Operators

It is desired to calculate the state sensitivity operator and the output sensitivity operator with respect to the parameter p. Several approaches to this problem may be taken.

In many nonlinear situations, the nonlinear equations can be linearized by limiting attention to small variations or perturbations about a nominal state solution. Although the system differential equations may be highly nonlinear, the differential equations describing these variations around the nominal can be considered to be linear to a first order approximation. This approach to linearizing the state model is to expand the functions $f_i(\underline{x}, \underline{u}, t)$, $i = 1, 2, \ldots, n$, and $g_j(\underline{x}, \underline{u}, t)$, $j = 1, 2, \ldots, \ell$, in a Taylor's series and delete the second and higher order terms. This yields

$$\dot{\underline{X}} = \underline{A}\,\underline{X} + \underline{B}\,\underline{U} \tag{2.3.11a}$$

$$\underline{Y} = \underline{C}\,\underline{X} + \underline{D}\,\underline{U} \tag{2.3.11b}$$

where

$$\underline{A} = \left[a_{ij}\right] = \left[\left.\frac{\partial f_i}{\partial x_j}\right|_{\underline{x}_n,\underline{u}_n}\right] \qquad \underline{C} = \left[c_{ij}\right] = \left[\left.\frac{\partial g_i}{\partial x_j}\right|_{\underline{x}_n,\underline{u}_n}\right]$$

$$\underline{B} = \left[b_{ij}\right] = \left[\left.\frac{\partial f_i}{\partial u_j}\right|_{\underline{x}_n,\underline{u}_n}\right] \qquad \underline{D} = \left[d_{ij}\right] = \left[\left.\frac{\partial g_i}{\partial u_j}\right|_{\underline{x}_n,\underline{u}_n}\right]$$

$$\underline{X} = \underline{x} - \underline{x}_n$$

$$\underline{U} = \underline{u} - \underline{u}_n$$

$$\underline{Y} = \underline{y} - \underline{y}_n$$

and $\underline{x}_n$, $\underline{u}_n$, $\underline{y}_n$ are the state, driver, and output vectors evaluated along the nominal solution. Since Equation (2.3.11) has linear form

the results of the previous discussion can be applied and the sensitivity operators for the linearized model can be evaluated from Equation (2.3.5).

A second method for finding the derivatives of interest entails no approximation as does the previous method (18). The state model of Equation (2.3.1) may be manipulated with the classical methods of differential calculus. By differentiating Equation (2.3.1a) with respect to the parameter p and assuming the vector $\underline{u}$ is independent of p, the following result is obtained.

$$\begin{bmatrix} \frac{\partial \dot{x}_1}{\partial p} \\ \frac{\partial \dot{x}_2}{\partial p} \\ \vdots \\ \frac{\partial \dot{x}_n}{\partial p} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1}{\partial p} \\ \frac{\partial x_2}{\partial p} \\ \vdots \\ \frac{\partial x_n}{\partial p} \end{bmatrix} + \begin{bmatrix} \frac{\partial f_1}{\partial p} \\ \frac{\partial f_2}{\partial p} \\ \vdots \\ \frac{\partial f_n}{\partial p} \end{bmatrix} . \qquad (2.3.12a)$$

This equation may be symbolized as

$$\underline{\dot{v}}^p(t) = \underline{J}\,\underline{v}^p(t) + \underline{Z}$$

where the gradient of $\underline{f}$ with respect to $\underline{x}$ is the Jacobian matrix $\underline{J}$, $\underline{v}^p(t)$ is the state sensitivity operator, and

$$\underline{Z} = \frac{\partial \underline{f}}{\partial p} .$$

This method is easily seen to reduce to that of Equations (2.3.5a) and (2.3.5b) when the functions $f_i(\underline{x}, \underline{u}, t)$, $i = 1,2, \ldots, n$, and $g_j(\underline{x}, \underline{u}, t)$, $j = 1, 2, \ldots, \ell$, of Equation (2.3.1) are linear in the components of

the $\underline{x}$ vector and $\underline{u}$ vector. The development of the algebraic output sensitivity operator is an exact parallel of the procedure shown above. The output sensitivity operator model is

$$\begin{bmatrix} \frac{\partial y_1}{\partial p} \\ \frac{\partial y_2}{\partial p} \\ \vdots \\ \frac{\partial y_\ell}{\partial p} \end{bmatrix} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial g_\ell}{\partial x_1} & \frac{\partial g_\ell}{\partial x_2} & \cdots & \frac{\partial g_\ell}{\partial x_n} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1}{\partial p} \\ \frac{\partial x_2}{\partial p} \\ \vdots \\ \frac{\partial x_n}{\partial p} \end{bmatrix} + \begin{bmatrix} \frac{\partial g_1}{\partial p} \\ \frac{\partial g_2}{\partial p} \\ \vdots \\ \frac{\partial g_\ell}{\partial p} \end{bmatrix} \qquad (2.3.12b)$$

or

$$\underline{w}^p(t) = \underline{J}_o \underline{v}^p(t) + \underline{Z}_o$$

where $\underline{J}_o$ is the gradient of $\underline{g}$ with respect to $\underline{x}$ and $\underline{Z}_o$ is $\frac{\partial \underline{g}}{\partial p}$.

Attention is now turned to the solution of Equations (2.3.11) and (2.3.12). The sensitivity operator solutions to the linearized nonlinear problem of Equation (2.3.11) may be found by the linear techniques of Section 2.3.1. However, in trying to find the solution to Equation (2.3.12) it should be noted that this is a nonlinear system of equations. Equation (2.3.12) and Equation (2.3.1a) may be solved simultaneously. If it is desirable to know the variation of the state vector for each parameter p_k, $k = 1, 2, \ldots, q$, parallel programs may be utilized to solve the $q + 1$ sets of equations or the problem may be repeated q times.

2.3.3 Example

In order to clarify the concept of the sentitivity operator, an

example problem is considered next. It is instructive to compare the predicted response with the actual response for a parameter variation of five percent. Consider the system of Figure 2. The differential equation is

$$\ddot{x} + a\,\dot{x} + b\,x = m(t) \qquad (2.3.13)$$

and

$$y = x \quad .$$

By letting

$$x_1 = x$$

$$x_2 = \dot{x}$$

then

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -b & -a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} m(t)$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} m(t)$$

and the transition matrix $\underline{\beta}(t)$ is given for $a^2 > 4b$ by

$$\underline{\beta}(t) = \frac{1}{\sqrt{a^2 - 4b}} \begin{bmatrix} (\lambda_1 - a)e^{-\lambda_1 t} - (\lambda_2 - a)e^{-\lambda_2 t} & e^{-\lambda_2 t} - e^{-\lambda_1 t} \\ b(e^{-\lambda_1 t} - e^{-\lambda_2 t}) & \lambda_1 e^{-\lambda_1 t} - \lambda_2 e^{-\lambda_2 t} \end{bmatrix}$$

where

$$\lambda_1 = a/2 + \tfrac{1}{2}\sqrt{a^2 - 4b}$$

$$\lambda_2 = a/2 - \tfrac{1}{2}\sqrt{a^2 - 4b} \quad .$$

For

$$m(t) = 0 \quad \text{and} \quad \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \underline{x}(0)$$

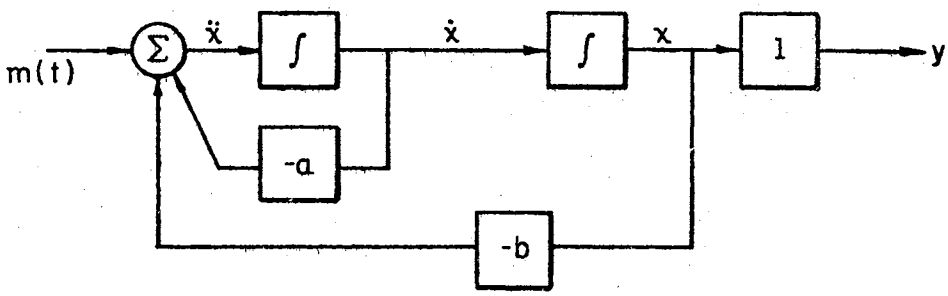


Figure 2. Example System

the solution for $\underline{x}(t)$ becomes

$$\underline{x}(t) = \underline{\beta}(t)\ \underline{x}(0) \quad .$$

From Equation (2.3.5a) the state sensitivity operator model is $\dot{\underline{v}}^p = \underline{A}\ \underline{v}^p + \underline{A}'\ \underline{x}$ and, hence, the state sensitivity operator solution may be written as

$$\underline{v}^p(t) = \underline{\beta}(t)\ \underline{v}^p(0) + \int_0^t \underline{\beta}(t-\lambda)\ \underline{A}'\ \underline{\beta}(\lambda)\ \underline{x}(0)\ d\lambda \quad .$$

For the values

$$a = 4 \qquad \text{and} \qquad b = 3$$

then

$$\underline{\beta}(t) = \tfrac{1}{2} \begin{bmatrix} 3e^{-t} - e^{-3t} & e^{-t} - e^{-3t} \\ 3e^{-3t} - 3e^{-t} & 3e^{-3t} - e^{-t} \end{bmatrix} \quad .$$

Consider now that the parameter of interest is the value of the element b where

$$b = 3(1 + \xi)$$

and

$$\xi = 0.05 \quad .$$

For

$$c_1 = 1 \qquad v_1^p(0) = 0$$

$$c_2 = 0 \qquad v_2^p(0) = 0$$

then

$$\begin{bmatrix} v_1^b(t) \\ v_2^b(t) \end{bmatrix} = \begin{bmatrix} -\int_0^t \beta_{12}(t-\lambda)\ \beta_{11}(\lambda)\ d\lambda \\ -\int_0^t \beta_{22}(t-\lambda)\ \beta_{11}(\lambda)\ d\lambda \end{bmatrix}$$

or

$$v_1^b(t) = -\tfrac{1}{4}\{e^{-t}(3t-2) + e^{-3t}(t+2)\}$$

$$v_2^b(t) = -\tfrac{1}{4}\{e^{-t}(-3t+5) + e^{-3t}(-3t-5)\} \quad .$$

The state sensitivity operator components are shown in Figure 3. As pointed out earlier in this section, the sensitivity operator may be used to predict the output when there exists a small variation in the parameter b. This predicted output is

$$\underline{x}(t) \approx \underline{x}(t)\Big|_{b_{nominal}} + \underline{v}^b(t)\Big|_{b_{nominal}} \cdot \xi b_{nominal} \quad . \qquad (2.3.14)$$

For ξ equal to 0.05 and $b_{nominal}$ equal to 3.00, the response x(t) is calculated from Equation (2.3.13) and from Equation (2.3.14) and presented in Figure 4. It is apparent that the agreement between the predicted and actual response is very good for this five percent variation in the value of the parameter b. Figure 5 presents the actual $\Delta x_1(t)$ and the $\Delta x_1(t)$ predicted by the sensitivity operator and Figure 6 presents the actual $\Delta x_2(t)$ and the predicted $\Delta x_2(t)$. It is seen that the agreement is very good for the five percent variation.

In the example above, the method of solving for the sensitivity operator by first finding the solution to the system and then integrating the sensitivity operator is presented. As pointed out earlier, it is also possible to augment the system state model by the sensitivity model as in Equation (2.3.6) and solve both systems simultaneously. The augmented system is

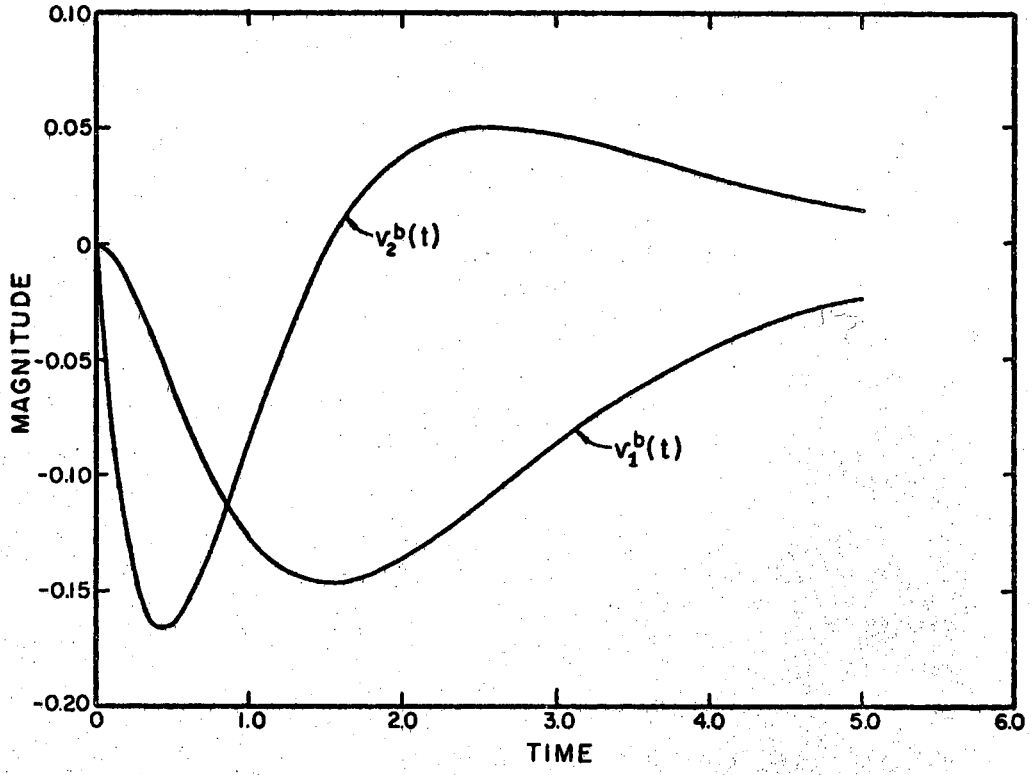


Figure 3. State Sensitivity Operators

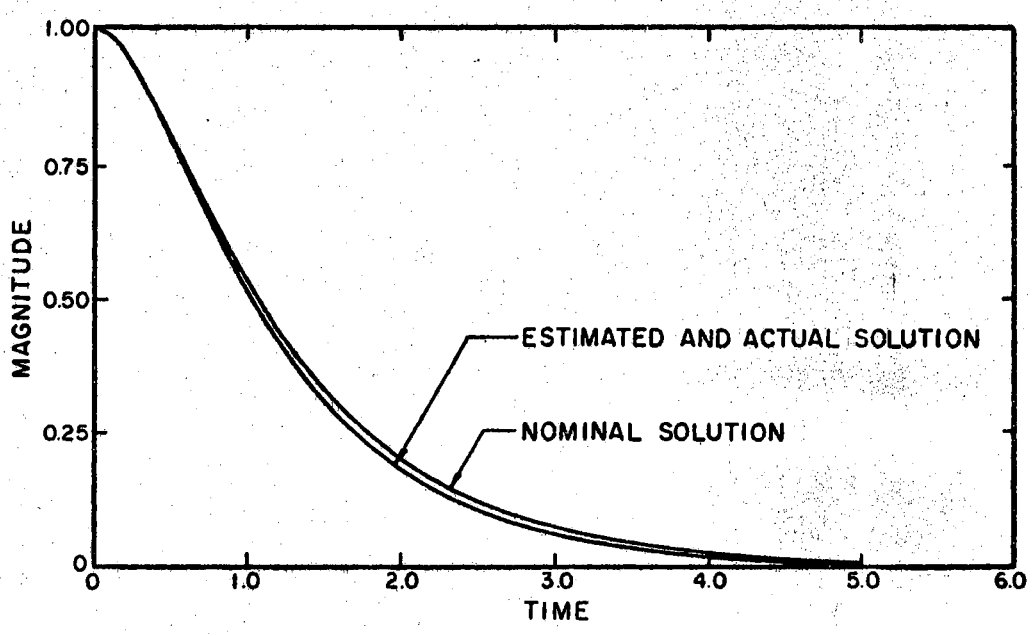


Figure 4. Estimated and Actual Solution for x(t)

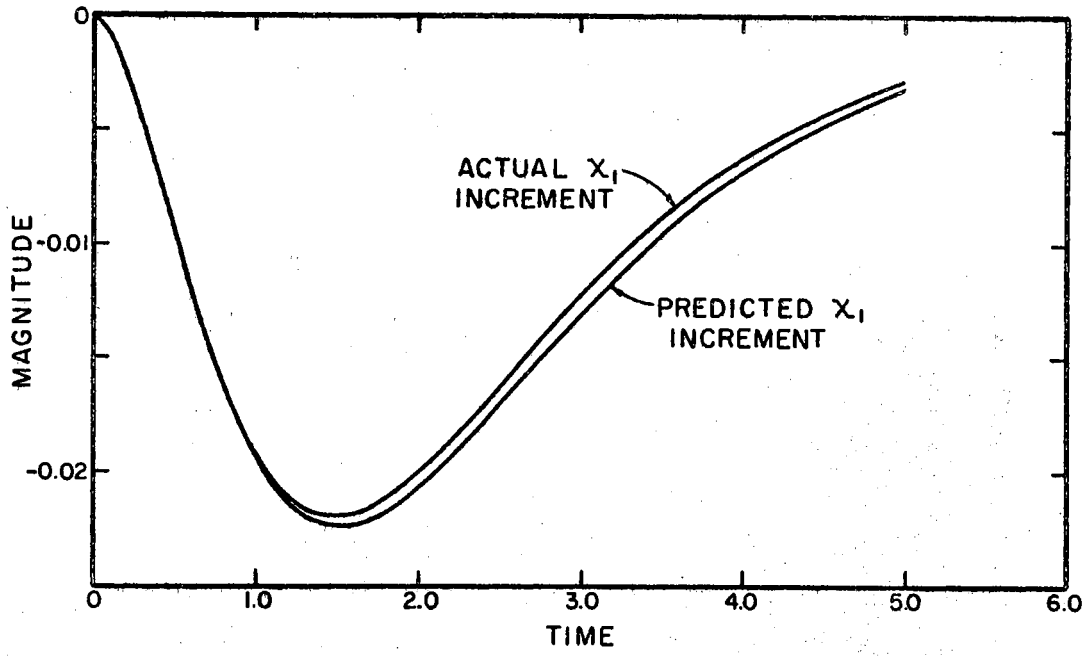


Figure 5. Predicted and Actual x_1 Increments

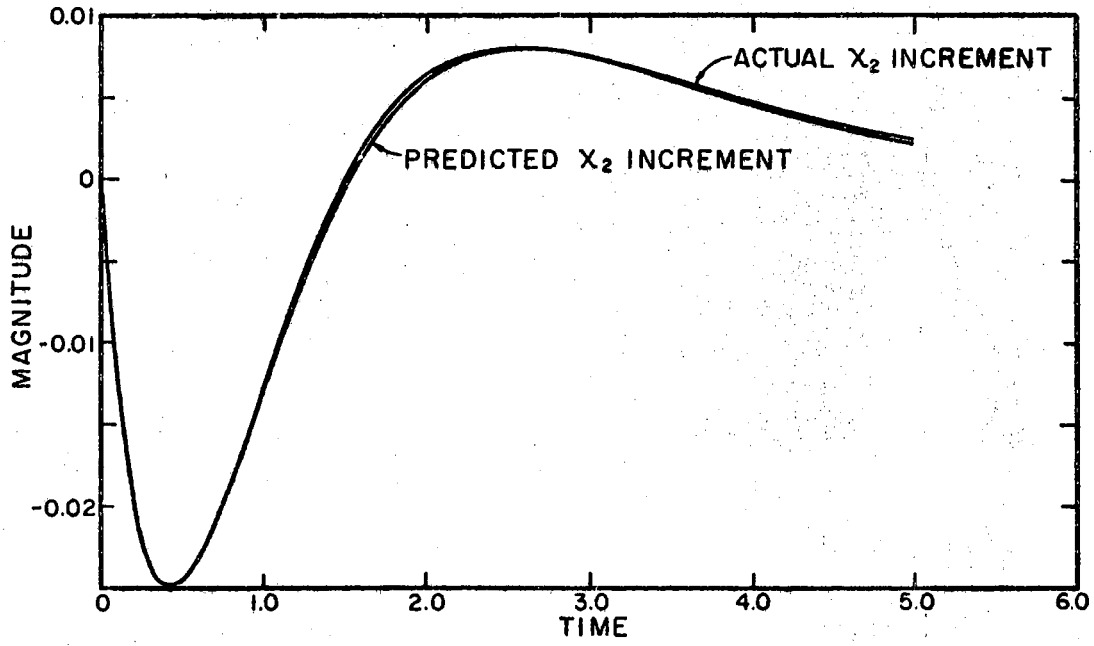


Figure 6. Predicted and Actual x_2 Increments

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{v}_1^b(t) \\ \dot{v}_2^b(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -b & -4 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & -b & -4 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ v_1^b(t) \\ v_2^b(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} m(t) \qquad (2.3.15a)$$

and

$$\begin{bmatrix} y \\ w^b(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ v_1^b(t) \\ v_2^b(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} m(t) \qquad (2.3.15b)$$

Equation (2.3.15) may be solved to yield the same functions as in the previous method by calculation of the transition matrix.

2.4 Time Domain Measures Based on Sensitivity Operators

Useful deterministic and statistical sensitivity measures that directly utilize sensitivity operators in their evaluation are presented in this section. Generally the purpose of sensitivity analysis is to aid in system design selection. Two or more designs may be found to satisfy the given performance criteria but have different sensitivity characteristics. The system designer is then faced with the task of selecting the best design on the basis of a sensitivity measure. He must evaluate this measure for each design in order to make meaningful comparisons. The automatic formulation and calculation of the sensitivity operators allows the designer to examine the sensitivity of many systems with a minimum of effort through the use of the time domain measures discussed in this section.

Suppose that the solutions to both the system state model and the sensitivity model have been found for k variable parameters. The variation in the state vector is evaluated at the parameter vector

$$\underline{p} = [p_1 \quad p_2 \quad \cdots \quad p_k] \tag{2.4.1}$$

and the change in the state vector $\underline{x}$, due to a small change in $\underline{p}$, is given by the n x k matrix

$$\frac{\partial \underline{x}}{\partial \underline{p}} = [\underline{v}^{p_1}(t) \quad \underline{v}^{p_2}(t) \quad \cdots \quad \underline{v}^{p_k}(t)]$$

where $\underline{v}^{p_i}(t)$ is defined in Definition 2.3.1. The general form of a useful deterministic sensitivity measure has been given by Siljak and Dorf (19)

$$I = \int_o^{t_f} \underline{K}(\underline{v}^{p_1}(t), \underline{v}^{p_2}(t), \ldots, \underline{v}^{p_k}(t))\, dt \quad .$$

A specific example of this type of index might be

$$I = \int_o^{t_f} (\underline{v}^{p_1}(t))^2 + (\underline{v}^{p_2}(t))^2 + \ldots + (\underline{v}^{p_k}(t))^2 \quad dt$$

where

$$(\underline{v}^{p_i}(t))^2 = \left[\underline{v}^{p_i}(t)\right]^T \cdot \underline{v}^{p_i}(t) \quad .$$

It should be noted that the state model approach has found wide application in optimal control problems. The optimal control problem is generally taken as that of minimizing a given performance index where

$$J = \int_o^{t_f} g(\underline{x}, \underline{u}, t)dt$$

subject to constraints on the control, $\underline{u}$. The function $g(\underline{x}, \underline{u}, t)$ is defined to produce desired behavior on the trajectory of the state (e.g., minimum error) and desired characteristics on the control $\underline{u}$. The system may, however, also be designed to take into account the sensitivity information (19). The approach is to define a new index such as

$$I = J + \mu J_s \tag{2.4.2}$$

where

J is the classical optimization index,

μ is the weighting factor, and

J_s is a specified sensitivity index.

The combined index may be solved by the standard methods of optimization. A particular example of a desirable index is that of quadratic functions of the variables. Since the sensitivities of certain state variables may be critical, a measure may be used which weights $\frac{\partial x_i}{\partial p_j}$ individually. An example of such a measure is

$$I = \int_0^{t_f} \left[\underline{x}^T \underline{M} \underline{x} + \underline{u}^T \underline{N} \underline{u} + (\underline{w}_1^T \underline{v}^{p_1})^2 + \ldots + (\underline{w}_k^T \underline{v}^{p_k})^2 \right] dt$$

where $\underline{w}_i^T = \left[w_1 \quad w_2 \quad \ldots \quad w_n \right]$ is the weighting vector for parameter p_i and $\underline{M}$ and $\underline{N}$ are positive definite weighting matrices for the state vector and the control vector. The optimal control law that minimizes this index results in an optimum system that is optimum with respect to both performance and sensitivity on a quadratic basis.

Kokotovic, Bingulac, and Medanic (20) present an example of a fifth order linear system that has been optimized using an index of the form given by Equation (2.4.2). The criterion I was minimized for

distinct values of μ varying between 0 and ∞ by using an analog computer to obtain a "μ trajectory" in parameter space linking the minimum of J and the minimum of J_s. Depending on the relative importance of the indices J and J_s the optimum fell nearer the minimum of J or the minimum of J_s. The investigation of the system performance along this trajectory fully answers the question concerning what can be achieved by parametric optimization with a fixed structure in regard to the minimization of sensitivity.

The measures presented above are based on deterministic sensitivity operators. The rest of this section deals with a statistical method of comparing various designs based on the mean square error. The measure is implemented by making use of a simple linear approximation of the mean and variance. The approximation calls for only the means and covariances of the circuit components and, hence, complete knowledge of the distribution function is not required.

The approach taken here most closely resembles the work of Broome and Young (21), and Breipohl and Campbell (16). Their basic idea uses the mean square error between a desired output and the actual output of the system. This is a good criterion for comparing different design approaches to the system. The approximation used here is simpler than Broome and Young's approximation. This approximation is the same as that used by Breipohl and Campbell in their study of transfer function sensitivity in the frequency domain, and it is also used by Breipohl and Grigsby (22) in their discussion on dependent variations of systems components.

Consider the system function, G (transfer function, impulse response, other response), which is a real function of k continuous

parameters and whose first partial derivatives are continuous,

$$G(\underline{p}) = g(p_1, p_2, \ldots, p_k).$$

Suppose the designer has arrived at a system design which ideally realizes the function G for an input $\underline{F}$. This ideal function is denoted G_o. Let G_N be the function evaluated at the nominal value of the parameter vector $\underline{p}$. The parameters are considered to be random variables. Then the mean square error between the desired function and the actual function G is

$$\begin{aligned} MSE &= E\left[(G - G_o)^2\right] \\ &= E\left[(G - \mu_G)^2\right] + (\mu_G - G_o)^2 \\ &= \sigma_G^2 + (\mu_G - G_o)^2 \end{aligned} \tag{2.4.3}$$

where

$E[\cdot]$ means "the expected value of $\cdot$",

$\mu_G = E[G]$, and

$\sigma_G^2 = E\left[(G - \mu_G)^2\right]$.

A linear approximation (16, 22) to the terms μ_G and σ_G^2 that is both practical and in some cases easy to achieve is

$$\mu_G \approx G_N \tag{2.4.4}$$

and

$$\sigma_G^2 \approx \sum_{i=1}^{k} \sum_{j=1}^{k} \left(\frac{\partial G(\underline{p})}{\partial p_i} \bigg|_{\underline{p} = \underline{\mu}} \right) \left(\frac{\partial G(\underline{p})}{\partial p_j} \bigg|_{\underline{p} = \underline{\mu}} \right) E\left[(p_i - \mu_{p_i})(p_j - \mu_{p_j})\right] \tag{2.4.5}$$

where

$$\underline{\mu} = (\mu_{p_1}, \mu_{p_2}, \ldots, \mu_{p_k})$$

$$\mu_{p_i} = E\left[p_i\right] \equiv \text{Nominal value of } p_i \ ,$$

$$E\left[(p_i - \mu_{p_i})(p_j - \mu_{p_j})\right] = \sigma^2_{p_i} \qquad \text{if } i = j$$

$$= COV\left[p_i p_j\right] \qquad \text{if } i \neq j,$$

and $COV\left[p_i\, p_j\right]$ is read "the covariance of the random variables p_i and p_j." If the random variables p_i and p_j are pairwise independent then

$$COV\left[p_i\, p_j\right] = 0 \qquad \text{for all } i \neq j.$$

In this case Equation (2.4.5) becomes

$$\sigma_G^{\,2} \approx \sum_{i=1}^{k} \left(\frac{\partial G(\underline{p})}{\partial p_i} \bigg|_{\underline{p} = \underline{\mu}} \right)^2 \sigma_{p_i}^{\,2} \ . \qquad (2.4.6)$$

It may happen that the parameters are not independent due to environmental dependence or component or manufacturing dependence. Environmental dependence is the dependence or correlation noted between parameter values due to environmental changes, e.g., changes noted in the values of resistors due to change in the ambient temperature. Component or manufacturing dependence is that correlation noted between parameter values when the parameters are of the same component, e.g., changes noted in parameters when transistors are interchanged. This case may be handled satisfactorily by the approximation in Equation (2.4.5) by evaluating the COV term for the dependent parameters by classical statistical techniques. These techniques will not be discussed here. The reader may refer to page twelve of the "Final Report on Probabilistic Systems Analysis" (22) for a discussion of the appropriate techniques to be used for this type of analysis.

The expected value terms of Equation (2.4.5) may be evaluated either by standard statistical means or perhaps obtained from the manufacturer's data. However, the partial derivative terms

$$\left.\frac{\partial G(\underline{p})}{\partial p_i}\right|_{\underline{p} = \underline{\mu}} \quad , \quad i = 1, 2, \ldots, k$$

still remain to be found. These partials lend themselves well to computer analysis since straight forward analytical techniques soon founder on the algebraic difficulties of this type analysis. These partial derivatives have the appearance of the sensitivity operator if the G function is identified as one of the system state variables or output variables. Previous efforts of Broome and Young (21), and Breipohl and Campbell (16) have been directed at consideration of $G(\omega)$, where ω is the radian frequency. By making use of sensitivity operators it becomes possible to use the approximation of Equation (2.4.5) in the time domain. Both types of terms needed for the variance approximation in Equation (2.4.5) and Equation (2.4.6) may be found and the mean square error of Equation (2.4.3) may be calculated.

It should be noted here that the formulation above in terms of sensitivity operators results in a MSE that is a function of time. Such a function does not serve well as a measure of goodness. Generally, a single number is more desirable as a measure. In this case it seems reasonable that an appropriate measure for each choice of component nominal values and tolerances may be given as

$$I = \int_o^{t_f} W(t) \cdot MSE \; dt \qquad (2.4.7)$$

where $W(t)$ is a general weighting function that the designer may

specify to emphasize the region or regions in which the time response is of the most critical nature for his design. The best design based on this measure then corresponds to the component nominal values and tolerances which minimize I.

This section has presented specific deterministic sensitivity measures which make direct use of the sensitivity operators. A new time domain statistical measure has also been discussed. This measure is based on the mean square error between a desired system function and its actual value. The automatic calculation of this measure is an important feature of the general purpose sensitivity analysis programs developed in this research activity.

2.5 Frequency Domain Measures Based on Sensitivity Operators

This section is concerned with the relation between the complex frequency domain and the sensitivity operators as defined earlier. If the sensitivity operator equations are formulated, then the matrices found in the formulation technique may be utilized to provide sensitivity information in the s-domain in terms of the measures defined in Section 2.2. The approach taken is similar to that of Morgan (23) for developing the pole-zero sensitivities of the transfer function matrix.

Consider the multivariable linear state model of Equation (2.3.2).

$$\dot{\underline{x}} = \underline{A}\,\underline{x} + \underline{B}\,\underline{u} \qquad (2.5.1a)$$

$$\underline{y} = \underline{C}\,\underline{x} + \underline{D}\,\underline{u} \qquad (2.5.1b)$$

where $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ are constant matrices for any particular choice of the parameter that are varying. The transfer function matrix, $\underline{P}(s)$,

may be found by taking the Laplace transform of Equation (2.5.1), assuming zero initial conditions, and expressing the transformed output $\underline{Y}(s)$ in terms of the transformed input $\underline{U}(s)$. Thus

$$\underline{Y}(s) = \underline{P}(s)\,\underline{U}(s)$$
$$\underline{P}(s) = \underline{C}\left[s\,\underline{I} - \underline{A}\right]^{-1}\underline{B} + \underline{D} \tag{2.5.2}$$

where

$\underline{I}$ = the identify matrix.

The development presented here differs from Morgan (23) in that the $\underline{D}$ term is not assumed to be zero.

The calculation of the transfer function matrix $\underline{P}(s)$ can be carried out directly on the computer by a method attributed to Faddeev (24), Gantmacher (25), and Frame (26), independently. Let

$$\underline{R}(s) = \text{Adjoint}\,(s\underline{I} - \underline{A})$$
$$= \underline{R}_o s^{n-1} + \underline{R}_1 s^{n-2} + \ldots + \underline{R}_{n-1} \tag{2.5.3}$$

and

$$g(s) = \left|s\underline{I} - \underline{A}\right|$$
$$g(s) = s^n - h_1 s^{n-1} - \ldots - h_n \tag{2.5.4}$$

so that $(s\underline{I} - \underline{A})^{-1} = \underline{R}(s)/g(s)$. The h_i's and $\underline{R}_i$'s may be found by the following process

$$\underline{A}_1 = \underline{A} \qquad h_1 = \text{tr}\,\underline{A}_1 \qquad \underline{R}_1 = \underline{A}_1 - h_1\,\underline{I} \tag{2.5.5}$$

$$\underline{A}_2 = \underline{A}\,\underline{R}_1 \qquad h_2 = \tfrac{1}{2}\,\text{tr}\,\underline{A}_2 \qquad \underline{R}_2 = \underline{A}_2 - h_2\,\underline{I} \tag{2.5.6}$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$\underline{A}_{n-1} = \underline{A}\,\underline{R}_{n-2} \qquad h_{n-1} = \frac{1}{n-1}\,\text{tr}\,\underline{A}_{n-1} \qquad \underline{R}_{n-1} = \underline{A}_{n-1} - h_{n-1}\,\underline{I} \tag{2.5.7}$$

$$\underline{A}_n = \underline{A}\,\underline{R}_{n-1} \qquad h_n = \frac{1}{n}\,\text{tr}\,\underline{A}_n \qquad \underline{R}_n = \underline{A} - h_n\,\underline{I} = \underline{O} \tag{2.5.8}$$

where

$$\text{tr } \underline{A} = \sum_{i=1}^{n} a_{ii} \quad .$$

It has been shown by Gantmacher (25, page 84) that

$$\underline{P}(s) = \frac{\sum_{i=0}^{n-1} \underline{M}_i \, s^{n-1-i}}{g(s)} + \underline{D}$$

where

$$\underline{M}_i = \underline{C} \, \underline{R}_i \, \underline{B} \qquad i = 0, 1, \ldots, n-1$$

$$\underline{R}_o = \underline{I}$$

and, hence,

$$\underline{P}(s) = \frac{1}{g(s)} \left\{ \underline{D} \, s^n + \sum_{i=0}^{n-1} (\underline{M}_i - h_{i+1} \, \underline{D}) \, s^{n-1-i} \right\} . \qquad (2.5.9)$$

Equation (2.5.8) provides a check on the computation of the h_i's and the $\underline{R}(s)$ matrix of Equation (2.5.3). Equation (2.5.9) may be used to find the transfer function matrix $\underline{P}(s)$.

Morgan has presented formulas for the differential change dh_i in the coefficients of the characteristic polynomial, $g(s)$, for a differential change $d\underline{A}$ in the parameters of the matrix $\underline{A}$. These formulas are true only for the case in which all the roots of $g(s)$, $s_1, s_2, \ldots, s_n$, are distinct. An unsolved problem is the non-distinct roots case. These formulas are

$$dh_1 = \underline{I} * d\underline{A} \qquad (2.5.10)$$

$$dh_2 = \underline{R}_1 * d\underline{A} \qquad (2.5.11)$$

$$\vdots$$

$$dh_n = \underline{R}_{n-1} * d\underline{A} \qquad (2.5.12)$$

where the asterisk indicates the inner product of two matrices, that is

$$\underline{A} * \underline{B} = \underline{a}_1\underline{b}_1^T + \underline{a}_2\underline{b}_2^T + \ldots + \underline{a}_n\underline{b}_n^T$$

where $\underline{a}_i$ is the i-th row of $\underline{A}$ and $\underline{b}_i$ is the i-th column of $\underline{B}$. These changes given in Equations (2.5.10), (2.5.11), and (2.5.12) may be used in Newton's formula for the approximation to the roots of a polynomial to provide the differential change in the zeros of the characteristic equation, g(s). Applying Newton's formula to Equation (2.5.4) one obtains

$$ds_i = \frac{1}{g'(s_i)}\left[dh_1\, s_i^{n-1} + dh_2\, s_i^{n-2} + \ldots + dh_n\right] \qquad (2.5.13)$$

where $g'(s_i)$ is the derivative of g(s) with respect to s, evaluated at s_i. Morgan has shown that Equation (2.5.13) may be rewritten as

$$ds_i = \left[\text{tr}\ \underline{R}(s_i)\right]^{-1}\left[\underline{R}(s_i)\right] * \left[d\underline{A}\right] .$$

This formula allows evaluation of the differential change in the location of the zero of the characteristic equation due to the differential change of the matrix $\underline{A}$.

In order to find the differential changes in the zeros of the numerator functions of the $\underline{P}(s)$ matrix, it is necessary to evaluate the differential changes in the numerator matrices of Equation (2.5.9). Let the coefficient of s^{n-1-i} be given by $\underline{F}_i$. Then

$$\underline{F}_i = \underline{M}_i - \underline{h}_{i+1}\underline{D} \qquad i = 0, 1, \ldots, n-1$$

and

$$\underline{F}_{-1} = \underline{D} \quad .$$

The differential change of $\underline{F}_{-1}$ is $d\underline{D}$ and the differential change of $\underline{F}_i$ is

$$d\underline{F}_i = d(\underline{M}_i - h_{i+1}\,\underline{D})$$

$$d\underline{F}_i = d\underline{M}_i - dh_{i+1}\,\underline{D} - h_{i+1}\,d\underline{D} \quad . \qquad (2.5.14)$$

Expressions for dh_i have been given in Equations (2.5.10), (2.5.11), and (2.5.12). Note that

$$\underline{M}_i = \underline{C}\,\underline{R}_i\,\underline{B} \qquad i = 0,\ 1,\ \ldots,\ n-1$$

and, hence,

$$d\underline{M}_i = d\underline{C}\,\underline{R}_i\,\underline{B} + \underline{C}\,d\underline{R}_i\,\underline{B} + \underline{C}\,\underline{R}_i\,d\underline{B} \quad .$$

For $i = 0$, $\underline{R}_o = \underline{I}$ and, hence, $d\underline{R}_o \equiv 0$. From Equations (2.5.7) and (2.5.8)

$$d\underline{R}_i = d\underline{A}_i - dh_i\,\underline{I} \qquad i = 1,\ 2,\ \ldots,\ n-1 \qquad (2.5.15)$$

and

$$d\underline{A}_i = d\underline{A}\,\underline{R}_{i-1} + \underline{A}\,d\underline{R}_{i-1} \quad . \qquad (2.5.16)$$

Now, successive applications of Equations (2.5.15) and (2.5.16) allow determination of each $d\underline{R}_i$ and, hence, each $d\underline{M}_i$. Equation (2.5.14) then allows determination of each $d\underline{F}_i$. Once these $d\underline{F}_i$ are found it is possible to apply Newton's formula for non-multiple zeros of the numerator functions. The result then is the differential change in the location of the zero due to the $d\underline{A}$, $d\underline{B}$, $d\underline{C}$, and $d\underline{D}$ variations.

Thus far this section has considered the linear time-invariant state model and shown how the transfer function matrix can be calculated. Formulas are given for the changes in the numerator and denominator coefficients of the transfer function matrix. Also

indicated is a method for finding the differential changes in the poles and zeros of the transfer functions due to differential changes in the $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$, matrices of Equation (2.5.1). The relationship between the analysis above and that of Section 2.3 is that the preceding method of this section provides a link between the classical sensitivity methods and those making use of sensitivity operators. Let the differential changes $d\underline{A}$, $d\underline{B}$, $d\underline{C}$, and $d\underline{D}$ become the matrices

$$\frac{d\underline{A}}{dp}, \frac{d\underline{B}}{dp}, \frac{d\underline{C}}{dp}, \text{ and } \frac{d\underline{D}}{dp} .$$

Then the calculations leading to the differential changes of the poles and zeros of the transfer functions of $\underline{P}(s)$ may be executed to yield the terms

$$\frac{dz_i}{dp} \quad \text{and} \quad \frac{d\lambda_i}{dp}$$

that are needed in the definition of the pole-zero sensitivity of Section 2.2. By multiplying these terms by the nominal value of the parameter p, these terms become the pole-zero sensitivities of the transfer function. After noting that the multiplying factor of the transfer function, K_o, is either some element of $\underline{D}$ or of $\underline{M}_i$, $0 \leq i \leq n-1$, it becomes possible to evaluate the term

$$S_p^{K_o} = p \frac{dK_o}{dp} .$$

Ur's formula, given in Equation (2.2.5), provides a method of evaluation of the standard Bode-Mason type sensitivity defined in Equation (2.2.2). Thus, the two standard definitions of sensitivity for the single element variation case in the frequency domain may be found from the sensitivity operator equations.

It should be obvious that this approach is definitely not the best approach if the transfer function is known explicitly as a function of the parameter of interest. In this case the direct differentiation of the transfer function should be performed or Newton's formula in Equation (2.5.13) may be applied to obtain pole-zero sensitivities. If time domain information is desired, the transfer function may be converted to a time-domain model in state variable form. Many recent books on state-variable methods such as Tou (27) and DeRusso, Roy, and Close (28) present standard techniques for this conversion.

If a system is described explicitly by a set of differential equations and the $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, and $\underline{D}'$ matrices are given, the approach under consideration becomes attractive for computer implementation. Appendix B presents a program designed to provide the pole-zero sensitivities from the state model and $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, and $\underline{D}'$ matrices.

The method of finding standard sensitivities becomes even more attractive if the differential equations and $\underline{A}'$, $\underline{B}'$, $\underline{C}'$ and $\underline{D}'$ matrices are automatically formulated for the user from simple graphical or tabular data. By restricting the research investigation to a class of electric networks the author has successfully automated a comprehensive model formulation and pole-zero sensitivity analysis program. This program is discussed in Chapter III.

As pointed out in Section 2.2 and Section 2.4, Breipohl and Campbell (16) make use of a sensitivity measure based on mean square error evaluated in the frequency domain. It is the purpose here to point out how the calculations discussed above may be used to provide the partial derivatives required for this measure in the frequency domain.

Letting $F_1(j\omega)$ and $F_2(j\omega)$ be the input and output, respectively, of the transfer function $H(j\omega)$, the following notations are established:

$$F_1 = |F_1| e^{j\Phi_1}$$

$$H = |H| e^{j\Phi_H}$$

$$F_2 = |F_2| e^{j\Phi_2} = |F_1||H| e^{j(\Phi_1+\Phi_H)} ,$$

$$F_{2o} = |F_{2o}| e^{j\Phi_{2o}} = \text{target value of output,}$$

$$H_o = |H_o| e^{j\Phi_{Ho}} = \text{target value of transfer function,}$$

and

$$E[\cdot] = \text{the expected value of the quantity in brackets.}$$

Then the mean square error can be expressed as in Section 2.4 for the magnitude of the output as

$$MSE_{|F_2|} = |F_1|^2 \left[\sigma^2_{|H|} + (|H_o| - \mu_{|H|})^2\right] , \qquad (2.5.17)$$

where

$$\sigma^2_{|H|} = E\left[(|H| - \mu_{|H|})^2\right] ,$$

and for the phase of F_2 as

$$MSE_{\Phi_2} = \sigma^2_{\Phi_H} + (\Phi_{Ho} - \mu_{\Phi_H})^2 \qquad (2.5.18)$$

where

$$\sigma^2_{\Phi_H} = E\left[(\Phi_H - \mu_{\Phi_H})^2\right] .$$

The mean and the variance of the magnitude and phase of the output function may be approximated by the procedure of Equation (2.4.4) and Equation (2.4.5) where the G function is now $G(\omega, \underline{\mu})$. Inspection of

the approximation to the variance reveals that interest should now be centered on the terms

$$\left.\frac{\partial |H(j\omega)|}{\partial p_i}\right|_{\underline{p} = \underline{\mu}} \quad \text{and} \quad \left.\frac{\partial \Phi_H}{\partial p_i}\right|_{\underline{p} = \underline{\mu}}$$

where

$$\underline{p} = \left[p_1, p_2, \ldots, p_k\right] ,$$

$$\underline{\mu} = \left[\mu_{p_1}, \mu_{p_2}, \ldots, \mu_{p_k}\right] ,$$

and

$$\mu_{p_i} = E\left[p_i\right] \quad .$$

These terms may be related to the sensitivity operators by means of the standard pole-zero sensitivity and the Bode-Mason sensitivity. The procedure for obtaining the standard pole-zero sensitivities has already been described in this section as has that for obtaining the Bode-Mason sensitivity. All that remains, then, is to relate the terms to the Bode-Mason sensitivity. This may be accomplished by remembering that

$$S_p^{H(j\omega)} = p \frac{dA(\omega)}{dp} + jp \frac{dPh(\omega)}{dp}$$

where

$$A(\omega) = \ell n \, |H(j\omega)|$$

and

$$Ph(\omega) = \arg H(j\omega) \quad ,$$

as shown in Section 2.2. It is then easily seen that

$$\left.\frac{\partial |H(j\omega)|}{\partial p_i}\right|_{\underline{p} = \underline{\mu}} = \left\{\frac{|H(j\omega)|}{p_i} \, \mathrm{Re}\left[S_p^{H(j\omega)}\right]\right\}\Bigg|_{\underline{p} = \underline{\mu}} \qquad (2.5.19)$$

and

$$\left.\frac{\partial \Phi_H}{\partial p_i}\right|_{\underline{p}=\underline{\mu}} = \left\{\frac{1}{p_i}\,\mathrm{Im}\left[S_p^{H(j\omega)}\right]\right\}\Bigg|_{\underline{p}=\underline{\mu}} . \qquad (2.5.20)$$

Thus the desired terms for Breipohl and Campbell's mean square error measure may be found by the relations given above. First, the standard pole-zero sensitivities are found; then the standard Bode-Mason sensitivity is evaluated for each frequency of interest. This information is then used in Equations (2.5.19) and (2.5.20) to provide the terms required to evaluate Equations (2.5.17) and (2.5.18).

This section has presented a method of obtaining the pole-zero sensitivities of the transfer function matrix $\underline{P}(s)$. Extensions have been made to provide the standard Bode-Mason sensitivity. From this point of view the process for automating the mean square error as a function of frequency has been discussed. Appendix B is devoted to a description of a FORTRAN IV language computer program for the IBM 7040 which calculates the pole-zero sensitivities by the process described above. An example problem illustrating the use of this program is also presented in Appendix B.

2.6 Summary

One of the major objectives of this chapter is to give a brief introduction to sensitivity theory and present many of the different measures of sensitivity cited in the literature. Most of these measures are based on first order derivatives or partial derivatives of the system function under consideration. These derivatives serve as a means for predicting the function variation due to changes in the parameters.

The frequency domain measures of sensitivity discussed in Section 2.2 generally fall into two classes, those based on single parameter variation and those in which many parameters are allowed to vary. Deterministic and probabilistic methods have been applied to this problem by various authors. Table I summarizes the important references within the chapter in terms of the characteristics of the various definitions of sensitivity.

The second major objective of this chapter is to introduce the sensitivity operators and illustrate their usefulness in the computer evaluation of many sensitivity measures. These operators are defined in Section 2.3 together with methods of solution and an example. Section 2.4 presents deterministic and probabilistic measures that make direct use of these time domain operators. The unifying link between the sensitivity operator equations and the classical frequency domain sensitivities is discussed in Section 2.5.

This chapter provides an analytical foundation and justification for the development of the computer programs discussed in Chapter III and Chapter IV. Even though the sensitivity operator concept is not new with this author, the automatic formulation and solution of these sensitivity models for a large class of systems represents a significant new application of the concept. Appendix A details a new technique for obtaining the simultaneous impulse response solutions for the linear time-invariant state and sensitivity models. An original probabilistic measure, the mean square error in the time domain, may be evaluated through the use of the sensitivity operators. A procedure for finding the pole-zero sensitivities of the transfer function matrix is illustrated in Appendix B along with documentation for a program

TABLE I

IMPORTANT REFERENCES IN CHAPTER II

		Frequency Domain	Page Number	Time Domain	Page Number
Deterministic	Single Parameter	Bode	8	Tomovic	16
		Mason	9		
		Pole-zero (Papoulis, Truxal, Horowitz, Ur, Calahan)	10		
		Morgan	39		
	Multiparameter	Goldstein and Kuo	11	Siljak and Dorf	33
		Lee	13	Kokotovic, Bingulac, Medanic	34
		Hakimi and Cruz	13		
		Huelsman	14		
Probabilistic		Hakimi and Cruz	15	Author	38
		Cluniess, Ross, Husson	15		
		Breipohl and Campbell	16		
		Broome and Young	35		
		Breipohl and Grigsby	35		

implementing this procedure. Extensions of this method have been suggested to provide the standard Bode-Mason sensitivity and a new process for automating the mean square error in the frequency domain has been presented.

CHAPTER III

COMPUTER-AIDED SENSITIVITY ANALYSIS OF LINEAR NETWORKS

3.1 Introduction

The previous chapter introduced the general theory of sensitivity analysis and discussed various definitions of sensitivity. Each of the different definitions have been found to be appropriate for certain cases. In all but the most trivial problems the tedious nature of hand calculation precludes evaluation of several such definitions in order to assure the designer the proper decisions have been reached. A computer-implemented approach is clearly indicated, but an investigation of the existing system analysis programs revealed none which provided direct evaluation of a wide selection of the more commonly used sensitivity definitions. Thus, the development of a design tool in the form of a general purpose sensitivity analysis program was selected to be the primary objective of this research activity.

The techniques selected for use in this study were based on the state-space model as the basic system description. This model was selected after extensive study of the previously existing analysis programs and their inherent capabilities for extension to provide sensitivity information. Extensive research has been performed in the development of algorithmic processes for network state-space model formulation and transient solution. However, no techniques had been

implemented for generating the sensitivity operator models and their associated solutions.

This chapter presents an application of the general theory of sensitivity analysis to a class of linear electrical networks. A program has been developed which automatically formulates both the state model and the sensitivity model for the class of networks under consideration and then provides sensitivity information in the form of either a general transient solution, a transient solution for an impulse input, or the pole-zero sensitivities for the network transfer function matrix.

Existing linear network analysis computer programs and their applicability to network sensitivity studies are discussed in Section 3.2. Section 3.3 defines the class of linear networks to be considered and Section 3.4 develops the formulation techniques used. The actual computer implementation of these techniques is discussed in Section 3.5, together with the solution techniques and the time-domain and frequency domain measures which the program provides.

3.2 Extant Linear Analysis Programs with Sensitivity Capabilities

Dertouzous (29) has suggested that:

> Techniques used in the several hundreds of 'programs' currently existing and surveyed in the literature, may be classified in several 'orthogonal' ways. They may be classified for example according to the type of networks that they treat, such as (1) linear resistive, (2) linear ladder-like, (3) linear (or 'slightly' nonlinear) RLC with (or without) dependent sources, (4) quasilinear resistive, or (5) nonlinear R, L, and C with constraints on topology and characteristic. Analysis techniques may also be classified according to their dynamics, i.e, (1) time domain (transient), (2) frequency domain, (3) static (dc) or according to special features such as sensitivity and optimization capabilities. The diversity (of approaches) is further aggravated since for a given network class and

> dynamics, there remains the formulation task. Formulations may be classified according to topology, i.e., as (1) nodal or mesh and (2) tree link, and according to dynamics, i.e., (1) state-variables and (2) system (transfer) functions.

Since there exist such a multitude of ways of classifying programs, it becomes highly unlikely that any two programs will make use of identical techniques even though developed for the same purposes. Indeed computer programs for electrical network analysis are still being developed and alternate techniques explored to determine their efficiency and accuracy. Many programs are not publicized and made available due to the present ineligibility of software for patent protection. Kuo (1), Dawson, Kup, and Magnuson (3), and Meissner (31) provide extensive surveys of existing programs.

Network analysis programs that have been developed with sensitivity capabilities include:

(a) ARINC, a mesh equation package written in FORTRAN for the IBM 7040 that handles both ac and dc analysis. This program provides a parameter modification scheme for use with a Monte Carlo tolerance evaluation of sensitivity (1).

(b) ECAP, a nodal analysis package written in FORTRAN for the IBM 1620, 7040, and 7090/94 computers providing ac, dc, and transient analysis. In the dc case this program provides partial derivatives of the network voltages with respect to the input parameters, and standard deviation and worst-case studies may be carried out through an automatic parameter variation facility. Allowable elements are R, L, C, switches, and linearly dependent sources (1,30).

(c) HYBRID, an IBM 7094 program which can handle linear networks

with R, L, and C elements, dependent or independent sources and any linear element described by immitance or hybrid matrices. This program is particularly well suited for parameter variation studies by repeated analysis due to its unique method of extracting variable parameters. In general, it has been found that this program is considerably more efficient than any general nodal analysis program if the number of nodes is large and the number of ports is few (1,30).

(d) RAPID1, a linear analysis program which can handle R, L, and C elements, linearly dependent and time dependent voltage and current sources for ac, dc, and transient analysis. Sensitivity information is provided for the transient and steady state solutions. This program is restricted to the class of networks defined in Section 3.3 and also makes use of the state model technique (32).

(e) LISA, a linear analysis program available on the IBM 7094 (33) and 360/65 (34). This program is based on nodal formulation techniques and can provide variations of the poles and zeros of the circuit to component variations.

Specialized programs have been written for many sensitivity studies of specific networks but these remain generally unpublished since they are too narrowly restricted to be of general interest. Examples of such programs may be found in the papers on probabilistic sensitivity (15,16).

The programs above illustrate four different approaches to the sensitivity analysis of linear networks: mesh, node, hybrid and state

model approach. These programs provide some sensitivity information, but they generally fall far short of providing an automated means of direct evaluation of a wide selection of the more commonly used sensitivity definitions. It was this situation then that prompted the development of the program VARYIT, one of the programs developed in this study.

Of the programs considered above, the RAPID1 program most closely duplicates VARYIT. Points of similarity are that the class of networks considered is the same as is the dynamic approach of developing the state model. The algorithms for formulating the sensitivity model are different and VARYIT provides an automated link with the frequency domain which RAPID1 does not consider. It should be noted that RAPID1 and VARYIT were developed at approximately the same time (35).

NASAP is a linear analysis program in the process of development since 1967 by twenty universities and ten industrial laboratories (36, 37, 38). This program is based on signal flow graphs, provides transfer functions, and has very good frequency domain sensitivity analysis capabilities. However, at present it is limited to very large computing facilities.

3.3 Terminology and Formulation of Network Equations

In this section attention is focused on the formulation of state models of linear time-invariant networks of two-terminal elements. The state model of a system of two-terminal elements is determined by the element values, their interconnection, and the choice of the state variables. It is usually desirable to choose state variables which

represent physically measurable quantities. Linear graph theory has been extensively applied to the problem of formulation of systems of equations by such authors as Koenig and Blackwell (39), Koenig, Tokad and Kesavan (40), and Seshu and Reed (41). These formulation techniques are not restricted to electrical systems or to any particular energy concept. It has long been recognized that analogues may be drawn between electrical components and mechanical, hydraulic, and mixed systems of components. Thus, a study of linear electrical networks implies considerable generality in the theory of linear systems. However, for ease of presentation, the remainder of this dissertation will deal only with electrical quantities and components.

It is assumed that the reader understands the basic notions of linear graph theory. If not, the reader should consult Seshu and Reed (41) which provides a comprehensive and highly readable introduction to the topic. However, since the notion of relative polarity of the variables can be based on a number of schemes, the convention adopted in this study will be cited. The arrowhead of the directed graph edge will be assumed to coincide with the direction of positive polarity of the current. This same arrowhead will indicate polarity of the voltage by the assumption that the tail of the arrow will indicate the positive terminal of the device for the voltage variable. It is then apparent that the simultaneous occurrence of positive or negative voltage and current variables for a given element will imply that the element is absorbing energy from the system. This notion is illustrated in Figure 7.

The method of formulation followed here is similar to that of Brown (42) in which inductors may be located in the network tree and

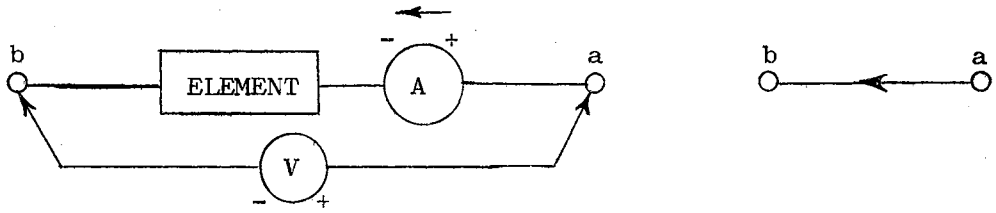


Figure 7. Polarity Convention

capacitors in the cotree. However, in order to simplify the form of the state model, it will be assumed here that a tree and cotree exist such that all capacitors and voltage drivers can be included in a tree and all inductors and current drivers can be included in the corresponding cotree. For this case it has been shown (42) that the state-space model will exist in the form of Equations (2.3.2a) and (2.3.2b). For this class, the network equations may be formulated as

$$\begin{bmatrix} \underline{C}_a & \underline{0} \\ \underline{0} & \underline{L} \end{bmatrix} \begin{bmatrix} \dot{\underline{E}}_C \\ \dot{\underline{J}}_L \end{bmatrix} = \begin{bmatrix} \underline{A}_{11} & \underline{A}_{12} \\ \underline{A}_{21} & \underline{A}_{22} \end{bmatrix} \begin{bmatrix} \underline{E}_C \\ \underline{J}_L \end{bmatrix} + \begin{bmatrix} \underline{B}_{11} & \underline{B}_{12} \\ \underline{B}_{21} & \underline{B}_{22} \end{bmatrix} \begin{bmatrix} \underline{E}_D \\ \underline{J}_D \end{bmatrix} \tag{3.3.1a}$$

and

$$\begin{bmatrix} \underline{I}_{CG} \\ \underline{V}_{TG} \end{bmatrix} = \begin{bmatrix} \underline{C}_{11} & \underline{C}_{12} \\ \underline{C}_{21} & \underline{C}_{22} \end{bmatrix} \begin{bmatrix} \underline{E}_C \\ \underline{J}_L \end{bmatrix} + \begin{bmatrix} \underline{D}_{11} & \underline{D}_{12} \\ \underline{D}_{21} & \underline{D}_{22} \end{bmatrix} \begin{bmatrix} \underline{E}_D \\ \underline{J}_D \end{bmatrix} \tag{3.3.1b}$$

where the symbols are defined in Figure 8. These equations may be more efficiently written as

$$\underline{CL}\ \dot{\underline{x}} = \underline{A}\ \underline{x} + \underline{B}\ \underline{u}$$

$$\underline{y} = \underline{C}\ \underline{x} + \underline{D}\ \underline{u}$$

where

$$\begin{bmatrix} \underline{I} & \underline{0} & \underline{0} & \underline{S}_{11} & \underline{S}_{12} & \underline{S}_{13} \\ \underline{0} & \underline{I} & \underline{0} & \underline{S}_{21} & \underline{S}_{22} & \underline{S}_{23} \\ \underline{0} & \underline{0} & \underline{I} & \underline{S}_{31} & \underline{S}_{32} & \underline{S}_{33} \end{bmatrix} \begin{bmatrix} \underline{I}_C \\ \underline{I}_D \\ \underline{I}_{TG} \\ \underline{J}_L \\ \underline{J}_D \\ \underline{I}_{CG} \end{bmatrix} = \underline{0} \qquad \begin{bmatrix} -\underline{S}_{11}^T & -\underline{S}_{21}^T & -\underline{S}_{31}^T & \underline{I} & \underline{0} & \underline{0} \\ -\underline{S}_{12}^T & -\underline{S}_{22}^T & -\underline{S}_{32}^T & \underline{0} & \underline{I} & \underline{0} \\ -\underline{S}_{13}^T & -\underline{S}_{23}^T & -\underline{S}_{33}^T & \underline{0} & \underline{0} & \underline{I} \end{bmatrix} \begin{bmatrix} \underline{E}_C \\ \underline{E}_D \\ \underline{V}_{TG} \\ \underline{V}_L \\ \underline{V}_D \\ \underline{V}_{CG} \end{bmatrix} = \underline{0}$$

CUTSET EQUATIONS — CIRCUIT EQUATIONS

$$\begin{bmatrix} \underline{I}_{TG} \\ \underline{I}_C \\ \underline{I}_{CG} \\ \underline{V}_L \end{bmatrix} = \begin{bmatrix} \underline{G}_T & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{C}_a & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{G}_C & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{L} \end{bmatrix} \begin{bmatrix} \underline{V}_{TG} \\ \frac{d\,\underline{E}_C}{d\,t} \\ \underline{V}_{CG} \\ \frac{d\,\underline{J}_L}{d\,t} \end{bmatrix}$$

COMPONENT EQUATIONS

WHERE $\underline{I}_C$ and $\underline{E}_C$ are the capacitor current and voltage vectors

$\underline{I}_D$ and $\underline{E}_D$ are the voltage-driver current and voltage vectors

$\underline{I}_{TG}$ and $\underline{V}_{TG}$ are the tree conductance current and voltage vectors

$\underline{J}_L$ and $\underline{V}_L$ are the inductor current and voltage vectors

$\underline{J}_D$ and $\underline{V}_D$ are the current-driver current and voltage vectors

$\underline{I}_{CG}$ and $\underline{V}_{CG}$ are the cotree conductance current and voltage vectors

$$\underline{A}_{11} = \underline{S}_{13}\ \underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{13}^T - \underline{S}_{13}\ \underline{G}_C\ \underline{S}_{13}^T \qquad \underline{C}_{11} = \underline{G}_C\ \underline{S}_{13}^T - \underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{13}^T$$

$$\underline{A}_{12} = \underline{S}_{13}\ \underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{31} - \underline{S}_{11} \qquad \underline{C}_{12} = -\underline{G}_C\ \underline{S}_{33}\ \underline{R}\ \underline{S}_{31}$$

$$\underline{A}_{21} = \underline{S}_{11}^T - \underline{S}_{31}^T\ \underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{13}^T \qquad \underline{C}_{21} = -\underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{13}^T$$

$$\underline{A}_{22} = -\underline{S}_{31}^T\ \underline{R}\ \underline{S}_{31} \qquad \underline{C}_{22} = -\underline{R}\ \underline{S}_{31}$$

$$\underline{B}_{11} = \underline{S}_{13}\ \underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{23}^T - \underline{S}_{13}\ \underline{G}_C\ \underline{S}_{23}^T \qquad \underline{D}_{11} = \underline{G}_C\ \underline{S}_{23}^T - \underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{23}^T$$

$$\underline{B}_{12} = \underline{S}_{13}\ \underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{32} - \underline{S}_{12} \qquad \underline{D}_{12} = -\underline{G}_C\ \underline{S}_{33}^T\ \underline{R}\ \underline{S}_{32}$$

$$\underline{B}_{21} = \underline{S}_{21}^T - \underline{S}_{31}^T\ \underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{23}^T \qquad \underline{D}_{21} = -\underline{R}\ \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{23}^T$$

$$\underline{B}_{22} = -\underline{S}_{31}^T\ \underline{R}\ \underline{S}_{32} \qquad \underline{D}_{22} = -\underline{R}\ \underline{S}_{32}$$

$$\underline{R} = \left[\underline{G}_T + \underline{S}_{33}\ \underline{G}_C\ \underline{S}_{33}^T \right]^{-1}$$

Figure 8. Definition of Terms for Equation (3.3.1)

$$\underline{CL} = \begin{bmatrix} \underline{C}_a & \underline{0} \\ \underline{0} & \underline{L} \end{bmatrix}$$

is a diagonal matrix with all diagonal entries positive. The solution of this set of equations may be used in the cutset, circuit, and component equations of Figure 8 to provide the complete solution for all network voltages and currents.

If the network has both circuits of capacitors only and cutsets of inductors only, then the model has the form of Equation (2.3.2). However, in general, the state models of networks violating the simplifying assumption do not have the simple form of Equation (2.3.2). Topological means are readily available to handle these cases.

3.4 Formulation of the Sensitivity Operators

In the preceding section the general form of the state model is given for the network class under consideration. It is now desired to develop the general form of the sensitivity operator equations for this class of networks. The techniques of Section 2.3 lead immediately to

$$\dot{\underline{v}}^p(t) = [\underline{CL}]^{-1} \underline{A}\, \underline{v}^p(t) + \underline{Z} \qquad (3.4.1a)$$

where

$$\underline{Z} = [\underline{CL}]^{-1}\{\underline{A}'\, \underline{x} + \underline{B}'\, \underline{u}\} + \frac{d[\underline{CL}]^{-1}}{dp} \{\underline{A}\, \underline{x} + \underline{B}\, \underline{u}\} \qquad (3.4.1b)$$

and

$$\underline{w}^p(t) = \underline{C}\, \underline{v}^p(t) + \underline{C}'\underline{x} + \underline{D}'\underline{u} \quad . \qquad (3.4.1c)$$

The cutset and circuit equations may be used together with the solutions to Equations (3.4.1a) and (3.4.1c) to yield complete knowledge of the first order partials of every voltage and current in the network.

Consider now formulation of the system state model and its associated sensitivity model for the parameter p. Inspection of Equation (3.4.1) indicates the only terms unavailable by direct numerical calculation are $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, $\underline{D}'$, and $\frac{d[\underline{CL}]^{-1}}{dp}$. Several methods may be used to obtain these terms.

The first method for finding the $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, and $\underline{D}'$ matrices is to consider the individual submatrices $\underline{A}_{11}$, $\underline{A}_{12}$, ..., $\underline{D}_{22}$ in terms of their defining expressions given in Figure 8. Matrix differentiation may be used directly on these expressions but a difficulty is encountered in finding $\underline{R}'$. This difficulty may be surpassed by differentiating the product

$$\underline{R}^{-1}\,\underline{R} = I$$

yielding

$$\underline{R}^{-1}\,\frac{d\underline{R}}{dp} + \frac{d\underline{R}^{-1}}{dp}\,\underline{R} = \underline{O}$$

so that

$$\underline{R}' \equiv \frac{d\underline{R}}{dp} = -\underline{R}\,\frac{d\underline{R}^{-1}}{dp}\,\underline{R}\,. \qquad (3.4.2)$$

Once the $\underline{R}'$ matrix is found the $\underline{A}'$, $\underline{B}'$, $\underline{C}'$ and $\underline{D}'$ matrices of Equation (3.4.1) may be evaluated. Although this method is direct and can be implemented on the computer it is not efficient when several nominal parameter values are given since the state and sensitivity models must be reformulated for each nominal parameter value.

A second method exists which does not necessitate reformulation of the models. In this approach the matrices are formulated as polynomial functions of the parameter p where the p is maintained in symbolic form, i.e., as a literal. Each submatrix may be formulated and then

combined to produce $\underline{A}(p)$, $\underline{B}(p)$, $\underline{C}(p)$, and $\underline{D}(p)$ in polynomial form. The elements of these matrices may then be differentiated to yield $\underline{A}'(p)$, $\underline{B}'(p)$, $\underline{C}'(p)$, and $\underline{D}'(p)$. At each value of p for which the models are desired the numerical value of p is inserted into $\underline{A}(p)$, $\underline{B}(p)$, $\underline{C}(p)$, $\underline{D}(p)$, $\underline{A}'(p)$, $\underline{B}'(p)$, $\underline{C}'(p)$, and $\underline{D}'(p)$.

The symbolic formulation procedure suggested above posed two major problems when computer aided analysis was envisioned. First, the symbolic formulation appeared to require large amounts of storage. Secondly, new techniques and algorithms were needed when dealing with the parameter in symbolic form.

The storage requirements will be discussed first. Consider the matrix polynomial representation scheme

$$\underline{F}(p) = \underline{F}_n p^n + \underline{F}_{n-1} p^{n-1} + \ldots + \underline{F}_o . \qquad (3.4.4)$$

The storage requirement for the k x k polynomial matrix of degree n $\underline{F}(p)$ is $(n+1)k^2$. During the formulation of the state model in terms of the variable p, it is necessary to find $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ as matrix polynomials. These matrices are seen from Figure 8 to depend strongly on the matrix

$$\underline{R} = \underline{R}(p) = \left[\underline{G}_T(p) + \underline{S}_{33}\, \underline{G}_c(p) \underline{S}_{33}^T\right]^{-1} . \qquad (3.4.5)$$

The parameter p may belong to either the tree or the cotree, but not to both. In general, the maximum degree of this matrix polynomial in the cotree conductance parameter is m-1 when there exists m resistances in the tree (see Appendix C). Thus, the storage required is

$$(m-1+1)m^2 = m^3$$

for the general case. Many problems of a reasonable size would exceed the 32,000 word memory of the 7040 computer, thus necessitating a very large amount of scratch pad manipulation using external storage. However, the initial concern over the appearance of high order polynomials was unfounded as the following result indicates. Proof of this theorem may be found in Appendix C.

3.4.1 Theorem

Consider an RLC electric network containing voltage and current sources. Let there exist a tree containing all capacitors and voltage sources and excluding all inductors and current sources. Then the network state model given by Equation (3.3.1) is such that no entry of the $\underline{A}$, $\underline{B}$, $\underline{C}$, or $\underline{D}$ matrices has numerator degree higher than three in any R, L, or C parameter or has denominator degree higher than one in any parameter.

This theorem guarantees that for every network in the class under consideration the maximum storage for each matrix polynomial need never exceed that required for numerator degree three. It is also shown in Appendix C that the maximum degree for $\underline{R}$ is in reality one. Thus, the true storage required becomes

$$(1 + 1)m^2 = 2m^2$$

instead of m^3. For example, let m be 20. Then an order of magnitude improvement in storage requirements is effected by using the results of Theorem 3.4.1. A similar reduction in storage requirements is applicable to $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$.

The second problem mentioned above is the development and implementation of computational techniques for use with symbolic parameters. Basic operations, such as addition, subtraction, multiplication, and

division, with polynomial elements must be developed before more advanced techniques such as differentiation and matrix inversion can be applied.

The most difficult algorithm implemented in this study was that of finding the inverse of the matrix $\underline{R}^{-1}$ of polynomial functions. Several approaches are possible. One approach is suggested by the topological relations of Seshu and Reed (41). It is possible to develop tree listing programs and thus form all needed terms by means of the formulae. Experience with tree-listing programs indicates this is not an attractive approach since even moderately large networks contain numerous trees. A second approach to the problem is to consider every parameter except the varying one in numerical form. Walden (43) has shown the $\underline{R}$ matrix is a positive definite, nonsingular matrix. This fact may be utilized to assure that the inversion by the bordering method of Faddevva (44) may be carried out. This process consists of finding the inverse of k x k submatrix in the upper left. This inverse is then used to generate the inverse of the (k+1) x (k+1) principal submatrix.

3.5 Computer Program VARYIT

Section 3.3 and Section 3.4 have considered a certain class of linear networks and presented the network state model and its associated sensitivity model. This section describes the program VARYIT which has been written to automatically formulate the state model and sensitivity model for this class of networks and then perform a solution in either the frequency or the time domain.

It is unnecessary to check to make sure a suitable tree exists before execution on the computer. If no tree exists which contains all

capacitors and voltage sources and excludes all inductors and current sources, the program will detect this condition and terminate processing. The user may circumvent this condition by one of two methods. When an improper cutset is detected a resistor of large magnitude may be paralleled with a current source or inductor leading to the isolated node or supernode. This resistor should be much larger than any existing resistor value so that it has a negligible influence on the network voltage and current solution. Similarly small resistances may be incorporated in series with capacitors and voltage sources without seriously degrading the solution. Normally a suitable tree will exist unless the elements are completely idealized. If losses in the storage elements are accounted for with resistors, the need for extra resistors will not arise.

The program is implemented in the FORTRAN IV language in the hope of providing the widest possible distribution and application. None of the techniques are restricted to any machine even though the program itself was developed on the Oklahoma State University Computer Center IBM 7040 (32K words, 5 tape drives). An attempt was made to always use symbolic names for each system unit in the hope that other centers with differing unit designations could easily make this program operational by modifying a minimum of statements. Additional versions of this program are available for the IBM 360 and the UNIVAC 1108.

The program is written in seven phases (not all of which are executed on any one problem). Phase 1 reads the input data, finds a suitable formulation tree, and calculates the cutset matrix. Phase 2 formulates the system state model submatrices $\underline{A}_{11}$, $\underline{A}_{12}$, ..., $\underline{D}_{22}$ of Figure 8 and stores these on tape. Phase 3 reads the submatrices,

arranges them into $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ matrices of Equation (3.3.1), evaluates them at the nominal value of the parameter p and stores the evaluated matrices on tape. Phase 3 also differentiates these matrices and stores the result on tape. Phase 4 is executed when a general time solution is desired by integration of the system state model and the sensitivity model. Phase 5 is executed when an impulse driver is specified at one of the driver positions. Phase 6 calculates the transfer function matrix and the pole and zero sensitivities of each element of the matrix. Phase 7 applies one measure of goodness if desired and prints, punches, or plots the output data. Figure 9 shows a schematic representation of the operations which VARYIT may be called on to perform.

In order to solve reasonably large networks VARYIT is dimensioned for a maximum of 80 elements. A total of 20 inductors and capacitors, 20 voltage and current drivers, 20 resistors in the cotree and 20 resistors in the tree (thus, 40 resistors total) may be included. A maximum of 40 nodes may exist in the network to be analyzed. No more than ten elements can be incident at a node in the network. Although these restrictions can be relaxed by modification of the program, the ultimate size of the network that may be solved is limited by the available storage in the computer. The number of elements may be increased by altering the appropriate DIMENSION statements.

This program may be used as a general analysis program by setting the number of varying parameters to zero. Time solutions, impulse solutions, and the transfer function matrix are then available outputs.

The input to this program is user oriented. In order to execute a problem the user must perform the following "desk" steps:

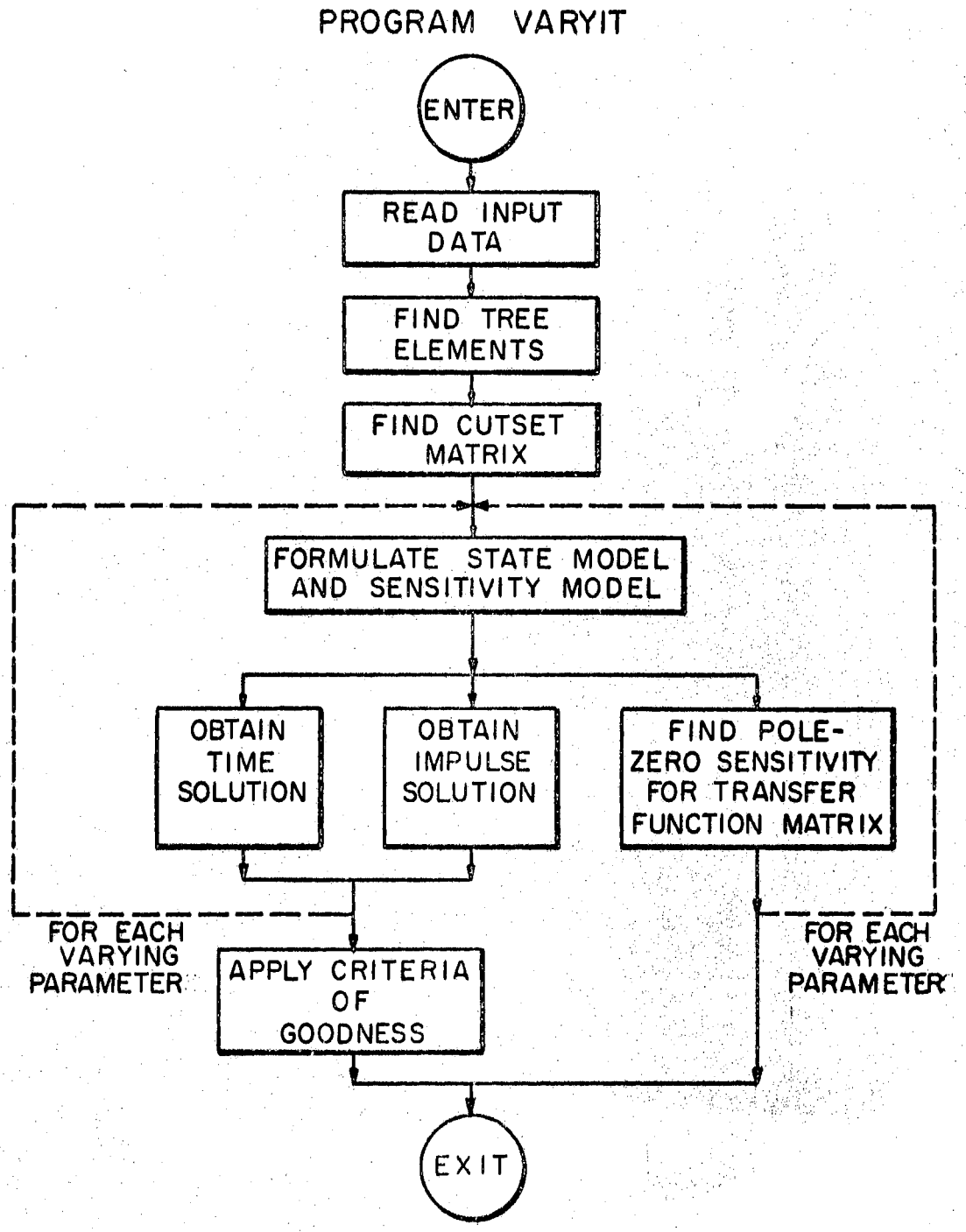


Figure 9. Operations Performed by VARYIT

(1) number the elements by type, resistors first, inductors second, current drivers third, capacitors fourth, and voltage drivers fifth;

(2) number the nodes of the network; and

(3) assign orientations to each element.

Then, in general, the input data consists of:

(1) an interconnection array listing elements incident to each node;

(2) an orientation list which lists nodes from which elements are oriented;

(3) nominal parameter values for R, L, and C elements;

(4) driving function types and parameters;

(5) a list of parameters which are to vary; and

(6) the type of solution desired.

More explicit information on the program, its parameters, and its input data is given in Appendix D. Chapter V presents examples of the various types of problems handled by VARYIT.

3.5.1 Tree Selection Algorithm

In order to formulate the submatrices $\underline{A}_{11}$, $\underline{A}_{12}$, ..., $\underline{D}_{22}$ from the equations of Figure 8, it is necessary to find a suitable formulation tree. The tree selection algorithm used in this program has evolved from an algorithm developed by Cummins and Thomason (45) and modified by Falk (46). The algorithm given by Falk produces a tree which contains certain specified elements of the network and excludes other specified elements. Initially all voltage drivers and capacitors are placed in the branch list of elements and all current drivers and

inductors are assigned to the chord list. The algorithm presented here differs from Falk in that it contains a test (Step 6) to insure the requirements for the existence of a tree are not violated as each element is added to the list of tree elements. Falk's algorithm assumed the existence of a suitable tree.

The steps of the algorithm are:

(1) Remove the elements from the network which are contained in the chord list.

(2) Select an element of the network branch list.

(3) Locate the nodes i and j to which the selected branch element is incident in the network. Remove the branch from the network and join together or identify the two nodes as one single node; label the combined node with the smaller of i and j.

(4) Remove from the network any self loop elements -- elements with both ends incident at the same node -- which are generated by the identification of two nodes in Step 3. Add these elements to the network chord list.

(5) Return to Step 2 until all elements of the branch list have been selected and removed from the network.

(6) If elements remain in the network, choose an element attached to the reduced node numbered 1 (if no element is now attached to this node exit to "No Tree Exists") and add the element to the network branch list. Go to Step 3. If no elements remain in the reduced network and the number of nodes has been reduced to one, the tree and cotree are defined by the branch and chord lists, respectively.

3.5.2 Cutset Matrix Formulation Algorithm

Having found the branch list and the chord list of elements, the program next must calculate the fundamental cutset matrix or the fundamental circuit matrix of Figure 8. The algorithm used in this program is identical to that proposed and automated by Falk (46). The algorithm itself is presented here only in the interest of completeness.

(1) Create a matrix of zeros with as many rows as the number of entries in the network branch list and as many columns as the number of entries in the network chord list. Select the first entry in the branch list as the first cutset branch.

(2) Determine one of the nodes to which the cutset branch is incident in the network and determine if the branch is oriented away from the node.

(3) List the node in the node list. List all the elements incident to the node, except the branch, in the cutset list.

(4) If any elements of the cutset list are branch elements, select one and remove it from the cutset list. Determine the other node to which it is incident and go to Step 3.

(5) When there are no branch elements in or remaining in the cutset list, remove both entries of all those elements which appear twice in the cutset list.

(6) The cutset list formed is the list of cutset chords corresponding to the initially selected cutset branch. Select the first chord in this list.

(7) Determine if the selected chord is oriented away from any of the nodes in the node list.

(8) If the orientation of the cutset branch (Step 2) and the

orientation of the chord (Step 7) are both away from or both toward any of the nodes in the node list, the cutset matrix entry corresponding to the branch and chord is +1; if either the branch or chord orientation is away from and the other is toward any of the nodes in the node list, the entry is -1.

(9) Place the entry determined in Step 8 in the matrix row corresponding to the cutset branch position in the network branch list and in the matrix column corresponding to the position in which the chord is located in the network chord list.

(10) Select the next chord in the cutset list and go to Step 7. When all chords have been selected, go to Step 11.

(11) Remove all entries from the node and cutset lists. Select the next branch in the branch list for the next cutset branch and go to Step 2. When all branches have been selected, the cutset matrix submatrix $\underline{S}$ is completed.

The complete cutset matrix representation is $\left[\underline{I}\ \underline{S}\right]$, where $\underline{I}$ is the unit matrix.

3.5.3 Formulation Algorithm

Program VARYIT generates the state model via the formulation algorithms presented in Section 3.3. The submatrices of Equation (3.3.1) are formed by implementing the relationships given in Figure 8. In order to formulate the sensitivity operators the $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ matrices are formulated as polynomial functions of the parameter p as suggested in Section 3.4. Theorem 3.4.1 is used to advantage in this process to decrease the storage requirements. The formulation procedure requires

the presence of only four matrix polynomials at any one time in the computer memory. The submatrices $\underline{A}_{11}$, $\underline{A}_{12}$, ..., $\underline{D}_{22}$ are calculated and stored on tape. They are retrieved and combined into the four matrices $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$. The matrices $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, and $\underline{D}'$ are found by differentiation of the respective matrices. At this time the nominal value of the parameter p is inserted and numerical values for the $\underline{A}$, $\underline{B}$, $\underline{C}$, $\underline{D}$, $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, and $\underline{D}'$ matrices are calculated. These matrices are stored on tape as inputs for the following time or frequency solution phases.

One additional task is performed in the formulation phases. Walden (43) contends, and the author concurs, that in the majority of cases the cutset matrix $\underline{S}$ is sparsely populated with non-zero elements. Thus, a saving in storage may be effected by storing only the non-zero elements of the $\underline{S}$ matrix along with the indices of their location in the matrix. This method also leads to a substantial reduction in execution time since only the operations associated with non-zero elements of the $\underline{S}$ matrix are performed.

3.5.4 Time Solution Technique

Program VARYIT provides the general time solution to the network state model and its associated sensitivity model as one of three solution options. The solution of the system state model requires the initial conditions be known for $\underline{E}_C$ and $\underline{J}_L$. In general, however, it is not convenient, or in some cases possible, to specify the capacitor voltages and inductor currents. In most electronic circuits it is usually possible to specify only the value of all power supplies and signal sources at t=0. The assumption usually made for such cases is that the circuit has achieved steady-state with the constant drivers

applied. This assumption implies that the derivatives of all variables must be equal to zero. The only derivatives of interest are those given by Equation (3.3.1a), since all others are linear combinations of these. By equating the left hand side of Equation (3.3.1a) to zero, it is possible to solve for the initial values of $\underline{J}_L$ and $\underline{E}_C$. This results in

$$\begin{bmatrix} \underline{E}_{CO} \\ \\ \underline{J}_{LO} \end{bmatrix} = - \begin{bmatrix} \underline{A}_{11} & \underline{A}_{12} \\ \\ \underline{A}_{21} & \underline{A}_{22} \end{bmatrix}^{-1} \begin{bmatrix} \underline{B}_{11} & \underline{B}_{12} \\ \\ \underline{B}_{21} & \underline{B}_{22} \end{bmatrix} \begin{bmatrix} \underline{E}_{DO} \\ \\ \underline{J}_{DO} \end{bmatrix} \tag{3.5.1}$$

or

$$\underline{x}_O = - \underline{A}^{-1} \underline{B} \, \underline{u}_O$$

where the "O" subscript is used to denote that these are the steady-state t=0 values. The inverse of the A matrix exists if the system has a finite steady-state solution to a step input, i.e., no zero eigenvalue. Equation (3.5.1) is used to provide the initial conditions for the solution when the steady-state assumption is desired; otherwise, initial conditions must be furnished.

In order to obtain the solution to the sensitivity model, Equation (3.4.1), the initial conditions for $\underline{v}^p(t)$ must be found or supplied as input data. If no initial conditions are supplied, program VARYIT computes the initial conditions for the sensitivity model from

$$\underline{v}^p(0) = - \underline{A}^{-1} \left\{ \left[- \frac{d(\underline{CL})}{dp} (\underline{CL})^{-1} \underline{A} + \underline{A}' \right] \underline{x}_O + \left[- \frac{d(\underline{CL})}{dp} (\underline{CL})^{-1} \underline{B} + \underline{B}' \right] \underline{u}_O \right\} . \tag{3.5.2}$$

Equation (3.5.2) is found by letting $\dot{\underline{x}}(0)$ and $d\dot{\underline{x}}/dp$ be zero. Thus, the sensitivity model itself may be said to have achieved a steady-state value $\underline{v}^p(0)$. If these requirements do not fit the case under consideration initial conditions may be computed and read in as input data.

The solution procedure used is the sequential method of Section 2.3.1 in which the $\underline{x}(t)$ and $\underline{y}(t)$ vectors of Equation (3.3.1) are obtained at each integration increment. These vector solutions are used in the cutset and circuit equations to yield the complete solution at each integration increment for all network currents and voltages. The complete network solution is stored on tape for use in the sensitivity model integration. The integration process used in this program makes use of the matrix exponential method given by Liou (47). This method provides a desired numerical accuracy when integrating from time t to time $(t + \Delta t)$. If the desired number of decimal places of accuracy is "LD", the PREC variable of Appendix D should be set at 10^{-LD}. Note that this method still allows round-off errors to accumulate during the integration process.

The state sensitivity operator solution integration procedure is to retrieve the complete network solution from tape storage to form the $\underline{Z}$ vector of Equation (3.4.1b). This $\underline{Z}$ vector is used as a driver vector in the integration process along with the transition matrix of the state model integration. By making use of Equation (3.4.1c) the output sensitivity operator may be found. Differentiation of the cutset and circuit equations of Figure 8 allows the complete solution to the network sensitivity model to be found by simple linear combinations of the output and state sensitivity operators. This complete solution

for all network variables and their sensitivity operators is calculated and stored on tape at each integration increment.

3.5.5 Impulse Response Solution Techniques

Program VARYIT provides the solution to the network state model and its associated sensitivity model when unit impulses are introduced at the driver inputs. This solution is found by the technique of augmentation given in Equations (2.3.6) and (2.3.7). Equation (2.3.10) yields the output state vector and the output sensitivity operator for an impulse function input at driver u_d. The reader should note that when an impulse function solution is desired the only non-zero driver is the impulse driver and the initial conditions on both the sensitivity operator equations and the state model equations are set equal to zero. Liou's method is also used in this type solution to insure the desired numerical accuracy. The user specifies those drivers which are to be impulse drivers and the program assumes one driver at a time is an input impulse and form the derivatives with respect to parameter p of each voltage and current in the network. By using the augmentation technique of Appendix A the integration is actually performed only once regardless of the number of impulse drivers. The complete solution to the network and the sensitivity model at each integration increment is stored on tape.

3.5.6 Pole-Zero Sensitivity

The third solution option provided by VARYIT is the calculation of the pole-zero sensitivities. Section 2.5 has discussed the techniques for calculation of the transfer function matrix $\underline{P}(s)$ and also the pole

and zero sensitivities of the individual elements of this matrix. Adaptation of the program of Appendix B into VARYIT was accomplished by inclusion of the steps necessary to convert the $\underline{A}$, $\underline{B}$, $\underline{C}$, $\underline{D}$, and $\underline{CL}$ matrices of Equation (3.3.1) to the $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ matrices of Equation (2.3.2). Similar steps are taken in order to obtain the corresponding $\underline{A}'$, $\underline{B}'$, $\underline{C}'$, and $\underline{D}'$ matrices for Equation (2.3.5). A decision was made to restrict the size of the maximum network which may be handled by this program to networks containing a total of 17 capacitors and inductors, 17 resistors and 17 voltage and current drivers. If this restriction is too stringent, the program may be split into phases to accomodate larger networks by utilizing the auxiliary storage capability of tape or disc machines.

3.5.7 Statistical Sensitivity Measure

Program VARYIT includes one statistical sensitivity measure as an output option. This measure is the variance approximation of Equation (2.4.6). If it is desired to apply this measure, the variance of each varying parameter $\sigma^2_{p_i}$, i=1,2, ..., k, must be included as input data. As pointed out in Section 2.4, these variances may be evaluated either by standard statistical means or obtained from the manufacturer. Different sets of parameter variances may be included so that multiple variance approximations may be studied as the individual parameter variances are allowed to change. In this manner a family of variance approximation curves may be formed and the effect of the individual parameter variances may be examined. This feature of the program may be used by the designer as an aid in the setting of tolerances on the parameters. If this portion of the program is executed, the program

will automatically plot the variance terms if desired.

3.5.8 Time Domain Output of the Program VARYIT

If either the time solution or the impulse solution is carried out, a tape is produced containing the complete network solution at each integration increment and the complete network sensitivity solution. This tape is available for removal and storage for processing at a later time. In this way multiple use of the sensitivity operators is possible without re-execution of the formulation and integration processes. The format of the output tape is given in Appendix D.

The output phase of the program has provision for several types of output. If a complete solution is to be printed at each integration instant, the word LIST should be entered in the appropriate position as given by the input sequence shown in Appendix D. Other options are more selective. For example, the user may specify the current through element number 11 by I11 and similarly the voltage across element 11 by V11. Then these variables may be either printed or punched as desired. Examples of the output are presented in Chapter V.

3.6 Summary

The primary objective of the research activity discussed in this chapter was to develop and implement a design tool that is capable of supplying sensitivity information for a variety of the most commonly used sensitivity definitions. This chapter has described the computer methodology and algorithmic processes used in the general purpose sensitivity analysis program VARYIT developed in this study. For a large class of linear networks or analogous non-electrical systems,

program VARYIT automatically formulates the state and sensitivity models from element values and interconnection data, and executes one or more solution options. These options provide sensitivity information either in the time or frequency domains and of either a deterministic or probabilistic nature. It is felt that this program and its outputs will be of value to the practicing engineer conducting optimal design selection or tolerancing studies.

Even though much study has previously been devoted to the state-space model, its formulation, and its solution, little consideration had been given to the automatic formulation and solution of the sensitivity model for impulse and transient analysis. A new theorem has been presented which is of great importance in decreasing the storage requirements for the formulation procedures. VARYIT implements a method of solving the sensitivity operator model for linear systems with impulse inputs. This new method assures desired number of places of numerical accuracy between time steps in the integration process. Also implemented is an extension of Breipohl's frequency domain measure of sensitivity to the time domain and a computational algorithm for obtaining the transfer function matrix and its associated pole-zero sensitivity information.

The algorithms of Sections 3.3 and 3.4 have been programmed and Section 3.5 is devoted to a discussion of the resulting program VARYIT. Examples of the use of VARYIT are presented in Chapter V and Appendix D. This program is user oriented and requires a minimum of input data. Even though this program was developed primarily as a sensitivity analysis aid, it may be used as a general analysis program either with or without sensitivity calculations.

CHAPTER IV

EXTENSION TO NONLINEAR NETWORKS

4.1 Introduction

The preceding chapter described the general purpose sensitivity analysis program VARYIT developed to automatically formulate and solve sensitivity models for a class of linear systems. The purpose of this chapter is to describe the methodology and algorithms developed for a program applicable to a large class of nonlinear systems. Utilization of these techniques in the new program VARNOL permits automatic problem formulation and solution for the state model and also the implementation of the sensitivity operators.

The extension of the sensitivity operator concept to computer-aided nonlinear analysis posed several problems. First, a suitable programming technique or program was needed to accomplish the formulation and solution of the nonlinear differential equation model of the system. Secondly, the development of a computational algorithm for generating and solving the nonlinear sensitivity operator equations was needed. Finally, the nonlinear analysis program and sensitivity algorithm had to be combined together to provide an operational general purpose program.

One of the major tasks that must be performed in obtaining sensitivity operators is the formulation and solution of the nonlinear network model. This is no easy task in itself but many programs have been

developed which achieve this to a more or less successful degree. Section 4.2 discusses these programs, their applicability to sensitivity analysis and their availability for this study. The particular nonlinear analysis programming technique selected for further development is described in Section 4.3.

The second problem area mentioned above is that of developing a computational algorithm for the generation and solution of the nonlinear sensitivity operators. Section 4.4 discusses the algorithms selected for implementation in program VARNOL.

The third problem area, that of combining the nonlinear analysis techniques and the sensitivity algorithm, has been successfully solved as demonstrated in Section 4.5. The program VARNOL developed during this study is described together with the nonlinear models included in the program. Further details of the program may be found in Appendix E.

4.2 Nonlinear Analysis Programs

Many programs have been developed to treat nonlinear elements in the general network analysis problem. However, at the initiation of this study, no programs were found that provided sensitivity information when nonlinear elements were included. Programs that include nonlinear analysis capabilities include:

(a) ECAP. This program may be used if the nonlinearity can be modeled by a piecewise linear, single-valued function of a network current or voltage. After breaking the nonlinearity into piecewise linear segments, the user must model each segment with a switch and source. This procedure is not desirable since an extremely large network may result

when a high degree of accuracy in representing the nonlinearity is desired.

(b) DCAP, PETAP. This is a series of programs developed by IBM which make use of highly developed transistor and diode models. Major difficulties were encountered when attempts were made to extend these programs to include additional nonlinear models (48).

(c) NET. This highly refined program developed by Malmberg (49) makes use of topological and matrix methods to solve networks including transistors, diodes, and linear elements. Since this program is not available in simple FORTRAN-like language, no possibility existed for modifying it for use at Oklahoma State University.

(d) PREDICT, SCEPTRE. These IBM programs are based on the state-space model and can handle a large class of linear, nonlinear, and time varying networks. Since PREDICT is written in a machine assembly language for the IBM 7090-94 machines, it was discarded from further consideration. SCEPTRE, the second generation of PREDICT, did not become available until the completion of the programming efforts in this study.

(e) AEDNET. This program, developed by Katzenelson (50), is capable of analyzing networks containing nonlinear RLC elements, dependent sources, and independent sources. It is written in AED-0 for the MAC computer at Massachusetts Institute of Technology.

(f) WALDEN. At Oklahoma State University a program has been developed by Walden (43) to analyze nonlinear networks. This

program handles a large class of nonlinearities, is written in FORTRAN, and is well documented.

Walden's techniques (43) for representing nonlinear elements were chosen as a suitable basis for the extension of the sensitivity operators into the nonlinear network domain. These techniques were selected for the following reasons:

(a) The state space model is the basic system description and the more widely publicized programs PREDICT, SCEPTRE, and AEDNET also use this basic approach. The technique evolved in this study may possibly be incorporated into these programs with modifications.

(b) A large class of nonlinear models has been successfully analyzed with Walden's techniques.

(c) The formulation of the network model and the sensitivity model may be carried out by the same algorithms developed for the linear program VARYIT discussed in Chapter III.

Because the original program developed by Walden was no longer operational at Oklahoma State University, it was necessary to re-implement the coding for the solution phases of this program.

Since the initiation of this study, Leeds, Grueneich, and Moore (32) have disclosed the development of RAPID2, a modification of RAPID1 to supply sensitivity information for circuits which include nonlinear energy storage elements. RAPID2 is restricted to the same class of network topologies discussed in Section 3.3 and nonlinear resistances are not allowed.

4.3 Inclusion of Nonlinear Elements in the State-Space Model

Many practical devices used in electronic networks have nonlinear terminal characteristics. If these devices are included in the network to be analyzed, the state-space model takes the form of Equation (2.3.1). Two methods of finding the time solution for such nonlinear systems are discussed in this section. Walden's approach, which is a combination of these two methods, is also described.

One method for finding the time solution is to represent the nonlinear system as a sequence of stepwise linear systems at each integration increment. The coefficients of the differential equations are reevaluated at each time step in the integration process and are assumed to be constant between integration steps. The integration increment must be smaller than the minimum time constant in the linearized system of equations before this assumption can be justified.

A second method utilizes dependent drivers. The nonlinear elements are represented as voltage or current drivers which are in turn functions of voltages and currents in the network. However, it does not suffice to simply evaluate this dependent driver step by step. Changes in the values of dependent drivers are propagated throughout the network and, hence, influence the value of all variables upon which the driver is dependent. Inconsistencies are then possible unless an iteration process is carried out at each integration step. A convergent iteration procedure that works for all nonlinearities is still unknown but certain classes of nonlinearities have been investigated (43).

Walden's approach for representing nonlinear devices considers a combination of the two methods above. The state model for the nonlinear network is written in the form

$$\underline{CL}\,\dot{\underline{x}} = \underline{A}\,\underline{x} + \underline{B}\,\underline{u} \tag{4.3.1a}$$

$$\underline{y} = \underline{C}\,\underline{x} + \underline{D}\,\underline{u} \tag{4.3.1b}$$

where

$$\underline{u} = \underline{f}(\underline{x},\underline{y},p,t) \tag{4.3.2}$$

$$\underline{CL} = \underline{F}(\underline{x},\underline{y},p,t) \tag{4.3.3}$$

with p a network parameter. Equation (4.3.1) has the same form as Equation (3.3.1) for the class of networks defined in Section 3.3.

Nonlinear energy storage elements, such as the inductors and capacitors in an electrical network, are represented by stepwise linear equivalent values in the integration process. It should be noted in Equation (3.3.1a) that all storage elements appear in the $\underline{CL}$ matrix and the $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ matrices are independent of the values of these storage elements. In general, it is necessary to invert the $\underline{CL}$ matrix at each step in the integration process. However, when there are no nonlinear storage elements the inverse may be computed only once since it remains constant. If no mutual inductances appear in the network the $\underline{CL}$ matrix is diagonal and, hence, can be easily inverted.

Nonlinear resistances appear in the matrices on the right of Equations (3.3.1a) and (3.3.1b). Thus, the stepwise linear method above would require reformulation of the system equations at each integration step. To avoid this costly process nonlinear resistors are represented as either a dependent voltage or current source. The evaluation of the source variable is carried out by a nonlinear side

equation which takes the form of a polynomial equation of up to fourth degree.

The inclusion of active nonlinear devices is accomplished in the same manner as nonlinear resistances. Side equations of a more general form have been incorporated to handle vacuum tubes and other devices. Space does not permit a thorough discussion of the iteration processes which are associated with the dependent driver approach. The interested reader should refer to Walden (43) for a detailed treatment.

4.4 Formulation and Solution Techniques for Sensitivity Operators for Nonlinear Networks

Section 4.3 presented the method Walden (43) used in obtaining the steady state and transient solution for a large class of nonlinear networks. This section considers a new problem, the formulation and solution of a sensitivity operator model for the network parameter p based on the model of Section 4.3. An approximation to the exact solution of the sensitivity model is shown.

The state sensitivity operator equation for the model of Section 4.3 may be seen to be

$$\underline{\dot{v}}^p = \frac{d\left[\underline{CL}\right]^{-1}}{dp}\left[\underline{A}\,\underline{x} + \underline{B}\,\underline{u}\right] + \left[\underline{CL}\right]^{-1}\left[\underline{A}'\,\underline{x} + \underline{A}\,\underline{v}^p + \underline{B}'\,\underline{u} + \underline{B}\,\underline{u}'\right] \tag{4.4.1}$$

from Equation (2.3.5). The main distinguishing characteristic of this equation is that, in general, $\underline{u}' \neq \underline{0}$ if any dependent drivers are included in the network. From Equation (4.3.2)

$$\underline{u}' = \frac{d\underline{f}}{dp} = \frac{\partial \underline{f}}{\partial \underline{x}} \underline{v}^p + \frac{\partial \underline{f}}{\partial \underline{x}} \underline{w}^p + \frac{\partial \underline{f}}{\partial p}$$

or

$$\underline{u}' = \underline{h}(\underline{v}^p, \underline{w}^p, p) \qquad (4.4.1a)$$

where

$$\frac{\partial \underline{f}}{\partial \underline{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

and $\partial \underline{u} / \partial \underline{y}$ is similarly defined. Note that $\underline{u}'$ depends on the state and output operators $\underline{v}^p$ and $\underline{w}^p$ which in turn depend on the value of $\underline{u}'$. Thus, if an exact solution to Equation (4.4.1) is desired, an iteration process must be implemented to obtain the consistent $\underline{u}'$. However, in the nonlinear case under consideration, the first derivatives of the variables with respect to the parameter p can serve only as an approximation to the true changes induced by the varying parameter. Thus, rather than investigate iteration criteria for another iteration, the solution to Equation (4.4.1) will be approximated. This approximation is developed as follows. Consider the term $(\underline{B}' \underline{u} + \underline{B} \underline{u}')$

$$\underline{B}' \underline{u} + \underline{B} \underline{u}' = (\underline{B} + \underline{B}') (\underline{u} + \underline{u}') - \underline{B} \underline{u} - \underline{B}' \underline{u}' .$$

If $\underline{u} \gg \underline{u}'$, then

$$\underline{B}' \underline{u} + \underline{B} \underline{u}' \simeq \underline{B}' \underline{u}$$

and hence Equation (4.4.1) may be written

$$\underline{\dot{v}}^{p}(t) \simeq \frac{d\left[\underline{CL}\right]^{-1}}{dp}\left[\underline{A}\ \underline{x} + \underline{B}\ \underline{u}\right] + \left[\underline{CL}\right]^{-1}\left\{\underline{A}'\underline{x} + \underline{A}\ \underline{v}^{p}(t) + \underline{B}'\underline{u}\right\} \quad . \tag{4.4.2}$$

The algebraic equation may be similarly shown to be

$$\underline{w}^{p}(t) = \underline{C}\ \underline{v}^{p}(t) + \underline{C}'\underline{x} + \underline{D}'\underline{u} \quad . \tag{4.4.3}$$

No iteration process is needed for the nonlinear storage elements since a stepwise linear approximation has been adopted for this type element. Thus, Equations (4.4.2) and (4.4.3) may be integrated without necessitating an iterative process under the assumption $\underline{u}(t) \gg \frac{d\underline{u}(t)}{dp}$. Example 5.3.1 of Chapter V contains plots of predicted and actual changes in the output voltage of a nonlinear network. The predicted changes were computed using the approximation above. Although the agreement is very good for this example, this will not always be the case. The program described in the next section computes the $\underline{u}'$ directly from Equation (4.4.1a) and compares it with the driver value. If the approximation used is not satisfied, a warning message is printed to alert the user.

4.5 Computer Program VARNOL

Section 4.3 and Section 4.4 have suggested a representation scheme for nonlinear elements and considered the sensitivity operator equations for this case. This section describes a program which automatically formulates the state model and the sensitivity model using these techniques and performs a time solution for both models.

The program is implemented in the FORTRAN IV language in the interest of the widest possible distribution and application. None of the techniques are restricted to any machine. Figure 10 is a functional block diagram of the five phase program VARNOL(VARy NOnLinear). Phase 1 reads the input data, processes it for later phases, selects a tree set of elements, and computes the cutset matrix for the network. Phase 2 and Phase 3 are identical to those used in VARYIT and perform the functions of formulation of the state and sensitivity models. Phase 4 performs the time solution for the nonlinear network state model and sensitivity models. Phase 5 serves dual purposes in that it applies the variance approximation as one possible criteria of sensitivity when desired and also prints, punches, or plots the output data. The output tape of Phase 4 is compatible with the output tape of VARYIT's Phase 4 and, hence, the same output phase may be used with both programs.

The restrictions placed on networks which may be solved by the present program are as follows. Allowable element types are linear and nonlinear inductors and capacitors, resistors, independent and dependent voltage and current drivers. Mutual inductances are not allowed. No more than eighty elements may be included in the network. A total of twenty inductors and capacitors, twenty voltage and current drivers (including the dependent drivers) and twenty resistors in the cotree and twenty resistors in the tree (thus, forty resistors total) may be included. A maximum of forty nodes may exist in the network to be analyzed and each node may have no more than ten elements incident to it. The ultimate size of the network to be solved is limited only by the available storage in the computer; the number of elements may be

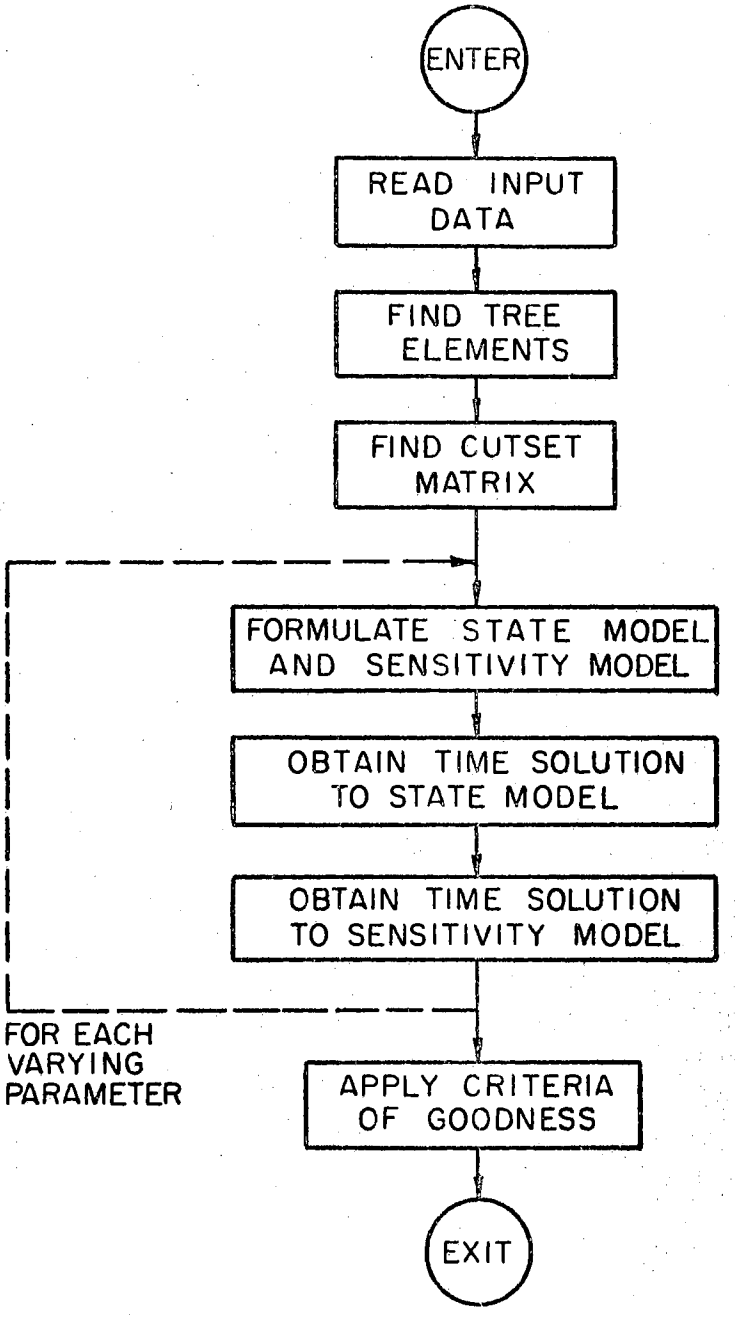


Figure 10. Operations Performed by VARNOL

increased by altering the appropriate DIMENSION statements.

This program may be used as a general analysis program, i.e., when no sensitivity information is desired, by setting the number of varying parameters to zero. In this manner the program may be used in modeling studies such as that conducted by Walden (43). When only general analysis is considered, the program bypasses all sensitivity model formulation and integration procedures.

The program VARNOL implements the following types of nonlinear models along with the required iteration techniques:

(a) nonlinear, dependent L and C values described by a polynomial equation of fourth degree;

(b) nonlinear L values described by a two-term power series in the arctangent for saturating iron core inductors;

(c) nonlinear R values to be described by a polynomial equation of fourth degree as a dependent voltage or current source;

(d) nonlinear dependent sources to be described by a two-variable power series of nine terms for simulation of a vacuum-tube triode plate circuit as a dependent source;

(e) nonlinear dependent sources to be described by a polynomial equation of fourth degree with any variable in the system the controlling variable;

(f) nonlinear dependent sources to be described by the semiconductor diode equation of the form

$$i = I_S(e^{bV} - 1)$$

where I_S and b are constants peculiar to each model or device; and

(g) nonlinear dependent sources to be described by a pair of piecewise linear equations,

$$y = a_1 x + b_1 \quad \text{for} \quad x > x_0$$

$$= a_2 x + b_2 \quad \text{for} \quad x < x_0$$

and

$$a_1 x_0 + b_1 = a_2 x_0 + b_2 \quad .$$

Appendix E presents in detail the parameters which characterize these nonlinearities. The models implemented in this program are those investigated by Walden (43) in his nonlinear analysis study. Inasmuch as the primary objective of this study has been the implementation of sensitivity operators, extensions of Walden's techniques have not been attempted.

The iteration processes implemented in VARNOL were originally developed in three distinct programs. VARNOL incorporates all of the above models into one program, but no iteration scheme has been developed to handle the general case in which arbitrary combinations of the models occur simultaneously in any one network. Instead nonlinear storage elements of type (a) and (b) may simultaneously occur in three particular classes of networks:

(1) the polynomial dependent drivers of model (c) and model (e);

(2) the triode model (d) and the piecewise linear equations (g); and

(3) the diode model (f).

The input data for VARNOL is approximately the same as that for VARYIT. Some effort has been expended in trying to user orient this program, but as is common with most nonlinear programs, the input data

remains relatively complex compared to that required for the linear program. The "desk" work that is necessary is:

(a) represent the nonlinearities by the appropriate method; dependent drivers and/or nonlinear storage elements;

(b) number the elements by type, resistors first, then inductors, current drivers, capacitors, and voltage sources;

(c) number the nodes of the network; and

(d) assign orientations to each element.

The input data to be entered then is essentially the same as that of VARYIT. More explicit information on the program and its input data is given in Appendix E.

Program VARNOL computes the time solution of the nonlinear network. An iteration procedure is implemented to compute initial conditions from the steady state assumption of Section 3.5.4. If desired, however, initial conditions for the state and sensitivity model may be specified. The initial conditions on the sensitivity model may also be calculated from Equation (3.5.2) if the sensitivity model is assumed to have reached steady state and the change in the drivers due to a parameter change is much less than the driver value itself. Equations (4.4.2) and (4.4.3) are used to calculate the sensitivity operators. The integration procedure chosen for VARNOL is the Runge-Kutta-Gill method given by Ralston and Wilff (51). It is necessary for the user to specify an integration increment and convergence criteria suitable for his particular problem. In order to assure stability of the integration the integration increment must be set to a value less than the shortest time constant of the system of equations.

The output of this program is the same as that provided by VARYIT for the general time solution. The variance approximation may be applied to any network variable when statistical independence of the components is assumed. Alternatively the complete network solution and the sensitivity partials may be printed or punched. As suggested in Section 3.4.8, any number of current or voltage meters may be incorporated in order to obtain selective print out or punch of the network variables. Multipass processing may be performed on the program output tape. This tape contains the complete network solution for every voltage and current at each integration increment and the complete sensitivity voltages and current for each varying parameter.

Even though no problems which were solved during the development, validation and subsequent use at Texas Instruments and Oklahoma State University have taken more than ten minutes to execute on the IBM 7040, it is felt that the techniques used in this program will achieve their full potentials only if the program is recoded with particular emphasis on decreasing solution time. Alternatively, the program storage requirements may be relaxed to take full advantage of the increased computer memory sizes available at most facilities. With larger computer memory banks much of the program linking and external input/output is unnecessary. However, even at facilities with extremely large memory banks, a tradeoff between memory allocated for use and execution time exists. The result of such a tradeoff is heavily dependent on the accounting procedures in use at the particular faciliy. Hence, no suggestions are included here as to the proper machine size and cost characteristics of this program.

4.6 Summary

This chapter has described the formal development of a design tool capable of efficiently supplying sensitivity information for a large class of nonlinear networks. A comprehensive study of general nonlinear analysis programs, such as those discussed in Section 4.2, led to the conclusion that Walden's state-space approach for the representation of nonlinear elements would provide a sound basis for such a design tool. Walden's program was limited to standard time-domain analysis with no capability for treating parameter variation or tolerance problems. The author has updated Walden's basic program and extended its capabilities to provide automatic formulation and solution of sensitivity operator models. Automatic tree selection and state-space model formulation provide desirable additional capability that was lacking in the original WALDEN program. The resulting program VARNOL provides an efficient, user-oriented, sensitivity analysis program applicable to a broad class of nonlinear networks and analogous non-electrical systems. Examples of the use of VARNOL are presented in Chapter V. Appendix E also discusses the input data requirements.

CHAPTER V

EXAMPLES OF SENSITIVITY ANALYSIS WITH VARYIT AND VARNOL

5.1 Introduction

This chapter demonstrates the use of VARYIT and VARNOL in sensitivity analysis. Three examples of linear network analysis are discussed in Section 5.2 illustrating, respectively, a general time solution, an impulse solution and a pole-zero analysis. Examples of nonlinear network analysis are presented in Section 5.3. These examples were chosen to illustrate the wide variety of problems for which VARNOL may be used and they represent a limited sample of the problems studied during this research activity.

5.2 Illustrations of the Use of VARYIT

Program VARYIT was used to compute the sensitivity measures for a number of networks. An example of each type of solution is presented below.

5.2.1 Example

In this example time solutions of the state model and sensitivity model are performed. It is assumed that all parameters are $\pm$ b percent of the nominal value and the distributions are such that the mean is equal to the nominal value, with the standard deviation M percent of

the mean. Consider constructing

$$Z(s) = \frac{(s + 2)(s + 5)}{(s + 1)(s + 3)}$$

as the input impedance to a linear network. Many alternative networks may be found by the classical synthesis techniques of network theory (52). Eight such networks are shown in Figure 11. Six of these networks are canonic forms, i.e., they contain the minimum number of elements required to realize Z(s). Since

$$V(s) = Z(s)\ I(s)$$

if a step current driver is used, the variation in the input impedance is reflected in the voltage across that driver. The ideal desired output G_o is the V(t) which results when all the parameters take on their mean values. Equation (2.4.3) for the mean square error reduces to

$$MSE \approx \sigma^2_{V(t)}$$

for this case. Program VARYIT may be utilized to directly compute MSE for these networks if the parameters are assumed to be statistically independent. If this assumption is not adequate VARYIT may still be used to provide the partial derivative terms of Equation (2.4.5). Once the mean square error has been obtained it becomes a fairly simple task to compute the measures of goodness of Equation (2.4.7) for each network.

The eight networks are easily encoded for analysis by VARYIT. Figure 12 shows Network A of Figure 11 with node numbers, element numbers, and orientation of elements assigned according to the rules

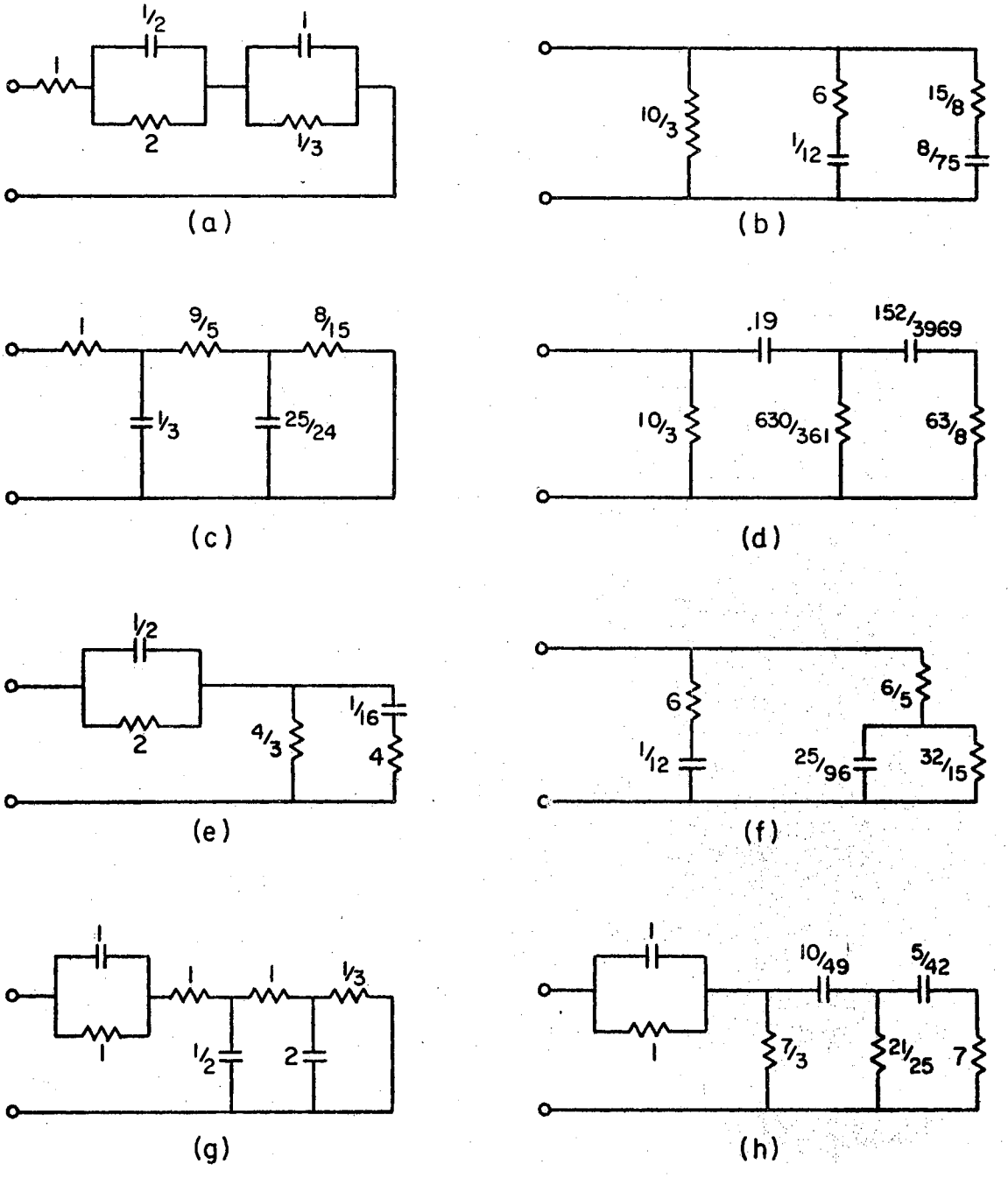


Figure 11. Alternative Networks Realizing Same Impedance

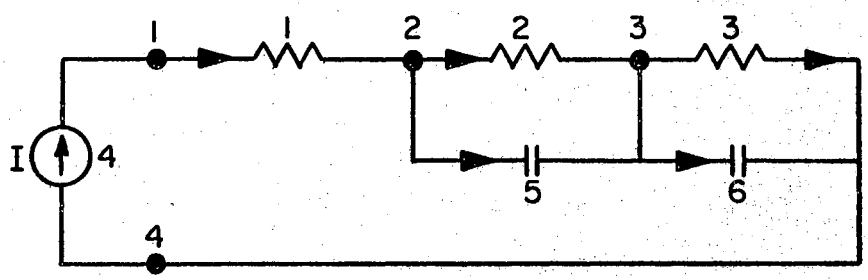


Figure 12. Network A With Assigned Conventions

of Section 3.5. This information is translated to cards by reference to Table VIII. The input data is shown in Table X where the sequence numbers correspond to those in Table IX. This input data is typical of that required when performing a transient time solution by integration of the state and sensitivity models. An abbreviated printed output is shown in Table II where all elements in the network have been allowed to vary. The standard deviation of each parameter has been assumed to be 0.1 of the mean value. Only the portion of the printed output referring to parameter 1 has been included. Figure 13 is a comparison of the mean square error of the six canonic networks, A, B, C, D, E, and F. Integration of the MSE for each of these networks with a uniform weighting yields the following ranking of networks on a multiparameter sensitivity basis: C (most desirable), A, E, F, B, and D (least desirable). Figure 14 compares the best canonic network with Networks G and H of Figure 11.

5.2.2 Example

In this example an impulse current driver is assumed to be present at the input to the networks of Figure 11. The calculations of Example 5.2.1 are repeated for this case. Table XI is the necessary input data for Network A. Figure 15 and Figure 16 are the plots of the output MSE for this case where the assumptions are the same as those of Example 5.2.1.

5.2.3 Example

In this example a pole-zero analysis is made for the networks of Figure 11. Table XII is the necessary input data for Network A. An

TABLE II

SAMPLE COMPUTER OUTPUT FOR TIME SOLUTION

```
NETWORK A

TOTAL NUMBER OF NETWORK ELEMENTS   6
 NUMBER OF VOLTAGE DRIVERS         0
 NUMBER OF CURRENT DRIVERS         1
 NUMBER OF CAPACITANCE ELEMENTS    2
 NUMBER OF INDUCTANCE ELEMENTS     0
 NUMBER OF CONDUCTANCE ELEMENTS    3
 NUMBER OF NODES   4

NETWORK TWO-TERMINAL ELEMENT CONNECTION ARRAY
 1    4   -0   -0
 1    2    5   -0
 2    3    5    6
 3    4    6   -0

ORIENTATION LIST
 1    2    3    4    2    3

NUMBER OF TREE BRANCHES     3
NUMBER OF COTREE CHORDS     3
TREE BRANCHES ARE     5     6     1
COTREE CHORDS ARE     4     2     3

CUTSET MATRIX
  -1   1   0
  -1   0   1
  -1   0   0

NOMINAL RESISTANCE VALUE FOR ELEMENT  1 IS  0.10000000E 01
NOMINAL RESISTANCE VALUE FOR ELEMENT  2 IS  0.20000000E 01
NOMINAL RESISTANCE VALUE FOR ELEMENT  3 IS  0.33333333E 00

CAPACITOR OR INDUCTOR NUMBER  5 VALUE IS  0.50000000E 00
CAPACITOR OR INDUCTOR NUMBER  6 VALUE IS  0.10000000E 01

LIST OF VARYING PARAMETERS
1    2    3    5    6

TIME SOLUTION REQUESTED

CAPACITOR OR INDUCTOR NUMBER  5 HAS AN INITIAL CONDITION OF  0.
CAPACITOR OR INDUCTOR NUMBER  6 HAS AN INITIAL CONDITION OF  0.

INTEGRATE TO T =  0.60000000E 01  IN STEPS OF 0.10000000E-01
          WITH PRECISION OF  0.10000000E-06

  0 SINE TYPE DRIVERS        1 WAVE TYPE DRIVERS
 KIND POSITION      PARAMETERS FOR DRIVER
    0    1   0.00000   0.00000   7.50000   0.00000   7.50000   1.00000

SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
SENSITIVITY INITIAL CONDITIONS FOR PARAMETER   1
              0.                0.
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
SENSITIVITY INITIAL CONDITIONS FOR PARAMETER   2
              0.                0.
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
SENSITIVITY INITIAL CONDITIONS FOR PARAMETER   3
              0.                0.
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
SENSITIVITY INITIAL CONDITIONS FOR PARAMETER   5
              0.                0.
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
SENSITIVITY INITIAL CONDITIONS FOR PARAMETER   6
              0.                0.
```

TABLE II (Continued)

```
NETWORK A
THE NUMBER OF THE VARYING PARAMETER IS     1
MATRIX INVERSION
      NUMBER OF DIMENSIONS =     1
      MATRIX OF DEGREES =      1
      R( 1, 1) =     0.0000    1.0000

      THE INVERSE OF R FOLLOWS

      NUMBER OF DIMENSIONS =    1
      R( 1, 1) =     1.0000
      DETERMINANT      0.0000    1.0000
      THE ORDER OF DETERMINANT IS =  1
      MATRIX OF DEGREES =      0

 D  22

POSITION
   1, 1            0.10000000E 01

 C  21

POSITION
   1, 1          -0.
   1, 2          -0.

 D  12

POSITION
   1, 1           0.
   2, 1           0.

 B  12

POSITION
   1, 1           0.              0.10000000E 01 X
   2, 1           0.              0.10000000E 01 X

 C  11

POSITION
   1, 1          -0.              0.50000000E 00 X
   1, 2          -0.
   2, 1          -0.
   2, 2          -0.              0.30000000E 01 X

 A  11

POSITION
   1, 1           0.             -0.50000000E 00 X
   1, 2           0.
   2, 1           0.
   2, 2           0.             -0.30000000E 01 X
```

TABLE II (Continued)

```
NETWORK A                    NUMBER OF VARYING PARAMETER    1

THE FOLLOWING COMPOSITE MATRICES HAVE A
COMMON DENOMINATOR =  0.                  0.10000000E 01 X

COMPOSITE A MATRIX

POSITION

   1, 1          0.                -0.50000000E 00 X
   1, 2          0.
   2, 1          0.
   2, 2          0.                -0.30000000E 01 X

COMPOSITE B MATRIX

POSITION

   1, 1          0.                 0.10000000E 01 X
   2, 1          0.                 0.10000000E 01 X

COMPOSITE C MATRIX

POSITION

   1, 1         -0.                 0.50000000E 00 X
   1, 2         -0.
   2, 1         -0.
   2, 2         -0.                 0.30000000E 01 X
   3, 1         -0.
   3, 2         -0.

COMPOSITE D MATRIX

POSITION

   1, 1          0.
   2, 1          0.
   3, 1          0.10000000E 01

THE FOLLOWING COMPOSITE MATRICES HAVE A
COMMON DENOMINATOR =  0.10000000E 01

COMPOSITE A MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1 -0.50000000E 00  0.
  2  0.              -0.30000000E 01

COMPOSITE B MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1  0.10000000E 01
  2  0.10000000E 01

COMPOSITE C MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1  0.50000000E 00 -0.
  2 -0.              0.30000000E 01
  3 -0.             -0.
```

TABLE II (Continued)

```
COMPOSITE D MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1  0.
  2  0.
  3  0.10000000E 01

THE FOLLOWING DIFFERENTIATED MATRIXES HAVE A
COMMON DENOMINATOR =  0.              0.10000000E 01 X  QUANTITY SQUARED

 DIFFERENTIATED COMPOSITE A MATRIX

POSITION

  1, 1          -0.
  1, 2           0.
  2, 1           0.
  2, 2          -0.

 DIFFERENTIATED COMPOSITE B MATRIX

POSITION

  1, 1           0.
  2, 1           0.

 DIFFERENTIATED COMPOSITE C MATRIX

POSITION

  1, 1           0.
  1, 2           0.
  2, 1           0.
  2, 2           0.
  3, 1           0.
  3, 2           0.

 DIFFERENTIATED COMPOSITE D MATRIX

POSITION

  1, 1           0.
  2, 1           0.
  3, 1          -0.10000000E 01

THE FOLLOWING COMPOSITE MATRICES HAVE A
COMMON DENOMINATOR =  0.10000000E 01

DIFFERERENTIATED COMPOSITE A MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1 -0.            0.
  2  0.           -0.

DIFFERERENTIATED COMPOSITE B MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1  0.
  2  0.

DIFFERERENTIATED COMPOSITE C MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1  0.            0.
  2  0.            0.
  3  0.            0.

DIFFERERENTIATED COMPOSITE D MATRIX EVALUATED AT PARAMETER =  0.10000000E 01

ROW
  1  0.
  2  0.
  3  0.10000000E 01
```

TABLE II (Continued)

```
THE FOLLOWING QUANTITIES ARE TO BE METERED
V  1
I  1
V  2
I  2
V  3
I  3
V  4
I  4
V  5
I  5
V  6
I  6

 TIME =        0.0000000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.                  I(  2) =  0.
V(  3) =  0.               I(  3) =  0.               V(  4) = -0.10000000E 01   I(  4) =  0.10000000E 01
V(  5) =  0.               I(  5) =  0.10000000E 01   V(  6) =  0.                  I(  6) =  0.10000000E 01

 TIME =        0.0500000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.97541183E-01     I(  2) =  0.48770592E-01
V(  3) =  0.46430473E-01   I(  3) =  0.13929142E 00   V(  4) = -0.11439717E 01     I(  4) =  0.10000000E 01
V(  5) =  0.97541183E-01   I(  5) =  0.95122941E 00   V(  6) =  0.46430473E-01     I(  6) =  0.86070858E 00

 TIME =        0.1000000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.19032522E 00     I(  2) =  0.95162612E-01
V(  3) =  0.86393578E-01   I(  3) =  0.25918074E 00   V(  4) = -0.12767188E 01     I(  4) =  0.10000000E 01
V(  5) =  0.19032522E 00   I(  5) =  0.90483738E 00   V(  6) =  0.86393578E-01     I(  6) =  0.74081926E 00

 TIME =        0.1500000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.27858414E 00     I(  2) =  0.13929207E 00
V(  3) =  0.12079017E 00   I(  3) =  0.36237051E 00   V(  4) = -0.13993743E 01     I(  4) =  0.10000000E 01
V(  5) =  0.27858414E 00   I(  5) =  0.86070793E 00   V(  6) =  0.12079017E 00     I(  6) =  0.63762949E 00

 TIME =        0.2000000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.36253861E 00     I(  2) =  0.18126930E 00
V(  3) =  0.15039560E 00   I(  3) =  0.45118682E 00   V(  4) = -0.15129342E 01     I(  4) =  0.10000000E 01
V(  5) =  0.36253861E 00   I(  5) =  0.81873070E 00   V(  6) =  0.15039560E 00     I(  6) =  0.54881319E 00

 TIME =        0.2500000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.44239857E 00     I(  2) =  0.22119928E 00
V(  3) =  0.17587726E 00   I(  3) =  0.52763177E 00   V(  4) = -0.16182758E 01     I(  4) =  0.10000000E 01
V(  5) =  0.44239857E 00   I(  5) =  0.77880072E 00   V(  6) =  0.17587726E 00     I(  6) =  0.47236823E 00

 TIME =        0.3000000

V(  1) =  0.10000000E 01   I(  1) =  0.10000000E 01   V(  2) =  0.51836370E 00     I(  2) =  0.25918185E 00
V(  3) =  0.19780954E 00   I(  3) =  0.59342861E 00   V(  4) = -0.17161732E 01     I(  4) =  0.10000000E 01
V(  5) =  0.51836370E 00   I(  5) =  0.74081815E 00   V(  6) =  0.19780954E 00     I(  6) =  0.40657139E 00
```

TABLE II (Continued)

THE FOLLOWING QUANTITIES SHOULD HAVE THEIR VARIANCES CALCULATED

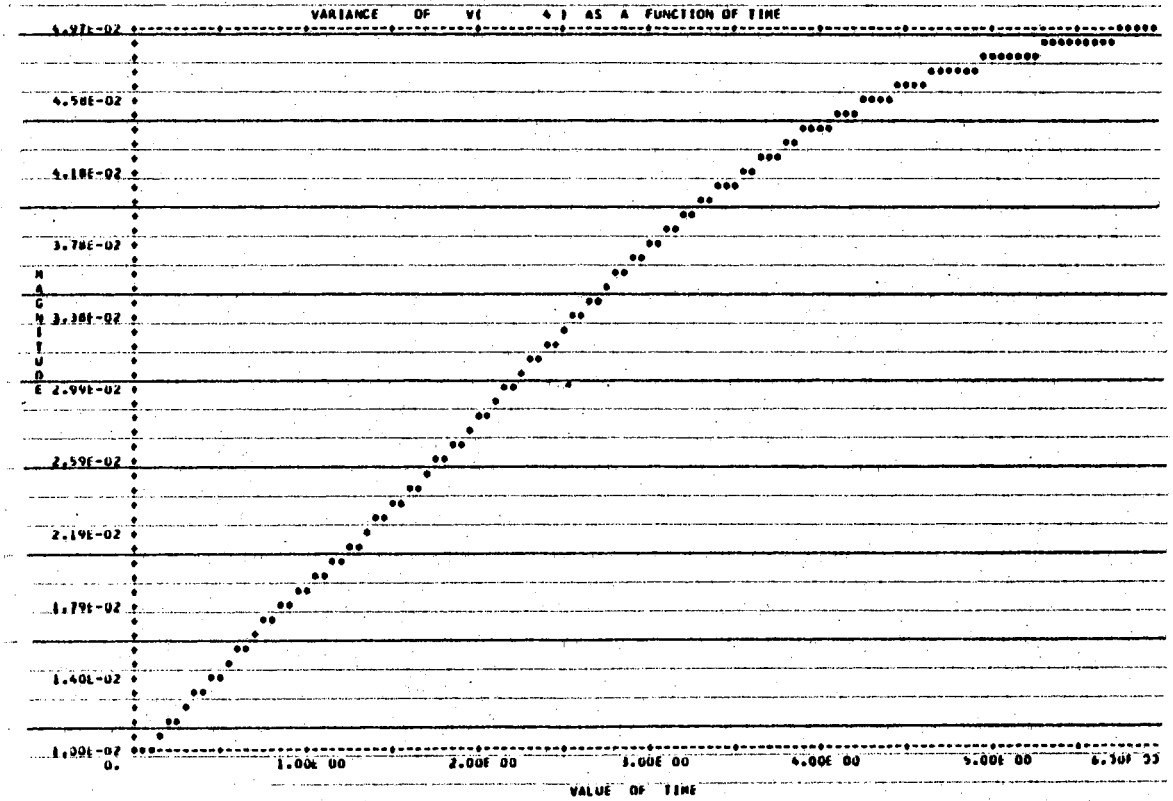
V 4

VARIANCE LIST FOR VARYING PARAMETERS

0.10000000E-01 0.40000000E-01 0.11111111E-02 0.25000000E-02 0.10000000E-01

VARIANCE FOR V(4)

TIME	VALUES					
0.00000000	0.10300300E-01	0.10110593E-01	0.10349810E-01	0.10776971E-01	0.11233804E-01	0.11722709E-01
0.30000000	0.12254404E-01	0.12787628E-01	0.13321806E-01	0.13850738E-01	0.14370945E-01	0.14879737E-01
0.59999999	0.15375841E-01	0.15858914E-01	0.16329325E-01	0.16787970E-01	0.17236123E-01	0.17675311E-01
0.89999999	0.18107209E-01	0.18533554E-01	0.18956074E-01	0.19376201E-01	0.19796201E-01	0.20216790E-01
1.19999999	0.20639468E-01	0.21065327E-01	0.21495279E-01	0.21930061E-01	0.22370229E-01	0.22816175E-01
1.49999999	0.23268130E-01	0.23726176E-01	0.24199271E-01	0.24660234E-01	0.25135804E-01	0.25616603E-01
1.79999998	0.26102179E-01	0.26592033E-01	0.27085584E-01	0.27582235E-01	0.28081340E-01	0.28582242E-01
2.09999996	0.29084264E-01	0.29586727E-01	0.30088956E-01	0.30590279E-01	0.31090045E-01	0.31587615E-01
2.39999995	0.32062375E-01	0.32573736E-01	0.33061136E-01	0.33544038E-01	0.34021964E-01	0.34494395E-01
2.69999993	0.34960944E-01	0.35421194E-01	0.35874775E-01	0.36321352E-01	0.36760625E-01	0.37192320E-01
2.99999991	0.37616200E-01	0.38032056E-01	0.38439707E-01	0.38839003E-01	0.39229817E-01	0.39612048E-01
3.29999989	0.39985624E-01	0.40350490E-01	0.40706614E-01	0.41053985E-01	0.41392610E-01	0.41722513E-01
3.59999987	0.42043735E-01	0.42356332E-01	0.42660373E-01	0.42955938E-01	0.43243124E-01	0.43522029E-01
3.89999986	0.43792770E-01	0.44059466E-01	0.44310268E-01	0.44557246E-01	0.44796602E-01	0.45028446E-01
4.19999984	0.45252978E-01	0.45470300E-01	0.45680582E-01	0.45883984E-01	0.46080666E-01	0.46270790E-01
4.49999976	0.46454512E-01	0.46632003E-01	0.46803612E-01	0.46968911E-01	0.47128655E-01	0.47282405E-01
4.79999971	0.47431510E-01	0.47574947E-01	0.47713252E-01	0.47846578E-01	0.47975080E-01	0.48098905E-01
5.09999967	0.48218197E-01	0.48333037E-01	0.48443743E-01	0.48550275E-01	0.48652824E-01	0.48751552E-01
5.39999962	0.48846698E-01	0.48937874E-01	0.49025772E-01	0.49110311E-01	0.49191606E-01	0.49269773E-01
5.69999957	0.49344911E-01	0.49417139E-01	0.49486553E-01	0.49553255E-01	0.49617362E-01	0.49678909E-01
5.99999952	0.49738046E-01	0.	0.	0.	0.	0.



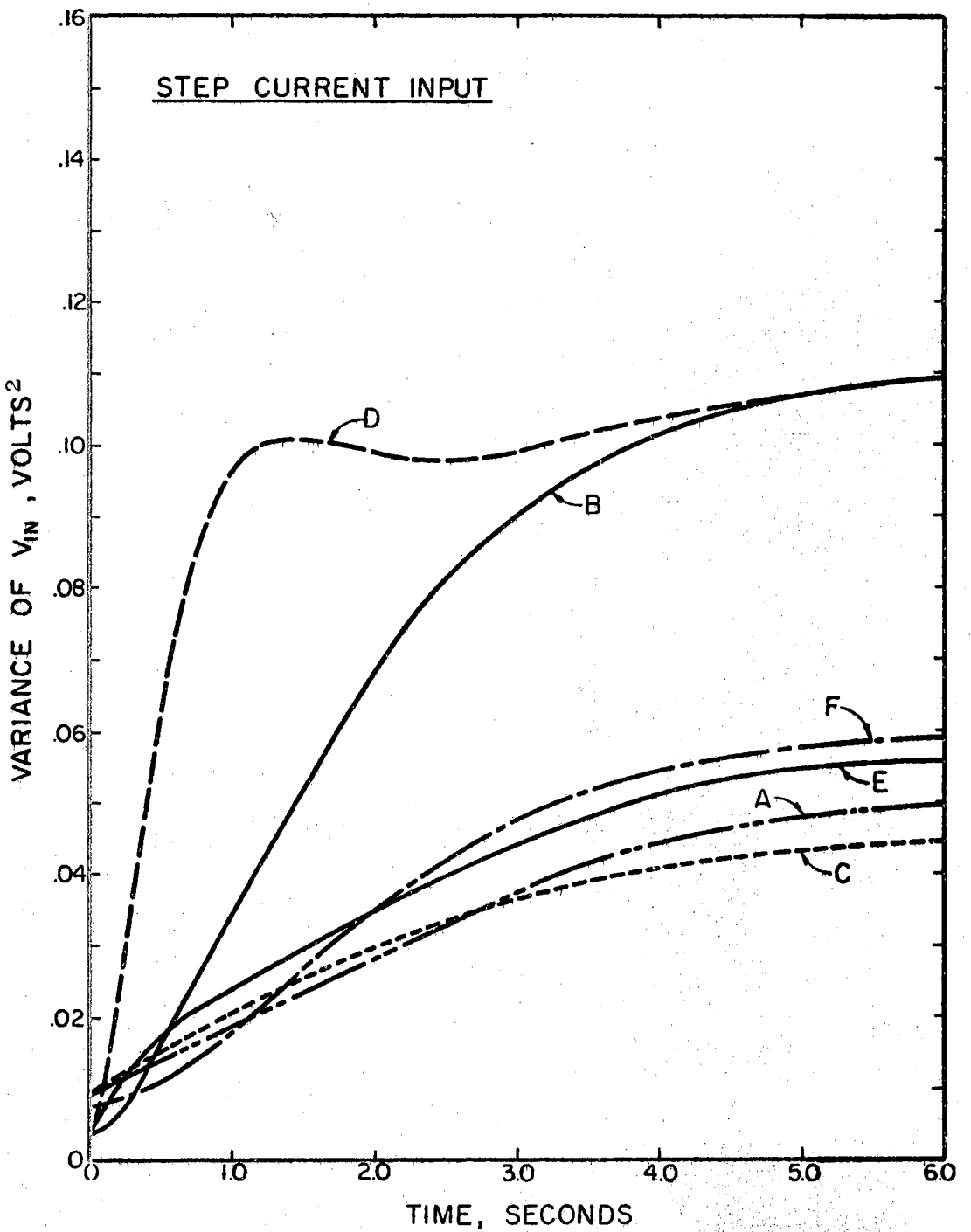


Figure 13. Mean Square Error for Canonic Networks

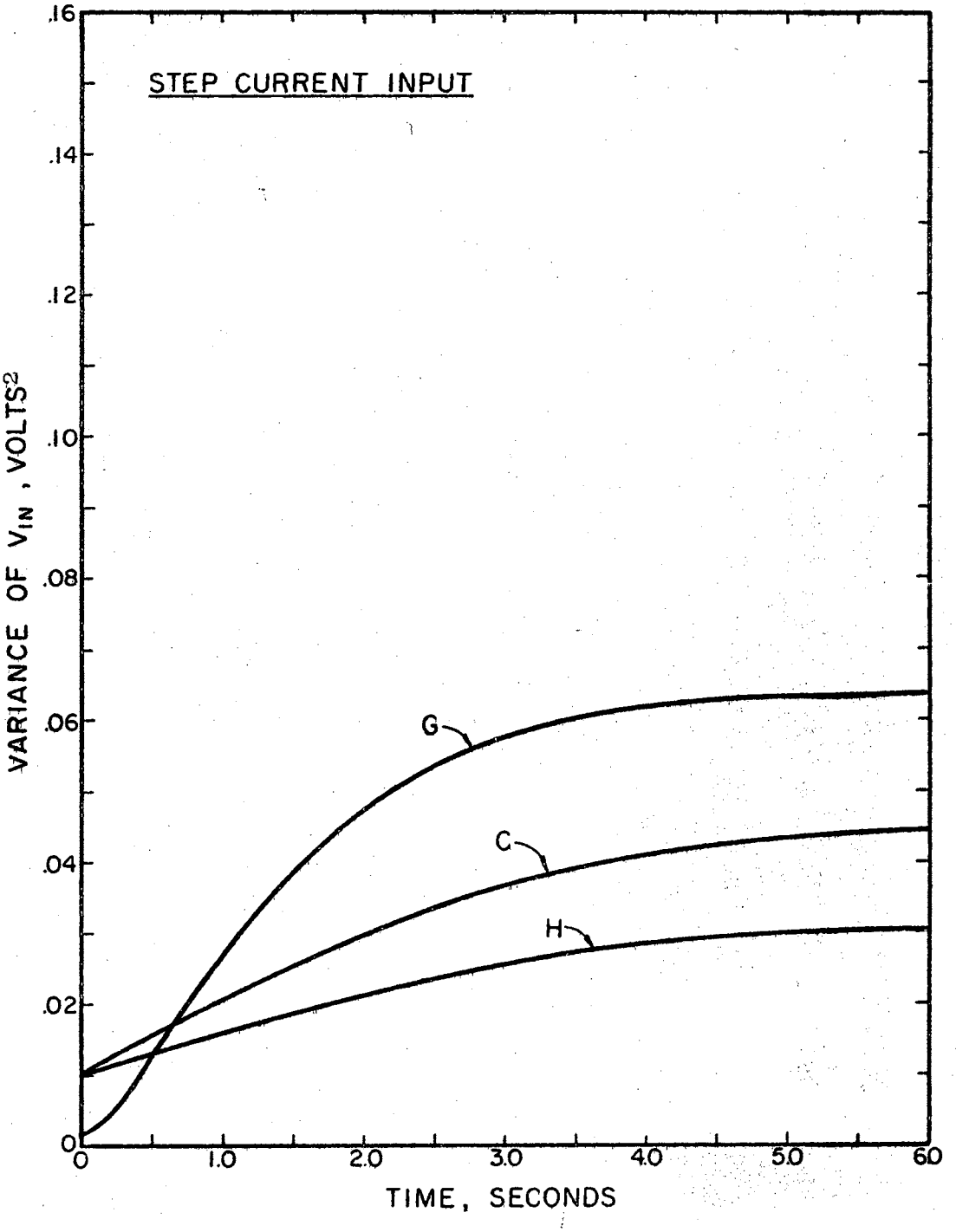


Figure 14. Mean Square Error for Non-Canonic Networks

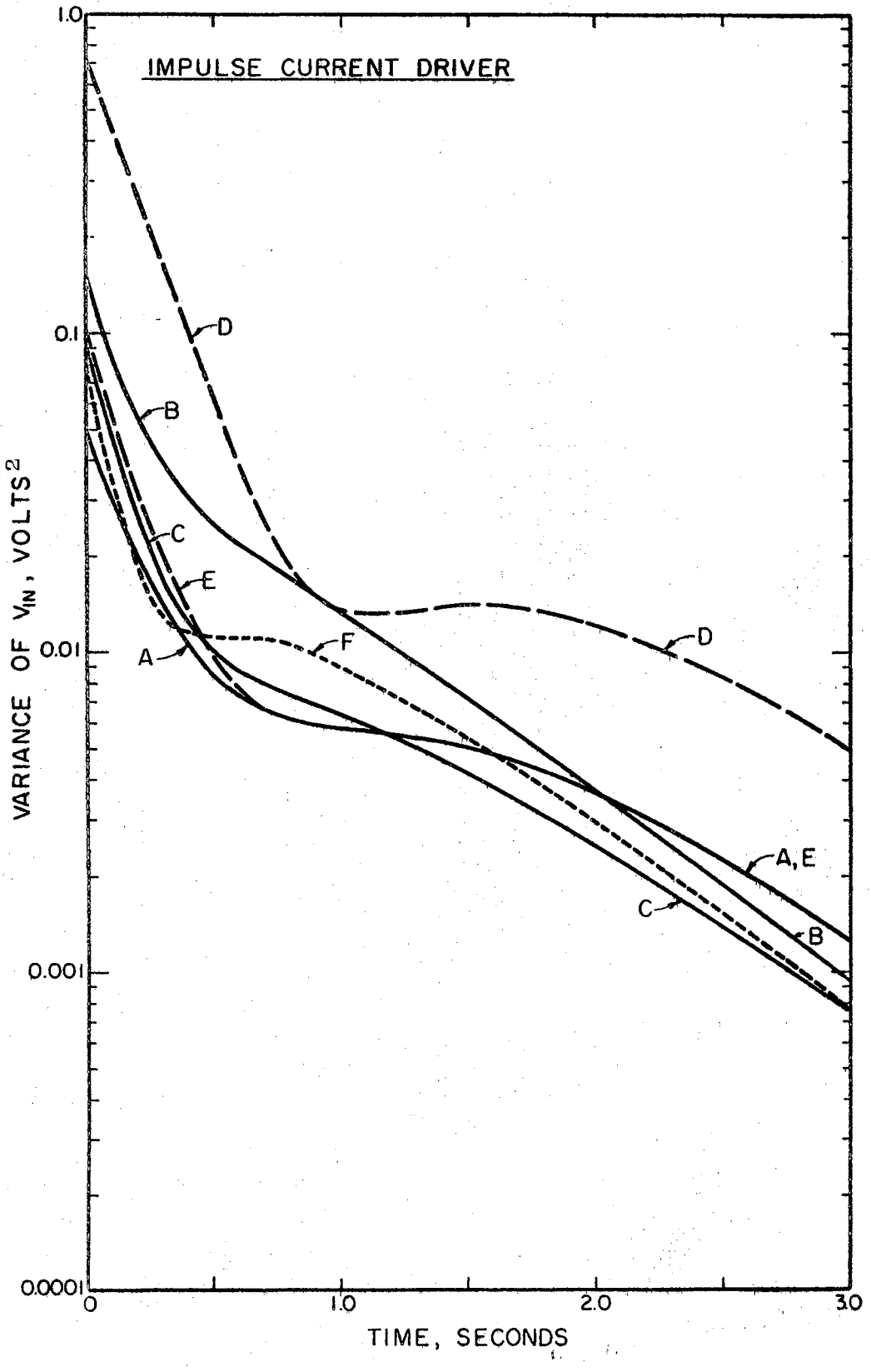


Figure 15. Mean Square Error for Canonic Networks

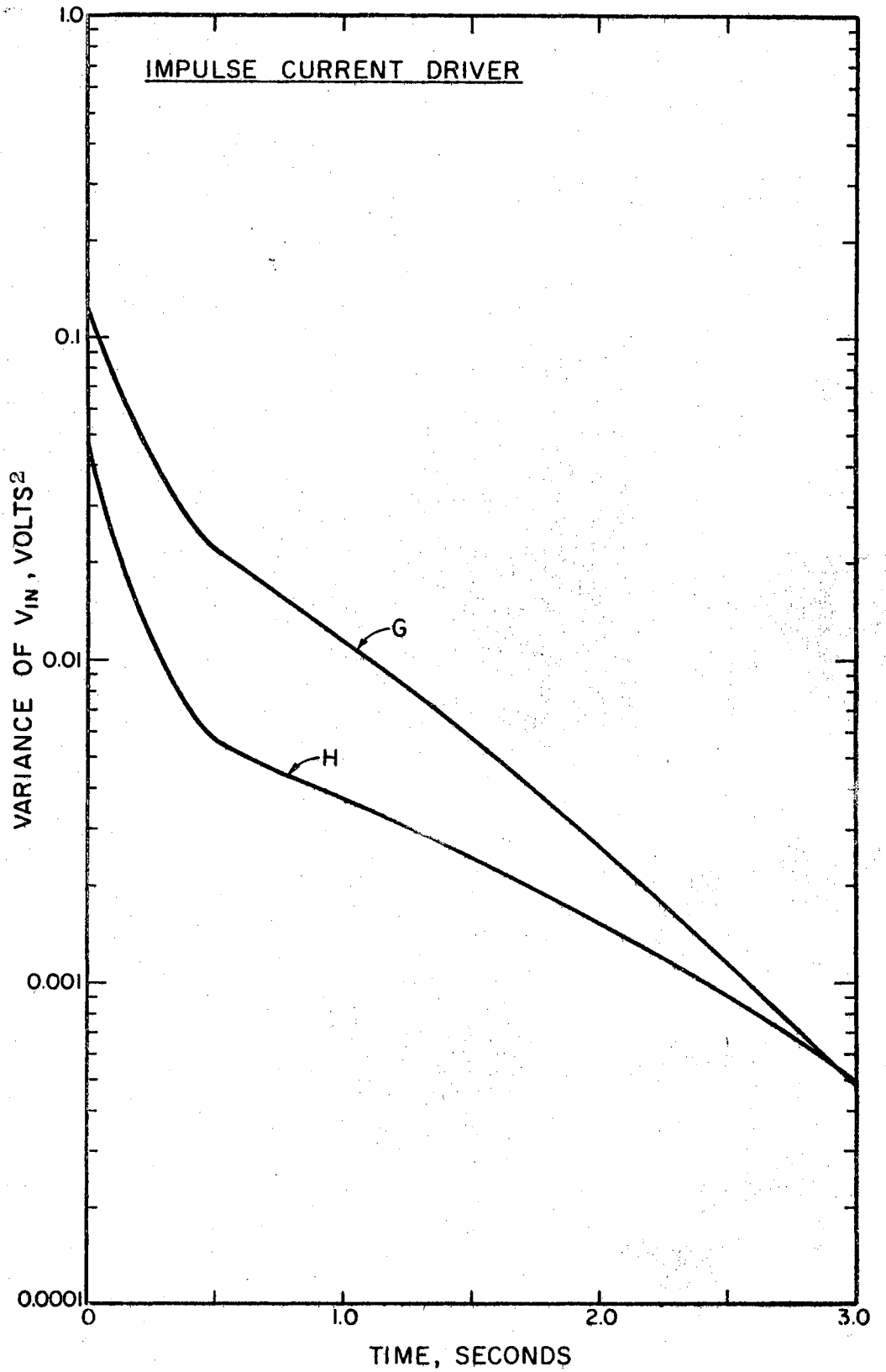


Figure 16. Mean Square Error for Non-Canonic Networks

abbreviated output for Network A formed by deleting the formulation output, as well as the sensitivities to all parameters except 1 and 2, is given in Table III. The symbols used are those of Section 2.5 and

$$\begin{bmatrix} \underline{I}_{CG}(s) \\ \underline{V}_{TG}(s) \end{bmatrix} = \underline{P}(s) \begin{bmatrix} \underline{E}_{D}(s) \\ \underline{J}_{D}(s) \end{bmatrix} .$$

The matrix polynomial $\underline{P}(s)$ is denoted in the output as the "transfer function matrix." The program provides the changes in the coefficients of the characteristic equation for this network as well as the changes in the numerator polynomial coefficients. After the roots of the characteristic equation are computed the real and imaginary parts of ds_i/dp are found. The sensitivity output is that computed by Equation (2.2.4). In this particular example only one driver is present and three resistors. Thus, $\underline{P}(s)$ is of dimension 3 x 1 and relates

$$\begin{bmatrix} I_2(s) \\ I_3(s) \\ V_1(s) \end{bmatrix} = \underline{P}(s)\ I_4(s) = \begin{bmatrix} p_{11}(s) \\ p_{21}(s) \\ p_{31}(s) \end{bmatrix} I_4(s) .$$

The characteristic equation zeros provide the pole sensitivities of the input impedance. These pole sensitivities are listed in Table IV for each varying parameter for the networks of Figure 11.

5.3 Illustrations of the Use of VARNOL

The program VARNOL has been used in the analysis of many circuits. A few examples of the types of problems considered are discussed in

TABLE III

SAMPLE COMPUTER OUTPUT FOR POLE-ZERO SOLUTION

```
NETWORK A

TOTAL NUMBER OF NETWORK ELEMENTS    6
 NUMBER OF VOLTAGE DRIVERS          0
 NUMBER OF CURRENT DRIVERS          1
 NUMBER OF CAPACITANCE ELEMENTS     2
 NUMBER OF INDUCTANCE ELEMENTS      0
 NUMBER OF CONDUCTANCE ELEMENTS     3
 NUMBER OF NODES   4

 NETWORK TWO-TERMINAL ELEMENT CONNECTION ARRAY
  1    4   -0   -0
  1    2    5   -0
  2    3    5    6
  3    4    6   -0

 ORIENTATION LIST
  1    2    3    4    2    3

 NUMBER OF TREE BRANCHES     3
 NUMBER OF COTREE CHORDS     3
 TREE BRANCHES ARE     5    6    1
 COTREE CHORDS ARE     4    2    3

 CUTSET MATRIX
   -1   1   0
   -1   0   1
   -1   0   0

 NOMINAL RESISTANCE VALUE FOR ELEMENT  1 IS  0.10000000E 01
 NOMINAL RESISTANCE VALUE FOR ELEMENT  2 IS  0.20000000E 01
 NOMINAL RESISTANCE VALUE FOR ELEMENT  3 IS  0.33333333E 00

 CAPACITOR OR INDUCTOR NUMBER  5 VALUE IS  0.50000000E 00
 CAPACITOR OR INDUCTOR NUMBER  6 VALUE IS  0.10000000E 01

 LIST OF VARYING PARAMETERS
 1    2    3    5    6

 POLE-ZERO SENSITIVITY REQUESTED
```

TABLE III (Continued)

```
 NETWORK A                THE NO OF THE VARYING PARAMETER IS     1

            H(  1)=   -0.40000000E 01

            H(  2)=   -0.30000000E 01

                  R MATRIXES FOLLOW
NO.
  1
   0.30000000E 01 0.
   0.              0.99999997E 00
  2
  -0.59604645E-07 0.
  -0.              0.29802322E-07


                  TRANSFER FUNCTION MATRIX COEFFICIENT OF S TO POWER    2
   0.
   0.
   0.10000000E 01

                  TRANSFER FUNCTION MATRIX COEFFICIENT OF S TO POWER    1
   0.10000000E 01
   0.30000000E 01
   0.40000000E 01

                  TRANSFER FUNCTION MATRIX COEFFICIENT OF S TO POWER    0
   0.30000000E 01
   0.29999999E 01
   0.30000000E 01

                  CHANGES IN COEFFICIENTS OF CHARACTERISTIC EQUATION.

                           DH(  1)=-0.
                           DH(  2)=-0.

CHANGES IN NUMERATOR POLYNOMIAL COEFFICIENTS IN DESCENDING ORDER
S TO POWER

  2
   0.
   0.
   0.10000000E 01
  1
   0.
   0.
   0.40000000E 01
 -0
   0.
   0.
   0.30000000E 01
```

TABLE III (Continued)

```
ROOT   1=-0.99999997E 00+J* 0.

        R EVALUATED AT SI

     0.39999999E 01-0.          0.          0.
     0.          0.             0.19999999E 01-0.

        RS/TRACE OF RS

     0.66666667E 00-0.          0.          -0.
     0.          -0.            0.33333333E 00-0.

ROOT   2=-0.30000000E 01+J* 0.

        R EVALUATED AT SI

     0.12000000E 02-0.          0.          0.
     0.          0.             0.99999999E 01-0.

        RS/TRACE OF RS

     0.54545455E 00-0.          0.          -0.
     0.          -0.            0.45454546E 00-0.

NO.       ROOT REAL       ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI          SENSITIVITY
  1    -0.99999997E 00 3.                  0.          0.          0.          0.
  2    -0.30000000E 01 0.                  0.          0.          0.          0.

NUMERATOR ELEMENT   1  1

        NUMERATOR IN DESCENDING ORDER
     0.10000000E 01 0.30000000E 01

CONSTANT MULTIPLIER IS  0.10000000E 01
SENSITIVITY OF CONSTANT MULTIPLIER IS  0.

NO.       ROOT REAL       ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI          SENSITIVITY
  1    -0.30000000E 01 0.                  -0.         0.          -0.         -0.

NUMERATOR ELEMENT   2  1

        NUMERATOR IN DESCENDING ORDER
     0.10000000E 01 0.99999996E 00

CONSTANT MULTIPLIER IS  0.30000000E 01
SENSITIVITY OF CONSTANT MULTIPLIER IS  0.

NO.       ROOT REAL       ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI          SENSITIVITY
  1    -0.99999996E 00 0.                  -0.         0.          -0.         -0.

NUMERATOR ELEMENT   3  1

        NUMERATOR IN DESCENDING ORDER
     0.10000000E 01 0.40000000E 01 0.30000000E 01

CONSTANT MULTIPLIER IS  0.10000000E 01
SENSITIVITY OF CONSTANT MULTIPLIER IS  0.10000000E 01

NO.       ROOT REAL       ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI          SENSITIVITY
  1    -0.99999997E 00 0.                  -0.         -0.         -0.         -0.
  2    -0.30000000E 01 0.                  0.          0.          0.          0.
```

TABLE III (Continued)

```
NETWORK A                THE NO OF THE VARYING PARAMETER IS    2

                CHANGES IN COEFFICIENTS OF CHARACTERISTIC EQUATION.

                                DH( 1)= 0.50000000E 00
                                DH( 2)= 0.15000000E 01

CHANGES IN NUMERATOR POLYNOMIAL COEFFICIENTS IN DESCENDING ORDER
S TO POWER

  2
    0.
    0.
    0.10000000E 01
  1
   -0.50000000E 00
   -0.
   -0.50000000E 00
 -0
   -0.15000000E 01
   -0.15000000E 01
   -0.15000000E 01


ROOT   1=-0.99999997E 00+J* 0.

        R EVALUATED AT S1

    0.39999999E 01-0.          0.          0.
    0.          0.             0.19999999E 01-0.

        RS/TRACE OF RS

    0.66666667E 00-0.          0.          -0.
    0.          -0.            0.33333333E 00-0.

ROOT   2=-0.30000000E 01+J* 0.

        R EVALUATED AT S1

    0.12000000E 02-0.          0.          0.
    0.          0.             0.99999998E 01-0.

        RS/TRACE OF RS

    0.54545455E 00-0.          0.          -0.
    0.          -0.            0.45454546E 00-0.

NO.      ROOT REAL      ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI      SENSITIVITY
 1   -0.99999997E 00 0.          0.33333334E 00 0.      0.16666667E 00 0.
 2   -0.30000000E 01 0.          0.27272728E 00 0.      0.13636364E 00 0.

NUMERATOR ELEMENT   1  1

        NUMERATOR IN DESCENDING ORDER
    0.10000000E 01 0.30000000E 01

CONSTANT MULTIPLIER IS  0.10000000E 01
SENSITIVITY OF CONSTANT MULTIPLIER IS -0.25000000E 00

NO.      ROOT REAL      ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI      SENSITIVITY
 1   -0.30000000E 01 0.          -0.          -0.      -0.          -0.

NUMERATOR ELEMENT   2  1

        NUMERATOR IN DESCENDING ORDER
    0.10000000E 01 0.99999996E 00

CONSTANT MULTIPLIER IS  0.30000000E 01
SENSITIVITY OF CONSTANT MULTIPLIER IS -0.

NO.      ROOT REAL      ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI      SENSITIVITY
 1   -0.99999996E 00 0.          0.50000000E 00-0.      0.25000000E 00 0.

NUMERATOR ELEMENT   3  1

        NUMERATOR IN DESCENDING ORDER
    0.10000000E 01 0.40000000E 01 0.30000000E 01

CONSTANT MULTIPLIER IS  0.10000000E 01
SENSITIVITY OF CONSTANT MULTIPLIER IS  0.50000000E 00

NO.      ROOT REAL      ROOT IMAG.      REAL AND IMAGINARY PARTS OF DSI      SENSITIVITY
 1   -0.99999997E 00 0.          0.37252901E-07-0.      0.18626451E-07 0.
 2   -0.30000000E 01 0.          0.45000000E 01-0.      0.22500000E 01 0.
```

TABLE IV

POLE SENSITIVITIES FOR CANONIC NETWORKS

Network	Parameter Number	Root	Sensitivity
A	1	-1	0.00
		-3	0.00
	2	-1	0.166
		-3	0.136
	3	-1	9.00
		-3	12.3
	5	-1	0.66
		-3	0.545
	6	-1	1.00
		-3	1.36
B	1	-1	0.045
		-3	0.042
	2	-1	0.020
		-3	0.024
	3	-1	0.126
		-3	0.165
	5	-1	0.777
		-3	0.818
	6	-1	0.888
		-3	1.09
C	1	-1	0.00
		-3	0.00
	2	-1	0.267
		-3	0.319
	3	-1	2.81
		-3	3.06
	5	-1	0.777
		-3	0.818
	6	-1	0.888
		-3	1.09
D	1	-1	0.045
		-3	0.042
	2	-1	0.098
		-3	0.088
	3	-1	0.014
		-3	0.019
	5	-1	0.684
		-3	0.588
	6	-1	0.982
		-3	1.320
E	1	-1	0.166
		-3	0.136

TABLE IV (Continued)

Network	Parameter Number	Root	Sensitivity
E	2	-1	0.141
		-3	0.192
	3	-1	0.047
		-3	0.064
	5	-1	0.666
		-3	0.545
	6	-1	1.00
		-3	1.36
F	1	-1	0.020
		-3	0.024
	2	-1	0.100
		-3	0.119
	3	-1	0.175
		-3	0.192
	5	-1	0.777
		-3	0.818
	6	-1	0.888
		-3	1.09

this section.

5.3.1 Example

A half wave rectifier and pi filter circuit shown in Figure 17 is to be analyzed for the effects that individual parameter variations induce in the output voltage. The diode is modeled by use of a dependent current source in which the current is given by the ideal semiconductor diode equation (model f of Section 4.5). For an input voltage of $120\sin(2\pi(60)t)$ the ac steady state output voltage across the load resistor exhibits a 0.46 percent ripple component as shown in Figure 18.

The sensitivity models were integrated with the initial conditions for the nominal network set equal to the steady state ac values (e.g., at t = 0 the voltage across C_2 is 17.39 volts) and initial conditions on the sensitivity models were set equal to zero. Figures 19 and 20 present a comparison between actual changes in the output voltage with respect to five percent changes in the inductor and load resistor from their nominal values and changes predicted from the sensitivity operator (see Equation 2.3.15).

The monotonic increase shown in Figure 20 indicates that the sensitivity model solution has not reached its steady state value since the filter requires approximately 0.15 seconds for transients to pass from its solution. When the sensitivity model is integrated over such a long period in this example, the predicted output bears little resemblance to the actual output. This is due to the build up of error in the approximation method of solving the sensitivity model. Errors in prediction of the output voltage for the increment in R_L in this

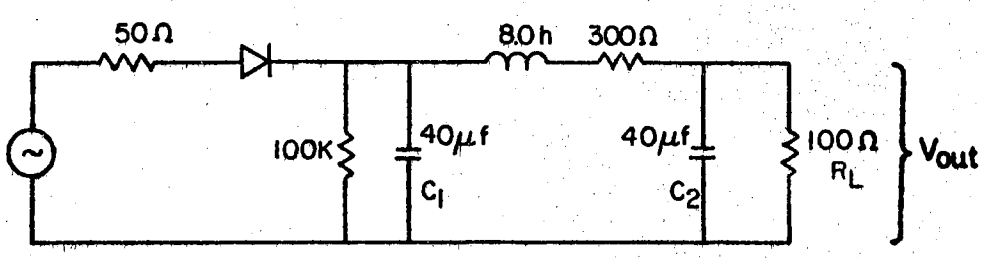


Figure 17. Circuit for Example 5.3.1

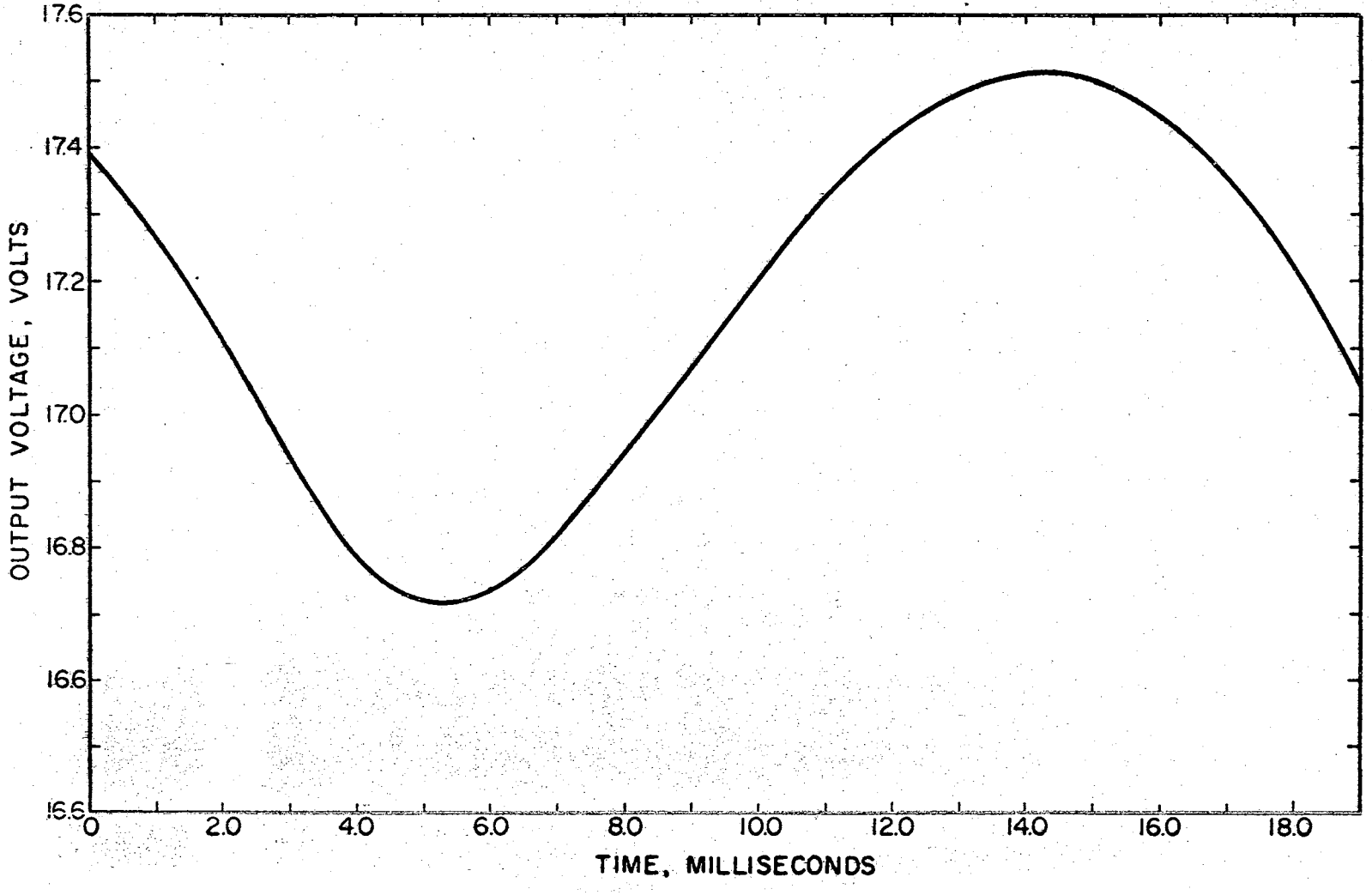


Figure 18. Output Voltage for Nominal Parameter Values for Example 5.3.1

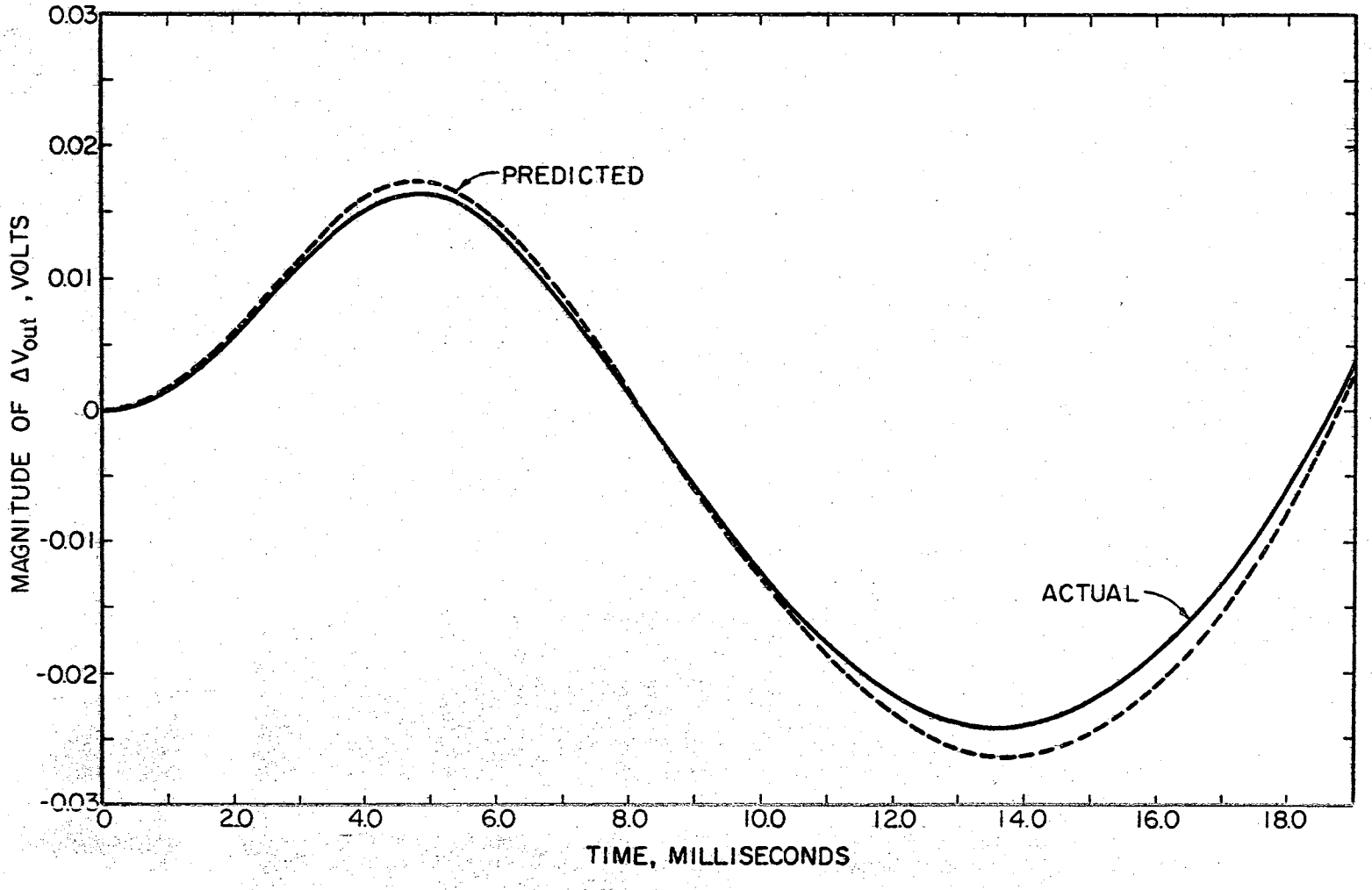


Figure 19. Comparison of Actual and Predicted Changes in Output Voltage for Five Percent Change in L

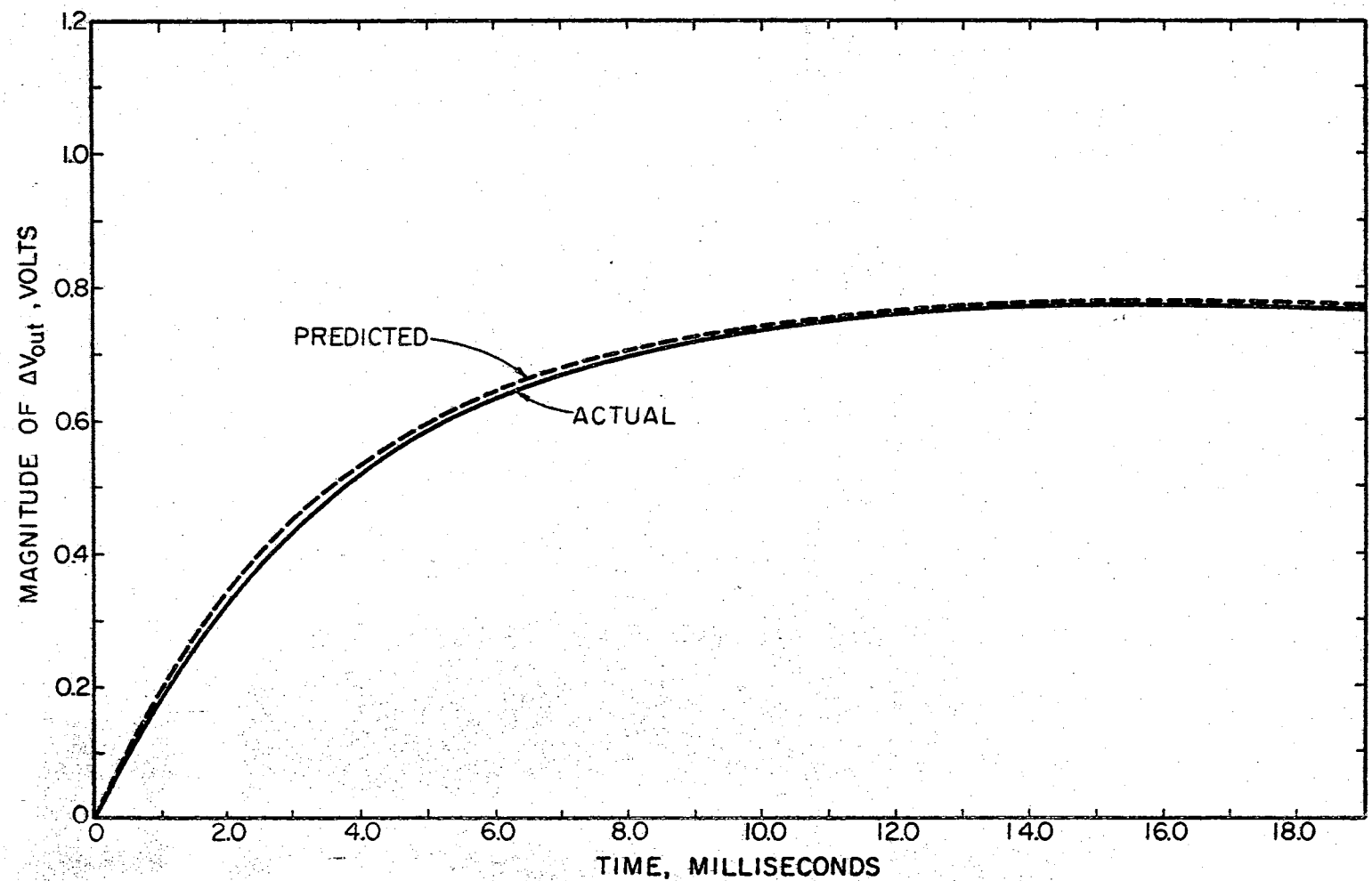


Figure 20. Comparison of Actual and Predicted Changes in Output Voltage for Five Percent Change in R_L

example did not exceed 3.5 percent of the true change over two time cycles of the input voltage.

This example illustrates several points. First, the sensitivity operators are very accurate prediction tools only in the vicinity of the time origin and it is, in general, difficult to predict in advance the interval over which they describe the system reasonably well. Secondly, it was found that by decreasing the step integration size the divergence tendencies of the sensitivity operator solutions could be lessened. This effect was not explored thoroughly in this study and no further discussion will be presented.

5.3.2 Example

A third order, low pass active filter is realized when the parameters in Figure 21 take on the values

$$R_g + R_1 = 0.5 \qquad C_1 = 1.44$$

$$R_2 = 1.0 \qquad C_2 = 1.743$$

$$R_3 = 1.0 \qquad C_3 = 0.404$$

with the transfer function for this circuit becoming

$$T(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{2}{s^3 + 1.75w_o s^2 + 2.15w_o^2 s + w_o^3}$$

where

$$w_o = 1.25 \text{ radians/sec.}$$

The network may be modeled as in Figure 21(b) where the dependent voltage driver is a linearly dependent driver with a gain of unity (model g of Section 4.5). It is desired to evaluate the step response

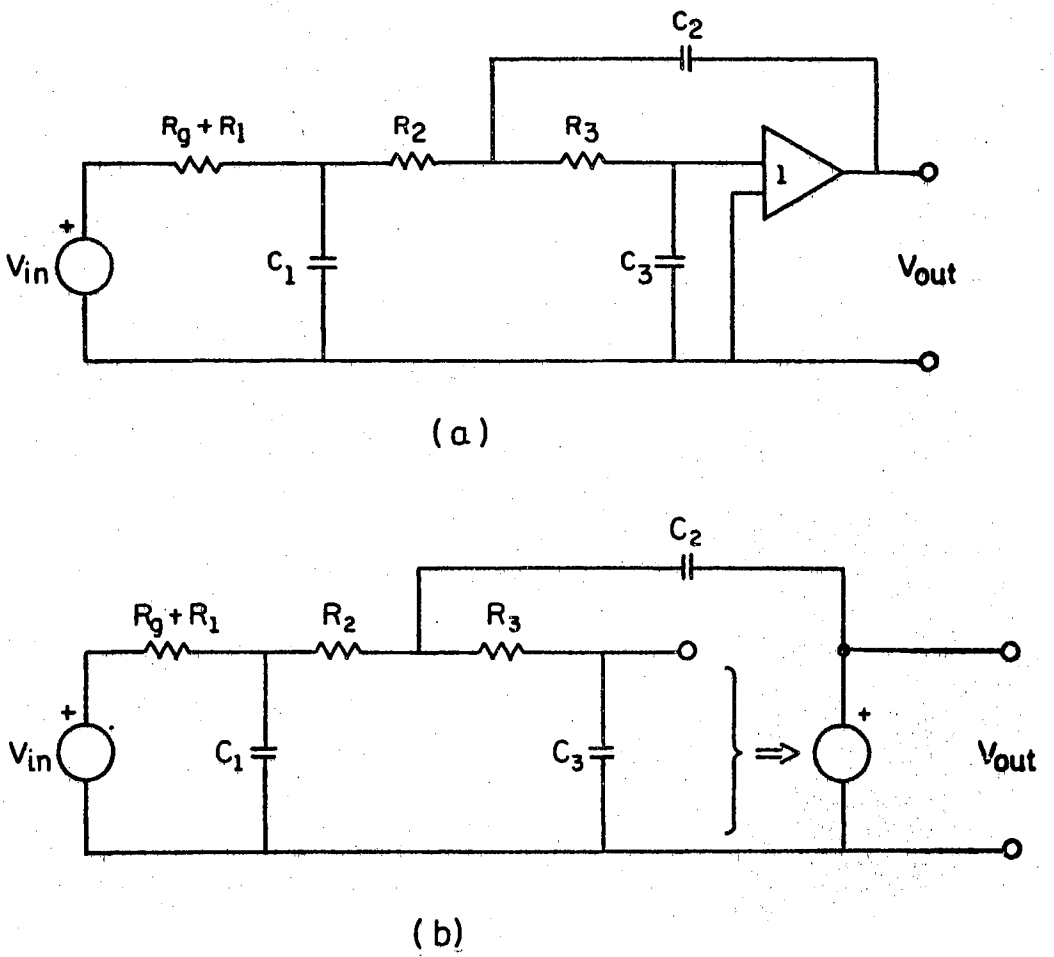


Figure 21. Filter Network (a) and Computer Model (b)

of this filter and evaluate the effects of multiparameter element variation. The step response is shown in Figure 22 and the variance of this output voltage is presented in Figure 23 for four different choices of component tolerances. The parameter values are assumed to be equally likely to be anywhere between the lower limit L and the upper limit U. The standard deviation for each parameter is easily seen to be

$$\sigma = \frac{U - L}{\sqrt{12}}$$

or if M is the tolerance and μ the mean

$$U = \mu + M\mu$$

$$L = \mu - M\mu$$

and

$$\sigma = .578\ M\mu \quad .$$

Four different sets of variances for the six varying parameters were supplied as input data in obtaining Figure 23. The designer may now directly apply his economic weighting factors (e.g., the relative costs of capacitor and resistor tolerances) when choosing the best set of tolerances for his application.

5.3.3 Example

An analog computer compensation network shown in Figure 24 was analyzed for the effects of imprecise resistors and capacitors. The high gain negative feedback in this circuit presented no problems in the analysis but the computer model contained a circuit of capacitors and voltage sources when no internal resistance for the input voltage

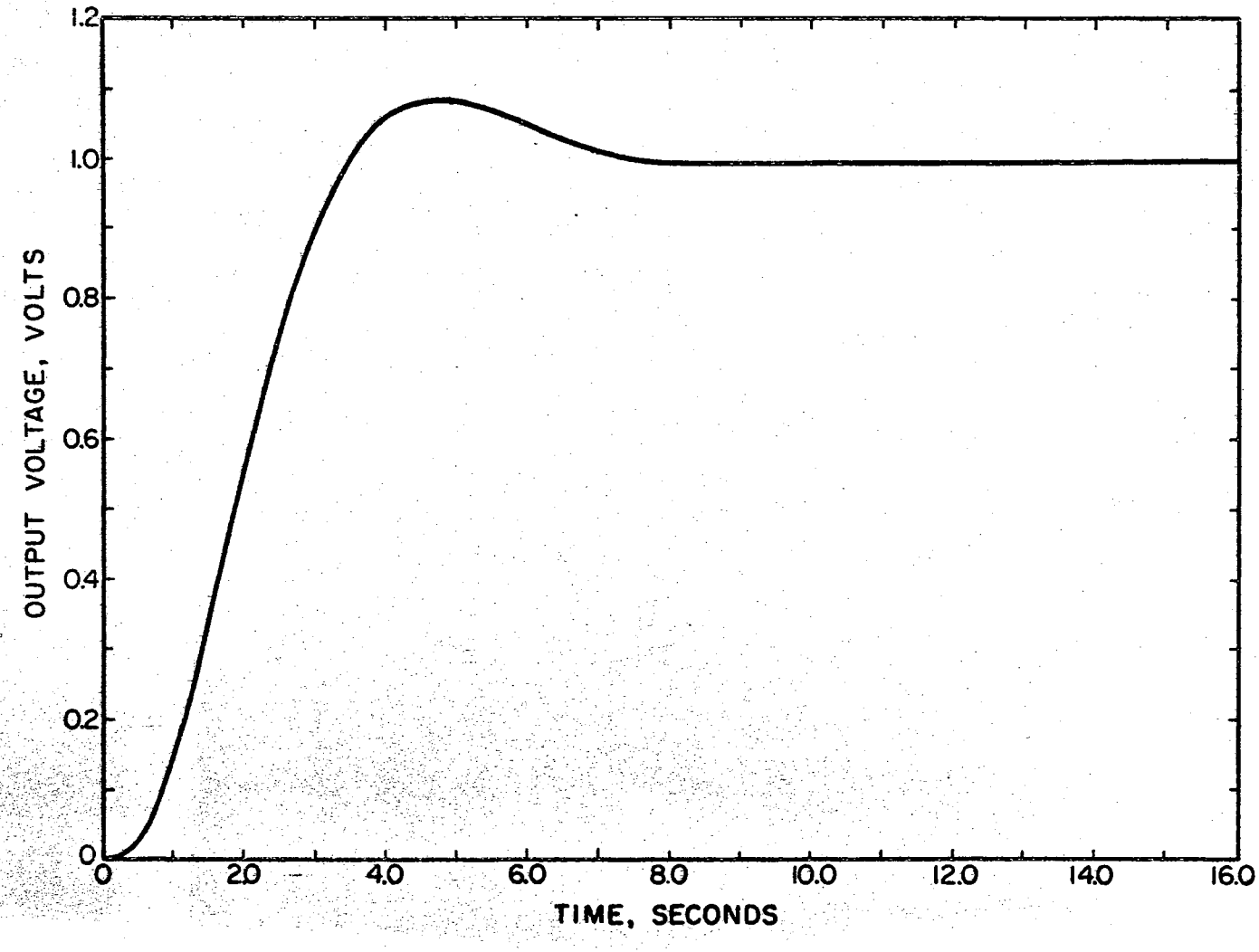


Figure 22. Step Response of Active Filter

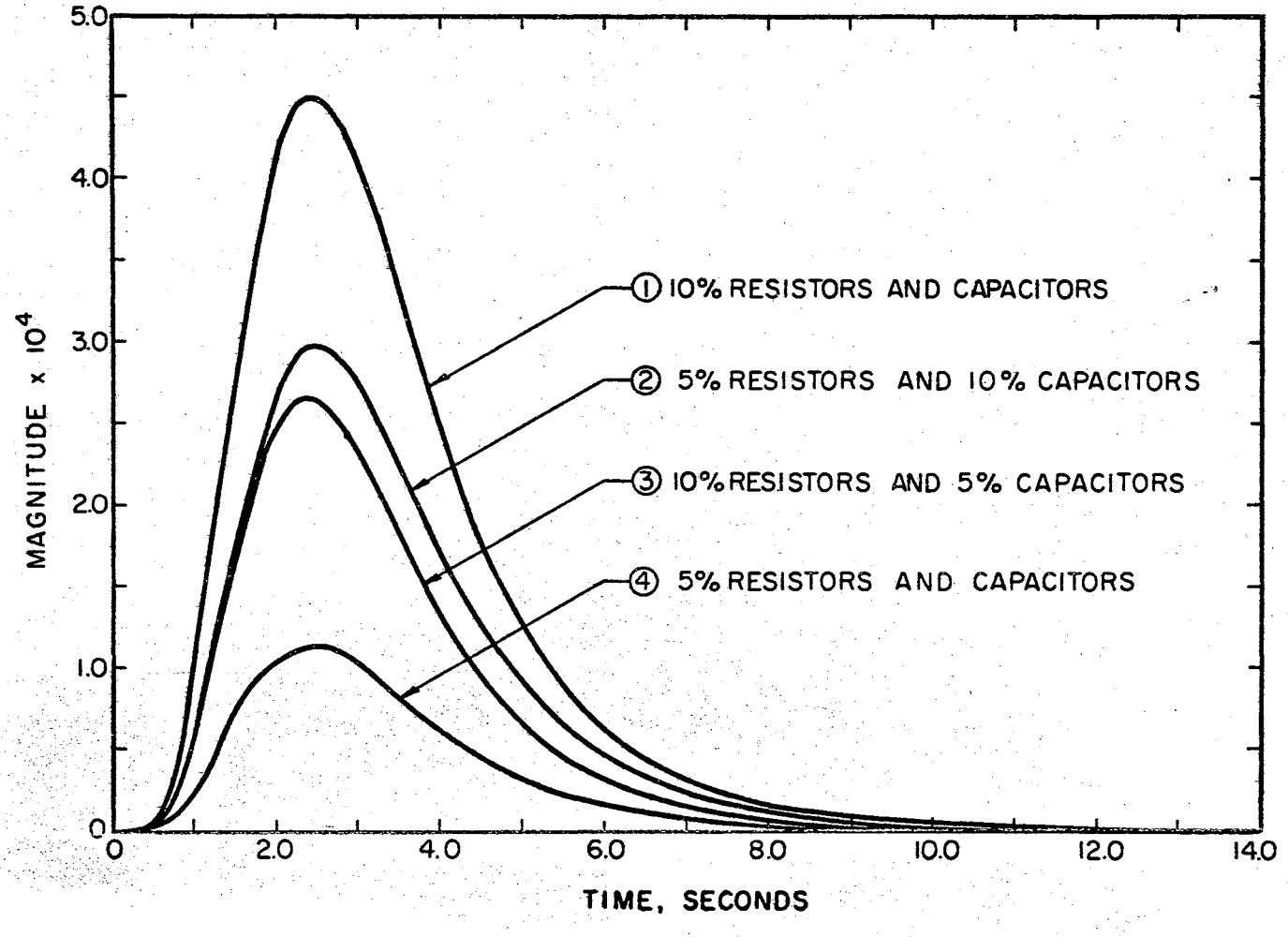


Figure 23. Variance of Output Voltage Due to Parameter Variations

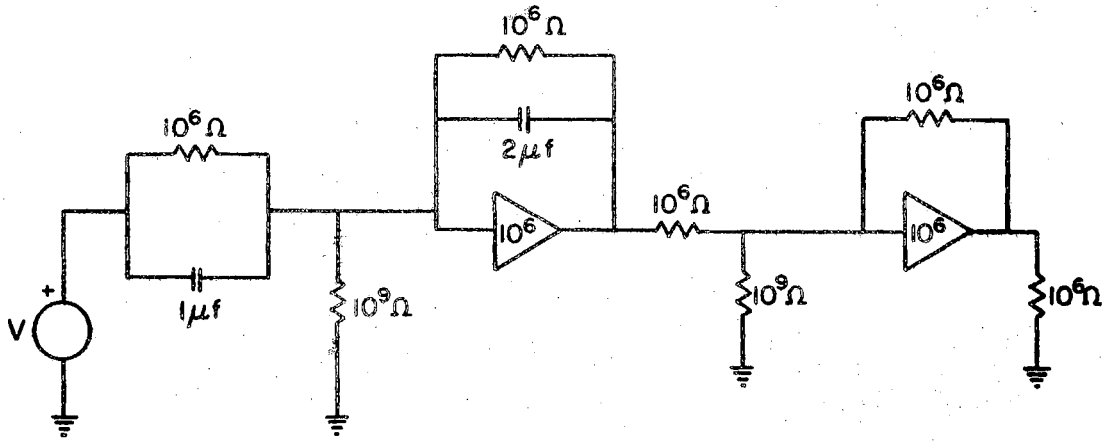


Figure 24. Compensation Network of Example 5.3.3

source was included. This problem was easily solved by including this source resistance. The smallest time constant for the network, and hence, the largest possible time increment for the integration, then became highly dependent on this choice of source resistance.

5.3.4 Example

A full wave rectifier and half wave rectifier were used with the same L-section filter circuit shown in Figure 25. It was necessary to modify Walden's iteration procedure for the diode nonlinearity to achieve convergence in the nonlinear solution process. Instead of using Equation (E.1.16) on page 100 of Walden's dissertation (43) for the iterated current values i_{n+1}, it was found to be desirable to substitute

$$i_{n+1} = i_n - (v_{n+1} - v_n) \frac{\Delta i}{\Delta v}$$

thus avoiding the possibility of diverging from the desired solution. The significance of this change is easily apparent by reference to Walden's appendix.

5.3.5 Example

The transformer equivalent circuit with shunt capacity from a transmission line shown in Figure 26 was analyzed. A saturating iron core inductor of the type given in model b of Section 4.5 was included and very close agreement of predicted and actual output changes were observed.

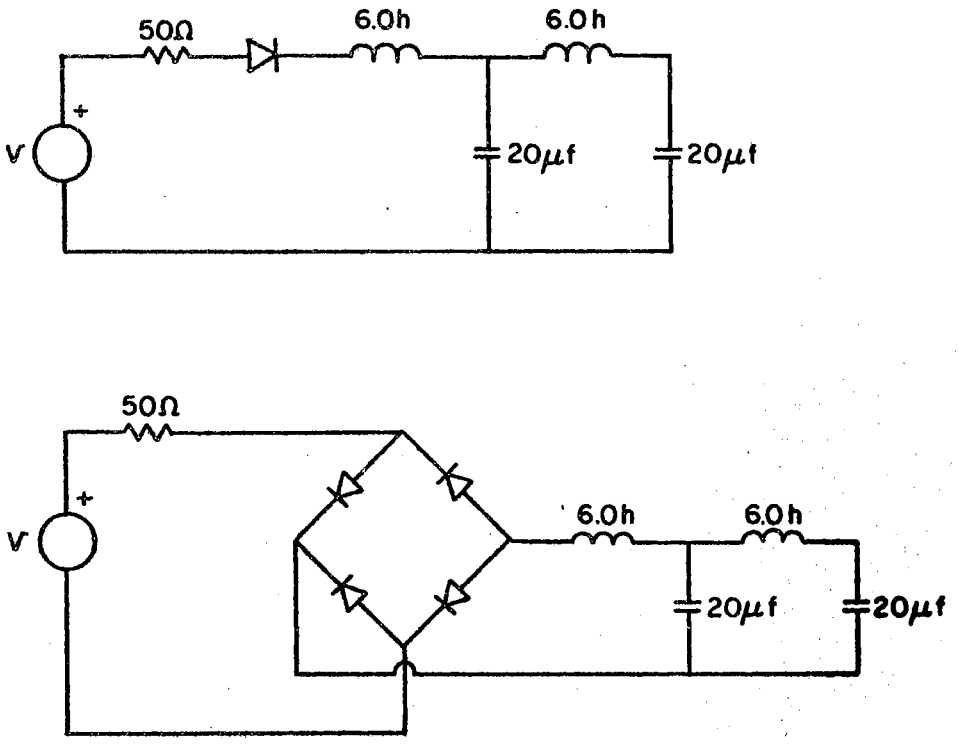


Figure 25. Rectifiers and L-Section Filters

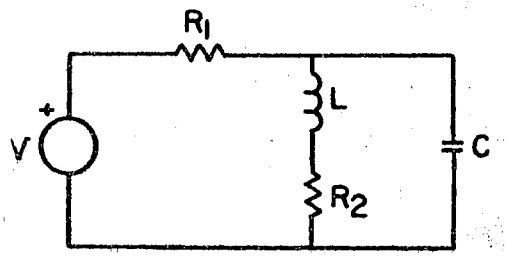


Figure 26. Circuit for
Example 5.3.5

5.3.6 Example

A mechanical system shown in Figure 27 was analyzed by utilizing the analogues of current and voltage. A third order nonlinear spring was included by making use of model a of Section 4.5. Close agreement between predicted and actual states occurred.

5.4 Conclusions

This chapter has demonstrated the application of the programs developed during this study to sensitivity analysis of both linear and nonlinear networks. The ease with with which a designer can achieve meaningful sensitivity measures for large classes of networks has been illustrated. These measures include classical pole-zero sensitivities and those based on the time domain models. The examples of Section 5.2 present sensitivity information in the form of the time domain mean square error criteria. This measure of sensitivity may be easily interpreted in terms of cost. Additional documentation and examples of input data describing the examples of this chapter are included in Appendices D and E.

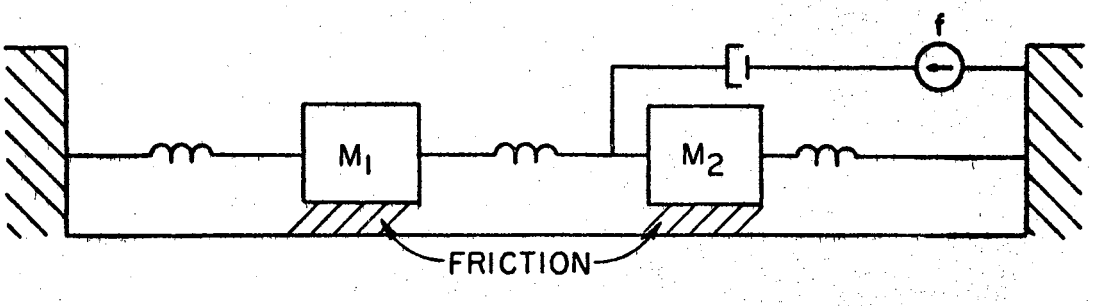


Figure 27. Circuit for Example 5.3.6

CHAPTER VI

SUMMARY AND CONCLUSIONS

6.1 Summary

This dissertation describes the research and development effort leading to new computer methodology for system sensitivity analysis. The motivation and primary objectives of this study are discussed in Chapter I. The general theory of system sensitivity performance analysis presented in Chapter II is applied to linear networks in Chapter III and nonlinear networks in Chapter IV. Examples of both linear and nonlinear network analysis by means of the design aid developed in this study are presented in Chapter V.

Chapter II suggests many different measures of sensitivity and methods of obtaining them. Deterministic and probabilistic measures based on classical transform methods and time domain methods are discussed. Multiple parameter variations are considered in a deterministic framework, but it seems much better to treat these variations in a probabilistic model.

One such probabilistic model discussed in Chapter II is a measure based on the expected mean square error from some desirable system function. If a simple approximation is made, the calculation of this measure requires statistical moments no higher than the second. This measure had previously been applied in the frequency domain but no consideration was given to the implementation of the required tedious

calculations. By introducing the sensitivity operator model the author has been successful in applying this mean square error measure to time domain system functions.

The elements of the sensitivity operator model provide all the information necessary for calculation of most of the frequency domain measures discussed in Chapter II. The unifying links between these time domain models and the frequency domain measures have been developed fully by the author.

Chapter III applies the theory of sensitivity presented in Chapter II to a large class of networks. A program, VARYIT, has been developed to illustrate the formulation procedure for the system state model and its sensitivity models for varying parameters, which are allowed to include varying resistors, capacitors, inductors, and current and voltage drivers. The problem of formulation of these models is discussed and a new theorem is given which allows a much more efficient allocation of storage in the computer. VARYIT has three options to provide either transient, impulse, or frequency domain sensitivity information.

As pointed out in Chapter II, the pole-zero sensitivities supplied by this program may be manipulated to provide most of the standard sensitivity measures. A new measure implemented in this program is the mean square error approximation in the time domain.

Chapter IV discusses the extension and application of the theory of Chapter II to certain classes of nonlinear networks. A nonlinear representation scheme based on dependent drivers and stepwise continuous storage elements is presented. Nine different types of nonlinearities are included in the program, VARNOL, developed in this

study to extend the capabilities of VARYIT to networks containing nonlinear elements.

Examples of the use of the new computer methodology for system sensitivity analysis are given in Chapter V. Output curves of the mean square error approximation are included in Section 5.2 for eight example networks all realizing the same input impedance. These curves were generated by the impulse and transient solutions options of VARYIT. Pole sensitivities are also presented for each of the example linear networks. Several examples of the use of the nonlinear network program VARNOL are included in Section 5.3 along with the solutions to their sensitivity operator models.

6.2 Conclusions

This study has accomplished the objectives set forth in Chapter I. A "user-oriented" computer design tool has been developed and implemented in FORTRAN language to supply accurate multiparameter sensitivity information. Many of the measures suggested in the past may be obtained from the program outputs by use of the links suggested in Chapter II. A very useful probabilistic measure, the mean square error, is calculated directly from an approximation making use of tolerancing data and the partial derivatives of the system variables.

Specific new contributions made during the development of this design tool are:

(a) development and implementation of an automatic formulation routine for obtaining the state and sensitivity operator models;

(b) development of a method of solving the sensitivity operator

model for linear systems with impulse inputs that insures a desired number of places of numerical accuracy between time steps in the integration process (Appendix A);

(c) statement and proof of a theorem regarding the order of the system parameters in the state-space model which allows a much more efficient utilization of computer memory (Appendix C);

(d) extension of Breipohl's frequency domain measure of sensitivity to the time domain with a partially automated method for obtaining Breipohl's measure in the frequency domain; and

(e) implementation in a design tool of a computational algorithm for obtaining the transfer function matrix and its associated pole-zero sensitivity information.

It should be emphasized that the primary objective in the research effort was to provide the practicing engineer with a design tool which he may use without acquiring a detailed knowledge of all aspects of sensitivity theory. He then is able to devote more of his time to the creative aspects of engineering problems.

6.3 Suggestions for Further Study

There are a number of areas in which further research would be fruitful and desirable. Several areas also exist in which the capabilities of this design tool may be coupled with previously reported work to provide a more comprehensive capability for network analysis and design.

It would be desirable to include networks belonging to a class larger than that implied by the model of Equation (3.2.1). The restriction on the existence of a tree containing all capacitors and voltage sources can be dropped if an individual is willing to build additional complexity into the formulation algorithms. Additional capability could also be achieved if models of multi-terminal components such as transformers and transducers were treated directly.

The mean square error in the time domain is a useful and meaningful concept but many designers are more accustomed to thinking in terms of the frequency domain. VARYIT presently provides the information necessary for computation of the mean square error in the frequency domain but does not include this measure as one of its output options. The technique for obtaining this measure is illustrated in Section 2.5 and could easily be incorporated into this program. Similarly the Bode-Mason sensitivity of Equation (2.2.2) may be computed by Ur's formula, Equation (2.2.5).

The models contained in the nonlinear program certainly do not exhaust the possibilities for modeling studies. The formulation phases may be used in studies to extend the types of allowable models. New iteration processes will need to be developed if the restriction on allowable types of models is dropped. No effort has been directed to this goal, but the difficulties do not appear to be insurmountable.

The optimization of sensitivity measures with respect to parameter variation should prove to be a fertile field of endeavor. This study has concentrated on the analysis problem almost exclusively. However, the researcher should be warned that the development of large scale network analysis tools, such as those developed in this study, is a task

that will involve not man-months but man-years and should not be attempted unless sufficient economic resources are available.

A SELECTED BIBLIOGRAPHY

1. Kuo, F. F. "Network Analysis by Digital Computer." Proceedings of the IEEE, Vol. 54-6 (1966), 820-829.

2. Bode, H. W. Network Analysis and Feedback Amplifier Design. Princeton: D. Van Nostrand, 1945.

3. Mason, S. J. "Feedback Theory-Some Properties of Signal Flow Graphs." Proceedings of the IRE, Vol. 41 (1953), 1144-1156.

4. Papoulis, A. "Displacement of the Zeroes of the Impedance Z(p) Due to Incremental Variations in the Network Elements." Proceedings of the IRE, Vol. 43 (1955), 79-82.

5. Truxal, J. G., and I. M. Horowitz. "Sensitivity Considerations in Active Network Synthesis." Proceedings Second Midwest Symposium on Circuit Theory. Michigan State University (1956), 6-1 to 6-11.

6. Horowitz, I. M. "Optimization of Negative Impedance Conversion Methods for Active RC Synthesis." Transactions of the IRE, Vol. CT-6 (1959), 296-303.

7. Ur, H. "Root Locus Properties and Sensitivity Relations in Control Systems." Transactions of the IRE, Vol. AC-5 (1960), 57-65.

8. Calahan, D. A. "Sensitivity Minimization in Active RC Synthesis." Transactions of the IRE, CT-9 (1962), 38-42.

9. Huelsman, L. P. "Active RC Synthesis with Prescribed Sensitivity." Proceedings of the National Electronic Conference, Vol. 16 (1960), 412-426.

10. Goldstein, A. J., and F. F. Kuo. "Multiparameter Sensitivity." Transactions of the IRE, Vol. CT-8 (1961), 177-178.

11. Calahan, D. A. Modern Network Synthesis. New York: Hayden, 1964.

12. Lee, S. C. " On Multiparameter Sensitivity." Proceedings Third Annual Allerton Conference on Circuit and System Theory (1965), 407-420.

13. Hakimi, S. L., and J. B. Cruz, Jr. "Measure of Sensitivity for Linear Systems with Large Multiparameter Variations." IRE WESCON Convention Records, Vol. 4 Pt. 2 (1960), 109-115.

14. Huelsman, L. P. "Network Analysis of Network Sensitivities." Proceedings of the National Electronic Conference, Vol. 21 (1965), 1-5.

15. Clunies-Ross, C., and S. S. Husson. "Statistical Techniques in Circuit Optimization." Proceedings of the National Electronics Conference, Vol. 18 (1962), 325-334.

16. Breipohl, A. M., and D. L. Campbell. "A Method for Comparing the Sensitivity of Circuits." Conference Record of the Ninth Midwest Symposium on Circuit Theory (1966), 4-1 to 4-12.

17. Tomovic, R. Sensitivity Analysis of Dynamic Systems. New York: McGraw-Hill, 1963.

18. Dorf, R. C. "System Sensitivity in the Time Domain." Proceedings Third Annual Allerton Conference on Circuit and System Theory (1965), 46-55.

19. Siljak, D. D., and R. C. Dorf. "On the Minimization of Sensitivity in Optimal Control Systems." Proceedings Third Annual Allerton Conference on Circuit and System Theory (1965), 225-229.

20. Kokotovic, P., S. Bingulac, and J. Medanic. "Some Approaches to the Reduction of Control System Sensitivity." Proceedings Third Annual Allerton Conference on Circuit and System Theory (1965), 212-224.

21. Broome, P. W., and F. V. Young. "The Selection of Circuit Components for Optimum Circuit Reproducibility." Transactions of the IRE, Vol. CT-9 (1962), 18-23.

22. Breipohl, A. M., and L. L. Grigsby. Final Report on Probabilistic System Analysis. Contract 50-7841. Sandia Corporation. Stillwater, Oklahoma: Oklahoma State University, 1966.

23. Morgan, B. S. "Computational Procedures for Sensitivity Coefficients in the Time Invariant Multivariable Systems." Proceedings of Third Annual Allerton Conference on Circuit and System Theory (1965), 252-258.

24. Faddeev, D. K., and V. N. Faddeeva. Computational Methods of Linear Algebra. San Francisco: W. H. Freeman, 1963.

25. Gantmacher, F. R. The Theory of Matrices, Vol. I. New York: Chelsea, 1959.

26. Frame, J. S. "A Simple Recursion Formula for Inverting a Matrix." (abstract) Bulletin of the American Mathematical Society, Vol. 55 (1949), 1045.

27. Tou, J. T. Modern Control Theory. New York: McGraw-Hill, 1964.

28. DeRusso, P. M., R. J. Roy, and C. M. Close. State Variables for Engineers. New York: John Wiley, 1965.

29. Dertouzous, M. L. "An Introduction to On-Line Circuit Design." Proceedings of the IEEE, Vol. 55-11 (1967), 1961-1971.

30. Dawson, D. F., F. F. Kuo, and W. G. Magnuson. "Computer-Aided Design of Electronic Circuits - - A Users Viewpoint." Proceedings of the IEEE, Vol. 55-11 (1967), 1946-1953.

31. Meissner, C. W., Jr. "An Annotated Bibliography of Computer Aided Circuit Analysis and Design." National Aeronautics and Space Administration, Langley Research Center NASA - 7023 (1968).

32. Leeds, J. V., G. L. Barksdale, G. Grueneich, and C. M. Moore. "Computer Analysis and Design Networks." Proceedings of the IEEE, Vol. 56-1 (1967).

33. Deckert, K. L., and E. T. Johnson. "User's Guide for LISA." IBM Technical Report TR 02.409 (1967).

34. Deckert, K. L., and E. T. Johnson. "LISA 360, A Program for Linear Systems Analysis." IBM Contributed Program Library 360D-16.4.009 (1968).

35. Pridgen, J. H. "Computer-Aided Network Sensitivity Analysis." A paper presented at Southwestern IEEE Conference in Dallas (1967).

36. Happ, W. W. "NASAP: Present Capabilities of a Maintained Program." IEEE Convention Record 1967, Vol. 15, Part 5, 64-88.

37. Murray-Lasso, M. A. "Inter-University Developed Programs." IEEE Transactions on Education, Vol. E-12, No. 4 (1969), 225-227.

38. Zobrist, G. W. "Signal Flow Graphs as an Aid in Network Analysis." IEEE Transactions on Education, Vol. E-12, No. 4 (1969), 235-242.

39. Koenig, H. E., and W. A. Blackwell. Electromechanical System Theory. New York: McGraw-Hill, 1961.

40. Koenig, H. E., I. Tokad, and M. Kesavan. Analysis of Discrete Physical Systems. McGraw-Hill: New York, 1967.

41. Seshu, S., and M. B. Reed. Linear Graphs and Electrical Networks. Reading, Massachusetts: Addison-Wesley, 1961.

42. Brown, D. P. "Derivative-Explicit Differential Equations for RLC Graphs." Journal of the Franklin Institute, Vol. 275 (1963), 503-514.

43. Walden, J. M. "Digital Computer Analysis of Systems with Non-Linear Elements." (unpub. Ph.D. dissertation, Oklahoma State University, 1966).

44. Faddeeva, V. N. Computational Methods of Linear Algebra. New York: Dover, 1959.

45. Cummins, R. L., and L. C. Thomason. "An Efficient Tree Listing Program." A paper presented at Southwestern IEEE Conference in Dallas, Texas (1964).

46. Falk, H. C. "n-Port Modeling of Large Linear Networks." (unpub. Ph.D. dissertation, Oklahoma State University, 1966).

47. Liou, M. L. "A Novel Method of Evaluating Transient Response." Proceedings of the IEEE, Vol. 54-1 (1966), 20-23.

48. Calahan, D. A. "Linear Network Analysis and Realization Digital Computer Programs." University of Illinois Engineering Experiment Station Bulletin 472 (1965).

49. Malmberg, A. G., and F. L. Cornwell. "NET-1 Network Analysis Program." Los Alamos Scientific Laboratory, University of California. Los Alamos, New Mexico, 1963.

50. Katzenelson, Jacob. "AEDNET: A Simulator for Nonlinear Networks." IEEE Proceedings, Vol. 54-11 (1966), 1536-1552.

51. Ralston, A., and H. S. Wilff. Mathematical Methods for Digital Computers. New York: John Wiley, 1960.

52. Weinberg, Louis. Network Analysis and Synthesis. New York: McGraw-Hill, 1962.

53. Smith, M. L. "Polynomial Root Extraction Program." SHARE PY POLY SDA 3332 (1965).

APPENDIX A

IMPULSE RESPONSE SOLUTION OF THE SYSTEM AND SENSITIVITY MODEL

The purpose of this appendix is to detail a new technique for obtaining the simultaneous impulse response solutions of the linear time-invariant state model and its sensitivity model. Consider the linear state model

$$\dot{\underline{x}} = \underline{A}\,\underline{x} + \underline{B}\,\underline{u} \tag{A.1}$$

$$\underline{y} = \underline{C}\,\underline{x} + \underline{D}\,\underline{u} \quad . \tag{A.2}$$

The solution to Equation (A.1) may be written as

$$\underline{x}(t) = e^{\underline{A}t}\,\underline{x}(0) + \int_0^t e^{\underline{A}(t-\lambda)}\,\underline{B}\,\underline{u}(\lambda)\,d\lambda \quad . \tag{A.3}$$

Let $\underline{x}(0) = \underline{0}$ and $\underline{\delta}_i(t)$ be an m x 1 vector each of whose elements is zero except for element i which is a unit impulse function, $\delta(t)$. For $\underline{u}(t) = \underline{\delta}_i(t)$, x(t) becomes

$$\underline{x}_i(t) = \text{i-th column of matrix } e^{\underline{A}t}\underline{B} \tag{A.4}$$

or

$$\underline{x}_i(t) = \underline{W}_i(t) \quad .$$

The solution to Equation (A.2) is then

$$\underline{y}_i(t) = \underline{C}\,\underline{W}_i(t) + \underline{D}\,\underline{\delta}_i(t) \quad .$$

Consider now that this process is repeated for i = 1, 2, ..., m and that the resulting solutions are arranged as follows:

$$\left[\underline{y}_1(t)\ \underline{y}_2(t)\ \cdots\ \underline{y}_m(t)\right] = \left[\underline{C}\,\underline{W}_1 + \underline{D}\,\underline{\delta}_1(t)\ \underline{C}\,\underline{W}_2 + \underline{D}\,\underline{\delta}_2(t)\ \cdots\ \underline{C}\,\underline{W}_m + \underline{D}\,\underline{\delta}_m(t)\right]$$

$$= \underline{C}\left[\underline{W}_1\ \underline{W}_2 \cdots \underline{W}_m\right] + \underline{D}\left[\underline{\delta}_1(t)\ \underline{\delta}_2(t) \cdots \underline{\delta}_m(t)\right]$$

$$= \underline{C}\, e^{\underline{A}t}\, \underline{B} + \underline{D}\, \delta(t) \quad . \tag{A.5}$$

By defining $\underline{H}(t) = \left[\underline{y}_1\ \underline{y}_2\ \cdots\ \underline{y}_m\right]$ Equation (A.5) then yields

$$\underline{H}(t) = \underline{C}\, e^{\underline{A}t}\, \underline{B} + \underline{D}\, \delta(t) \quad . \tag{A.6}$$

Thus the i-th column of $\underline{H}(t)$ is the output vector of the state model solution for a unit impulse at the i-th input.

Discussion will now be centered upon evaluating the changes, $\underline{H}'(t)$, in the impulse response matrix $\underline{H}(t)$ for a differential change in the value of the parameter p. These changes can be found by

$$\underline{H}'(t) = \frac{d\,\underline{C}}{dp}\, e^{\underline{A}t}\, \underline{B} + \underline{C}\, \frac{d\, e^{\underline{A}t}}{dp}\, \underline{B} + \underline{C}\, e^{\underline{A}t}\, \frac{d\underline{B}}{dp} + \frac{d\,\underline{D}}{dp}\, \delta(t) \quad . \tag{A.7}$$

Consider the augmented system of Equations (A.6) and (A.7)

$$\begin{bmatrix} \underline{H}(t) \\ \underline{H}'(t) \end{bmatrix} = \begin{bmatrix} \underline{C} & \underline{0} \\ \frac{d\underline{C}}{dp} & \underline{C} \end{bmatrix} \begin{bmatrix} e^{\underline{A}t} & 0 \\ \frac{de^{\underline{A}t}}{dp} & e^{\underline{A}t} \end{bmatrix} \begin{bmatrix} \underline{B} \\ \frac{d\underline{B}}{dp} \end{bmatrix} + \begin{bmatrix} \underline{D} \\ \frac{d\underline{D}}{dp} \end{bmatrix} \delta(t)$$

$$= \underline{P}\, \underline{\theta}(t)\, \underline{R} + \underline{M}\, \delta(t) \quad . \tag{A.8}$$

Note that the impulse response matrix and its derivative, with respect to p, may be calculated if the $\underline{\theta}(t)$ matrix can be found. The solution for the matrix $\underline{\theta}(t)$ may be calculated by noting that the transition

matrix $e^{\underline{\underline{A}}t}$ satisfies

$$\frac{d}{dt} e^{\underline{\underline{A}}t} = \underline{\underline{A}}\, e^{\underline{\underline{A}}t} \tag{A.9}$$

and, hence,

$$\frac{d}{dt}\frac{de^{\underline{\underline{A}}t}}{dp} = \underline{\underline{A}}\frac{de^{\underline{\underline{A}}t}}{dp} + \frac{d\underline{\underline{A}}}{dp} e^{\underline{\underline{A}}t} \quad . \tag{A.10}$$

Equations (A.9) and (A.10) may be written as

$$\frac{d}{dt}\begin{bmatrix} e^{\underline{\underline{A}}t} \\ \frac{de^{\underline{\underline{A}}t}}{dp} \end{bmatrix} = \begin{bmatrix} \underline{\underline{A}} & 0 \\ \frac{d\underline{\underline{A}}}{dp} & \underline{\underline{A}} \end{bmatrix}\begin{bmatrix} e^{\underline{\underline{A}}t} \\ \frac{de^{\underline{\underline{A}}t}}{dp} \end{bmatrix} \tag{A.11}$$

The solution of Equation (A.11) is

$$\begin{bmatrix} e^{\underline{\underline{A}}t} \\ \frac{de^{\underline{\underline{A}}t}}{dp} \end{bmatrix} = \exp\left\{\begin{bmatrix} \underline{\underline{A}} & \underline{\underline{0}} \\ \frac{d\underline{\underline{A}}}{dp} & \underline{\underline{A}} \end{bmatrix} t\right\}\begin{bmatrix} e^{\underline{\underline{A}}t} \Big|_{t=0} \\ \frac{de^{\underline{\underline{A}}t}}{dp}\Big|_{t=0} \end{bmatrix}$$

where

$$e^{\underline{\underline{A}}t}\Big|_{t=0} = \underline{\underline{I}}$$

and

$$\frac{de^{\underline{\underline{A}}t}}{dp}\Big|_{t=0} = \underline{\underline{0}} \quad .$$

Expansion of the term

$$e^{\begin{bmatrix} \underline{\underline{A}} & \underline{\underline{0}} \\ \frac{d\underline{\underline{A}}}{dp} & \underline{\underline{A}} \end{bmatrix} t}$$

by the series definition of

$$e^{\underline{\underline{A}}t} = \sum_{k=0}^{\infty} \frac{(\underline{\underline{A}}t)^k}{k!}$$

reveals that

$$\begin{bmatrix} e^{\underline{\underline{A}}t} & \underline{\underline{0}} \\ \frac{de^{\underline{\underline{A}}t}}{dp} & e^{\underline{\underline{A}}t} \end{bmatrix} = \exp \begin{bmatrix} \underline{\underline{A}} & \underline{\underline{0}} \\ \frac{d\underline{\underline{A}}}{dp} & \underline{\underline{A}} \end{bmatrix} t = \underline{\underline{\theta}}(t) \quad . \tag{A.12}$$

Thus Equation (A.12) may be inserted in Equation (A.8) to yield

$$\begin{bmatrix} \underline{H}(t) \\ \underline{H}'(t) \end{bmatrix} = \begin{bmatrix} \underline{\underline{C}} & \underline{\underline{0}} \\ \frac{d\underline{\underline{C}}}{dp} & \underline{\underline{C}} \end{bmatrix} \underline{\underline{\theta}}(t) \begin{bmatrix} \underline{\underline{B}} \\ \frac{d\underline{\underline{B}}}{dp} \end{bmatrix} + \begin{bmatrix} \underline{\underline{D}} \\ \frac{d\underline{\underline{D}}}{dp} \end{bmatrix} \delta(t) \tag{A.13}$$

where $\underline{\underline{\theta}}(t)$ is defined by Equation (A.12). The i-th column of the matrices $\underline{H}(t)$ and $\underline{H}'(t)$ of Equation (A.13) then provides a means of finding the output vector $\underline{y}(t)$ and $\frac{d\,\underline{y}(t)}{dp}$ when an impulse function is applied to the i-th input.

A computer program in FORTRAN IV language has been written which accepts the linear state model and its sensitivity operator equations as input. The output of this program is then the calculation of the $\underline{H}(t)$ and $\underline{H}'(t)$ matrices of Equation (A.13). The program does not yield the true values at t = 0 due to the absence of the impulse terms. However, at any time greater than zero the solution is correct. This program is written as a subroutine for incorporation into larger programs of the type discussed elsewhere in this dissertation. A flow chart of this process is given in Figure 28.

The arguments to be supplied the subroutine are as follows. A, B,

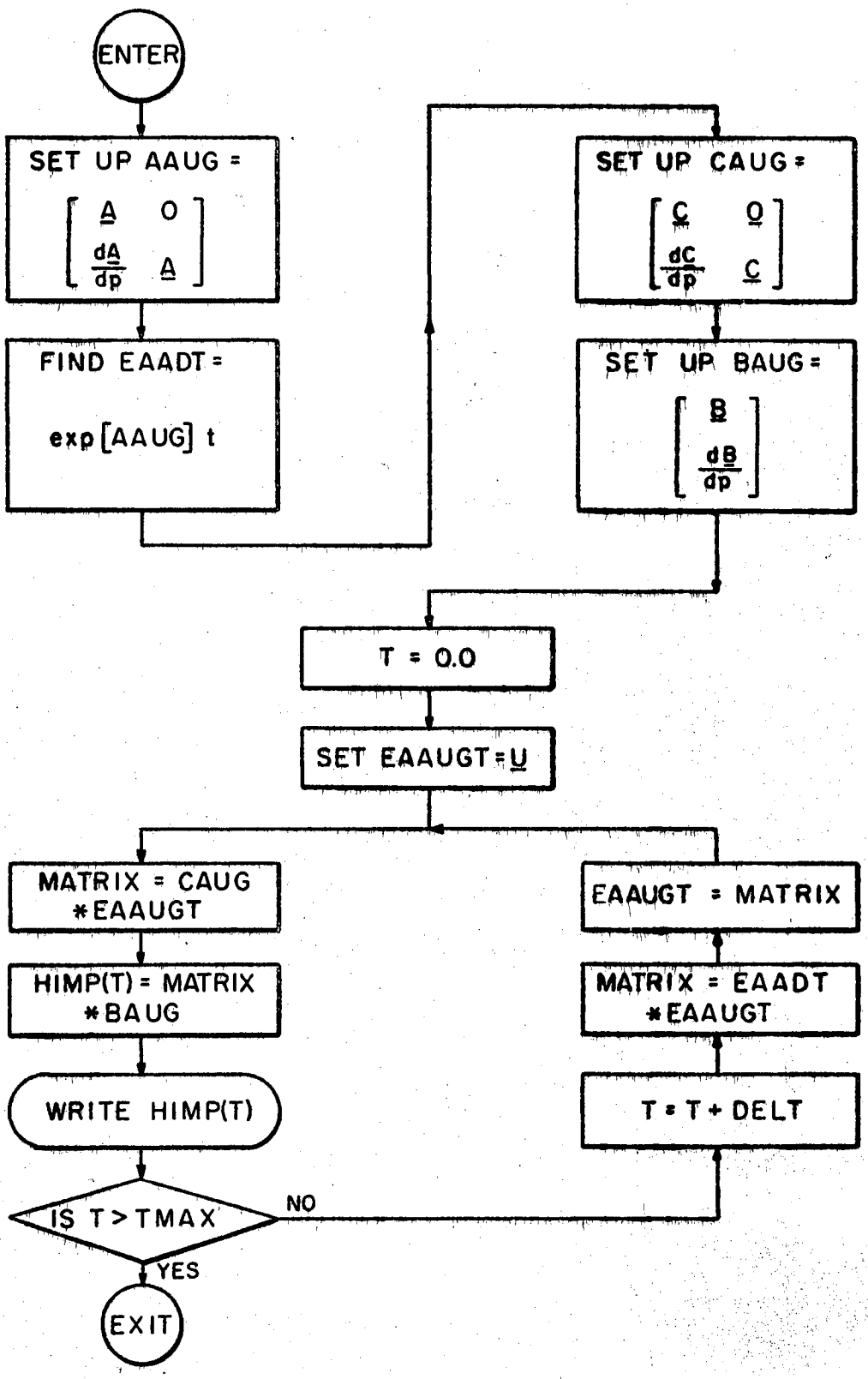


Figure 28. Flow Chart for Computation of Impulse Response Solution

and C are the matrices of the state model of Equations (A.1) and (A.2). APRIME, BPRIME, and CPRIME are the differential changes in the A, B, and C matrices. TMAX is the maximum time for which the solutions will be evaluated. PREC is the number of significant places for which the matrix exponential solution $\underline{\theta}(t)$ will be calculated. The other arguments are explained by COMMENT cards in the program listing.

TABLE V

PROGRAM LISTING FOR THE IMPULSE RESPONSE SOLUTION

```
C       THIS SUBROUTINE CALCULATES THE IMPULSE RESPONSE OF THE STATE MOD-
C       EL AND FINDS THE SENSITIVITY MODEL RESPONSE ALSO. THIS IS DONE
C       BY AUGMENTING. LIOU'S METHOD IS USED TO ASSURE ACCURACY.
C       A HAS DIMENSIONS II*II, B HAS DIMENSIONS II*KK, C HAS DIMENSIONS
C       JJ*II.
        SUBROUTINE IMPULS (A,B,C,APRIME,BPRIME,CPRIME,II,JJ,KK,TMAX,DELT,
     1                     PREC)
1       FORMAT (10X,5E15.8/(15X,5E15.8))
2       FORMAT (/20X,24HIMPULSE RESPONSE MATRIX  /20X,4HT = ,E15.8)
        REAL MATRIX
        DIMENSION A(10,10),B(10,10),C(10,10),APRIME(10,10),BPRIME(10,10),
     1            CPRIME(10,10),AAUG(20,20),BAUG(20,10),MATRIX(20,20),
     2            HIMP(20,20),EAADT(20,20),EAAUGT(20,20),CAUG(20,20)
        IA = II + 1
        IB = 2 * II
        IC = 2 * JJ
        T = 0.0
C       SET UP AAUG MATRIX BY AUGMENTING A MATRIX WITH APRIME MATRIX.
        DO 50 I = 1,II
        DO 50 J = 1,II
        IG = II + I
        IH = II + J
        AAUG(I,J) = A(I,J)
        AAUG(IG,IH) = A(I,J)
        AAUG(I,IH) = 0.0
50      AAUG(IG,J) = APRIME(I,J)
C       CALCULATE THE MATRIX EXPONENTIAL FOR AUGMENTED MATRIX.
        CALL MEXPON (AAUG,DELT,PREC,EAADT,IB)
C       MEXPON IS A MATRIX EXPONENTIAL ROUTINE USING LIOU'S TEST.
C       SET UP THE CAUG MATRIX.
        DO 70 I = 1,JJ
        DO 70 J = 1,KK
        IG = JJ + I
        IH = KK + J
        CAUG(I,J) = C(I,J)
        CAUG(I,IH) = 0.0
        CAUG(IG,J) = CPRIME (I,J)
70      CAUG (IG,IH) = C(I,J)
C       SET UP THE BAUG MATRIX.
        DO 80 I = 1,II
        DO 80 J = 1,KK
        IG = II + I
        BAUG(I,J) = B(I,J)
80      BAUG(IG,J) = BPRIME(I,J)
C       SET EAAUGT = U.
        DO 90 I = 1,IB
        DO 90 J = 1,IB
        EAAUGT(I,J) = 0.0
        IF (I.EQ.J) EAAUGT(I,J) = 1.0
90      CONTINUE
C       SET MATRIX = CAUG * EAAUGT.
100     DO 110 I = 1,IC
        DO 110 J = 1,IB
        MATRIX(I,J) = 0.0
        DO 110 K = 1,IB
110     MATRIX(I,J) = MATRIX(I,J) + CAUG(I,K) * EAAUGT(K,J)
        FIND THE IMPULSE RESPONSE MATRIX HIMP(T) = MATRIX * BAUG.
        DO 120 I = 1,IC
        DO 120 J = 1,KK
        HIMP(I,J) = 0.0
        DO 120 K = 1,IB
120     HIMP(I,J) = HIMP(I,J) + MATRIX(I,K) * BAUG(K,J)
```

TABLE V (Continued)

```
C       WRITE OUT THE IMPULSE RESPONSE MATRIX AND THE CHANGES IN THE IMP-
C       ULSE MATRIX WITH RESPECT TO THE PARAMETER.
        WRITE (6,2) T
        DO 140 I = 1,IC
140     WRITE (6,1) (HIMP(I,J),J=1,KK)
        IF (T.GE.TMAX) RETURN
        T = T + DELT
C       FORM MATRIX = EAADT * EAAUGT
        DO 150 I = 1,IB
        DO 150 J = 1,IB
        MATRIX(I,J) = 0.0
        DO 150 K = 1,IB
150     MATRIX(I,J) = MATRIX(I,J) + EAADT(I,K) * EAAUGT(K,J)
C       SET EAAUGT = MATRIX
        DO 160 I = 1,IB
        DO 160 J = 1,IB
160     EAAUGT(I,J) = MATRIX(I,J)
        GO TO 100
        END
```

APPENDIX B

PROGRAM FOR FINDING THE POLE-ZERO SENSITIVITIES

The purpose of this appendix is to describe a program which has been written to furnish pole-zero sensitivity information for the multivariable system described by a linear time-invariant state model. The relations used in this calculation are described in Section 2.5 and will not be described further here.

In the flow-chart for this program, Figure 29, the oval shaped symbols signify input or output steps and are explained by COMMENT or FORMAT statements in the listing. Contained within the ovals for data inputs are numbers which correspond to the "location in sequence" shown in Table VI which describes the sequencing of the input data. In an effort to make the program self-explanatory, COMMENT statements are inserted in the listing following Figure 29 in locations which correspond roughly to the inputs to blocks in the flow-chart. Additional comments have been included to aid the reader in interpreting the program.

The program listing is incomplete in that the subroutine that extracts the roots of the polynomial equations in s is not included. This subroutine, EXTRAC, is a library program for the extraction of roots of polynomials (53). The polynomial must be of degree greater than one and less than 100. Miller's method is used to iterate the

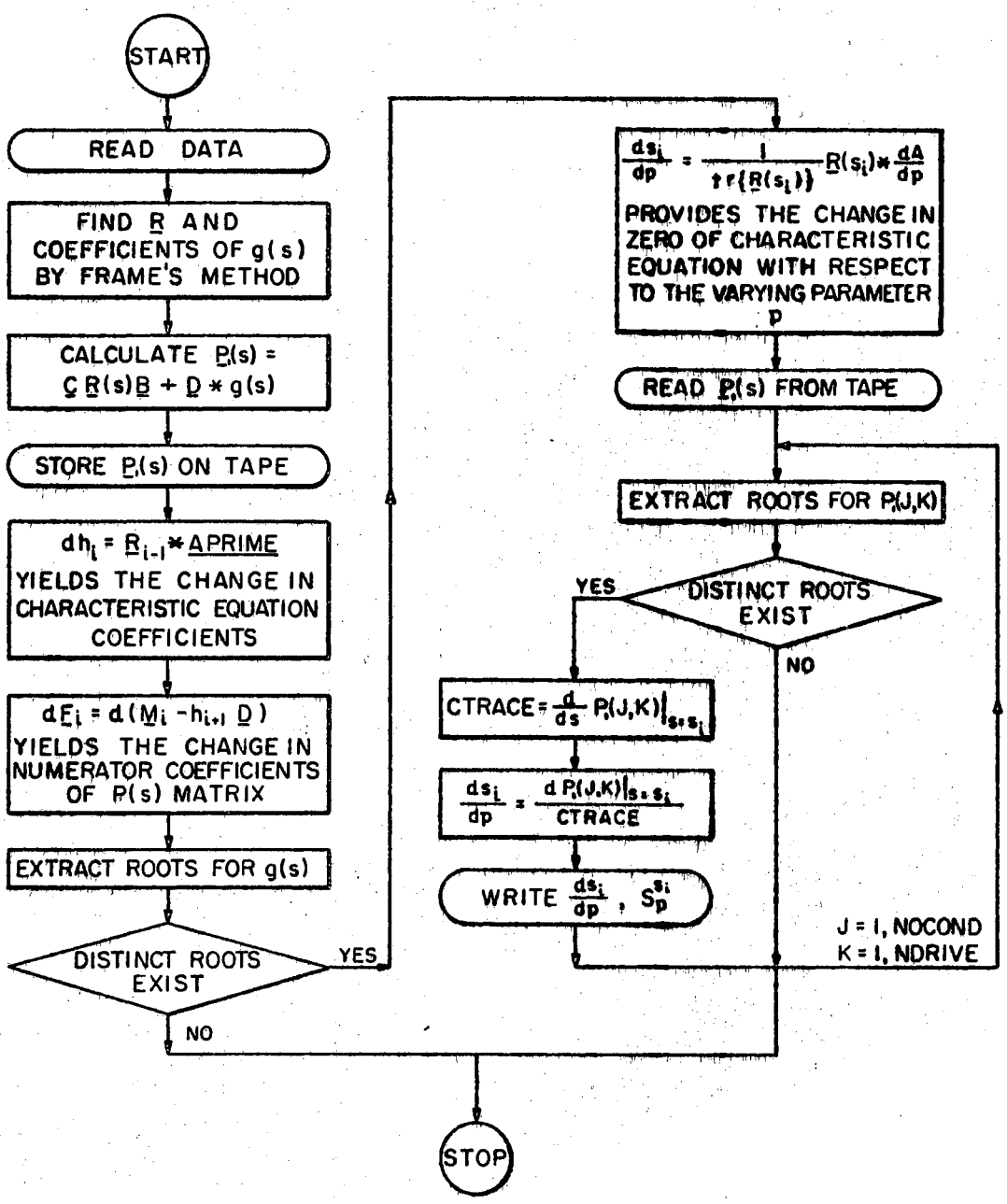


Figure 29. Flow Chart for Pole-Zero Sensitivity Program

TABLE VI

DESCRIPTION OF INPUT DATA FOR THE POLE-ZERO SENSITIVITY

Location in Sequence	Number of Cards	Description of Contents	Fortran Variables	Format
1	1	Dimension of $\underline{A}$ is IQ*IQ; Dimension of $\underline{B}$ is IQ*NDRIVE; Dimension of $\underline{C}$ is NOCOND*IQ	IQ, NOCOND, NDRIVE	10I5
2	IQ	Elements of $\underline{A}$ by row (one row per card)	A(I,J)	5E15.8
3	IQ	Elements of $\underline{B}$ by row (one row per card)	B(I,J)	5E15.8
4	NOCOND	Elements of $\underline{C}$ by row (one row per card)	C(I,J)	5E15.8
5	NOCOND	Elements of $\underline{D}$ by row (one row per card)	D(I,J)	5E15.8
6	IQ	Elements of differential $\underline{A}$ by row (one row per card)	APRIME (I,J)	5E15.8
7	IQ	Elements of differential $\underline{B}$ by row (one row per card)	BPRIME (I,J)	5E15.8
8	NOCOND	Elements of differential $\underline{C}$ by row (one row per card)	CPRIME (I,J)	5E15.8
9	NOCOND	Elements of differential $\underline{D}$ by row (one row per card)	DPRIME (I,J)	5E15.8
10	1	The nomial value of the parameter	PARAM	5E15.8

roots to an accuracy of 10^{-5}. In order to increase the convergence rate Newton's method is used to refine the roots to an accuracy of 2×10^{-8}. These accuracies are made possible by the use of double precision calculation at all critical states (53). The computation for extraction of roots with this program is extremely fast and is output limited in nature.

While EXTRAC performs its purpose admirably, the core storage requirement of 10,000 words consumes almost half of the memory available for computation on the IBM 7040 and limits the maximum number of state variables to 18, of drivers to 18, and of output variables to 18. These maximums may not be increased unless the pole-zero program is broken into several phases. This problem may be somewhat alleviated by including a root extractor program that is more economical from the standpoint of memory requirements.

In order to illustrate the use of this program an example problem is now considered. Let

$$\underline{A} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & -2 & -2 \\ 1 & 0 & 0 \end{bmatrix} \qquad \underline{B} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & -1 \end{bmatrix} \qquad \underline{C} = \begin{bmatrix} 3 & -1 & 2 \\ 2 & 1 & 1 \end{bmatrix}$$

$$\underline{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \underline{A}' = \begin{bmatrix} 0 & -0.1 & 0 \\ 0 & 0.2 & 0 \\ 0.1 & 0 & 0 \end{bmatrix} \qquad \underline{B}' = \begin{bmatrix} 0 & 0.1 \\ 0 & 0 \\ 0.3 & 0 \end{bmatrix}$$

$$\underline{C}' = \begin{bmatrix} 0 & -0.1 & 0 \\ 0.2 & 0 & 0 \end{bmatrix} \qquad \underline{D}' = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} .$$

Then

$$\underline{R}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \underline{R}_1 = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 0 & -2 \\ 1 & 0 & 2 \end{bmatrix} \qquad \underline{R}_2 = \begin{bmatrix} 0 & 0 & 2 \\ -2 & 0 & 0 \\ 2 & -1 & -1 \end{bmatrix}$$

and

$$h_1 = -2 \qquad h_2 = 1 \qquad h_3 = 2 \ .$$

Note

$$\underline{P}(s) = \frac{1}{s^3 + 2s^2 - s - 2} \begin{bmatrix} 7(s^2+3s+2) & 0 \\ 7(s^2+1) & 2(s^2+s-2) \end{bmatrix}$$

and

$$s_1 = 1 \qquad s_2 = -1 \qquad s_3 = -2$$

$$\text{tr } \underline{R}(s_1) = 6 \qquad \text{tr } \underline{R}(s_2) = -2 \qquad \text{tr } \underline{R}(s_3) = 3$$

yields

$$\frac{ds_1}{dp} = 0.1167 \qquad \frac{ds_2}{dp} = -0.25 \qquad \frac{ds_3}{dp} = 0,333$$

$$\frac{dh_1}{dp} = 0.20 \qquad \frac{dh_2}{dp} = 0.10 \qquad \frac{dh_3}{dh} = 0,40$$

$$\frac{d\underline{R}_1}{dp} = \begin{bmatrix} -0.2 & -0.1 & 0 \\ 0 & 0 & 0 \\ 0.1 & 0 & -0,2 \end{bmatrix} \qquad \frac{d\underline{R}_2}{dp} = \begin{bmatrix} 0 & 0 & 0.2 \\ -0.2 & 0 & 0 \\ 0 & -0,2 & -0.1 \end{bmatrix}$$

$$\frac{d\underline{M}_0}{dp} = \begin{bmatrix} 0.4 & 0.2 \\ 0.5 & 0.4 \end{bmatrix} \qquad \frac{d\underline{M}_1}{dp} = \begin{bmatrix} 0.3 & 0.5 \\ -1,3 & 0.3 \end{bmatrix} \qquad \frac{d\underline{M}_2}{dp} = \begin{bmatrix} 2.0 & 0.2 \\ 2.4 & -1.1 \end{bmatrix}$$

and for numerator element 1,1

$$S_p^{K_o} = 4.0 \qquad S_p^{-1} = -3.0 \qquad S_p^{-2} = 4.28j$$

for numerator element 2,1

$$S_p^{K_o} = 5.0 \qquad S_p^{i} = .928 + j\cdot 135 \qquad S_p^{-i} = .928 - j\cdot 135 \ ;$$

for numerator element 2,2

$$S_p^{K_o} = 4.0 \qquad S_p^{1} = .666 \qquad S_p^{-2} = - .166$$

TABLE VII

PROGRAM LISTING FOR THE POLE-ZERO SENSITIVITY PROGRAM

```
C       THIS PROGRAM COMPUTES THE TRANSFER FUNCTION MATRIX FOR A LINEAR
C       STATE MODEL. IT ALSO CALCULATES THE DIFFERENTIAL CHANGES IN THE
C       LOCATION OF THE POLES AND ZEROES DUE TO DIFFERENTIAL CHANGES IN
C       THE MATRICES A,B,C, AND D.
1       FORMAT (1H1,9HA MATRIX  )
2       FORMAT (5X,5E15.8/(10X,5E15.8))
3       FORMAT (5E15.8)
4       FORMAT (10I5)
5       FORMAT (1H0,14X,2HH(,I3,2H)=,3X,E15.8)
6       FORMAT (/20X,18HR MATRIXES FOLLOW  /3X,3HNO.)
7       FORMAT (//20X,51HTRANSFER FUNCTION MATRIX COEFFICIENT OF S TO POW
     1ER   ,I3)
8       FORMAT (/10X,18HR EVALUATED AT SI  /)
9       FORMAT (1H0,5HROOT ,I3,1H=,E15.8,3H+J*,E15.8)
10      FORMAT (//10X,31HFIRST COEFFICIENT TOO SMALL FOR ,I3,1H,,I3)
11      FORMAT (//20X,51HCHANGES IN COEFFICIENTS OF CHARACTERISTIC EQUATI
     1ON.  /)
12      FORMAT (30X,3HDH(,I3,2H)=,E15.8)
13      FORMAT (    I4,5X,2E15.8,5X,2E15.8,5X,2E15.8)
14      FORMAT (/1H0,3HNO.,8X,9HROOT REAL,6X,10HROOT IMAG.,7X,32HREAL AND
     1 IMAGINARY PARTS OF DSI  ,10X,14H   SENSITIVITY  )
15      FORMAT (1H0,18HNUMERATOR ELEMENT  ,2I3//)
16      FORMAT (10X,29HNUMERATOR IN DESCENDING ORDER /5(5X,5E15.8))
17      FORMAT (1H1,5X)
18      FORMAT (5X,8E15.8/(10X,8E15.8))
19      FORMAT (/10X,16HRS/TRACE OF RS  /)
20      FORMAT (//1H ,9HB MATRIX  )
21      FORMAT (/10X,30HTHE SYSTEM HAS MULTIPLE ROOTS   )
22      FORMAT (//1H ,9HC MATRIX  )
23      FORMAT (//1H ,9HD MATRIX  )
24      FORMAT (1H0,65HCHANGES IN NUMERATOR POLYNOMIAL COEFFICIENTS IN DE
     1SCENDING ORDER   /1X,11HS TO POWER  /)
25      FORMAT (//1H ,14HAPRIME MATRIX )
26      FORMAT (//1H ,14HBPRIME MATRIX )
27      FORMAT (//1H ,14HCPRIME MATRIX )
28      FORMAT (//1H ,14HDPRIME MATRIX )
29      FORMAT (/1H ,23HCONSTANT MULTIPLIER IS ,E15.8)
30      FORMAT (  1H ,38HSENSITIVITY OF CONSTANT MULTIPLIER IS ,E15.8)
        REAL MWK
        REAL MDCRB
        DIMENSION POLYNM(20),H(20),DH(20)
        DIMENSION A(17,17),B(17,17),C(17,17),D(17,17),APRIME(17,17),
     1  BPRIME(17,17),CPRIME(17,17),DPRIME(17,17),MWK(17,17),
     2  R(17,17,17),DRI(17,17),CRB(18,17,17),MDCRB(18,17,17)
        EQUIVALENCE (CRB(1,1,1),MDCRB(1,1,1))
C       READ THE INPUT MATRICES AS DATA.
        READ (5,4) IQ,NOCOND,NDRIVE
        REWIND 4
        WRITE (6,1)
        DO 31 I = 1,IQ
        READ (5,3) (A(I,J),J=1,IQ)
31      WRITE (6,2) (A(I,J),J=1,IQ)
        IQV = 1
        DO 32 I = 1,IQ
        DO 32 J = 1,IQ
32      MWK(I,J) = A(I,J)
        WRITE (6,20)
        DO 33  I = 1,IQ
        READ (5,3) (B(I,J),J=1,NDRIVE)
33      WRITE (6,2) (B(I,J),J=1,NDRIVE)
        WRITE (6,22)
        DO 34 I = 1,NOCOND
```

TABLE VII (Continued)

```
      READ (5,3) (C(I,J),J=1,IQ)
34    WRITE (6,2) (C(I,J),J=1,IQ)
      WRITE (6,23)
      DO 35 I = 1,NOCOND
      READ (5,3) (D(I,J),J=1,NDRIVE)
35    WRITE (6,2) (D(I,J),J=1,NDRIVE)
      WRITE (6,25)
      DO 36 I = 1,IQ
      READ (5,3) (APRIME(I,J),J=1,IQ)
36    WRITE (6,2) (APRIME (I,J),J=1,IQ)
      WRITE (6,26)
      DO 37 I = 1,IQ
      READ (5,3) (BPRIME(I,J),J=1,NDRIVE)
37    WRITE (6,2) (BPRIME(I,J),J=1,NDRIVE)
      WRITE (6,27)
      DO 38 I = 1,NOCOND
      READ (5,3) (CPRIME(I,J),J=1,IQ)
38    WRITE (6,2) (CPRIME(I,J),J=1,IQ)
      WRITE (6,28 )
      DO 39 I = 1,NOCOND
      READ (5,3) (DPRIME(I,J),J=1,NDRIVE)
39    WRITE (6,2) (DPRIME(I,J),J=1,NDRIVE)
      READ (5,3) PARAM
C     CALCULATE THE COEFFICIENTS OF THE CHARACTERISTIC EQUATION AND
C     THE R MATRICES BY FRAME'S METHOD.
40    TRACE = 0.0
      QV = IQV
      DO 41 I = 1,IQ
41    TRACE = TRACE + MWK(I,I)
      H(IQV)' = TRACE/QV
      DO 42 I = 1,IQ
42    MWK(I,I) = MWK(I,I) - H(IQV)
C     SET R(IQV,I,J) = MWK(I,J)
      DO 43 I = 1,IQ
      DO 43 J = 1,IQ
43    R(IQV,I,J) = MWK(I,J)
      IF (IQV.GE.IQ) GO TO 50
C     SET MWK = A * MR(IQV,I,J)
      DO 44 I = 1,IQ
      DO 44 J = 1,IQ
      MWK(I,J) = 0.0
      DO 44 K = 1,IQ
44    MWK(I,J) = MWK(I,J) + A(I,K) * R(IQV,K,J)
C     INCREMENT IQV
      IQV = IQV + 1
      GO TO 40
C     WRITE OUT COEFFICIENTS OF THE CHARACTERISTIC EQUATION.
50    DO 55 I = 1,IQ
55    WRITE (6,5) I,H(I)
C     WRITE OUT THE R MATRICES.
      WRITE (6,6)
      DO 60 IQV = 1,IQ
      WRITE (6,4) IQV
      DO 60 I = 1,IQ
60    WRITE (6,2) (R(IQV,I,J),J=1,IQ)
      DO 65 I = 1,IQ
65    H(I) = -H(I)
C     CALCULATE THE TRANSFER FUNCTION MATRIX CRB + D.
C     FIND CRB FOR HIGHEST ORDER S, IQ.
      DO 71 I = 1,NOCOND
      DO 71 J = 1,NDRIVE
      CRB(1,I,J) = D(I,J)
```

TABLE VII (Continued)

```
      CRB(2,I,J) = 0.0
      DO 70 K = 1,IQ
70    CRB(2,I,J) = CRB(2,I,J) + C(I,K) * B(K,J)
71    CRB(2,I,J) = CRB(2,I,J) + D(I,J) * H(1)
C     FORM R * B , STORE IN MWK.
      IA = IQ - 1
      DO 81 IQV = 1,IA
      IB = IQV + 2
      IC = IQV + 1
      DO 75 I = 1,IQ
      DO 75 J = 1,NDRIVE
      MWK(I,J) = 0.0
      DO 75 K = 1,IQ
75    MWK(I,J) = MWK(I,J) + R(IQV,I,K) * B(K,J)
C     FORM CRB = C * MWK FOR S TO IQ-IQV+1 POWER.
      DO 81 I = 1,NOCOND
      DO 81 J = 1,NDRIVE
      CRB(IB,I,J) = 0.0
      DO 80 K = 1,IQ
80    CRB(IB,I,J) = CRB(IB,I,J) + C(I,K) * MWK(K,J)
81    CRB(IB,I,J) = CRB(IB,I,J) + H(IC) * D(I,J)
C     NOTE THAT CRB HAS FIRST ELEMENTS CORRESPONDING TO S RAISED TO IQ.
C     NOTE THAT R(1,I,J) CORRESPONDS TO S TO IQ-2 POWER.
C     WRITE OUT TRANSFER FUNCTION MATRIX.
      WRITE (4) (((CRB(I,J,K),I=1,IB),J=1,NOCOND),K=1,NDRIVE)
      REWIND 4
      DO 85 I = 1,IB
      IA = IQ + 1 - I
      WRITE (6,7) IA
      DO 85 J = 1,NOCOND
85    WRITE (6,18) (CRB(I,J,K),K=1,NDRIVE)
C     FIND THE CHANGES IN COEFFICIENTS OF CHARACTERISTIC EQUATION BY
C     DH(I) = R(I-1) * APRIME.
      DO 90 I = 1,IQ
      DO 90 J = 1,IQ
      MWK(I,J) = 0.0
      IF (I.EQ.J) MWK(I,I) = 1.0
90    CONTINUE
      DO 91 I = 1,IQ
91    DH(I) = 0.0
      DO 92 I = 1,IQ
      DO 92 J = 1,IQ
92    DH(1) = DH(1) + MWK(I,J) * APRIME(J,I)
      IA = IQ - 1
      DO 93 IQV = 1,IA
      IB = IQV + 1
      DO 93 I = 1,IQ
      DO 93 J = 1,IQ
93    DH(IB) = DH(IB) + R(IQV,I,J) * APRIME(J,I)
      WRITE (6,11)
      DO 94 I = 1,IQ
      WRITE (6,12) I,DH(I)
94    DH(I) = - DH(I)
C     FIND CHANGES IN NUMERATOR COEFFICIENTS.
C     SET MDCRB(1) = DPRIME, S TO POWER IQ
      DO 300 I = 1,IQ
      DO 300 J = 1,NDRIVE
300   MDCRB(1,I,J) = DPRIME(I,J)
C     SET UP TERM FOR S TO POWER IQ-1
      DO 302 I = 1,NOCOND
      DO 302 J = 1,NDRIVE
      MDCRB(2,I,J) = 0.0
```

TABLE VII (Continued)

```
      DO 301 K = 1,IQ
301   MDCRB(2,I,J) = MDCRB(2,I,J) + CPRIME(I,K) * B(K,J)+ C(I,K) *
     1  BPRIME(K,J)
302   MDCRB(2,I,J) = MDCRB(2,I,J) + DH(I) * D(I,J) + H(I) * DPRIME(I,J)
C     SET MWK = APRIME
      DO 305 I = 1,IQ
      DO 305 J = 1,IQ
305   MWK(I,J) = APRIME(I,J)
      IA = IQ - 1
      DO 350 IQV = 1,IA
      IB = IQV + 2
      IC = IQV + 1
C     SET UP DRI = DAI - DH(I) * U
      DO 310 I = 1,IQ
      DO 310 J = 1,IQ
      DRI(I,J) = MWK(I,J)
      IF (I.EQ.J) DRI(I,I) = DRI(I,I) - DH(IQV)
310   CONTINUE
C     SET MWK = R(I) * B
      DO 311 I = 1,IQ
      DO 311 J = 1,NDRIVE
      MWK(I,J) = 0.0
      DO 311 K = 1,IQ
311   MWK(I,J) = MWK(I,J) + R(IQV,I,K) * B(K,J)
C     SET MDCRB(IB) = CPRIME * MWK
      DO 312 I = 1,NOCOND
      DO 312 J = 1,NDRIVE
      MDCRB(IB,I,J) = 0.0
      DO 312 K = 1,IQ
312   MDCRB(IB,I,J) = MDCRB(IB,I,J) + CPRIME(I,K)*MWK(K,J)
C     SET MWK = DRI * B
      DO 315 I = 1,IQ
      DO 315 J = 1,NDRIVE
      MWK(I,J) = 0.0
      DO 315 K = 1,IQ
315   MWK(I,J) = MWK(I,J) + DRI(I,K) * B(K,J)
C     SET MDCRB(IB) = C * MWK + MDCRB(IB)
      DO 320 I = 1,NOCOND
      DO 320 J = 1,NDRIVE
      DO 320 K = 1,IQ
320   MDCRB(IB,I,J) = MDCRB(IB,I,J) + C(I,K) * MWK(K,J)
C     SET MWK = C * R(I)
      DO 325 I = 1,NOCOND
      DO 325 J = 1,IQ
      MWK(I,J) = 0.0
      DO 325 K = 1,IQ
325   MWK(I,J) = MWK(I,J) + C(I,K) * R(IQV,K,J)
C     SET MDCRB(IB) = MWK * BPRIME + MDCRB(IB)
      DO 335 I = 1,NOCOND
      DO 335 J = 1,NDRIVE
      DO 330 K = 1,IQ
330   MDCRB(IB,I,J) = MDCRB(IB,I,J) + MWK(I,K) * BPRIME(K,J)
335   MDCRB(IB,I,J) = MDCRB(IB,I,J) + DH(IC)*D(I,J) + H(IC)*DPRIME(I,J)
C     SET UP NEW DAI MATRIX AND STORE IN MWK
      DO 340 I = 1,IQ
      DO 340 J = 1,IQ
      MWK(I,J) = 0.0
      DO 340 K = 1,IQ
340   MWK(I,J) = MWK(I,J) + APRIME(I,K)*R(IQV,K,J) + A(I,K)*DRI(K,J)
350   CONTINUE
      WRITE (6,24)
      DO 360 IQV = 1,IB
```

TABLE VII (Continued)

```
      IA = IQ - IQV + 1
      WRITE (6,4) IA
      DO 360 I = 1,NOCOND
360   WRITE(6,18) (MDCRB(IQV,I,J),J = 1,NDRIVE)
C     THIS PART OF PROGRAM FINDS THE DSI  ALONG WITH THE ROOTS
      COMPLEX RS,SN,SR,SI,ROOTS,CTRACE,DSI,SENS
      DIMENSION RS(17,17),ROOTI(20),ROOTR(20),DSI(20)
      WRITE (6,17)
      CALL EXTRAC (H,IQ,ROOTR,ROOTI)
      WRITE (6,17)
C     TEST FOR MULTIPLE ROOTS.
      IG = IQ - 1
      DO 95 IQV = 1,IG
      IC = IQV + 1
      DO 95 I = IC,IQ
      IF (ROOTR(IQV).EQ.ROOTR(I).AND.ROOTI(IQV).EQ.ROOTI(I)) GO TO 96
95    CONTINUE
      GO TO 99
96    WRITE (6,21)
      CALL EXIT
C     EVALUATE R MATRIX AT ROOT IQV.
99    DO 200 IQV = 1,IQ
      SR = (1.0,0.0) * ROOTR(IQV)
      SI = (0.0,1.0)*ROOTI(IQV)
      ROOTS = SR + SI
      WRITE (6,9) IQV,ROOTS
      SN = ROOTS
      IA = IQ - 1
C     SET UP CONSTANT TERM OF RS MATRIX.
      DO 100 I = 1,IQ
      DO 100 J = 1,IQ
100   RS(I,J) = (1.0,0.0)*R(IA,I,J)
      K = IA
110   K = K - 1
      DO 120 I = 1,IQ
      DO 120 J = 1,IQ
120   RS(I,J) = R(K,I,J) * SN + RS(I,J)
      SN = SN * ROOTS
      IF (K.GT.1) GO TO 110
      DO 140 I = 1,IQ
140   RS(I,I) = RS(I,I) + SN
      WRITE (6,8)
      DO 145 I = 1,IQ
145   WRITE (6,18)(RS(I,J),J=1,IQ)
C     FORM TRACE.
      CTRACE = (0.0,0.0)
      DO 150 I = 1,IQ
150   CTRACE = RS(I,I) + CTRACE
      DO 160 I = 1,IQ
      DO 160 J = 1,IQ
160   RS(I,J) = RS(I,J)/CTRACE
      WRITE (6,19)
      DO 165 I = 1,IQ
165   WRITE (6,18) (RS(I,J),J=1,IQ)
C     FORM CROSS PRODUCT
      DSI(IQV) = (0.0,0.0)
      DO 170 I = 1,IQ
      DO 170 J = 1,IQ
170   DSI(IQV) = RS(I,J) * APRIME(J,I) + DSI(IQV)
200   CONTINUE
      WRITE (6,14)
      DO 220 IQV = 1,IQ
```

TABLE VII (Continued)

```
      SENS = PARAM * DSI(IQV)
220   WRITE (6,13) IQV,ROOTR(IQV),ROOTI(IQV),DSI(IQV),SENS
C     MAKE USE OF R TO STORE THE TRANSFER FUNCTION MATRIX ITSELF
C     SINCE R IS NO LONGER NEEDED HERE.
      IG = IQ + 1
      READ (4) (((R(I,J,K),I=1,IG),J=1,NOCOND),K=1,NDRIVE)
C     FIND THE NUMERATOR ZEROES AND THEIR CHANGES WITH RESPECT TO X.
      DO 600 J = 1,NOCOND
      DO 600 K = 1,NDRIVE
      WRITE (6,15 ) J,K
C     SET UP COEFFICIENT ARRAY POLYNM FOR THE J,K ELEMENT.
      DO 500 I = 1,IG
500   POLYNM(I) = R(I,J,K)
      IC = IQ
501   IF (POLYNM(1).NE.0.0) GO TO 503
      IC = IC - 1
      IF (IC.EQ.0) GO TO 600
      IA = IC + 1
      DO 502 I = 1,IA
502   POLYNM(I) = POLYNM(I+1)
      GO TO 501
C     AT THIS TIME IC IS DEGREEE OF NUMERATOR TERM.
503   IF (ABS(POLYNM(1)).LT.0.00000002) WRITE (6,10 ) J,K
      DO 510 I = 2,IA
510   POLYNM(I) = POLYNM(I)/POLYNM(1)
      POLSAV = POLYNM(1)
      POLYNM(1) = 1.0
      IH = IC + 1
      WRITE (6,16 ) (POLYNM(I),I=1,IH)
      IF(IC-1) 600,520,530
C     EXTRACT THE NUMERATOR ROOTS.
520   ROOTR(1) = -POLYNM(2)
      ROOTI(1) = 0.0
      GO TO 540
C     MAKE THIS POLYNM CORRESPOND TO THE H ARRAY PREVIOUSLY.
530   DO 535 I = 1,IC
535   POLYNM(I) = POLYNM(I+1)
      CALL EXTRAC (POLYNM,IC,ROOTR,ROOTI)
C     RESTORE POLYNM ARRAY.
      DO 536 I = 1,IC
      IA = IC + 2 - I
536   POLYNM(IA) = POLYNM(IA-1)
      POLYNM(1) = 1.0
C     TEST FOR MULTIPLE ROOTS
      IMP = IC - 1
      DO 537 IQV = 1,IMP
      IMP1 = IQV + 1
      DO 537 I = IMP1,IC
      IF (ROOTR(IQV).EQ.ROOTR(I).AND.ROOTI(IQV).EQ.ROOTI(I)) GO TO 538
537   CONTINUE
      GO TO 540
538   WRITE (6,21)
      GO TO 600
C     CALCULATE THE SENSITIVITY  OF  CONSTANT K MULTIPLIER.
540   IANY = IQ - IC + 1
      SENCON = PARAM * MDCRB(IANY,J,K)
      WRITE (6,29) POLSAV
      WRITE (6,30) SENCON
C     CALCULATE DERIVATIVE WITH RESPECT TO S,AND STORE IN
C     CTRACE= IC*M1*S**(IC-1) + (IC-1)*M2*S**(IC-2) + ... +MIC
      WRITE (6,14)
      DO 600 I = 1,IC
```

TABLE VII (Continued)

```
      CTRACE = (0.0,0.0)
      ROOTS = (1.0,0.0)*ROOTR(I) + (0.0,1.0)*ROOTI(I)
      SN = (1.0,0.0)
      DO 550 IA = 1,IC
      IB = IC - IA + 1
      QV = IA
      CTRACE = POLYNM(IB) * SN * QV       + CTRACE
550   SN = SN * ROOTS
C     FORM NUMERATOR FOR NEWTON'S FORMULA.
C     N = M1 *S**(IC) + M2*S**(IC-1) + M3*S**(IC-2) + ... + MIC+1.
C     DN = DM1*S**IC + DM2*S**(IC-1) + ... + DMIC+1.
      SN = (1.0,0.0)
      IB = IQ + 1
      DSI(I) = (0.0,0.0)
      DO 570 IA = 1,IB
      IQV = IQ - IA + 2
      DSI(I) = DSI(I) + MDCRB(IQV,J,K)*SN
570   SN = SN * ROOTS
C     APPLY NEWTON'S FORMULA.
      DSI(I) =-DSI(I)/(CTRACE*POLSAV)
      SENS = PARAM * DSI(I)
      WRITE (6,13) I , ROOTR(I),ROOTI(I),DSI(I),SENS
600   CONTINUE
      STOP
      END
```

APPENDIX C

A THEOREM ON THE DEGREE OF PARAMETERS IN THE STATE MODEL MATRICES

This appendix develops a theorem of particular importance in this study for decreasing computer storage requirements. One of the major objectives of this study was to develop a design tool with automatic formulation and solution capabilities for both the state and sensitivity operator models. The state model formulation procedure of Section 3.3 was selected for implementation. A polynomial representation of each state model matrix was desired to allow parameter values to be changed without reformulation of the basic models. When polynomial representations are used the maximum degree of the polynomials to be encountered becomes very important since it determines the storage required. The following theorem is developed in this appendix.

Theorem: Consider an electric network of resistors, inductors, capacitors and voltage and current source elements. If there exists a tree containing all capacitors and voltage sources and excluding all inductors and current sources, then the state model of Equation (3.3.1) is such that no element of the $\underline{A}$, $\underline{B}$, $\underline{C}$, or $\underline{D}$ matrices has numerator degree higher than three in any parameter or has denominator degree higher than one in any parameter.

Proof: Let an electric network N consist of resistors, inductors, capacitors, voltage sources, and current sources such that some tree T exists which contains all voltage drivers and capacitors and excludes all current drivers and inductors. The tree T is a connected subgraph of the connected network graph which contains all vertices of the graph but does not contain any circuits. The tree T will be symbolized

$$T = (e_1, e_2, \ldots, e_n)$$

or

$$T = (S_C, S_E, S_{TG})$$

where

S_C is the set of elements corresponding to capacitors,

S_E is the set of elements corresponding to voltage drivers, and

S_{TG} is the set of elements corresponding to tree conductances.

The elements not belonging to the tree T belong to the cotree CT symbolized

$$CT = (S_L, S_J, S_{CG})$$

where

S_L is the set of elements corresponding to inductors,

S_J is the set of elements corresponding to current drivers, and

S_{CG} is the set of elements corresponding to cotree conductances.

The matrix state model formulation procedure given in Figure 8 may be developed by manipulation of the cutset, circuit and component equations written using the tree T. Additional notation employed in this appendix is given in Figure 8. If it can be shown that the

elements of the matrix

$$\underline{R} = \left[\underline{G}_T + \underline{S}_{33}\,\underline{G}_C\,\underline{S}_{33}^T\right]^{-1}$$

can be expressed as

$$r_{ij} = \frac{a_1 p + a_2}{a_3 p + a_4} \qquad (C.1)$$

where a_1, a_2, a_3, and a_4 are independent of any single conductance parameter p, then the theorem will follow directly from the formulation rules.

Consider the matrix

$$\underline{G} = \underline{R}^{-1} = \underline{G}_T + \underline{S}_{33}\,\underline{G}_C\,\underline{S}_{33}^T \quad . \qquad (C.2)$$

It will be shown first that $|\underline{G}|$ is a function of no more than degree one in the conductance variables of the network. Secondly, the cofactors Δ_{ij} of the i, j elements of $\underline{G}$ will be shown to be likewise functions of no more than degree one in the conductance variables. Since

$$\underline{R} = \underline{G}^{-1} = \frac{\text{Adj } \underline{G}}{|\underline{G}|}$$

these two statements will imply

$$r_{ij} = \frac{a_1 p + a_2}{a_3 p + a_4}$$

is a true representation of each element.

Consider now the network N' formed from network N by removing the current sources and inductors and shorting the terminals of the voltage sources and capacitors before removing them. Removing the chords consisting of inductors and current sources does not change the tree

of the resulting network in any manner (44, page 16). However, the process of shorting and removing capacitors and voltage sources does modify the tree since the number of vertices is decreased by one for each element shorted and removed.

The reduction process for capacitors and voltage sources cannot introduce a circuit of tree resistors belonging to T in the reduced network N' as shown by the following reasoning. If a circuit of tree resistors belonging to T occurred in the network N', then in N a circuit consisted of capacitors and/or voltage sources and tree resistors. However, since the capacitors and voltage sources belong to T, there must have been a circuit of tree elements in N. This contradicts the definition of the tree T. Therefore, all tree resistors in N remain in the tree of N'.

In the process of shorting and removing capacitors and voltage sources, self-loop conductance elements may be introduced in network N'. Resistors contained in a circuit made up of one resistor and capacitors and/or voltage sources appear in the reduced network as self-loop elements. Since the capacitors and voltage sources belong to the tree T, these self-loop elements in the reduced network must belong to the cotree of the original network N. Also, these elements form circuits in N' and, hence, they belong to any cotree of N'.

Assume first that no self-loop conductances occur in N'. The subgraph of N' corresponding to the tree conductance elements of N is a valid tree of N' since it is connected, contains all the vertices, and does not contain any circuits. This tree T' contains every tree conductance of N and the cotree CT' contains every cotree conductance of N. The fundamental cutset equations formulated in terms of the

tree T' are

$$\begin{bmatrix} \underline{A} \end{bmatrix} \begin{bmatrix} \underline{I}_{TG} \\ \underline{I}_{CG} \end{bmatrix} = \underline{O}$$

where

$$\underline{A} = \begin{bmatrix} \underline{I} & \underline{S}_{33} \end{bmatrix}$$

and the fundamental circuit equations are

$$\begin{bmatrix} -\underline{S}_{3}^{T} & \underline{I} \end{bmatrix} \begin{bmatrix} \underline{V}_{TG} \\ \underline{V}_{CG} \end{bmatrix} = \underline{O} \quad .$$

The matrices $\underline{S}_{33}$ and $\underline{S}_{33}^{T}$ are identical to the corresponding submatrices found, respectively, in the cutset and circuit equations of Figure 8. This can be seen by setting to zero the variables $\underline{J}_L$, $\underline{J}_D$, $\underline{E}_C$, $\underline{E}_D$ in these equations which is equivalent to the reduction process yielding N'. It should be clear that the conductance matrices $\underline{G}_T$ and $\underline{G}_C$ for the reduced network N' are identical to those for N.

The node-admittance matrix of N' can be written as

$$\begin{aligned} \underline{Y} &= \begin{bmatrix} \underline{I} & \underline{S}_{33} \end{bmatrix} \begin{bmatrix} \underline{G}_T & \underline{O} \\ \underline{O} & \underline{G}_C \end{bmatrix} \begin{bmatrix} \underline{I} \\ \underline{S}_{33}^{T} \end{bmatrix} \\ &= \begin{bmatrix} \underline{A} \end{bmatrix} \begin{bmatrix} \underline{G}_T & O \\ O & \underline{G}_C \end{bmatrix} \begin{bmatrix} \underline{A}^{T} \end{bmatrix} \quad . \end{aligned} \tag{C.3}$$

But upon comparing this matrix with Equation (D.2) it follows that $\underline{G} = \underline{Y}$. Thus, $\underline{G}$ is the node-admittance matrix of the network N'.

Seshu and Reed (41, page 157) state that for a passive network containing no mutual inductances and no self-loop elements the node

admittance determinant is a linear function of any parameter admittance. Thus, for the case where no self-loops exist in N', it has been shown that $|\underline{G}|$ is a function of no more than degree one in any conductance variable of the network.

Consider now the cofactors Δ_{ij} of the i,j-th element of the node-admittance matrix $\underline{G}$. Seshu and Reed (41, page 161) also show the linear-in-any-one-parameter property of Δ_{ij}. Thus, the cofactors of the node-admittance matrix $\underline{G}$ can have degree no greater than one in any parameter when no self-loops exist.

Return now to the case in which self-loops exist in the reduced network N'. These self-loops were generated by circuits consisting of one cotree resistor and capacitors and/or voltage sources in the original network N. Note that each row of the fundamental circuit equations of Figure 8 corresponds to a circuit including only one cotree element and tree elements. No tree resistors exist in the circuits corresponding to self-loop elements. Thus, in the fundamental circuit equation submatrix $-\underline{S}_{33}^{T}$ there exists a row of zeros corresponding to the location of each self-loop element in the reduced network. Since a row of zeros exists in $-\underline{S}_{33}^{T}$ and a column of zeros in $\underline{S}_{33}$ the product

$$\underline{S}_{33}\ \underline{G}_{C}\ \underline{S}_{33}^{T}$$

is independent of the self-loop conductance parameters. Also, since the self-loop elements belong to CT' the matrix $\underline{G}_{T}$ is independent of these self-loop parameters. Thus, the matrix $\underline{G}$ of Equation (C.2) is independent of the self-loop parameters and its determinant and cofactors are in fact of degree zero in the self-loop conductance parameters.

Consider now deleting all self-loop elements from N' to produce network N". Since these elements belong to the cotree CT', they may be removed without affecting the existence of the tree T' and T' is also a tree of network N". Removal of these self-loop elements may be performed by deleting the self-loop zero rows of $\underline{S}_{33}^{T}$ and the corresponding columns of $\underline{S}_{33}$ to form $\underline{S}''_{33}$. It is then apparent that the matrix $\underline{A}''$,

$$\underline{A}'' = \begin{bmatrix} \underline{I} & \underline{S}''_{33} \end{bmatrix}$$

is a fundamental cutset matrix of the resistive network with no self-loops. The matrix $\underline{G}$ of Equation (C.3) including self-loops may also be expressed as

$$\underline{G} = \begin{bmatrix} \underline{A}'' \end{bmatrix} \begin{bmatrix} \underline{G}_T & \underline{0} \\ \underline{0} & \underline{G}''_C \end{bmatrix} \begin{bmatrix} \underline{A}'' \end{bmatrix}^T \qquad (C.4)$$

where $\underline{G}''_C$ is the cotree parameter matrix from which all self-loop conductances have been deleted. The theorems of Seshu and Reed invoked earlier may be applied to Equation (C.4). Thus, when self-loops occur the determinant of $\underline{G}$ and its cofactors are functions of no more than degree one in any conductance parameter in the network.

Thus, each element of the $\underline{R}$ matrix of Figure 8 may be expressed as shown in Equation (C.1). Examination of the submatrix formulation rules of Figure 8 indicates that tree conductances will occur in the numerator of the state model matrices $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ with a maximum degree of one since the incidence submatrices and $\underline{G}_C$ are independent of tree conductances. However, cotree conductances may occur with a maximum degree of three in the numerator. The denominator is unchanged

by the formulation rules and, hence, the maximum denominator degree in any parameter is one.

An alternate proof of this theorem has been suggested by Dr. C. M. Bacon and Dr. Rao Yarlagadda. This proof uses

$$\underline{G} = \begin{bmatrix} \underline{G}_T & \underline{S}_{33} \underline{G}_C \end{bmatrix} \begin{bmatrix} \underline{I} \\ \underline{S}_{33}^T \end{bmatrix} = \underline{P}\,\underline{Q} \quad .$$

Since $\underline{G}_T$ and $\underline{G}_C$ are diagonal, the matrix $\underline{P}$ is such that any one resistor value is found in only one column. By the Binet-Cauchy Theorem for finding the determinant of the square product of non-square matrices (41, page 156) it follows directly that $|\underline{G}|$ is a degree one in any conductance parameter. The Binet-Cauchy Theorem may also be applied to the evaluation of the minor M_{ij} of the i,j-th element of $\underline{G}$ by noting that this matrix corresponds to deleting row i of $\underline{P}$ and column j of $\underline{Q}$. The remaining statements in the proof are identical to those above.

APPENDIX D

DETAILS OF PROGRAM VARYIT

This appendix presents in greater detail the operations described in Section 3.5. The presentation is tabular for ease of reference. A brief summary of the following figures and tables follows:

(1) Fortran Variable Names and Definitions

(2) Input Data Preparation Chart

(3) Input Data for Time Solution

(4) Input Data for Impulse Solution

(5) Input Data for Pole-Zero

(6) Time Dependent Source Parameter Definitions

(7) Output Tape Formats

A few comments should be made to clarify the following tables and figures. The FORTRAN variables defined in item (1) above are only a partial list of those used in the program itself. Dimensioned variables---program lists, arrays, vectors, matrices, and polynomial matrices--are indicated by NAME(I), NAME(I,J), and NAME(I,J,K). The sequence numbers of the Input Data Preparation Chart refer to the first column of item (3) above. Program decks are available from the Engineering Computing Laboratory at Oklahoma State University upon request. A separate addendum document containing flow charts and listings will be furnished with these decks.

TABLE VIII

FORTRAN VARIABLE NAMES AND DEFINITIONS FOR VARYIT

A(I,J) -- The matrix A defined by Equation 3.2.1a.

APRIME (I,J) -- The derivative of the matrix A with respect to the parameter p.

A61(I), A62(I), A63(I), A64(I) -- Parameters describing the sinusoidal drivers (Type 6) of current or voltage.

A71(I), A72(I), A73(I), A74(I), A75(I), A76(I) -- Parameters describing pulse drivers (Type 7) of current or voltage.

B(I,J) -- The matrix B of Equation 3.2.1a.

BOTNOD -- The second node located to which a tree element is incident.

BRANCH -- The tree element of the cutset.

BPRIME (I,J) -- The derivative of the matrix B with respect to the parameter p.

C(I,J) -- The matrix C of Equation 3.2.1b.

CLVALU (I) -- Vector containing the diagonal elements of the matrix CL of Equation 3.2.

COND (I) -- Vector containing the nominal conductance values for resistance elements.

CPRIME (I,J) -- The derivative of the matrix C with respect to the parameter p.

CRDSET (I) -- The list of elements belonging to the cotree.

CURVAR -- Program flag to indicate that an inductor or current driver varies.

D(I,J) -- The matrix D of Equation 3.2.1b.

DELTAT -- The integration increment for state model solution.

TABLE VIII (Continued)

DET (I) -- The denominator polynomial for A, B, C, and D, matrices.

DPRIME (I,J) -- The derivative of the matrix D with respect to the parameter p.

E(I) -- Driver vector evaluated at time T.

FLAG -- Program flag generated by Phase 1 to direct MAIN link of VARYIT to proper solution phase.

FLAG10 -- Program flag indicating Phase 4 of VARYIT is in the sensitivity integration mode when FLAG10 = 1.

FLAG20 -- Input data flag to terminate "batch" problem mode of operation and return control to system monitor.

G1(I,J,K), G2(I,J,K), G3(I,J,K), G4(I,J,K) -- Matrices used for storing polynomial matrices in the formulation processes.

GNVARY -- Variable used to store the nominal value of the varying conductance during formulation.

IFLAG -- Program flag generated by Phase 3 and utilized in the evaluation of matrices at nominal values.

INDEX1(I) -- Row index number for a non-zero element of the cutset matrix S.

INDEX2(I) -- Column index number for a non-zero element of the cutset matrix S.

INITL -- If INITL = 1, initial conditions are supplied for the state model integration.

INITL2 -- If INITL2 = 1, initial conditions are supplied for the sensitivity operator integration.

INSEN(I) -- Alphabetic array used as flag to indicate that initial conditions are to be read in for the sensitivity operator integration.

TABLE VIII (Continued)

INSIGN(I) -- Variable indicating the sign of the non-zero element of the cutset matrix $\underline{S}$ which lies in row INDEX1(I) and column INDEX2(I).

INTREE -- Logical variable which indicates the varying parameter belongs to the tree set when true.

IPUNCH -- Variable used as program flag to indicate the network solutions should be outputted on punch medium and also which phases were executed.

LAST -- Program flag whose value gives information to indicate when the final varying parameter analysis is complete.

LCVALU(I) -- Vector obtained from CLVALU by interchanging inductor parameter values with the capacitor values.

LCX1(I) -- Vector containing the initial conditions for the state model integration.

LETTIM -- Program flag which when equal to one indicates a state model integration is to be performed.

LIST -- Alphabetic variable which is used to set program flags to indicate whether complete time solution is to be printed.

LISTDC -- Program flag which forces complete time solution to be printed when its value is one.

LISTDR(I) -- List of element numbers corresponding to drivers which are impulse drivers.

LTCAPT -- The element number of the capacitor with the largest number.

LTCTSR -- The element number of the current driver with the largest number.

LTINDT -- The element number of the inductor with the largest number.

MAXDEG -- The maximum number of two-terminal elements incident to any node of the network.

TABLE VIII (Continued)

MDET -- The degree of the DET polynomial.

MM6(I) -- Number of position in the driver vector for type 6 drivers.

MM7(I) -- Number of position in the driver vector for type 7 drivers.

N6 -- Number of type 6 voltage or current drivers.

N7 -- Number of type 7 voltage or current drivers.

NCARD -- Symbolic name of the card reader unit.

NDRIVE -- Number of drivers present in the network.

NMETER -- Number of voltage and current meters to be used in printing the output solution.

NN6(I), NN7(I) -- Number indicating which type driver corresponds to the parameters stored in the accompanying parameter array.

NOCAIN -- Number of capacitors and inductors of the network.

NOCAPS -- Number of capacitors of the network.

NOCDST -- Number of elements belonging to the selected cotree.

NOCLLT -- Variable indicating position in the CL matrix where varying parameter p occurs.

NOCOND -- Number of conductance elements of the network.

NOCTSR -- The number of current drivers in the network.

NODRLT -- Number indicating position in the driver list where the varying parameter occurs.

NOGCOT -- Number of conductances appearing in the cotree.

NOGTRE -- Number of conductances appearing in the tree.

NOIMP -- Number of impulse drivers to be analyzed.

NOINDS -- Number of inductors of the network.

NONODE -- Number of nodes of the network.

NONWEL -- Number of two-terminal elements of the network.

NOPAR -- Number of the varying parameter p.

TABLE VIII (Continued)

NOTRST -- Number of elements belonging to the selected tree.

NOVTSR -- Number of voltage drivers of the network.

NPLOT -- Program flag indicating no plot of the variance is desired if NPLOT = 0.

NPRINT -- Symbolic name of the printer unit.

NPUNCH -- Symbolic name of the punch unit.

NS -- Number of non-zero entries in the cutset matrix $\underline{S}$.

NTH -- The number of increments between printouts of the metered network variables.

NTIMES -- Variable indicating the total number of sets of variances for the varying parameters to be used in forming the variances.

NTWKCN (I,J) -- Network connection array with one row for each node number and element number list of all two-terminal elements connected to each node.

NUMVAR -- Number of parameters for which sensitivity calculations are to be made.

NVAR -- Number of parameters allowed to vary to be used in calculation of the variance.

NVARY -- Number indicating the position in the $\underline{G}_T$ or $\underline{G}_C$ matrices when the varying parameter is a resistor.

ORIENT (I) -- The network element orientation list which lists nodes from which elements are oriented.

PARTDR (I) -- Array which contains the derivative of the driver vector with respect to the varying parameter p.

PARTLC (I) -- Array which contains the derivative of the inductor-capacitor vector with respect to the varying parameter p.

PREC -- The precision limit to be used in calculation of the matrix exponential.

TABLE VIII (Continued)

PROBNO (I) -- Block of storage for user specified problem title up to 24 characters.

S(I,J) -- The cutset equation coefficient matrix $\underline{S}$.

SAV(I,J) -- Matrix containing one row for each varying parameter and each column is solution at time t_j to sensitivity model multiplied by appropriate parameter variance.

T -- Value of time variable during state model integration.

TMAX -- The maximum value of time for which the solutions will be found.

TRESET(I) -- The list of element numbers corresponding to elements placed in the tree.

TOPNOD -- The first node located to which the tree element is incident.

VARIAN(I) -- Array formed by summing columns of SAV to yield the variance of the system function for the multiple parameter variation case.

VAR(I) -- Array containing variances of each varying parameter in same order as VARY(I).

VARY(I) -- Array containing list of element numbers which are to be used in computing sensitivity information.

VOLVAR -- Program flag to indicate that a capacitor or voltage driver vary.

WHICH -- Alphabetic variable used to store information as to which of the many options are desired.

Y(I) -- Array which contains the complete network solution at time T for all voltages and currents as well as the sensitivity operator solutions (see Output Tape Format).

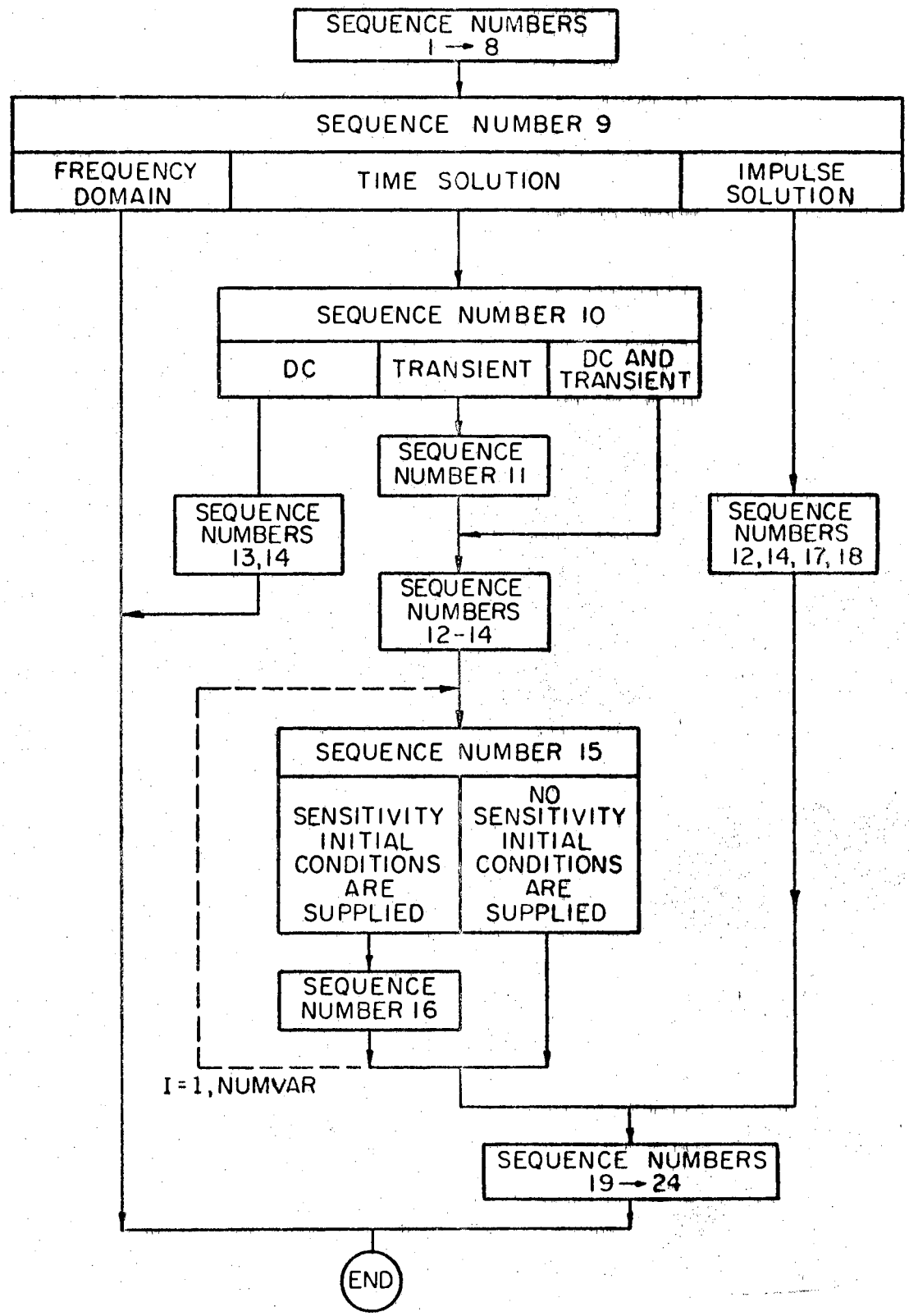


Figure 30. Input Data Preparation Chart

TABLE IX

DESCRIPTION OF INPUT DATA FOR PROGRAM VARYIT

Location in Sequence	Number of Cards	Description of Contents	Fortran Variables	Format
1	1	99999 only if last problem to be done, otherwise blank	FLAG20	I5
2	1	Identification label for problem	PROBNO	4A6
3	1	Number of nodes, network elements, voltage sources, current sources, capacitors, and inductors	NONODE, NONWEL, NOVTSR NOCTSR, NOCAPS, NOINDS	16I5
4	NONODE	List of elements incident at node I	(NTWKCN(I,J),J=1,10)	16I5
5	1	List of node numbers from which each element i is oriented	ORIENT(I)	16I5
6	NOCOND	N=5, Resistance of element NO = VALUE	N, NO, VALUE	2I5, E15.8
7	NOCAIN	N=6, Capacitance or inductance of element NO is VALUE	N, NO, VALUE	2I5, E15.8
8	1	N=7, Number of varying parameters	N, NUMVAR	16I5
	1	N=7, List of varying parameter numbers	N,(VARY(I),I=1,NUMVAR)	16I5
9	1	Specify type solution desired: TIME SOLUTION, IMPULSE SOLUTION, FREQUENCY DOMAIN SOLUTION	WHICH	A6

TABLE IX (Continued)

Location in Sequence	Number of Cards	Description of Contents	Fortran Variables	Format
10	1	Specify type time solution: DC, DC AND TRANSIENT, TRANSIENT	WHICH	A6
11	NOCAIN	N=10, Initial condition on capacitor or inductor NO = VALUE	N, NO, VALUE	2I5, E15.8
12	1	N=11, Maximum solution time value, integration increment, desired precision	N, TMAX, DELTAT, PREC	I5, 5E15.8
13	NDRIVE	N=12, Driver NO is NTYPE driver and NKIND, driver parameters (see Table)	N, NO, NTYPE, NKIND, (VALU(IK),IK=1,6)	I2, I3, I3, I2, 7F10.5
14	1	LIST prints all values during integration; NOLIST suppresses printing	IWHICH	A6
15	1	SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED or else NO SENSITIVITY INITIAL CONDITIONS	INSEN	8A6
16	NOCAIN	N=17, Initial condition on sensitivity model element NO = VALUE	N, NO, VALUE	2I5, E15.8
17	1	Number of driver positions having impulse input	NOIMP	16I5
18	1	List of driver numbers with impulse inputs	LISTDR(I)	16I5

TABLE IX (Continued)

Location in Sequence	Number of Cards	Description of Contents	Fortran Variables	Format
19	1	Prints answers at times NTH steps apart; IPUNCH=1 punches out answers	NTH, IPUNCH	16I5
20	1	Number of meters for which it is desired to punch or print answers	NMETER	16I5
21	NMETER	AIB=V means voltage meter on element NO and AIB=I means current meter on element NO	AIB, NO	A1, I4
22	1	Number of variance computations required, NPLOT=1 means plot variance at steps of NTH, NTIMES is number of different variance sets for NUMVAR parameters	NVAR, NPLOT, NTIMES	16I5
23	NVAR	AIB=V means variance of voltage of element NO and AIB=I means variance of current of element NO is to be found	AIB, NO	A1, I4
24	NTIMES	Variances of elements in VARY list	(VAR(I),I=1,NUMVAR)	5F15.8

TABLE X

INPUT DATA FOR TIME SOLUTION

```
                                                                  Sequence Number
                    EXECUTE THE FOLLOWING PROBLEM                          (1)
NETWORK A                                                                  (2)
    4    6    0    1    2    0                                            (3)
    1    4
    1    2    5
                                                                           (4)
    2    3    5    6
    3    4    6
    1    2    3    4    2    3                                            (5)
    5    1 1.0          +0
    5    2 2.0          +0                                               (6)
    5    3 .33333333    +0
    6    5 0.5          +0
                                                                           (7)
    6    6 1.0          +0
    7    5
    7    1    2    3    5    6                                            (8)
TIME SOLUTION                                                              (9)
TRANSIENT                                                                 (10)
   10    5 0.0          +0
                                                                          (11)
   10    6 0.0          +0
   11 6.0          +0 0.01        +0 0.0000001    +0                     (12)
12  4  7 0 0.0        0.0        7.5                 7.5        1.0       (13)
NOLIST                                                                    (14)
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
   17    5 0.0
   17    6 0.0
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
   17    5 0.0
   17    6 0.0                                                           (15)
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED                                &
   17    5 0.0                                                           (16)
   17    6 0.0
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
   17    5 0.0
   17    6 0.0
SENSITIVITY INITIAL CONDITIONS ARE SUPPLIED
   17    5 0.0
   17    6 0.0
    5    0                                                               (19)
   12                                                                    (20)
V   1
I   1
V   2
I   2                                                                    (21)
V   3
I   3
V   4
I   4
```

TABLE X (Continued)

V	5							
I	5							(21)
V	6							
I	6							
	1	1	1					(22)
V	4							(23)
	0.01		0.04	0.0011111111	0.0025	0.01		(24)

TABLE XI

INPUT DATA FOR IMPULSE SOLUTION

	Sequence Number
EXECUTE THE FOLLOWING PROBLEM	(1)
NETWORK A	(2)
	(3)
4 6 0 1 2 0	
1 4	
1 2 5	(4)
2 3 5 6	
3 4 6	
1 2 3 4 2 3	(5)
5 1 1.0 +0	
5 2 2.0 +0	(6)
5 3 .33333333 +0	
6 5 0.5 +0	
6 6 1.0 +0	(7)
7 5	
7 1 2 3 5 6	(8)
IMPULSE SOLUTION	(9)
11 3.0 +0 0.05 +0 0.0000001	(12)
NOLIST	(14)
1	(17)
4	(18)
1 0	(19)
0	(20)
1 1 1	(22)
V 4	(23)
0.01 0.04 0.0011111 0.0025 0.01	(24)

TABLE XII

INPUT DATA FOR POLE-ZERO

	Sequence Number
EXECUTE THE FOLLOWING PROBLEM	(1)
NETWORK A	(2)
4 6 0 1 2 0	(3)
1 4	
1 2 5	(4)
2 3 5 6	
3 4 6	
1 2 3 4 2 3	(5)
5 1 1.0 +0	
5 2 2.0 +0	(6)
5 3 .33333333 +0	
6 5 0.5 +0	
6 6 1.0 +0	(7)
7 5	
7 1 2 3 5 6	(8)
FREQUENCY DOMAIN SOLUTION	(9)

TABLE XIII

TIME DEPENDENT SOURCE PARAMETER DEFINITIONS

NTYPE = 6

$$\text{Driver}(t) = A \sin[2\pi f(t - T_1) + \phi] \qquad t \geq T_1$$

$$= 0 \qquad t < T_1$$

and NKIND = 0 => Clear previous value, then add new value.

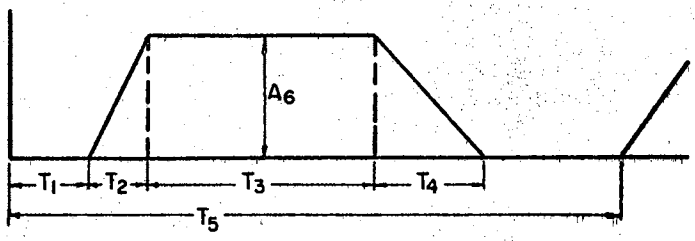
= 1 => Multiply previous value by new value.

= 2 => Add to previous value the new value.

where VALU(1) = A VALU(3) = T_1

VALU(2) = f VALU(4) = ϕ

NTYPE = 7



$$\text{Driver}(t) = 0 \qquad 0 \leq t \leq T_1$$

$$= A_6\left(\frac{t - T_1}{T_2}\right) \qquad T_1 \leq t \leq T_1 + T_2$$

$$= A_6 \qquad (T_1 + T_2) \leq t \leq (T_1 + T_2 + T_3)$$

$$= A_6\left[1 - \frac{t - T_1 - T_2 - T_3}{T_4}\right] \qquad (T_1 + T_2 + T_3) \leq t \leq (T_1 + T_2 + T_3 + T_4)$$

$$= 0 \qquad (T_1 + T_2 + T_3 + T_4) \leq t \leq T_5$$

and NKIND = 0 => Clear previous value, then add new value.

= 1 => Add to previous value the new value.

= 2 => Multiply previous value by new value.

where VALU(1) = T_1 VALU(4) = T_4

VALU(2) = T_2 VALU(5) = T_5

VALU(3) = T_3 VALU(6) = A_6

TABLE XIV

OUTPUT TAPE FORMAT

Time Solution

Each record consists of
T,(Y(I), I=1,2*NONWEL)

where

$$Y = \begin{bmatrix} J_L \\ I_{CG} \\ I \\ E_C \\ V_{TG} \\ V \\ E_L \\ V_{CG} \\ V_I \\ I_C \\ I_{TG} \\ I_V \end{bmatrix}$$

J_L	NOINDS
I_{CG}	NOGCOT
I	NOCTSR
E_C	NOCAPS
V_{TG}	NOGTRE
V	NOVTSR
E_L	NOINDS
V_{CG}	NOGCOT
V_I	NOCTSR
I_C	NOCAPS
I_{TG}	NOGTRE
I_V	NOVTSR

Cotree Currents

Tree Voltages

Cotree Voltages

Tree Currents

Impulse Solution

Each record consists of
T, J,(Y_a(I), I=1,4*NONWEL)

where

J = Number of driver in impulse list

$$Y_a = \begin{bmatrix} Y \\ Y' \end{bmatrix}$$

APPENDIX E

DETAILS OF PROGRAM VARNOL

This appendix presents in greater detail the operations described in Section 4.5. Since VARNOL uses phases 2, 3, and 7 of VARYIT and much of phase 1, most of the FORTRAN variables are common to both programs. The main differences in input data are due to the presence of nonlinear elements which must be described. The types of dependent drivers and nonlinear storage elements which are allowed are shown in Table XV where the parameters for each source are related to the input data of Table XVI.

Program decks for VARNOL are available from the Engineering Computing Laboratory at Oklahoma State University upon request. A separate addendum document containing flow charts and listings will be furnished with these decks.

TABLE XV

NONLINEARITY REPRESENTATIONS INCLUDED IN VARNOL

Type 1

$$\text{Driver (t)} = A_0 + A_1 Y_k + A_2 Y_k^2 + A_3 Y_k^3 + A_4 Y_k^4$$

where Y_k = Voltage or current through element number k.

NO = Element number of dependent source being described as dependent on Y_k.

NTYPE = 0 Test value for convergence criteria is formed by using Y_k.

= 1 Test value for convergence criteria is formed by using $|Y_k|$.

NKIND = 0 Test value for convergence criteria is unchanged in sign.

= 1 Test value for convergence criteria is changed in sign.

AMPS = V Y_k is the voltage across element k.

= I Y_k is the current across element k.

NTEST = Element number k.

VALU(1) = A_0

VALU(2) = A_1

VALU(3) = A_2

VALU(4) = A_3

VALU(5) = A_4

DELTAO = Starting increment for parameter changes during convergence testing.

Type 2

This driver option is currently not used.

TABLE XV (Continued)

Type 3

$$\text{Driver (t)} = \left[A_1Y_m + A_2Y_m^2 + A_3Y_m^3 + A_4Y_n + A_5Y_nY_m + A_6Y_nY_m^2 + A_7Y_n^2 + A_8Y_n^2Y_m + A_9Y_n^3\right]\left[\frac{Y_m}{Y_m + B}\right]$$

where Y_k = Voltage or current through element number k

NO = Element number of dependent source being described as dependent on Y_n and Y_m.

for card 1

AMPS = V Y_n is the voltage across element n.

= I Y_n is the current through element n.

NTEST = Element number of element n.

VALU(1) = A_1

VALU(2) = A_2

VALU(3) = A_3

VALU(4) = A_4

VALU(5) = A_5

for card 2

AMPS = V Y_m is the voltage across element m.

= I Y_m is the current through element m.

NTEST = Element number of element m.

VALU(1) = A_6

VALU(2) = A_7

VALU(3) = A_8

VALU(4) = A_9

VALU(5) = B

TABLE XV (Continued)

$$\text{Driver}(t) = I_s(e^{bY_k} - 1)$$

where Y_k = Voltage or current through element number k.

NO = Element number for source being described by exponentially dependent driver.

AMPS = V Y_k is the voltage across element k.

= I Y_k is the current through element k.

NTEST = Element number k

VALU(1) = I_s

VALU(2) = Increment to be used in iteration for convergence.

"b" is held constant at 38.5.

TABLE XV (Continued)

Type 5

$$\text{Driver (t)} = A_1 Y_k + A_2 \qquad Y_k > X_1$$

$$= A_3 Y_k + A_4 \qquad Y_k < X_1$$

with

$$A_1 X_1 + A_2 = A_3 X_1 + A_4$$

NO = Element number of source being described by the piecewise continuous model.

AMPS = V Y_k is the voltage across element k.

= I Y_k is the current across element k.

NTEST = Element number k.

VALU(1) = A_1

VALU(2) = A_2

VALU(3) = A_3

VALU(4) = A_4

VALU(5) = X_1

Type 6

This driver is the same as that of Figure E.2.

Type 7

This driver is the same as that of Figure E.3.

TABLE XV (Continued)

Nonlinear Storage Elements

Two types of nonlinear storage elements are present. They are distinguished by the variable TYPELC.

TYPELC = 1.0

Iron Core Inductor Model

$$L = \frac{C^2}{Y_k^2 + C^2} \left[A + 2B \tan^{-1} \frac{Y_k}{C} \right]$$

where L is the value of the storage element.

NO = Element number of storage element being described by this model.

AMPS = V Y_k is voltage across element k.

= I Y_k is current across element k.

NTEST = Element number k.

VALU(1) = A

VALU(2) = 2B

VALU(3) = C

VALU(6) = 1.0

TABLE XV (Continued)

TYPELC = 0.0

Polynomial Dependency Model

$$L \text{ or } C = A_1 + A_2 Y_k + A_3 Y_k^2 + A_4 Y_k^3 + A_5 Y_k^4$$

where L or C is the value of the storage element.

NO = Element number of storage element being described by this model.

AMPS = V Y_k is the voltage across element k.

= I Y_k is the current through element k.

NTEST = Element number k.

VALU(1) = A_1

VALU(2) = A_2

VALU(3) = A_3

VALU(4) = A_4

VALU(5) = A_5

VALU(6) = 0.0

TABLE XVI

DESCRIPTION OF INPUT DATA FOR PROGRAM VARNOL

Location in Sequence	Number of Cards	Description of Contents	FORTRAN Variables	Format
1	1	Identification label for problem	PROBNO	4A6
2	1	Number of nodes, network elements, voltage sources, current sources, capacitors, and inductors	NONODE,NONWEL,NOVTSR NOCTSR,NOCAPS,NOINDS	16I5
3	NONODE	List of elements incident at node I	(NTWKCN(I,J),J=1,10)	16I5
4	1	List of node numbers from which each element I is oriented.	(ORIENT(I),I=1,NONWEL)	16I5
5	NOCOND	Resistance of element NO is VALUE	NO,VALUE	I5,E15.8
6	NOCAIN	Capacitance or inductance of element NO is VALUE	NO,VALUE	I5,E15.8
7	1	Number of varying parameters	NUMVAR	I5
	1	List of varying parameter numbers	(VARY(I),I=1,NUMVAR)	16I5
8		Number of type 1 through 7 drivers and number of dependent inductors and capacitors	N1,N2,N3,N4,N5 N6,N7,NODEPL	16I5

TABLE XVI (Continued)

Location in Sequence	Number of Cards	Description of Contents	FORTRAN Variables	Format
		N=1, Element number, class, and kind of source	N,NO,NTYPE,NKIND	16I5
9	N1	AMPS=I means this source is dependent on current through element NTEST; AMPS=V means this source is dependent on voltage through element NTEST; VALU is parameters for this type of driver	AMPS,NTEST,(VALU(IK),JK=1,5)	A1,I4,5E15.8
		N=1, starting increment for convergence test	N,DELTAO	I5,2F15.8
		N=3, Element number for type 3 source	N,NO	2I5
10	N3	Same as sequence number 9 and contains parameters A_1 through A_5	AMPS,NTEST,(VALU(IK),IK=1,5)	A1,I4,5E15.8
		Same as sequence number 9 and contains parameters A_1 through A_9	AMPS,NTEST,(VALU(IK),IK=1,5)	A1,I4,5E15.8
11	N4	N=4, Element number for type 4 source	N,NO	2I5
		Same as sequence number 9	AMPS,NTEST,(VALU(IK),IK=1,2)	A1,I4,2E15.8
12	N5	N=5, Element number for type 5 source	N,NO	2I5
		Same as sequence number 9	AMPS,NTEST,(VALU(IK),IK=1,5)	A1,I4,5E15.8

TABLE XVI (Continued)

Location in Sequence	Number of Cards	Description of Contents	FORTRAN Variables	Format
13	N6	N=6, Element number for type 6 driver, kind, parameters for this type driver	N,NO,NTYPE,(VALU(IK), IK=1,4)	3I5,6E10.4
14	N7	N=7, Element number for type 7 driver, kind, parameters for this type driver	N,NO,NTYPE,(VALU(IK), IK=1,6)	3I5,6E10.4
15	NODEPL	N=8, Element number for dependent capac-itor or inductor, remainder similar to sequence number 9	N,NO,AMPS,NTEST,(VALU(IK), IK=1,6)	2I5,1X,A1,I3, 6E10.4
16	Variable	Element number NO has initial value DV	NO,DV	I5,F15.8
17	1	Blank card signifying end of initial driver values (Insert only if sources exist).	NO,DV	I5,F15.8
18	1	N=12, Numerical convergence criteria EPS for dependent sources (Omit if no dependent sources exist).	N,EPS	I5,F15.8
19	1	N=13; LETTIM=1 implies only transient solution desired; LISTDC=1 implies printing complete solution at each instant of time; LISTT=1 implies a listing of convergence criteria as iteration proceeds.	N,LETTIM,LISTDC,LISTT	4I5

TABLE XVI (Continued)

Location in Sequence	Number of Cards	Description of Contents	FORTRAN Variables	Format
20	1	N=14, maximum time and increment for integration (Omit if LETTIM≠1)	N,TMAX,DELTAT	I5,2F15.8
21	1	N=15, INITL=1 implies initial conditions are supplied for capacitors and inductors (If NOCAIN=0 skip to sequence number NOTE).	N,INITL	2I5
22	NOCAIN	N=16, Element number NO has initial value DV	N,NO,DV	2I5,E15.8
23 NUMVAR	1	N=17, INITL=1 implies initial conditions are supplied for capacitors and inductors for sensitivity integration	N,INITL	2I5
	NOCAIN	N=17, Element number NO has initial value DV (Omit if INITL≠1)	N,NO,DV	2I5,E15.8

NOTE: The data for the output phase should be prepared as Sequence Numbers 19 to 24 in Table E.2.

VITA

Jack Hammond Pridgen

Candidate for the Degree of

Doctor of Philosophy

Thesis: COMPUTATION OF SENSITIVITY MEASURES

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Paris, Texas, August 23, 1941, the son of Jack L. K. and Ruth Pridgen.

Education: Attended grade school and high school in Texarkana, Texas; graduated from Texas High School in June, 1959; received the Bachelor of Science degree from Southern Methodist University, with a major in Electrical Engineering, in August, 1963; received the Master of Science degree from Oklahoma State University, with a major in Electrical Engineering in May, 1965; completed requirements for the Doctor of Philosophy degree at Oklahoma State University in May, 1970.

Professional Experience: Employed by Texas Instruments Incorporated in Dallas, Texas, as co-operative student while attending Southern Methodist University and Summer Development engineer at Texas Instruments during summers of 1963 and 1964 in space systems and airborne radar groups. Employed by School of Electrical Engineering at Oklahoma State University as a graduate research assistant from September, 1965 through November, 1966. Employed by Texas Instruments Missiles and Ordnance Systems Department since May, 1967.

Professional Organizations: Member of IEEE, Sigma Tau, Eta Kappa Nu, and Phi Kappa Phi.