DIGITAL COMPUTER ANALYSIS OF SYSTEMS WITH NONLINEAR ELEMENTS

By

JACK MILFRED WALDEN

Bachelor of Science
South Dakota School of Mines and Technology
Rapid City, South Dakota
1944

Master of Science
Oklahoma State University
Stillwater, Oklahoma
1962

Submitted to the Faculty of the Graduate School of
the Oklahoma State University
in partial fulfillment of the requirements
for the degree of
DOCTOR OF PHILOSOPHY
May, 1966

DIGITAL COMPUTER ANALYSIS OF SYSTEMS

WITH NONLINEAR ELEMENTS

Thesis Approved:

D. A. Blackwell
Thesis Adviser

Paul C. McCollum

Richard L. Cummins

William J. Lewis

J. Wayne Johnson

J. H. Boggs
Dean of the Graduate School

PREFACE

The motivation for this study was provided by a strong personal conviction that the analysis and/or design of large physical systems is almost hopeless without the aid of a digital computer. Without such aid, the system must be idealized to such an extent that the final results are of questionable value. As our technology develops, the complexity of the systems we create will increase so that this problem becomes more serious.

I will readily admit to a bias in my viewpoint on the importance of digital computing in modern technology. It is my personal conviction that the role of computers in engineering today is limited more by our lack of understanding of how to use them than by shortcomings in the computers themselves. Rapid developments in computer hardware and software now in progress will increase their versatility far beyond our ability to use them unless considerable study and research improve our capabilities by orders of magnitude. It is my hope that, in some small way, this study will help us in our use and understanding of this powerful tool.

The original ideas for this study were developed when I attended the 1963 Summer Session on "Analysis and Design of Discrete Physical Systems" at Michigan State University, sponsored by the National Science Foundation. I wish to express my gratitude for the opportunity to attend this session.

To Dr. W. A. Blackwell, Chairman of my advisory committee, I wish to express my special thanks for his counsel, guidance, and encouragement during my doctoral program. His assistance and advice were invaluable aid at all times.

To Professors P. A. McCollum, R. L. Cummins, L. W. Johnson, and W. J. Leivo, the other members of my committee, I owe a debt of gratitude for their counsel and guidance in my entire study program.

To Dr. William L. Hughes, Head of the School of Electrical Engineering, I am deeply indebted. His enthusiasm, help, and encouragement during my graduate studies has been unfailing. Without his assistance, this program would have been impossible.

To Dr. Dale Grosvenor, Director of the University Computing Center, I express thanks for wholehearted cooperation and assistance.

The Ford Foundation has provided financial assistance for which I wish to express my appreciation.

Finally, a special note of thanks to my wife, Nancy, and my family, whose good cheer and never-failing encouragement made this project possible--and worthwhile.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Motivation. The high-speed digital computer is exerting an ever-increasing influence on every phase of engineering and science. Its utilization has become a matter of paramount concern for every engineer.

The early applications of the digital computer in science and engineering were in the areas of most obvious benefit. The reduction and treatment of large amounts of data and the solution of mathematical functions to an accuracy never before possible received first attention.

As the use and understanding of the digital computer developed, the areas of application broadened. Numerical computation techniques improved, and the solution of broader and broader classes of mathematical systems became possible and practical. Concurrently, our technology developed and the systems of interconnected and interrelated components became more complex. As the size of interconnected systems of all kinds--electrical, mechanical, electromechanical, hydraulic, and pneumatic--increased, the conventional techniques for analyzing their characteristics and performance became inadequate. The sheer size of systems made pencil-and-paper solution impractical.

Of even greater importance was the inability of the conventional analysis techniques to predict observed phenomena of instability, surges, and breakdowns. It became increasingly apparent that the idealizing of elements into linear equivalents in system models was the source of

many of the analysis problems. The whole world of analysis was built on linear models, so that a very real and awesome enigma presented itself--how can the analysis of nonlinear systems be carried out when the systems are so complicated that even the linear analysis is impractical?

The answer may well lie in proper and efficient use of the digital computer. In this area, the known techniques are rudimentary, and the known applications are limited. But the prospects are encouraging in the light of present computer capacities and presently known numerical methods. The adaptation of many efficient and orderly linear analysis techniques may be possible. Transformation methods for the solution of the differential equations of linear systems has been highly developed. Brown (1) has developed a method for the application of the LaPlace transform to some nonlinear systems.

The development of numerical methods for the approximate integration of differential equations has been highly developed, and many efficient and highly accurate algorithms for this purpose are generally known (2 - 10). The solution of nonlinear algebraic equations has received much attention, and many methods are also generally known. Application of these methods for the direct numerical solution of the equations of a mathematical model by digital computer offers great promise. However, for practical use by the engineer, this approach involves several serious problems: (a) how can the equations of the mathematical model be determined in an efficient and orderly way, (b) how can the solution be implemented on a computer, and (c) how can the results of this solution process--a column of numbers--be interpreted and understood? The motivation for this study lies in the first two of these--a desire

to study and develop techniques for utilizing the digital computer in both the formulation and solution of nonlinear systems. The answer to the last problem--how to understand and interpret the results--can only come from experience and familiarity. Until the first two have been answered, this is impossible.

1.2 Scope of This Study. The direct objectives of this study have been fourfold: (a) to develop an algorithm suitable for direct programming on a digital computer for the formulation of the equations of a suitable mathematical system model, (b) to explore methods of introducing and including in this model some nonlinear elements, (c) to develop suitable mathematical models for the description of a selected class of nonlinear elements, (d) to explore some of the well-known numerical techniques and their applicability in the solution of the models that have been developed.

The state-space model was selected as the basis for this study. This requires the determination of an independent set of normal form (first-order) differential equations and an accompanying set of algebraic equations. For convenience in the manipulation of these equation sets, they are developed in matrix notation; and, wherever possible, the methods of matrix algebra are utilized. This is an advantage not only in compactness and efficiency of notation, but in direct utilization of the operations in computer programming. Published works by Koenig, Tokad, and Kesavan (11), Koenig and Blackwell (12), and Zadeh and Desoer (13) provide an adequate background in state-space and modeling, particularly for linear systems.

The techniques of linear graph theory and topology developed by

Seshu and Reed (14) were selected as the method of formulation of the equations of the system. The efficiency and power of these techniques make possible a truly methodical process for the formulation of the necessary equations.

A necessary part of the work connected with this study was the writing of a computer program to test and evaluate the ideas and methods. This program has been written entirely in the Fortran language for use on a medium-sized computer.

The development of equations describing the nonlinear elements was accompanied by additional computer programming to evaluate the coefficients of the model for actual physical devices.

CHAPTER II

BACKGROUND OF COMPUTER SYSTEM ANALYSIS

2.1 Linear Systems. Undoubtedly, many workers in various parts of the country in research laboratories, universities, and computer centers have developed digital computer programs for the analysis of linear electrical and/or mechanical systems. However, very little of this work is published and available to other workers. Among the published works in this area, two are particularly valuable for reference. They represent two entirely different approaches--one through the s-domain (complex frequency) and the other through the time domain and direct integration of the differential equations.

Typical of work in the s-domain is the analysis program developed by Calahan (15) at the University of Illinois. The program is set up for the solution of linear electrical circuits containing passive elements, time-dependent sources, and linear voltage-dependent current sources. Topological methods based on tree-products for creation of the s-domain transfer functions are utilized. The poles and zeroes of this transfer function are obtained by a polynomial factoring routine. From these, the unit-step and unit-impulse response can be obtained. Provisions for direct integration of a set of differential equations deduced from the transfer function is provided. Similar work has also been done by Cummins and Thomason (16) at Oklahoma State University. Their work includes implementation of an improved tree-listing

algorithm developed by Minty (17). Transfer function analysis methods are very efficient for linear systems, but their usefulness in the treatment of nonlinear systems is questionable and relatively unknown.

Typical of work directly in the time domain is a program (MISSAP) developed by Reid at Michigan State University. Unfortunately, the sponsor of this project has not allowed publication of the work, and little is known about it. The program is based on solution of the state-space model and is intended to handle only linear passive elements, time-dependent sources, and internally-dependent linear sources. Provisions for non-ideal transformers and long transmission lines are also incorporated. Topological methods are employed in the determination of the state-space equations. Matrix row and column manipulations coupled with matrix rank evaluation are employed to determine a tree of the linear graph. The program is designed with a simple and convenient input data format to encourage use by the widest possible class of users.

A program similar to MISSAP has been implemented by IBM for various models of their computers and is distributed under the program name ECAP. ECAP has some additional features for the study of worst-case analysis and power supply failure analysis.

2.2 Systems with Nonlinear Elements. Programs for the analysis of nonlinear systems have dealt primarily with the analysis of transistor switching circuits, in which the only nonlinear elements are transistors or diodes.

Most of these programs have been based on work done by Branin (20) at the IBM Product Development Laboratory under the program name

DCAP-PETAP. Further development has been carried on by Malmberg (21) at the Los Alamos Scientific Laboratory of the University of California under the program name NET. The basic algorithm uses topological and matrix methods. The matrix manipulations are carried out as a variation of Kron's method of matrix tearing (22). The nonlinear transistor model in the program includes voltage-dependent current sources based on the Ebers-Moll equations for the transistor (23) and nonlinear voltage- and current-dependent capacitors. Provisions for evaluation of the variation of α, the forward-current transfer ratio, under dynamic conditions are included.

The differential equations of the system are divided into two sets in both programs. One of the sets contains the nonlinear equations of the transistors only. The other set contains the linear equations associated with the remainder of the circuit--that is, all components except the transistors themselves. Integration of the equations alternates between these two sets. While one set is being integrated, all of the variables defined by the other set are assumed to remain constant. Because of the entirely different characteristics of the two sets of equations, entirely different integration routines are utilized for their solution. It is frequently necessary to make many small integration steps in the solution of the nonlinear equations between much larger steps in integration of the linear equations. IBM has apparently stopped any further development on the DCAP-PETAP program, but Malmberg has continued his work and has several versions of the program for different computers.

A very powerful program for the analysis of linear and/or nonlinear mechanical systems has been developed by Hart (19) and co-workers at

the General Motors Research Center under the program name DYANA. This program is intended for the solution of a wide range of mechanical systems (translational and/or rotational) with a wide variety of linear or nonlinear constraints.

Two basic versions of the program have been developed--one for transient-time response and the other for steady-state sinusoidal response. The program is written in IBM 704 Symbolic Language so that it is usable only on that model of computer. The output of the program is novel in that it is not directly a solution to the system. Rather, it is a set of statements which comprise a Fortran Source program, which is then compiled and executed by a standard Fortran Compiler to obtain the actual solution. Thus, the DYANA program, in itself, is not a solution program, but a compiler to assemble a program which can solve the particular system specified.

The DYANA program sets up the solution of a system of differential equations which are not in normal form, which are integrated by standard integration subroutines incorporated in the Fortran System itself. The constraints on the system, either linear or nonlinear, are incorporated by the formulation of a system of side equations in the form of Lagrangian-multiplier equations of constraint. From these constraint equations, various parameters of the system are evaluated after each step in the integration.

The program is able to solve electrical or mixed electrical-mechanical systems only by analogy with mechanical components. This restricts its usefulness for an engineer interested in electrical or mixed systems. However, this program is one of the most powerful and versatile programs yet developed.

CHAPTER III

DEVELOPMENT OF THE ALGORITHM FOR COMPUTER ANALYSIS

The state-space model is a set of normal-form differential equations and a set of algebraic equations. For any given system it may be possible to obtain a number of different state-space models. The differences lie in the choice of the state variables. There is usually considerable latitude possible in this choice, and the final selection may be determined by a variety of factors. One of the most common restrictions placed on this choice, for instance, is that the variables selected shall represent physically measurable quantities.

For the purposes of this study, it was necessary to develop an algorithm suitable for computer implementation. The realization of this aim, in itself, imposes restrictions on the choice of state variables. The particular form of the model is such as to implement the application of the method to the analysis of systems containing some nonlinear elements. This, too, imposed further restrictions on the choice of variables.

3.1 Theoretical Background: State-Space Models and Topological Methods. The state-space model, in its most general form, appears as a set of differential equations which, in matrix notation, appear as

$$\dot{\underline{\psi}} = \underline{A}\,\underline{\psi} + \underline{B}\,\underline{F} \qquad (3.1.1)$$

where ψ is the vector of state variables, and $\underline{F}$ is the vector of forcing functions. Also a part of the state-space model is the algebraic matrix equation

$$\underline{W} = \underline{C}\,\underline{\psi} + \underline{D}\,\underline{F} \qquad (3.1.2)$$

where $\underline{W}$ is the vector of variables associated with non-storage (memoryless) elements of the system. Zadeh and Desoer (13) have established the existence of such a model for a time-invariant, linear system, which will be accepted without proof here. For such a system, all of the entries in the matrices $\underline{A}$, $\underline{B}$, $\underline{C}$, and $\underline{D}$ of Equations 3.1.1 and 3.1.2 are real constants or zero.

In the establishment of a specific state model for a system, it is necessary to select one specific set of variables for ψ before the entries in the matrices can be determined. In the analysis of physical systems, it is usually convenient and desirable to choose state variables which directly represent physically measurable quantities, if this is possible. Koenig, Tokad, and Kesavan (11) have established that this is possible in a restricted class of systems. For the purposes of this study, only such systems will be considered. This restriction will be considered in more detail.

In any linear, time-invariant system, the passive elements comprising the system can be represented mathematically in one of the following forms:

$$x_1 = r\,y_1 \quad (\text{or } y_1 = g\,x_1) \qquad (3.1.3)$$

$$x_2 = \ell\,\dot{y}_2 \qquad (3.1.4)$$

$$y_3 = c\,\dot{x}_3 \qquad (3.1.5)$$

In these equations, x and y are functions of time; and the dot notation is used to represent differentiation with respect to time. The parameters r, g, ℓ, and c are constants.

x is known as an <u>across</u> variable, and y is known as a <u>through</u> variable. Typically, voltage in electrical systems and translational or rotational velocity in mechanical systems are across variables. Electric current, translational force, and rotational torque are through variables. Equation 3.1.3 represents the dissipation or non-storage (memoryless) elements, and Equations 3.1.4 and 3.1.5 represent the energy storage elements.

In a system comprised of many two-terminal passive elements, these equations when collected and written in matrix form,

$$\underline{X}_1 = \underline{R}\,\underline{Y}_1 \quad (\text{or } \underline{Y}_1 = \underline{G}\,\underline{X}_1) \qquad (3.1.6)$$

$$X_2 = \underline{L}\,\dot{\underline{Y}}_2 \qquad (3.1.7)$$

$$Y_3 = \underline{C}\,\dot{X}_3 \qquad (3.1.8)$$

serve to describe the terminal relations for all of the passive elements. In these equations, the $\underline{X}$'s are vectors of across variables, the $\underline{Y}$'s are vectors of through variables, and $\underline{R}$, $\underline{L}$, $\underline{C}$, and $\underline{G}$ are square diagonal matrices.

Also a part of an active system are drivers, which establish specific conditions for either the across or through variable at their terminals. They are grouped according to which variable is specified at their terminals. Thus, in association with the

equations of the system are two additional vectors of time functions

$$F_x(t), \text{ the across drivers,} \qquad (3.1.9)$$

$$F_y(t), \text{ the through drivers,} \qquad (3.1.10)$$

where the elements of these vectors are specified functions of time, which may be constants.

The mathematical model of the system is determined by the interconnection scheme of the two-terminal elements described by Equations 3.1.6 through 3.1.10. The concepts of linear graph theory have been applied to the problem of formulating the appropriate equations of the system (11, 12, 14) as a convenient and methodical process for the development of the model from the pattern of interconnection of the elements.

The application of linear graph theory to the description of the interconnection scheme of a given physical system is simple and straightforward. For the reader untrained in this notion, Seshu and Reed (14) provide a comprehensive introduction for the treatment of electrical networks. References 11 and 12 provide extension of the technique to other physical systems. A particular concept of importance in this study is that of the directed linear graph, in which the notion of polarity assumptions for the through and across variables is included in the graph.

Since the notion of relative polarity of the through and across variable can be based on a number of possible variations, we will establish here the assumed scheme for the purposes of this study. The arrowhead of the directed graph edge will be assumed to coincide with

the direction of positive polarity for the through variable. This same arrowhead will indicate polarity of the across variable by the assumption that the tail of the arrow will indicate the positive terminal of the device for the across variable. From this notation system, it is apparent that the simultaneous occurrence of positive through and across variables for a given element will imply that the element is absorbing energy from the system. This notion is illustrated in an example for a two-terminal electrical device in Figure 3.1.1.

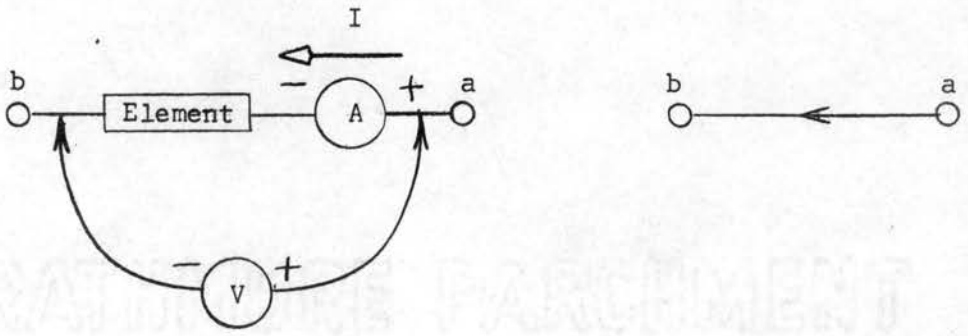


Figure 3.1.1. Electrical Example of Polarity Convention for Across and Through Variables

In the application of linear-graph theory, the fundamental concept of a tree and associated co-tree of the graph are of utmost importance. For concise and detailed definitions and properties of the tree and co-tree, the reader is referred to Seshu and Reed (14, Chapters 2 - 5). A brief description of these concepts is adequate here.

A tree of a linear graph can be described by two of its properties:

(a) A path exists from each node of the graph to every other node

of the graph, traveling along only edges which are in the tree.

(b) The edges in the tree form no closed paths within the graph.

The edges of the graph which are in a particular tree (there may be many trees in any given graph) are called branches of the tree. When the word branch is used in referring to an edge of a graph, it is understood that this edge is in a particular tree.

The co-tree corresponding to a particular tree of a graph is the set of all edges which are not in the tree. These edges are referred to as the chord-set. When an element of the graph is referred to as being a chord of a tree, it is understood that it is a member of the set of edges in the co-tree corresponding to that tree.

In a general treatment of the properties of the trees and associated co-trees of a linear graph, it is necessary to consider the case for both nonseparable and separable graphs. By definition (14, p. 35), a graph G is nonseparable if every subgraph of G has at least two nodes in common with its complement. All other graphs are separable. In the interest of clarity and brevity in the development of the formulation algorithm, only the case of the nonseparable graph will be considered here. However, nothing in the methods utilized will require that only systems represented by nonspearable or one-part graphs can be treated. For further treatment, the interested reader is referred to references 14 and 11 for discussion of the more general case.

The trees and associated co-trees of a graph possess many useful properties. However, two are of particular importance in the formulation of the state-space model. They may be summarized in the

following:

(a) If there are n nodes in a one-part graph, then there are (n - 1) edges in the tree. There is a unique node or supernode associated with each edge of the tree. For each of these nodes or supernodes, an independent equation can be written which states that the sum of the through variables into (or out of) that node is zero. Each such equation will, in fact, relate the through variable of one branch (the branch corresponding to that node) in terms of the through variables of the chords. Therefore, in a one-to-one relationship with the branches of a tree, (n - 1) independent equations can be generated relating the through variables. (The above statements hold true for a p-part graph if n - p is substituted for n - 1.)

(b) If there are e edges in a graph, then there are e - (n - 1) = e - n + 1 edges in a co-tree of the graph. There is a unique closed path around the edges of the graph associated with each chord. This path traverses only that particular chord and branches of the tree. For each of these paths, an independent equation can be written which states that the sum of the across variables around that path is zero. Each such equation will, in fact, relate the across variable of one chord (the chord associated with the path) in terms of the across variables of the branches. Therefore, in a one-to-one relationship with the chords of a tree, e - n + 1 independent equations can be generated relating the across variables. (The above statements hold true for a p-part graph if e - n + p is substituted for

e - n + 1.)

These properties of the tree and co-tree provide a powerful method for the generation of two sets of independent equations relating the across and through variables. These equations and the terminal equations of the elements when properly manipulated yield the state-space model equations.

It was stated earlier (pages 9 and 10) that the state model in the form of Equations 3.1.1 and 3.1.2 would always exist when certain restrictions are placed on the interconnection pattern of the elements. These restrictions can now be more precisely set down, in terms of the properties of linear graphs and their trees and associated co-trees.

Theorem 3.1.1. The state-space model of a system will exist in the form of Equations 3.1.1 and 3.1.2 if it is possible to find a tree and associated co-tree of the linear graph of the system which has the following properties:

(a) All of the across-drivers are contained in the tree.

(b) All of the elements of the type described by Equation 3.1.8 are contained in the tree.

(c) All of the through-drivers are contained in the co-tree.

(d) All of the elements of the type described by Equation 3.1.7 are contained in the co-tree.

This theorem is stated without proof here, since it is adequately developed in the references (14, Chapter 6 and 11, Chapters 6 - 8).

This restriction has been arbitrarily adopted for the purposes of this study and is not a necessary limitation to the general applicability of the method. It is possible to utilize similar topological methods

in the generation of a state-space model for a system which does not meet this restriction, but the model does not have the simple form of Equations 3.1.1 and 3.1.2. It should also be mentioned again that the restriction of this discussion to the case of a one-part graph is not necessary for validity of the method. It has been adopted only for purposes of discussion.

It is also worth noting at this point that nothing in the procedures discussed will limit their application to any particular energy system. Terminal equations in the form of Equations 3.1.6 through 3.1.10 can be written for the elements of any linear system--electrical, mechanical, hydraulic, mixed, or otherwise. It is always possible to describe the interconnection of these elements in terms of a linear graph. In the case of electrical circuits, the linear graph is immediately obvious from the actual physical interconnection of the components. In some other systems, this relationship is not so obvious, but it is possible to develop a linear graph which properly describes the system. For further details in this area, the reader is referred to references 11 and 12, where these matters are treated in considerable detail. For the remainder of this study, discussion will be directed to the treatment of electrical networks. However, the methods and algorithm remain general in nature--only the terminology will be specialized.

The restrictions imposed by Theorem 3.1.1 can be stated in terms more familiar and meaningful to the electrical engineer. They are:

(a) All of the voltage drivers must be in the tree.

(b) All of the capacitors must be in the tree.

(c) All of the current drivers must be in the co-tree.

(d) All of the inductors must be in the co-tree.

The resistors of the network may be arbitrarily allocated to the tree or co-tree as desired.

Stated in yet another way: restrictions (a) and (b) require that no closed path may consist of only capacitors and/or voltage drivers, and (c) and (d) require that no node shall exist to which only current drivers and/or inductors are connected.

At this point, three necessary concepts have been established:

(a) The fundamental notions of linear graph theory and their application in the generation of two sets of independent equations relating the through and across variables of the system,

(b) The classification of the elements of the system and the terminal equations which describe them, and

(c) Some restrictions which must be placed upon the allowable topology of the system.

From these, the algorithm for development of the state-space equations in the form of Equations 3.1.1 and 3.1.2 can be implemented.

3.2 Development of the Algorithm for Formulation of the State-Space Equations. Before proceeding into the actual development of the algorithm, it is necessary to firmly establish a scheme of notation, which will be used henceforth. Wherever possible, standard and accepted electrical notation and symbols will be utilized.

The across variable in electrical networks is electrical voltage. The standard symbol E will be used to denote such quantities. When underlined, $\underline{E}$ will denote a vector of such variables. The through variable in electrical networks is current, and the standard symbol I

will be used. A vector of such variables will be $\underline{I}$. Voltage (across) drivers will be symbolized by V, and current (through) drivers by I.

The role of a particular variable is very largely determined by whether it is associated with the branch or chord of a tree, so that it is important that the notation should reflect this property. All variables will carry the subscript B or C to denote whether they are associated with a branch or chord, respectively. Thus, I_B will denote a current for an element in the tree, and E_C will denote a voltage for an element in the co-tree, and so on.

As a result of the restrictions placed upon the topology of the network, it is possible to divide the branches and chords into subsets as a further aid. This will be done in the following way:

B1 - will denote all capacitive elements

B2 - will denote all resistive elements in the tree

B3 - will denote all voltage drivers

C1 - will denote all inductors

C2 - will denote all resistive elements in the co-tree

C3 - will denote all current drivers

With this notation established, and the elements divided into subsets as indicated, Equations 3.1.6 to 3.1.10 can be written in the form:

$$\underline{E}_{B2} = \underline{R}_B \, \underline{I}_{B2} \qquad (3.2.1)$$

$$\underline{I}_{C2} = \underline{G}_C \, \underline{E}_{C2} \qquad (3.2.2)$$

$$\underline{E}_{C1} = \underline{L} \, \dot{\underline{I}}_{C1} \qquad (3.2.3)$$

$$\underline{I}_{B1} = \underline{C} \, \dot{\underline{E}}_{B1} \qquad (3.2.4)$$

$$\underline{E}_{B3} = \underline{V}\,(t) \qquad (3.2.5)$$

$$\underline{I}_{C3} = \underline{I}\,(t) \quad . \qquad (3.2.6)$$

These six equations are commonly referred to as the volt-ampere (VA) equations of the elements. These equations hold for the elements regardless of their manner of interconnection. From them, by taking into account the interconnection pattern, the state-space equations are developed.

Mathematical information concerning the interconnection of the network is contained in the two sets of equations obtained from the tree and co-tree of the graph.

As a result of the way in which the tree is related to the remainder of the graph, one, and only one, of the branches of the tree will connect to the node or supernode with which it is associated. Mathematically, this is evidenced by the previously stated property of the equation summing the currents at that node to zero (Kirchoff's Current Law). These equations, one-by-one, relate a single branch current in terms of the chord currents. There are (n - 1) such equations, which can be written in matrix form

$$\underline{Q}\,\underline{I} = 0 \ , \qquad (3.2.7)$$

where $\underline{Q}$ is a rectangular matrix of (n - 1) rows and e columns, whose only entries are 0, +1, or -1. If the entries in $\underline{I}$ are ordered properly, this equation can be stated in the following partitioned form

$$\begin{bmatrix} \underline{U} & \underline{Q}_C \end{bmatrix} \begin{bmatrix} \underline{I}_B \\ \underline{I}_C \end{bmatrix} = 0 \qquad (3.2.8)$$

where $\underline{U}$ is an (n - 1) x (n - 1) unit matrix, $\underline{Q}_C$ is an (n - 1) x (e - n + 1) matrix containing only entries of 0, +1, or -1, $\underline{I}_B$ is the vector of branch currents, and $\underline{I}_C$ is the vector of chord currents. This partition is possible because of the basic property of the node equations set down above.

Consider now the (e - n + 1) equations summing voltages around closed paths in one-to-one correspondence with the edges in the chord set. Each such closed path involves one, and only one, edge which is in the chord set. All other edges traversed are members of the tree set. This is again, a consequence of the manner in which the tree (or co-tree) of the network was defined. Therefore, each of these equations serves to define a single chord voltage in terms of branch voltages.

The (e - n + 1) equations summing voltages to zero in matrix form can be written

$$\underline{P}\,\underline{E} = 0 \ , \qquad (3.2.9)$$

where $\underline{P}$ is a rectangular matrix of (e - n + 1) rows and (e) columns, whose only entries are 0, +1, or -1. If the entries in $\underline{E}$ are ordered properly, this equation can be stated in partitioned form as

$$\begin{bmatrix} \underline{P}_B & \underline{U} \end{bmatrix} \begin{bmatrix} \underline{E}_B \\ \underline{E}_C \end{bmatrix} = 0 \ , \qquad (3.2.10)$$

where $\underline{U}$ is a unit matrix of order (e - n + 1), $\underline{P}_B$ is a rectangular matrix (e - n + 1) x (n - 1) containing only entries of 0, +1, or -1, $\underline{E}_B$ is the vector of branch voltages, and $\underline{E}_C$ is the vector of chord voltages. At this point, a remarkable property of these matrices must be noted--the matrix $\underline{P}_B$ is the negative transpose of the matrix $\underline{Q}_C$

(14, Chapter 5). Thus, the two sets of equations summing currents or voltages to zero are not independent but are, in fact, completely dependent in that knowledge of one completely specifies the other. It now becomes apparent that either $\underline{P}_B$ or $\underline{Q}_C$ completely state all of the necessary mathematical information on the interconnections of the system of elements.

Returning to Equation 3.2.8, the indicated matrix multiplication can be carried out to obtain

$$\underline{I}_B + \underline{Q}_C \underline{I}_C = 0 \ , \qquad (3.2.11)$$

which can then be solved for $\underline{I}_B$ to yield

$$\underline{I}_B = -\underline{Q}_C \underline{I}_C \ . \qquad (3.2.12)$$

Similar operations can be performed on Equation 3.2.10 to obtain

$$\underline{E}_C = -\underline{P}_B \underline{E}_B \ . \qquad (3.2.13)$$

At this point, it is expeditious to rearrange the entries of the four vectors $\underline{E}_B$, $\underline{E}_C$, $\underline{I}_B$, and $\underline{I}_C$ (with, of course, accompanying row and column changes in $\underline{P}_B$ and $\underline{Q}_C$) so that the grouping according to type of elements contained in B1, B2, B3, C1, C2, and C3 is satisfied. When this has been performed, Equation 3.2.12 can be written in the form

$$\underline{I}_B = -\underline{S} \underline{I}_C \ , \qquad (3.2.14)$$

and similarly, Equation 3.2.13 becomes

$$\underline{E}_C = \underline{S}^T \underline{E}_B \ . \qquad (3.2.15)$$

The symbol $\underline{S}$ denotes the rearranged $\underline{Q}_C$ matrix, and $\underline{S}^T$ is the transpose

of $\underline{S}$.

The entries of the four vectors have been properly ordered so that Equation 3.2.14 can be written in the following completely partitioned form:

$$\begin{bmatrix} \underline{I}_{B1} \\ \underline{I}_{B2} \\ \underline{I}_{B3} \end{bmatrix} = - \begin{bmatrix} \underline{S}_{11} & \underline{S}_{12} & \underline{S}_{13} \\ \underline{S}_{21} & \underline{S}_{22} & \underline{S}_{23} \\ \underline{S}_{31} & \underline{S}_{32} & \underline{S}_{33} \end{bmatrix} \begin{bmatrix} \underline{I}_{C1} \\ \underline{I}_{C2} \\ \underline{I}_{C3} \end{bmatrix} . \qquad (3.2.16)$$

Similarly, Equation 3.2.15 becomes:

$$\begin{bmatrix} \underline{E}_{C1} \\ \underline{E}_{C2} \\ \underline{E}_{C3} \end{bmatrix} = \begin{bmatrix} \underline{S}_{11}^T & \underline{S}_{21}^T & \underline{S}_{31}^T \\ \underline{S}_{12}^T & \underline{S}_{22}^T & \underline{S}_{32}^T \\ \underline{S}_{13}^T & \underline{S}_{23}^T & \underline{S}_{33}^T \end{bmatrix} \begin{bmatrix} \underline{E}_{B1} \\ \underline{E}_{B2} \\ \underline{E}_{B3} \end{bmatrix} . \qquad (3.2.17)$$

From these two partitioned equations, six matrix equations can be written, of which a typical one would be

$$[\underline{I}_{B1}] = - [\underline{S}_{11} \quad \underline{S}_{12} \quad \underline{S}_{13}] \begin{bmatrix} \underline{I}_{C1} \\ \underline{I}_{C2} \\ \underline{I}_{C3} \end{bmatrix} . \qquad (3.2.18)$$

These six equations, which describe mathematically the interconnection of the system, together with the terminal relations expressed by Equations 3.2.1 through 3.2.6, can be manipulated to obtain a state-space model of the system.

Slightly rearranged, Equation 3.2.4 can be written

$$\underline{C}\,\dot{\underline{E}}_{B1} = \underline{I}_{B1} \quad . \tag{3.2.19}$$

Writing Equation 3.2.18 in a slightly more convenient form yields

$$\underline{I}_{B1} = -\underline{S}_{11}\,\underline{I}_{C1} - \underline{S}_{12}\,\underline{I}_{C2} - \underline{S}_{13}\,\underline{I}_{C3} \quad . \tag{3.2.20}$$

Substitution of Equation 3.2.20 into Equation 3.2.19 leads to

$$\underline{C}\,\dot{\underline{E}}_{B1} = -\underline{S}_{11}\,\underline{I}_{C1} - \underline{S}_{12}\,\underline{I}_{C2} - \underline{S}_{13}\,\underline{I}_{C3} \quad . \tag{3.2.21}$$

Using Equations 3.2.2 and 3.2.6 gives

$$\begin{aligned} \underline{C}\,\dot{\underline{E}}_{B1} = {} & -\underline{S}_{11}\,\underline{I}_{C1} \\ & -\underline{S}_{12}\,\underline{G}_{C}\quad \underline{E}_{C2} \\ & -\underline{S}_{13}\,\underline{I}\,(t) \quad . \end{aligned} \tag{3.2.22}$$

Rewriting the second line of Equation 3.2.17

$$\underline{E}_{C2} = \underline{S}_{12}^{T}\,\underline{E}_{B1} + \underline{S}_{22}^{T}\,\underline{E}_{B2} + \underline{S}_{32}^{T}\,\underline{E}_{B3} \quad . \tag{3.2.23}$$

Substituting this into Equation 3.2.22 yields

$$\begin{aligned} \underline{C}\,\dot{\underline{E}}_{B1} = {} & -\underline{S}_{11}\,\underline{I}_{C1} \\ & -\underline{S}_{12}\,\underline{G}_{C}\quad \underline{S}_{12}^{T}\,\underline{E}_{B1} \\ & -\underline{S}_{12}\,\underline{G}_{C}\quad \underline{S}_{22}^{T}\,\underline{E}_{B2} \\ & -\underline{S}_{12}\,\underline{G}_{C}\quad \underline{S}_{32}^{T}\,\underline{E}_{B3} \\ & -\underline{S}_{13}\,\underline{I}\,(t) \quad . \end{aligned} \tag{3.2.24}$$

Substitution of Equation 3.2.5 into Equation 3.2.24 gives

$$\underline{C}\,\dot{\underline{E}}_{B1} = -\underline{S}_{11}\,\underline{I}_{C1}$$

$$-\underline{S}_{12}\,\underline{G}_C\;\;\underline{S}_{12}^T\,\underline{E}_{B1}$$

$$-\underline{S}_{12}\,\underline{G}_C\;\;\underline{S}_{22}^T\,\underline{E}_{B2}$$

$$-\underline{S}_{12}\,\underline{G}_C\;\;\underline{S}_{32}^T\,\underline{V}\,(t)$$

$$-\underline{S}_{13}\,\underline{I}\,(t) \quad . \qquad (3.2.25)$$

The major problem remaining in obtaining an equation which will lead to the state-space form is the elimination of the variable $\underline{E}_{B2}$.

Writing Equation 3.2.1, for convenience,

$$\underline{E}_{B2} = \underline{R}_B\,\underline{I}_{B2} \quad . \qquad (3.2.26)$$

Rewriting the second row of Equation 3.2.16 yields

$$\underline{I}_{B2} = -\underline{S}_{21}\,\underline{I}_{C1} - \underline{S}_{22}\,\underline{I}_{C2} - \underline{S}_{23}\,\underline{I}_{C3} \quad . \qquad (3.2.27)$$

Substituting, Equation 3.2.26 becomes

$$\underline{E}_{B2} = -\underline{R}_B\,\underline{S}_{21}\,\underline{I}_{C1}$$

$$-\underline{R}_B\,\underline{S}_{22}\,\underline{I}_{C2}$$

$$-\underline{R}_B\,\underline{S}_{23}\,\underline{I}_{C3} \quad . \qquad (3.2.28)$$

Substitution of Equation 3.2.2 into this yields

$$\underline{E}_{B2} = -\underline{R}_B\,\underline{S}_{21}\,\underline{I}_{C1}$$

$$-\underline{R}_B\,\underline{S}_{22}\,\underline{G}_C\;\;\underline{E}_{C2}$$

$$-\underline{R}_B\,\underline{S}_{23}\,\underline{I}\,(t) \quad . \qquad (3.2.29)$$

Rewriting the second row of Equation 3.2.17 in the form of Equation 3.2.23 gives

$$\underline{E}_{C2} = \underline{S}_{12}^{T} \underline{E}_{B1} + \underline{S}_{22}^{T} \underline{E}_{B2} + \underline{S}_{32}^{T} \underline{E}_{B3} \quad . \tag{3.2.30}$$

Substituting Equation 3.2.30 into Equation 3.2.29 leads to

$$\begin{aligned} \underline{E}_{B2} = & -\underline{R}_{B} \underline{S}_{21} \underline{I}_{C1} \\ & -\underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \quad \underline{S}_{12}^{T} \underline{E}_{B1} \\ & -\underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \quad \underline{S}_{22}^{T} \underline{E}_{B2} \\ & -\underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \quad \underline{S}_{32}^{T} \underline{E}_{B3} \\ & -\underline{R}_{B} \underline{S}_{23} \underline{I}(t) \quad . \end{aligned} \tag{3.2.31}$$

Substituting Equation 3.2.5 into Equation 3.2.31 and collecting the terms in $\underline{E}_{B2}$ on the left side results in

$$\begin{aligned} \underline{E}_{B2} + \underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \underline{S}_{22}^{T} \underline{E}_{B2} = & \\ & -\underline{R}_{B} \underline{S}_{21} \underline{I}_{C1} \\ & -\underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \quad \underline{S}_{12}^{T} \underline{E}_{B1} \\ & -\underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \quad \underline{S}_{32}^{T} \underline{V}(t) \\ & -\underline{R}_{B} \underline{S}_{23} \underline{I}(t) \quad . \end{aligned} \tag{3.2.32}$$

The expression premultiplying $\underline{E}_{B2}$ can be written as

$$\underline{R}_{1} = \underline{U} + \underline{R}_{B} \underline{S}_{22} \underline{G}_{C} \underline{S}_{22}^{T} \quad . \tag{3.2.33}$$

Taking the inverse of $\underline{R}_{1}$ (see Appendix A for proof of existence of this

inverse) and calling the inverse $\underline{R}_2$, Equation 3.2.32 can be written in the form

$$\begin{aligned}\underline{E}_{B2} = &-\underline{R}_2\ \underline{R}_B\ \underline{S}_{21}\ \underline{I}_{C1} \\ &-\underline{R}_2\ \underline{R}_B\ \underline{S}_{22}\ \underline{G}_C\ \ \underline{S}_{12}^T\ \underline{E}_{B1} \\ &-\underline{R}_2\ \underline{R}_B\ \underline{S}_{22}\ \underline{G}_C\ \ \underline{S}_{32}^T\ \underline{V}\ (t) \\ &-\underline{R}_2\ \underline{R}_B\ \underline{S}_{23}\ \underline{I}\ (t) \quad .\end{aligned} \tag{3.2.34}$$

Returning now to Equation 3.2.25, substitution of Equation 3.2.34 yields

$$\begin{aligned}\underline{C}\ \dot{\underline{E}}_{B1} = &-\underline{S}_{11}\ \underline{I}_{C1} \\ &-\underline{S}_{12}\ \underline{G}_C\ \ \underline{S}_{12}^T\ \underline{E}_{B1} \\ &+\underline{S}_{12}\ \underline{G}_C\ \ \underline{S}_{22}^T\ \underline{R}_2\ \ \underline{R}_B\ \underline{S}_{21}\ \underline{I}_{C1} \\ &+\underline{S}_{12}\ \underline{G}_C\ \ \underline{S}_{22}^T\ \underline{R}_2\ \ \underline{R}_B\ \underline{S}_{22}\ \underline{G}_C\ \ \underline{S}_{12}^T\ \underline{E}_{B1} \\ &+\underline{S}_{12}\ \underline{G}_C\ \ \underline{S}_{22}^T\ \underline{R}_2\ \ \underline{R}_B\ \underline{S}_{22}\ \underline{G}_C\ \ \underline{S}_{32}^T\ \underline{V}\ (t) \\ &+\underline{S}_{12}\ \underline{G}_C\ \ \underline{S}_{22}^T\ \underline{R}_2\ \ \underline{R}_B\ \underline{S}_{23}\ \underline{I}\ (t) \\ &-\underline{S}_{12}\ \underline{G}_C\ \ \underline{S}_{32}^T\ \underline{V}\ (t) \\ &-S_{13}\ I\ (t) \quad .\end{aligned} \tag{3.2.35}$$

At this time it is worthwhile to call attention to the character of the variables which appear on the right-hand side of Equation 3.2.35. Note that only four vector variables appear. Of these, two are the drivers of the system, $\underline{I}\ (t)$ and $\underline{V}\ (t)$. The other two are the current vector of

the inductive elements in the chord set, $\underline{I}_{C1}$, and the voltage vector of the capacitive elements in the tree set, $\underline{E}_{B1}$. Note that the derivative of $\underline{E}_{B1}$ appears on the left side of Equation 3.2.35 so that this equation is beginning to take the form of the state-space equations.

Returning to the terminal equations of the elements, Equation 3.2.3 can be written in the form

$$\underline{L}\,\dot{\underline{I}}_{C1} = \underline{E}_{C1} \qquad (3.2.36)$$

By a process of substitution and elimination, which is exactly parallel to the one just completed (which will not be carried out here), an equation in the same form as Equation 3.2.35 can be obtained. It is

$$\begin{aligned}\underline{L}\,\dot{\underline{I}}_{C1} = {} & +\underline{S}_{11}^{T}\,\underline{E}_{B1} \\ & -\underline{S}_{21}^{T}\,\underline{R}_{2}\quad\underline{R}_{B}\,\underline{S}_{21}\,\underline{I}_{C1} \\ & -\underline{S}_{21}^{T}\,\underline{R}_{2}\quad\underline{R}_{B}\,\underline{S}_{22}\,\underline{G}_{C}\quad\underline{S}_{12}^{T}\,\underline{E}_{B1} \\ & -\underline{S}_{21}^{T}\,\underline{R}_{2}\quad\underline{R}_{B}\,\underline{S}_{22}\,\underline{G}_{C}\quad\underline{S}_{32}^{T}\,\underline{V}\,(t) \\ & -\underline{S}_{21}^{T}\,\underline{R}_{2}\quad\underline{R}_{B}\,\underline{S}_{23}\,\underline{I}\,(t) \\ & +\underline{S}_{31}^{T}\,\underline{V}\,(t) \qquad . \end{aligned} \qquad (3.2.37)$$

Note that here, too, the only variables on the right side of the equation are the drivers, $\underline{I}\,(t)$ and $\underline{V}\,(t)$, and the variables, $\underline{E}_{B1}$ and $\underline{I}_{C1}$, as in the case of Equation 3.2.35.

Since the only differential equations of the system were stated in Equations 3.2.3 and 3.2.4, and Equations 3.2.35 and 3.2.37 are versions of these equations, these two matrix equations are the complete

set of differential equations describing the system. Since all variables on the right side of both equations are the same, they may be adjoined into a single matrix equation. In this equation, the variables on the left side are $\underline{E}_{B1}$ and $\underline{I}_{C1}$; and they appear only as the first derivative with respect to time. On the right side, the variables are $\underline{E}_{B1}$ and $\underline{I}_{C1}$ and the drivers of the system, $\underline{I}\,(t)$ and $\underline{V}\,(t)$. This adjoined set of equations does, in fact, become a set of differential equations in normal form and are the differential equations of the state-space model.

In Equation 3.2.23, part of the set of algebraic equations which are a part of the state-space model was obtained. The remaining set is developed in the process of elimination leading to Equation 3.2.37. These equations are

$$\begin{aligned}
\underline{I}_{C2} = {} & +\underline{G}_C\, \underline{S}_{12}^T\, \underline{E}_{B1} \\
& -\underline{G}_C\, \underline{S}_{22}^T\, \underline{R}_2 \quad \underline{R}_B\, \underline{S}_{21}\, \underline{I}_{C1} \\
& -\underline{G}_C\, \underline{S}_{22}^T\, \underline{R}_2 \quad \underline{R}_B\, \underline{S}_{22}\, \underline{G}_C \quad \underline{S}_{12}^T\, \underline{E}_{B1} \\
& -\underline{G}_C\, \underline{S}_{22}^T\, \underline{R}_2 \quad \underline{R}_B\, \underline{S}_{22}\, \underline{G}_C \quad \underline{S}_{32}^T\, \underline{V}\,(t) \\
& -\underline{G}_C\, \underline{S}_{22}^T\, \underline{R}_2 \quad \underline{R}_B\, \underline{S}_{23}\, \underline{I}\,(t) \\
& +\underline{G}_C\, \underline{S}_{32}^T\, V\,(t) \qquad\qquad (3.2.38)
\end{aligned}$$

Since again the variables on the right of Equations 3.2.34 and 3.2.38 are the same, $\underline{E}_{B1}$, $\underline{I}_{C1}$, $\underline{V}\,(t)$, and $\underline{I}\,(t)$, these two sets can be adjoined into a single matrix equation, which becomes the complete set of algebraics which are a part of the state-space model.

The equations would be far too cumbersome to be meaningful if written out in the form in which they have been developed to this point, so this will not be attempted. It is sufficient that it has been possible to develop the state-space equations, both differential and algebraic, by matrix manipulations of the original terminal equations and the linear graph equations which represent the interconnections.

3.3 Adaptation of the Formulation Algorithm for Computer Programming. The state-space equations as obtained in the form of Equations 3.2.34, 3.2.35, 3.2.37, and 3.2.38 are not at all well adapted for implementation by computer programming, particularly in the case of a limited-size computer. The arithmetic operations indicated are lengthy and require random access to portions of the $\underline{S}$ matrix. It is expedient to introduce some transformations and operations which will reduce this complexity. At the same time, a final form of the four equations is obtained which places in evidence the true state-space form of the equations.

The approach selected is to search out repeated combinations of two matrices in the equations and to redefine these combinations as another single matrix. As a start, it is apparent in the equations that the following combinations appear frequently:

$$\underline{R} = \underline{R}_2 \, \underline{R}_B \tag{3.3.1}$$

$$\underline{1} = \underline{G}_C \, \underline{S}_{12}^T \tag{3.3.2}$$

$$\underline{2} = \underline{G}_C \, \underline{S}_{22}^T \tag{3.3.3}$$

$$\underline{3} = \underline{G}_C \, \underline{S}_{32}^T \qquad (3.3.4)$$

Many other combinations could be selected, but it is most useful for the present to make step-by-step substitutions and redefine useful combinations as they appear, rather than making many substitutions at the beginning. If Equations 3.3.1 through 3.3.4 are substituted wherever possible, the modified equations become

$$\begin{aligned}
\underline{L}\,\dot{\underline{I}}_{C1} = {} & +\underline{S}_{11}^T \, \underline{E}_{B1} \\
& -\underline{S}_{21}^T \, \underline{R} \quad \underline{S}_{21} \, \underline{I}_{C1} \\
& -\underline{S}_{21}^T \, \underline{R} \quad \underline{S}_{22} \, \underline{1} \quad \underline{E}_{B1} \\
& -\underline{S}_{21}^T \, \underline{R} \quad \underline{S}_{22} \, \underline{2} \quad \underline{V}(t) \\
& -\underline{S}_{21}^T \, \underline{R} \quad \underline{S}_{23} \, \underline{I}(t) \\
& +\underline{S}_{31}^T \, \underline{V}(t) \,.
\end{aligned} \qquad (3.3.5)$$

$$\begin{aligned}
\underline{C}\,\dot{\underline{E}}_{B1} = {} & -\underline{S}_{11} \, \underline{I}_{C1} \\
& -\underline{S}_{12} \, \underline{1} \quad \underline{E}_{B1} \\
& +\underline{S}_{12} \, \underline{2} \quad \underline{R} \quad \underline{S}_{21} \, \underline{I}_{C1} \\
& +\underline{S}_{12} \, \underline{2} \quad \underline{R} \quad \underline{S}_{22} \, \underline{1} \quad \underline{E}_{B1} \\
& +\underline{S}_{12} \, \underline{2} \quad \underline{R} \quad \underline{S}_{22} \, \underline{3} \quad \underline{V}(t) \\
& +\underline{S}_{12} \, \underline{2} \quad \underline{R} \quad \underline{S}_{23} \, \underline{I}(t) \\
& -\underline{S}_{12} \, \underline{3} \quad \underline{V}(t) \\
& -\underline{S}_{13} \, \underline{I}(t) \,.
\end{aligned} \qquad (3.3.6)$$

$$\underline{E}_{B2} = -\underline{R}\,\underline{S}_{21}\,\underline{I}_{C1}$$

$$-\underline{R}\,\underline{S}_{22}\,\underline{1}\quad\underline{E}_{B1}$$

$$-\underline{R}\,\underline{S}_{22}\,\underline{3}\quad\underline{V}(t)$$

$$-\underline{R}\,\underline{S}_{23}\,\underline{I}(t) \quad . \qquad (3.3.7)$$

$$\underline{I}_{C2} = +\underline{1}\,\underline{E}_{B1}$$

$$-\underline{2}\,\underline{R}\quad\underline{S}_{21}\,\underline{I}_{C1}$$

$$-\underline{2}\,\underline{R}\quad\underline{S}_{22}\,\underline{1}\quad\underline{E}_{B1}$$

$$-\underline{2}\,\underline{R}\quad\underline{S}_{22}\,\underline{3}\quad\underline{V}(t)$$

$$-\underline{2}\,\underline{R}\quad\underline{S}_{23}\,\underline{I}(t)$$

$$+\underline{3}\,V(t) \quad . \qquad (3.3.8)$$

Recalling also the definition of the matrix $\underline{R}_2$ as the inverse of the matrix $\underline{R}_1$, which is defined

$$\underline{R}_1 = \underline{U} + \underline{S}_{22}\,\underline{G}_C\,\underline{S}_{22}^T \quad ,$$

this can be defined in terms of the matrix combinations as

$$\underline{R}_1 = \underline{U} + \underline{S}_{22}\,\underline{2} \quad . \qquad (3.3.9)$$

At this point, define six new combinations:

$$\underline{4} = \underline{S}_{12}\,\underline{1} \qquad \underline{6} = \underline{S}_{12}\,\underline{3}$$

$$\underline{5} = \underline{S}_{12}\,\underline{2} \qquad \underline{7} = \underline{S}_{22}\,\underline{1}$$

$$\underline{8} = \underline{S}_{22}\,\underline{2} \qquad \underline{9} = \underline{S}_{22}\,\underline{3} \quad . \tag{3.3.10}$$

Substitution of these into the Equations 3.3.5 through 3.3.8 is routine and will not be presented here. When this is performed, it becomes expedient to define four new combinations which appear frequently:

$$\underline{14} = \underline{R}\,\underline{S}_{21} \qquad \underline{15} = \underline{R}\,\underline{7}$$

$$\underline{16} = \underline{R}\,\underline{9} \qquad \underline{17} = \underline{R}\,\underline{S}_{23} \quad . \tag{3.3.11}$$

Following this substitution, further simplification requires the definition of a set of twelve matrices:

$$\underline{18} = \underline{S}_{21}^{T}\,\underline{14} \qquad \underline{22} = \underline{5}\,\underline{14} \qquad \underline{26} = \underline{2}\,\underline{14}$$

$$\underline{19} = \underline{S}_{21}^{T}\,\underline{15} \qquad \underline{23} = \underline{5}\,\underline{15} \qquad \underline{27} = \underline{2}\,\underline{15}$$

$$\underline{20} = \underline{S}_{21}^{T}\,\underline{16} \qquad \underline{24} = \underline{5}\,\underline{16} \qquad \underline{28} = \underline{2}\,\underline{16}$$

$$\underline{21} = \underline{S}_{21}^{T}\,\underline{17} \qquad \underline{25} = \underline{5}\,\underline{17} \qquad \underline{29} = \underline{2}\,\underline{17} \quad . \tag{3.3.12}$$

Substitution of these combinations into the equations allows the definition of one final group of matrices which become the actual coefficients of the equations in their simplest form. These matrices are:

$$\underline{34} = -\,\underline{18} \qquad \underline{38} = \underline{22} - \underline{S}_{11}$$

$$\underline{35} = -\,\underline{19} + \underline{S}_{11}^{T} \qquad \underline{39} = \underline{23} - \underline{4}$$

$$\underline{36} = -\,\underline{20} + \underline{S}_{31}^{T} \qquad \underline{40} = \underline{24} - \underline{6}$$

$$\underline{37} = -\,\underline{21} \qquad \underline{41} = \underline{25} - \underline{S}_{13}$$

$$\underline{42} = -\underline{14} \qquad \underline{46} = -\underline{26}$$

$$\underline{43} = -\underline{15} \qquad \underline{47} = -\underline{27} + \underline{1}$$

$$\underline{44} = -\underline{16} \qquad \underline{48} = -\underline{28} + \underline{3}$$

$$\underline{45} = -\underline{17} \qquad \underline{49} = -\underline{29} \quad . \qquad (3.3.13)$$

The Equations 3.3.5 through 3.3.8 now become, in their simpler form,

$$\underline{L}\,\dot{\underline{I}}_{C1} = \underline{34}\,\underline{I}_{C1} + \underline{35}\,\underline{E}_{B1} + \underline{36}\,\underline{V}(t) + \underline{37}\,\underline{I}(t) \ ,$$

$$\underline{C}\,\dot{\underline{E}}_{B1} = \underline{38}\,\underline{I}_{C1} + \underline{39}\,\underline{E}_{B1} + \underline{40}\,\underline{V}(t) + \underline{41}\,\underline{I}(t) \ ,$$

$$\underline{E}_{B2} = \underline{42}\,\underline{I}_{C1} + \underline{43}\,\underline{E}_{B1} + \underline{44}\,\underline{V}(t) + \underline{45}\,\underline{I}(t) \ ,$$

$$\underline{I}_{C2} = \underline{46}\,\underline{I}_{C1} + \underline{47}\,\underline{E}_{B1} + \underline{48}\,\underline{V}(t) + \underline{49}\,\underline{I}(t) \ . \qquad (3.3.14)$$

The variables on the right of these equations are all the same, and similarly ordered, so that the equations can be adjoined into two matrix equations, which are the equations of the state-space model. The differential equations of the state model thus become

$$\begin{bmatrix} L & 0 \\ 0 & C \end{bmatrix} \begin{bmatrix} \dot{I}_{C1} \\ \dot{E}_{B1} \end{bmatrix} = \begin{bmatrix} 34 & 35 \\ 38 & 39 \end{bmatrix} \begin{bmatrix} I_{C1} \\ E_{B1} \end{bmatrix} + \begin{bmatrix} 36 & 37 \\ 40 & 41 \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} \qquad (3.3.15)$$

and the algebraic equation is

$$\begin{bmatrix} E_{B2} \\ I_{C2} \end{bmatrix} = \begin{bmatrix} 42 & 43 \\ 46 & 47 \end{bmatrix} \begin{bmatrix} I_{C1} \\ E_{B1} \end{bmatrix} + \begin{bmatrix} 44 & 45 \\ 48 & 49 \end{bmatrix} \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} \ . \qquad (3.3.16)$$

If the inverse of the LC matrix on the left of Equation 3.3.15 is obtained, the final form of the equations in normal form can be developed by premultiplication of the two matrices on the right by this inverse. In the most general system this inverse must be obtained by regular inversion routines. An almost trivial situation pertains for electrical circuits when there are no mutual inductances or capacitances. For this situation, these matrices are diagonal, and the inverse is trivial. A similar situation may arise for other energy systems.

It is worthwhile, at this point, to draw attention to a characteristic of the matrices 34 through 49 of Equations 3.3.15 and 3.3.16 which will be of value later. Review of the procedures utilized to generate these matrices reveals that the value of their entries depends only upon the entries of the interconnection matrix S and the resistance values in the circuit. Therefore, the matrices on the right of Equations 3.3.15 and 3.3.16 remain fixed if the resistance values and topology of the network remain unchanged. This characteristic is used to advantage in the implementation of the computer program. Values of the capacitors and inductors and specifications of the voltage and current drivers can be changed without the necessity for regeneration of the model. However, if any resistor, or the topology, is changed, it is necessary to regenerate the entire model.

The system of equations defined in Equations 3.3.15 and 3.3.16 can be solved to obtain what can be termed a "complete" solution for the system. In this process, a solution for either the across or through variable of every element in the network is obtained. The through variable (current) will be defined for every element in the co-tree and the across variable (voltage) for every element in the tree.

However, frequently a particular variable of interest is not one of this set. It is sometimes possible to select a suitable tree and co-tree which will solve this dilemma. For instance, if the voltage across a particular resistor is of interest, it should be placed in the tree if this is possible. However, this technique is not generally applicable. For instance, if the current through a particular capacitor is of interest, a real dilemma is apparent since a capacitor must always be placed in the tree, and the voltage across its terminals is the direct variable in the state-space equations.

In order to obtain a solution for the "complementary" variables to those defined in the state-space model, it is possible to return to the terminal equations of the elements, Equations 3.2.1 through 3.2.4. This has several disadvantages. First, Equations 3.2.3 and 3.2.4 for the storage elements are differential equations. Secondly, this does not provide a complete answer as these equations do not define the currents for voltage drivers or the voltages for current drivers.

There is a desirable alternative to this process, defined by Equations 3.2.14 and 3.2.15, which relate chord and branch through and across variables in terms of the $\underline{S}$ matrix. The solution for the state-variables defined by Equations 3.3.15 and 3.3.16, together with the known values of the voltage and current drivers, provide a complete solution for all chord currents and branch voltages. Application of Equations 3.2.14 and 3.2.15 provides a completely algebraic method to obtain a total solution for all variables in the network. Recalling that all entries in the $\underline{S}$ matrix are 0, +1, or -1, this obviously is a simple and straightforward process. For this reason, the computer program provides the $\underline{S}$ matrix during the solution process so that a

total solution to the system can be obtained.

3.4 The Determination of Initial Conditions. In order to obtain a definite solution to the differential equation of Equation 3.3.15, it is necessary to specify a complete set of initial conditions. For this equation, the initial values of I_{C1} and E_{B1} and the t = 0 value of all drivers are required. These values will completely define all quantities on the right side of Equation 3.3.15 so that the derivatives of the differential variables can be calculated. In the case of an electrical circuit, it is necessary to know the initial voltages of all capacitors and voltage drivers and the initial currents of all inductors and current drivers.

However, in the case of many electrical circuits of interest, this manner of specification of initial conditions is inconvenient or impossible. A typical example is encountered in the analysis of most electronic circuits. For these circuits, it is usually possible to specify only the value of all power supplies (which are elements in $\underline{V}(t)$ and $\underline{I}(t)$) and signal sources at t = 0. The assumption is then made that the circuit has achieved steady-state with these drivers applied. From these specifications, the analysis program should generate the steady-state solution for all variables in the network. The capacitor voltages and inductor currents from this solution then provide the normal set of initial conditions for the differential equations. From these initial conditions, the time-solution to the system can proceed.

With the knowledge that the network is in steady-state, Equation 3.3.15 can be used to obtain the complete solution. The specification

of steady-state implies that the derivatives of all variables must be equal to zero. The only derivatives of consequence are those defined by Equation 3.3.15, since all others are linear combinations of these. Equating the right-hand side of Equation 3.3.15 to zero leads to

$$\left[\begin{array}{c|c} 34 & 35 \\ \hline 38 & 39 \end{array}\right] \left[\begin{array}{c} I_{Clo} \\ \hline E_{Blo} \end{array}\right] + \left[\begin{array}{c|c} 36 & 37 \\ \hline 40 & 41 \end{array}\right] \left[\begin{array}{c} V(t)_o \\ \hline I(t)_o \end{array}\right] = 0 \quad , \qquad (3.4.1)$$

where the o subscript is used to denote that these are the steady-state $t = 0$ values.

The inverse of the matrix composed of submatrices 34, 35, 38, 39 must exist; since it has previously been established that a solution to the system exists. Therefore, Equation 3.4.1 can be written in the following form:

$$\left[\begin{array}{c} I_{Clo} \\ \hline E_{Blo} \end{array}\right] = -\left[\begin{array}{c|c} 34 & 35 \\ \hline 38 & 39 \end{array}\right]^{-1} \left[\begin{array}{c|c} 36 & 37 \\ \hline 40 & 41 \end{array}\right] \left[\begin{array}{c} V(t)_o \\ \hline I(t)_1 \end{array}\right] \quad . \qquad (3.4.2)$$

Solution of this equation yields the steady-state value of the inductor currents and capacitor voltages from the known values of the voltage and current drivers.

The formulation program provides the matrix on the right of Equation 3.4.2 for use when this method for determination of initial conditions is required.

CHAPTER IV

ADAPTATION OF THE ALGORITHM FOR SYSTEMS CONTAINING NONLINEAR ELEMENTS

The algorithm for formulation of the state-space equations, as outlined in Chapter III, is based on a linear system model. Cognizance of this must be maintained in any attempt to apply the algorithm to the solution of a system containing nonlinear elements. Two basically different ways to approach this problem seem worthwhile.

The first method consists of representation of the nonlinear system by a sequence of piecewise linear systems. Step by step, as the solution advances, the coefficients of the system equations are re-calculated to conform to the nonlinear constraints. Such a procedure assumes that between steps the system is linear and the coefficients remain constant. Therefore, it is necessary that step sizes be kept small enough that this assumption is not violated.

A second method consists of representation of the nonlinear elements by voltage or current drivers which are functions of variables in the system. If all other elements in the system are linear, the coefficients of the state-space equations, as formulated by the algorithm, remain constant. Such a procedure, however, involves more than a simple step-by-step evaluation of the dependent driver. Any change in the value of this driver is propagated throughout the entire network and influences the value of all variables--including the one (or more) upon which the

driver is dependent. It is, therefore, necessary to utilize an iteration process at each step in the solution to arrive at a consistent value for the dependent driver. The success of this method hinges entirely on the convergence of the iteration process.

A more useful analysis scheme would include both of these methods so that the broadest possible class of nonlinear elements could be treated. For this study, a combination of both methods has been explored.

The procedures involved in the implementation of these analysis methods are relatively straightforward. However, in actual use they involve a vast amount of repetitive numerical evaluation. For this reason, they are completely unsuitable for desk calculation for anything but the very simplest networks. However, the procedures involved are very methodical so that their implementation by computer programming is simplified. This has been the aim of this work--to develop methods suitable for computer programming with no consideration given to hand calculation.

4.1 Background. The number of computer analysis programs for nonlinear systems has been relatively limited, as noted in earlier chapters. Undoubtedly, many people have performed analyses of specific nonlinear systems of many types. However, such work has not been published to any extent so that little can be learned from these efforts. They also tend to be very specialized for a given type of nonlinear element.

The only published nonlinear analysis program of general nature appears to be the General Motors DYANA program. In this program the models generated are not in state-space equation form, and linear graph

concepts are not used. However, the method is based on piecewise linear analysis in which the coefficients of the system are recalculated at each step to conform to the nonlinear constraints. In this respect, the DYANA program lends support to the piecewise-linear method of analysis.

The other well-known analysis programs--PETAP and NET--are both based on the same analytic procedures. In these programs, the system is separated into two parts. Each part is represented by a set of differential equations--one linear, the other nonlinear. Each set is integrated separately. The solution procedure alternates between them so that, in effect, the solution for one provides initial conditions and forcing functions for the solution of the other. No iteration processes are involved, but it is necessary to make many small steps in the integration of the nonlinear system for one large step in integration of the linear system. This process, in effect, substitutes for the iteration process. Because of these methods, the PETAP and NET1 programs are quite different from the methods proposed for this study.

However, it is felt that the solution methods of PETAP and NET1 do support the general technique of the use of driving functions which are functions of variables in the system, rather than time alone. Also, both programs utilize iteration methods of the general type proposed for this study for evaluation of the initial conditions in the network.

4.2 Inclusion of Nonlinear Elements in the State-Space Model. The incorporation of nonlinear storage elements--the inductors and capacitors in an electrical network--follows the first method outlined in the preface to this chapter. The process of recalculation of the coefficients

of the system equations is made quite simple because of the form of Equation 3.3.15.

The parameters associated with all storage elements appear as entries in the LC matrix premultiplying the left side of Equation 3.3.15. In the case of a system with only linear storage elements, the entries in this matrix are constant. In this case, the matrix inverse can be developed once at the beginning of the solution process and will remain unchanged.

In the case of nonlinear storage elements, the entries in the LC matrix which correspond to these elements will change. It is then necessary to obtain the inverse of this matrix for each step in the solution process.

In either the linear or nonlinear case, if no mutual inductances or capacities are involved, this is a trivial matter, since the matrix is diagonal. In the more general case, it will be necessary to obtain a full inverse.

The model, as presented in Equations 3.3.15 and 3.3.16, is particularly adapted to inclusion of nonlinear storage elements since the matrices on the right of these equations remain unchanged.

The inclusion of nonlinear resistances is not so straightforward as the inclusion of storage elements. Because of the way the model is formulated, a change in the value of a resistor will require complete recalculation of the matrices on the right of Equations 3.3.15 and 3.3.16. This is a lengthy process, since it involves going back through the entire formulation procedure at every step in the solution.

For this reason, no attempt was made to include nonlinear resistances directly into the system equations by direct recalculation

of coefficients. Instead, these elements were included by use of the second method outlined in the preface to this chapter. Each nonlinear resistor is represented by a dependent voltage or current driver. The choice of representation is not restricted, and either may be used.

The drivers representing nonlinear resistances become a part of the vector of drivers symbolized by $\underline{V}(t)$ or $\underline{I}(t)$. It is necessary to implement an accompanying iteration process, as outlined previously. Topologically, these drivers are treated like any other drivers. The evaluation of the driver is carried out by a nonlinear "side-equation" which is a polynomial function of fourth degree.

The inclusion of active nonlinear elements, such as tubes or transistors, is little different from the procedure for nonlinear resistances. They are represented in the circuit by dependent drivers and are treated the same as those for nonlinear resistances.

In addition to the polynomial side-equation which is generalized to allow any variable to be the controlling variable, other models are included. For the representation of a vacuum tube, it is necessary to include a driver which is a function of two other variables in the network, rather than just one. A transistor is treated in much the same way, except that its model is comprised of two dependent drivers, each of which is a separate function of two variables. A model is also included for the semiconductor diode, which is an exponential function of one variable. These models are all represented by side equations and require the use of suitable iteration processes for the determination of a consistent solution.

In summary, then, nonlinear elements are included in two ways:

(a) nonlinear storage elements are represented on a piecewise linear basis by step-by-step changes in equation coefficients, and (b) nonlinear resistors and active elements are represented by dependent drivers which require a suitable iteration process to obtain a consistent solution.

4.3 Model Specifications for Selected Elements. Having selected a basic method of representation for the elements of interest, it was necessary to determine specific equations for each one. Some guidance was provided by the ideal models derived on theoretical grounds. It was discovered that most such models are, in fact, poor representations of the elements for large-signal (global) conditions. Models which have served quite adequately for all small-signal (local) representations may fail completely when subjected to a wide range of variation of the terminal variables.

The representation of an iron-core inductor provides a good example. It is common to base the representation of such a device on the B-H curve, which is directly related to the flux-current curve. Polynomial approximation of this curve by curve-fitting with the least-squares error criterion is commonly employed. In some cases, it is useful to utilize fractional-power representations or an inverse polynomial (i.e., fit the H-B curve).

However, for the case at hand, it is necessary to obtain a functional relationship between the value of inductance, L, and the current, not between flux and current. There is a direct relationship involved, evidenced by the following:

$$e_L = \frac{d\lambda}{dt} = \frac{d\lambda}{di}\,\frac{di}{dt} = L\,\frac{di}{dt} \quad ,$$

so that

$$L = \frac{d\lambda}{di} \quad . \tag{4.3.1}$$

It would seem that if a functional approximation for the flux-current curve (λ versus i) could be obtained that it would be possible to perform an analytic differentiation with respect to i and obtain a suitable representation of the inductance-current relationship.

A number of trial representations were attempted using the least-squares criterion for various polynomials, inverse polynomials, and fractional power series for the B-H curve of a common magnetic core material. From these models the coefficients were substituted into the corresponding equation for the inductance-current relationship. The results were completely unsatisfactory and yielded very poor representations of known L-i relationships.

Several alternate functional relationships were tested, and one was encountered which provides a realistic evaluation of the L-i curve from curve-fitting applied to the B-H curve.

The equation of the useful model is of the form

$$\lambda = A \tan^{-1} \left(\frac{i}{C} \right) + B \left[\tan^{-1} \left(\frac{i}{C} \right) \right]^2 \tag{4.3.2}$$

The constant C must be selected by prior inspection of the flux-current curve, since it is not possible to determine the argument of a transcendental function by least-squares criterion. However, once C is selected, the values of A and B can be determined by curve-fitting.

The choice of C by inspection is not difficult. Since the arctangent approaches $\pi/2$ as the argument approaches infinity, $\pi/2$ (scaled) represents the saturation condition. For argument of 1, the

arctangent has value $\pi/4$ or 1/2 the saturation value. Thus, it is only necessary to pick C so that (i/C) = 1 for the current required to produce 1/2 saturation flux. In those materials where complete saturation does not occur, it is suitable to select the value of current where downward inflection of the flux-current curve begins, which indicates the start of partial saturation.

Primary interest still lies in the L-i curve, which can be deduced by differentiation of Equation 4.3.2:

$$L = \frac{d\lambda}{di} = \frac{C}{i^2 + C^2} \left[A + 2B \tan^{-1} \left(\frac{i}{C} \right)\right] . \qquad (4.3.3)$$

Values of A, B, and C obtained for Equation 4.3.2 and then used in Equation 4.3.3 provide a very suitable representation of the L-i relationship. Appendix B contains an illustrative example and sample curves.

A second example of the failure of a theoretical model under large-signal conditions was encountered in treating the vacuum triode. The ideal model is:

$$i_b = K \left[e_b + \mu e_c\right]^n \qquad (4.3.4)$$

where K, μ, and n are constants of the model. On theoretical grounds, the value of n is 1.5 and should remain constant. However, it is well known that the value of n may lie anywhere in the range of 1.2 - 1.8 for practical devices.

This model of the vacuum triode has served for many years as a sound and useful basis for small-signal analysis of electronic circuits. However, an exhaustive investigation of this equation revealed that it failed completely to provide a realistic representation of the

large-signal characteristics of the device. It became apparent that the exponent, n, is not a constant but is some obscure function of both e_c and e_b. No way was found to determine what this functional relationship was or might be.

A search for an acceptable representation revealed that a two-variable power series, including all powers and cross-products up to the third degree in each variable, was adequate. The constant term was deliberately omitted from the series so that the plate current would be zero when both e_c and e_b were zero. This model lends itself very directly to least-squares curve-fitting, and the results are excellent over the entire operating range of the device when an adequate number (around 150) of data points are used.

Some difficulty was experienced with this model when attempts were made to extend it to represent the region of positive grid voltage. In this region, for low values of plate voltage, the curves of constant grid voltage are concave downward. For all other regions, the curves are concave upwards. It was found that this situation could be accounted for by multiplying the entire series representation by a factor of the form

$$i_b = [\text{power series}] \frac{e_b}{e_b + E} \qquad (4.3.5)$$

where E is a constant selected by inspection prior to the curve-fitting process. The choice of E is somewhat influenced by knowledge of the expected region of operation. Several trial-and-error attempts were usually necessary before an acceptable result was obtained.

Appendix C contains more information on this model and several examples of the large-signal characteristics that it yields.

One additional nonlinear model of considerable interest was incorporated. A piecewise linear model consisting of two straight lines forced to pass through a specified common point was included. The linear relationship used depends upon whether the argument variable is greater or less than common point value. This two-part piecewise linear model was included for preliminary test of the concept of piecewise linear representations in analysis procedures of this kind. It is possible that multipart, piecewise linear models can be extremely useful in representing devices for which other representations cannot be found. With this idea in mind, the simplest form was included for evaluation.

CHAPTER V

PROGRAM IMPLEMENTATION OF FORMULATION ALGORITHM AND NONLINEAR ANALYSIS

In the interest of widest possible distribution and application, and the easiest understanding of the programming methods, it was decided that the entire program should be implemented in the FORTRAN IV language. By this means, none of the techniques or developments would be restricted to any one machine, and the programming time and effort could be greatly reduced. The price paid is in terms of operating efficiency and efficient machine utilization. With the exception of the work by Reid at Michigan State (8), all other programs have been implemented in symbolic or machine language and distribution and use of the ideas and techniques by others is thus seriously restricted.

In the interest of handling at least a moderately complicated system (electrical circuit), it was decided to write the program so that sixty elements could be included. Ten elements of each kind as described in the six Equations 3.2.1 through 3.2.6 can be included in the circuit. For an electrical circuit, ten inductors, ten capacitors, ten resistors in the tree, ten resistors in the co-tree (thus twenty resistors, total), ten voltage drivers, and ten current drivers are allowed. The nonlinear, dependent storage elements are counted as part of these categories; and the nonlinear drivers are included as part of the vector of voltage and/or current drivers.

The restrictions as to the number of elements were determined by the available storage in the computer. If implemented on a larger machine with more storage, the number of elements can be increased by merely altering the appropriate DIMENSION statements in the FORTRAN program.

5.1 Programming the Formulation Algorithm. The formulation algorithm, as developed in Chapter III, is designed for direct computer implementation. The operations are carried out in the manner specified by the definition of the numbered-matrix combinations. The matrices, as defined by these steps, are created one by one in simple matrix multiplications or additions; and the result is immediately stored on tape for later recall as required. While relatively inefficient and time consuming, this process was forced by the lack of adequate memory storage in the computer. By this technique, during the formulation stage, the computer is required to have in memory only three matrices--the multiplicand, multiplier, and the product. These are 10 x 10 arrays, as determined by the desire to handle up to ten of any one of the six kinds of elements.

In order to avoid excessive tape manipulations, the various operations are not carried out in the exact order in which they are defined in Chapter III. As a preliminary planning aid, a flow chart outlining this process, shown in Figure 5.1.1, was constructed. This chart provides a guide for planning an orderly execution of the various processes. It was also necessary to build up a history chart of the status of each tape machine at each step in the process so that the read, write, backspace, and rewind operations would be completed in proper

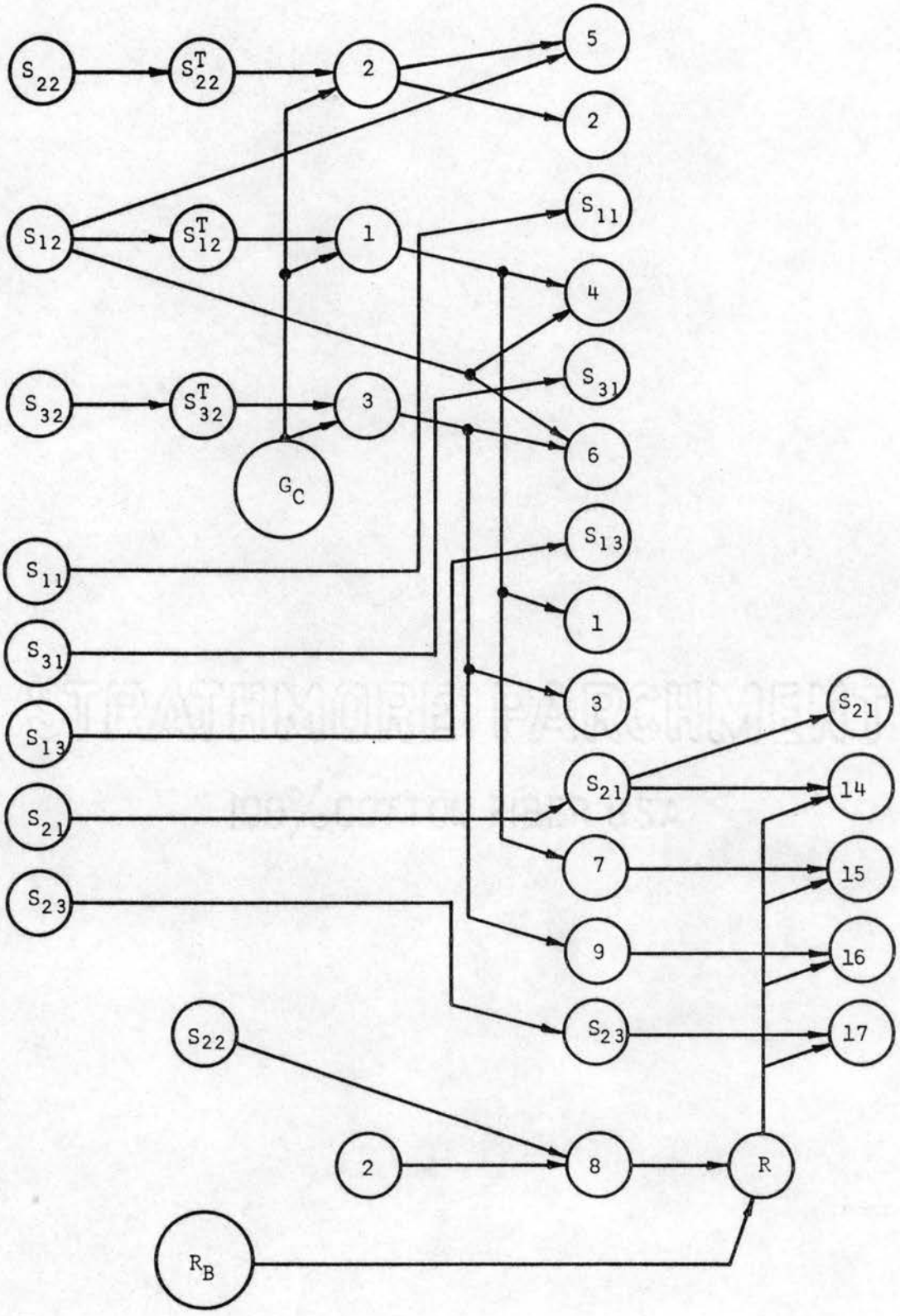


Figure 5.1.1(a). Generation Process, Phase 1, of Formulation Program

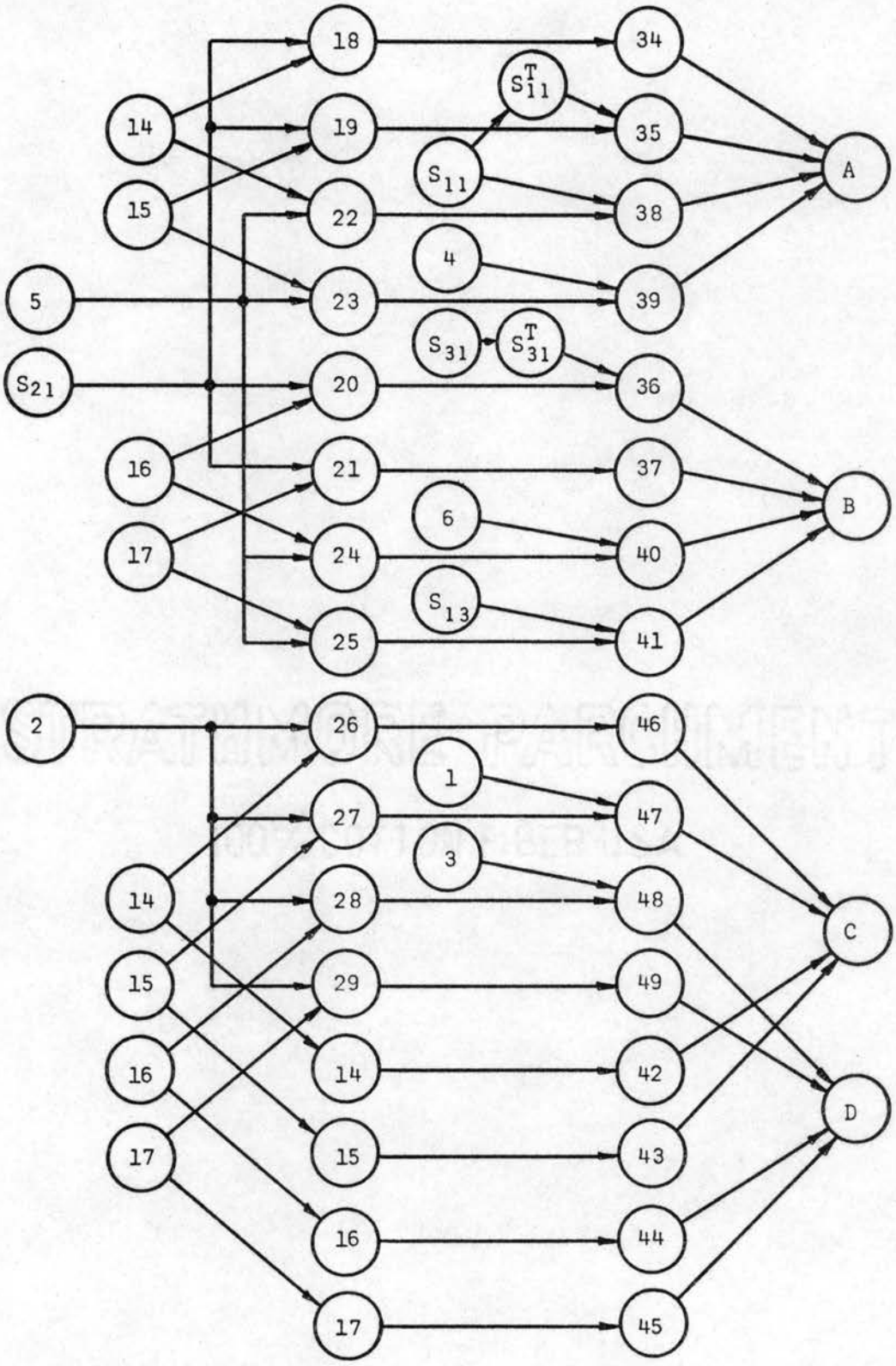


Figure 5.1.1(b). Generation Process, Phase 2, of Formulation Program

order. Experienced programmers are quite aware of the need for such planning and record-keeping, and it is mentioned here only as a reminder and word of caution to any inexperienced programmers who might attempt to write a similar program.

A serious complication for the orderly execution of the formulation process by FORTRAN programming rests on a peculiarity of the Fortran system itself. While the program is intended to be able to handle all six types of elements, it is possible that in any given network there may be no elements of a given type or types. For instance, a particular network may not include any inductors. In the equations of the system, this means that a number of the matrices involved do not exist--i.e., they have at least one zero dimension. The operations on all of these matrices are handled by DO-loops in the Fortran language. However, because of the particular way in which DO-loops are implemented, entry into the loop with a zero upper limit results in one execution of the DO-loop. This is a quite erroneous operation, since it is performed on an array which does not exist! In order to allow for the absence of any one of the particular types of elements, it is necessary to test for this condition before entering every operation which involves a sub-matrix whose size is determined by the number of such elements.

This process "cascades" down through the program. For example, the lack of inductors in a network reduces the $\underline{I}_{C1}$ vector to zero length. This, in turn, implies that there are zero columns in the S_{11}, S_{21}, and S_{31} partitions of the $\underline{S}$ matrix and zero rows in their transposes. Thus, any product which involves one of these six sub-matrices cannot exist. In this particular case, the matrices $\underline{14}$, $\underline{18}$, $\underline{19}$, $\underline{20}$, $\underline{21}$, $\underline{22}$, $\underline{26}$, $\underline{34}$, $\underline{35}$, $\underline{36}$, $\underline{37}$, $\underline{38}$, $\underline{42}$, and $\underline{46}$ do not exist.

Similar results occur for other conditions--and become even more complicated when two or three types of elements are missing.

It was arbitrarily decided to restrict this condition and require that every network have at least one resistor in both the tree and co-tree. This implies that the submatrix $\underline{S}_{22}$ will always exist. This is not a very desirable restriction and occasionally requires the addition of an unwanted resistor to a network. The influence of this resistor on the solution can be made negligible, since it can be made quite large or quite small, depending on whether it is in the co-tree or tree. This situation will normally arise only when 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 formulation program is a single-line program so that a conventional functional flow chart has little meaning. The only decisions and branching involved are those necessary to bypass operations when certain element types are missing, as noted previously. However, the order in which the various matrices are created and placed on the tapes during the program operation is of interest. An alternative form of flow chart to exhibit this process was developed to clarify the program manipulations. This flow chart is presented in Appendix D.

The program makes extensive use of subroutine subprograms to perform the actual matrix manipulations and the numerous read and write operations. This is advantageous because of the repetitious nature of the operations. However, this technique was not carried to an extreme, and a number of similar subroutines are used with different names for the mnemonic value of those names in clarifying program operation. This technique results in reduction of the main program to the role of an

"executive" routine which calls the subroutines in proper sequence. The main program also performs the tests required for bypassing of the appropriate routines when element types are missing from the circuit, as mentioned previously.

Though the extensive use of subroutine subprograms greatly reduced the program storage requirements, the large number of operations to be performed far exceeded the capabilities of the machine that was available. To overcome this, it was necessary to break the formulation program into four phases called MAKES, MATRIX1, MATRIX2, and MATRIX4. MAKES is specialized and performs the function of reading from cards the entries in the $\underline{S}$ matrix and writing onto tape the various submatrices required in later steps and in the execution phase for the time solution. The remaining three phases perform the actual formulation process, leaving the necessary matrices for the system on three tapes, as follows:

Tape Unit #4 - Matrices 42 through 49 for the solution of Equation 3.3.16,

Tape Unit #5 - Matrices 34 through 41 for the solution of Equation 3.3.15,

Tape Unit #6 - Matrix of Equation 3.4.2 for the development of the initial conditions from given t = 0 driver values.

These matrices are adjoined to yield equations in the following form:

$$\underline{LC}\,\underline{\dot{X}} = \underline{A}\,\underline{X} + \underline{B}\,\underline{E}$$

$$\underline{Y} = \underline{C}\,\underline{X} + \underline{D}\,\underline{E}$$

$$\underline{X}_o = \underline{A}^{-1}\,\underline{B}\,\underline{E}_o \quad . \tag{5.1.1}$$

5.2 Programming the Time Solution Including Nonlinear Elements.

With the matrices of the equations of the state-space model formulated and written onto tapes, the time-solution phases of the program can be initiated. Using the techniques outlined in Chapter IV, the following nonlinear models are implemented:

(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, as shown by Equation 4.1.2,

(c) Nonlinear R values (dissipation elements) 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 polynomial equation of fourth degree with any variable in the system as the controlling variable,

(e) 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-current source, as explained in Appendix C,

(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,

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

$$\begin{aligned} y &= a_1 x + b_1 \quad \text{for } x > x_o \\ y &= a_2 x + b_2 \quad \text{for } x < x_o \end{aligned} \tag{5.2.1}$$

with

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

Provisions are necessary for the iteration of the initial conditions when nonlinear sources are included. These are implemented to allow for either the specification of initial conditions on the storage elements and time-dependent sources, or the alternative scheme for the solution of the entire system with only sources specified to their t = 0 value.

The most desirable situation would be to have all of these features included in one program so that the user could call upon any or all of them for any one system or circuit. However, the restrictions of limited core memory in the available computer ruled out such a possibility. The only alternative was to write several similar integration-iteration programs with one or more of the models included in each. These programs are preceded by a routine which picks the proper program for solution--or rejects the circuit if an invalid combination is encountered.

As the program now stands, there are three distinct time-solution programs. The polynomial model for nonlinear inductances or capacitances and the iron-core inductor model are incorporated into each of these three programs. This allows nonlinear storage elements to be present in all circuits.

The first time-solution program has, in addition, the nonlinear dependent sources of the polynomial type. Iteration of the nonlinear elements is performed by a stepwise progression toward the convergence value. The user provides suggested values for this step size as input data. At each step in the iteration, the dependent variable is

calculated from the value of the control variable. This calculated value is compared with the value of that variable in the circuit solution. If the circuit value is too low, a step upward is made, and the solution for the entire network is reevaluated. The dependent variable is again recalculated, compared, and so on. If the step takes the controlled variable too far so that the calculated value is on the opposite side after the step, the step size is reduced by a factor of ten; and a step backward is taken. This iteration is continued until the convergence test indicates that the percentage change in the length of each of the six vectors of variables (the r.m.s. value), such as E_{B2}, is less than some specified convergence factor. This terminates the iteration process, and the program passes on to the next integration step after setting the value of the step size in the iteration process upward by a factor of ten. This is necessary in order to force a new iteration to convergence after the integration step has been taken. Otherwise, the steps at the beginning of the next iteration are likely to be so small that a false indication of convergence will be obtained.

This iteration method is obviously almost an "exhaustive search" technique. It has a tendency to be rather slow, requiring many iteration steps if the changes in the circuit are rapid from step to step of the integration. This was improved somewhat by inclusion of a routine to increase the step size if more than ten steps are taken in one direction without passing through the convergence value. This same iteration method was tried with the other nonlinear sources but did not perform well.

Most of the shortcomings of this method of iteration lie in the convergence test. The r.m.s. sum provides a check of over-all convergence

in the whole network. However, it is very insensitive when the elements in any one vector have a wide range of magnitudes--the r.m.s. sum of a 100-unit variable and a 1-unit variable is little influenced by changes in the 1-unit variable!

The failure of this method for the vacuum tube model was caused by the large values of resistors common in such circuits. Resistors of a million ohms are not uncommon. With such resistance values in the circuit, an arbitrary step of 1-10,000-th of an ampere in current through this resistor results in a 100-volt jump in the voltage across the resistor. Such changes frequently resulted in voltage values outside the valid region for the vacuum tube model. When this occurred, the solution went into violent oscillation or diverged. This could sometimes be avoided by putting limits on the allowable range of the model, but this became unduly cumbersome and was not fail-safe. For the straight polynomial model of dependent sources, this step-by-step iteration converges for tremendous ranges of coefficients in the polynomial and is quite useful for this reason.

The second time-solution program includes the two-variable vacuum tube model (e) and the piecewise linear model (g), in addition to the nonlinear storage elements. The iteration method used in this program is called the "modified method of successive approximations" by Ledley (3) and McCracken and Dorn (5). The normal method of successive approximations, where

$$x_{n+1} = f(x_n) \qquad (5.2.2)$$

fails to converge for $|f'(x_n)| > 1$, where $f'(x_n)$ is the slope of the function $f(x)$ at x_n. The modified method achieves convergence over a

much wider range by the process

$$x_{n+1} = q\, x_n + (1 - q)\, f(x_n) , \qquad (5.2.3)$$

where q is given by

$$q = m/(m - 1) , \qquad (5.2.4)$$

and m is given by

$$m = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} . \qquad (5.2.5)$$

Convergence for this iteration process is tested by evaluating

$$\text{error} = \left| \frac{x_{n+1} - x_n}{x_{n+1} + x_n} \right| \qquad (5.2.6)$$

and continuing the iteration until this error term is less than some specified convergence factor. This iteration method has proved to be very useful for the vacuum triode model. The only failures have occurred in circuits which are inherently very unstable circuits, such as multivibrators. The response of such circuits, when they are idealized, is actually discontinuous. These discontinuities result from ignoring small inductances and stray capacities which are present in the actual circuit.

The third time-solution program includes the semiconductor exponential model (f), along with the nonlinear inductor and capacitor models. The iteration method used in this program is an adaptation of the Newton-Raphson method for finding the zeroes of a function. This method is used in the PETAP and NET programs, but a modification has been developed and tested which is worthy of mention here.

Shown in Figure 5.2.1 is a sketch of the exponential characteristic curve of a semiconductor diode and the straight line which represents the terminal characteristics of a linear circuit external to the diode. In the analysis of a system containing more than one semiconductor diode or other nonlinear element, the assumption of linearity for the external circuit for any one of the diodes is questionable. However, during the iteration for one of the diodes, the remaining nonlinear elements are represented by fixed sources. Thus, on a piecewise basis it is valid to represent the circuit in this manner. The straight-line representation of a complicated linear circuit is based on the Thevenin equivalent representation.

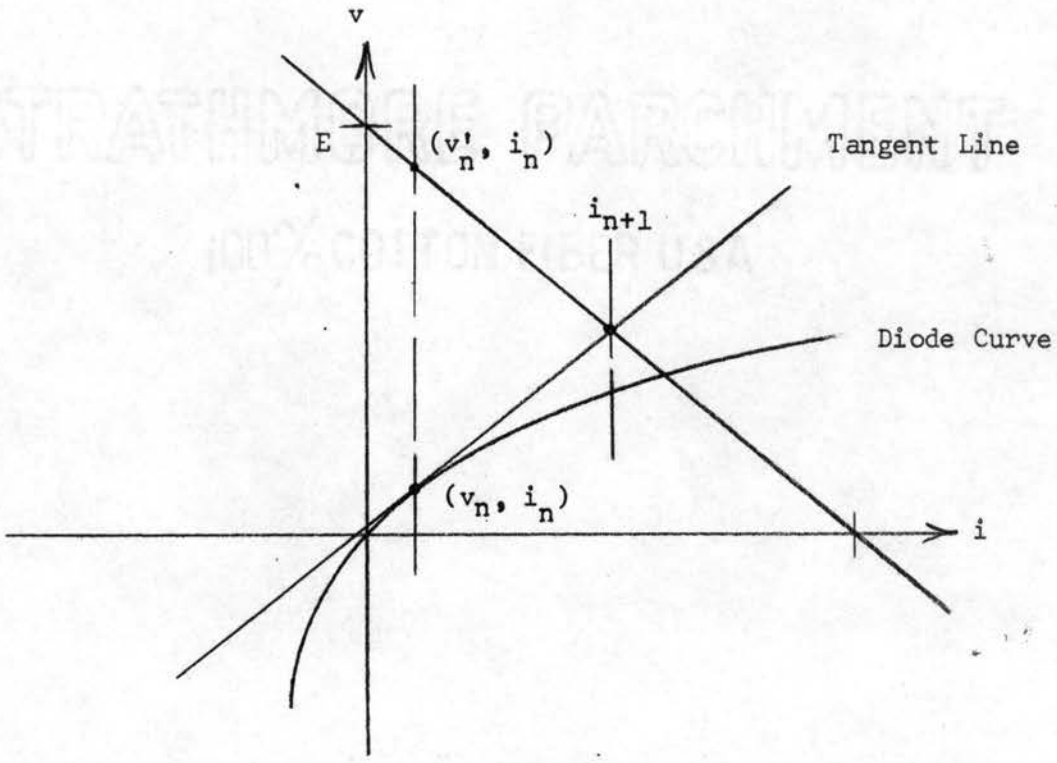


Figure 5.2.1. Current-Voltage Curves for Newton-Raphson Method, Region 1, $v > 0$

The equation defining the semiconductor diode represents the current as a function of the voltage across the device so that it is represented by a dependent-current driver. However, the iteration process being considered seeks to obtain a new value of i, rather than the controlling variable v, so that the normal diode equation

$$i = I_s \, (e^{bv} - 1) \tag{5.2.7}$$

must be written in the inverse form

$$v = \frac{1}{b} \ln \left(\frac{I_s + i}{I_s} \right) , \tag{5.2.8}$$

which is the equation of the diode curve of Figure 5.2.1.

At the n-th step in the iteration, the current for the driver representing the diode has been set at i_n. For this value of current, the external circuit yields a voltage v_n', as illustrated by the point (v_n', i_n). For the diode, this current results from a voltage of v_n, as represented by the point (v_n, i_n). Obviously, for the case shown, these two values are widely different so that additional iteration is required to approach the simultaneous solution represented by the point (v_o, i_o) at the intersection of the two curves. Utilizing the Newton-Raphson method of iteration, the tangent to the diode curve is constructed through the point (v_n, i_n). The value of current at the intersection of this tangent line and the external circuit line is taken as the value for the next step in the iteration. Thus, for the (n + 1) iteration, the current generator is set to the value i_{n+1} as shown in the figure, and the process is then repeated until suitable agreement is achieved in the two values of voltage.

For a truly linear circuit, the slope of the line representing the

circuit is constant, as all resistors and passive elements in the circuit remain fixed. For such a circuit, it is possible to determine at the beginning of the analysis the value of this slope and use it throughout the analysis. The only variable of the representation of the external circuit is then the intercept, E, with the v-axis. However, the case is not so simple when nonlinear elements are included in this external circuit, as they introduce changing values of resistance. Thus, at any one step in the process, the external circuit can be considered linear on a piecewise basis; but, from step to step, allowance must be made for change of not only the intercept but the slope of the line. To carry out the nonlinear analysis, some method must be devised for determining this line at every step in the iteration.

For this purpose, it is convenient to determine the equation of the line by finding two points on it. One point is already known--(v'_n, i_n). An additional point could be obtained by temporarily setting i to zero and recalculating the external circuit to obtain the intercept E. However, since it is assumed that the circuit is only piecewise linear, it is desirable to change the value of i no more than necessary to obtain this point. Thus, the value of i is changed only a small amount, Δi, to obtain the second point on the line. With this information, the equation of the line can be obtained. The tangent line to the diode curve can be determined, and their intersection then gives the new value of i, i_{n+1}. The equations utilized in this iteration process are developed in Appendix E.

The basic Newton-Raphson method just outlined is utilized in the PETAP and NET programs, except for the additional feature of reevaluation of the slope at each iteration, which is original to this study.

The iteration process just presented is valid only for values of current and voltage greater than zero. This is graphically illustrated by Figure 5.2.2, which is the situation for negative voltage applied to the diode.

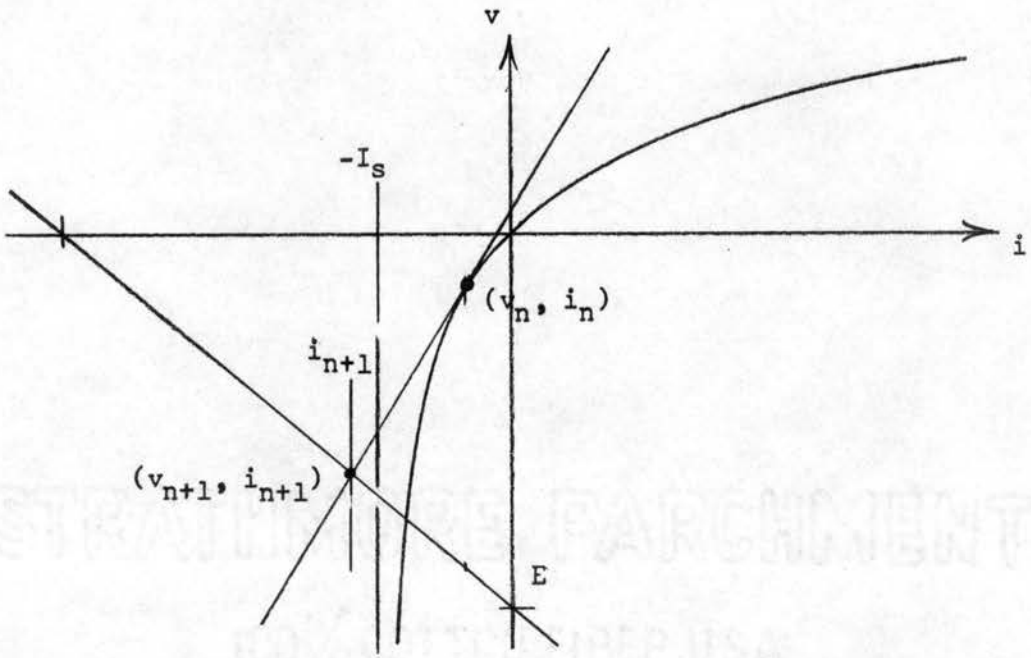


Figure 5.2.2. Voltage-Current Curves for the Newton-Raphson Method, Region 2, $v < 0$

Again, the new iterate value of i is assumed to be the intersection point (v_{n+1}, i_{n+1}). However, for large values of negative voltage applied to the diode, the current approaches asymptotically the value $-I_s$ and under no circumstances can exceed this value. It is obvious from the illustration that the value i_{n+1} represents an impossible current for the device. This same failing arises in the iteration process utilized in PETAP and NET. The procedure selected by the writers of these programs to overcome this failing is to arbitrarily set i_{n+1}

to some fraction of $-I_s$ and then to improve this by the method of successive approximations. In most realistic situations, this process will successfully converge to an acceptable value of i. However, in certain cases the convergence is very slow, and in some cases it has failed completely.

For the purposes of this study, another Newton-Raphson type of iteration process for this situation was developed. The basis for this process is the interchange of the roles of v and i in the iteration. This is illustrated in Figure 5.2.3, which is essentially the same as Figure 5.2.2 except that the v and i axes have been interchanged. In this process, a new value of v will be iterated, rather than i. However, since the element representing the device is a current driver, it is necessary to convert this new iteration of v to an equivalent i, which is accomplished by use of the diode equation

$$i_{n+1} = I_s \left(e^{bv_{n+1}} - 1\right) \quad . \tag{5.2.8}$$

This process is illustrated in Figure 5.2.3, and the equations are developed in Appendix E. This iteration method seems to offer convergence as positive and rapid as that for the positive values. Theoretically, since there are no inflection points or double inflection points in the diode equation, it should converge for all values without any trouble. In this respect, it is believed to be superior to the method of successive iterations.

Since there is a complete change of iteration methods as the voltage crosses the axis, it is necessary to perform tests before attempting the iteration. A pathological case arises when, for instance, the value of current is still at i_n, but the time-dependent drivers

have changed sufficiently for the voltage across the diode to be of opposite sign. Neither iteration method, as implemented, will function satisfactorily in this case. This problem was alleviated by setting the value of i to zero when this situation is detected and restarting the iteration. This technique has proved to be satisfactory, but further testing may reveal that other measures are necessary.

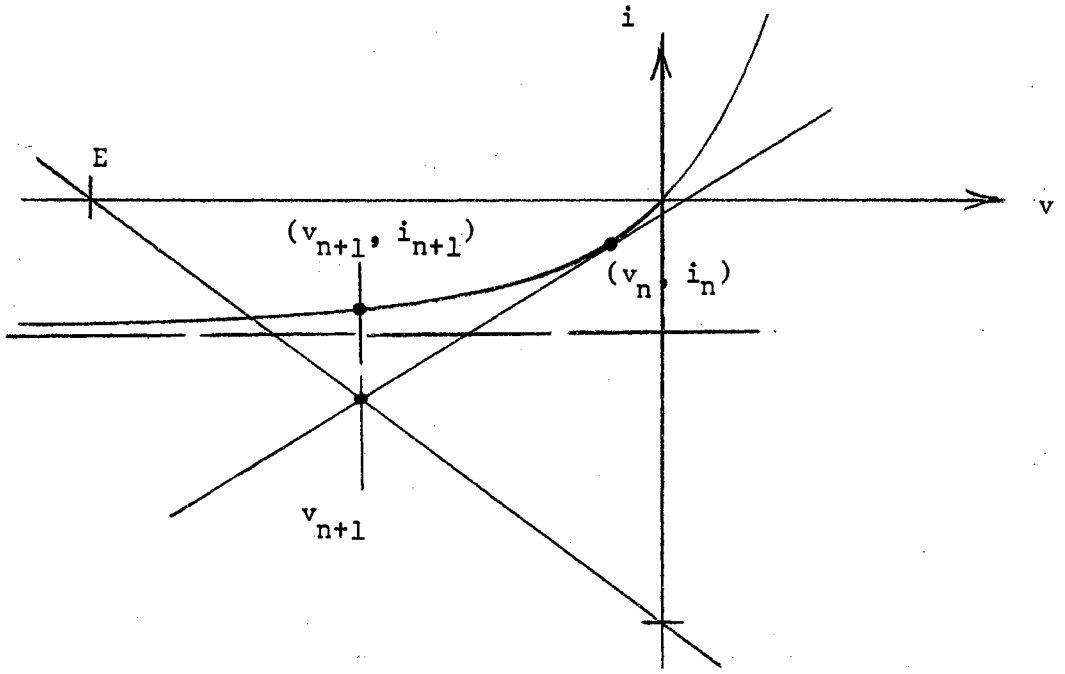


Figure 5.2.3. Modified Voltage-Current Curves for Newton-Raphson Method, Region 2, $v < 0$

Convergence for this iteration process is tested by the same method as for the method of modified successive approximations, as illustrated by Equation 5.2.6.

The over-all logic of the three time-analysis programs is basically the same. The differences in them are principally in the iteration processes. The flow diagram of the major logic operations is illustrated

in Figure 5.2.4.

Each of the analysis programs is actually divided into two phases: the first phase loads the data; the second is the actual integration-iteration phase. Even so, inadequate memory capacity forced the optimization of every process to the absolute minimum of data storage and operating instructions. Many desirable features were left out because it was impossible to implement them in the available memory.

No memory is available for other than the bare minimum of output writing operations. A complete total solution to the network is obtained after every step in the integration-iteration process, and this total solution is written in one logical record (in internal machine format). This output record is accumulated on tape; and at the completion of the analysis phase, a separate output program is called into operation to translate this data into acceptable printer output. Any part or all of the variables may be written, and other special processing may be done at this time since the total solution is available. Such special calculations as instantaneous power of any element, integration of average powers, and so on, may be incorporated into this output phase. It is felt that this is actually a desirable mode of operation.

One special memory-saving technique which is utilized in this program is worthy of mention. The $\underline{S}$ matrix, in its direct and transposed form, is a necessary part of data required to generate the total solution, as outlined in Section 3.3. This matrix is a 30 by 30 matrix when the program is implemented for ten elements in each of the six categories. However, the contents of the $\underline{S}$ matrix consist only of

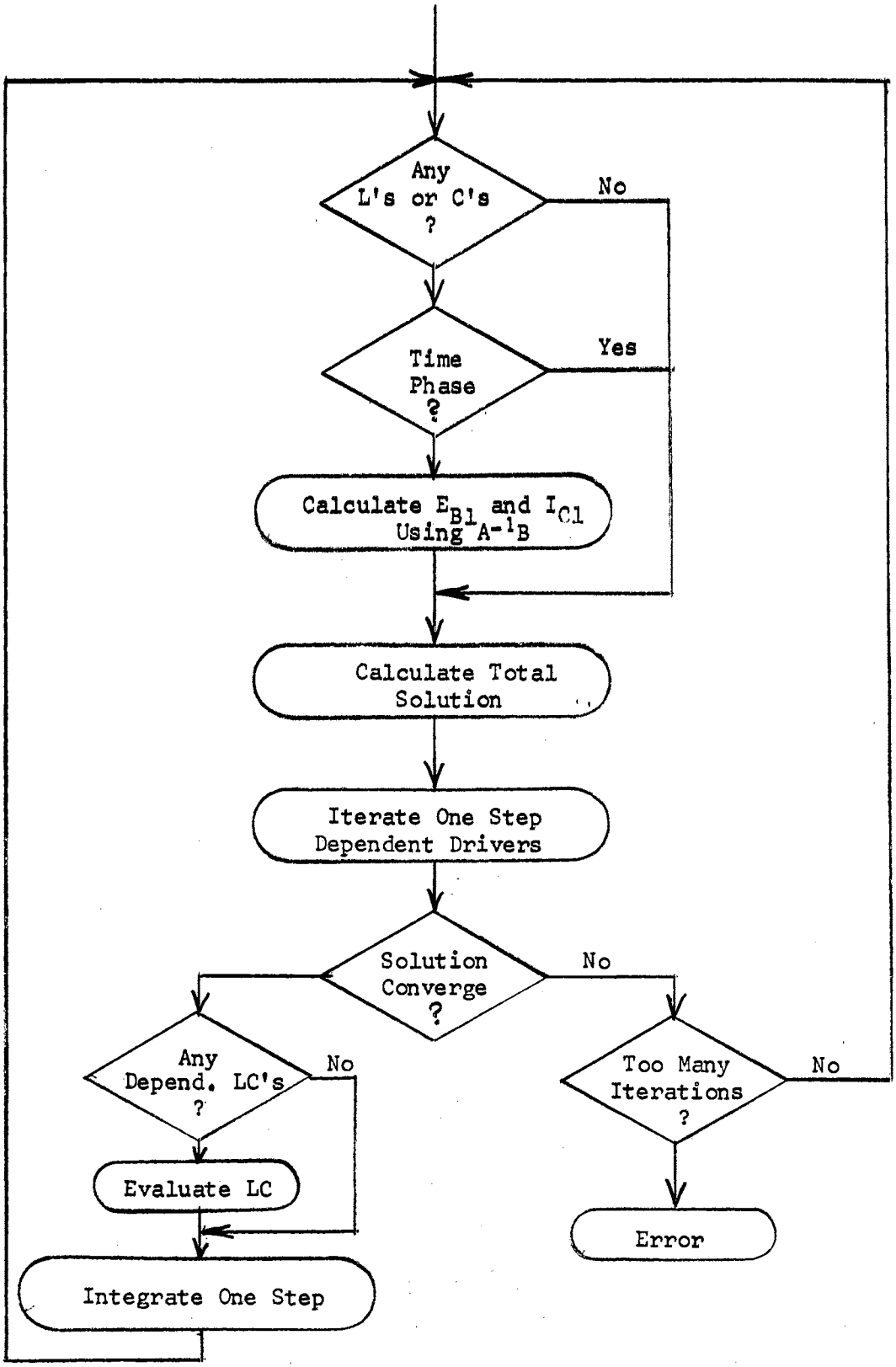


Figure 5.2.4. Flow Chart of Time-Solution Programs

entries 0, 1, or -1. This matrix is used in the calculation of floating-point numbers. If stored in its entirety with the normal eight digits of floating-point precision, it would require 9,000 digits of storage. For computation, the only entries of significance are the non-zero entries. A survey was made of the $\underline{S}$ matrix for some 25 typical circuits, and it was found that the content of non-zero entries was 50 percent or less. This is, of course, entirely determined by the relative number of nodes and elements in the network. On this basis, considerable memory could be conserved by storing the subscript for each non-zero entry and the +1 or -1 value in integer form. As compiled, the program uses an integer-number field size of only three digits so that the storage of these three entries requires only nine digits. At the time of computation, the integer number is converted to floating point. The matrix multiplication is performed using the stored subscripts to multiply only the non-zero entries. In addition to saving considerable core space for data storage, the running time was materially improved by the reduced number of arithmetic operation required. Provisions are made in the DIMENSION statements for storing 125 non-zero entires, which should be adequate for most circuits.

In general, a philosophy of minimum input data has been followed throughout the program. All data storage areas are initialized to zero, and only non-zero entries are read as input data. This technique reduces the amount of input data which must be provided.

The program has been tested by the processing of a variety of circuits containing many combinations of linear and the nonlinear elements. Some of the more interesting examples are:

(1) Steady-state (DC) analysis of an operational amplifier with

feedback (amplifier gain: 10^6)

(2) Steady-state (DC) analysis of a large network containing 41 elements--3 capacitors, 19 resistors, 7 inductors, 6 current drivers, and 6 voltage drivers

(3) Steady-state analysis of a ladder network of 14 resistors excited by a DC source

(4) Transient analysis of a passive network of four resistors, three capacitors, and two saturating iron-core inductors with an initial charge on one capacitor

(5) Semiconductor diode and saturating iron-core inductor excited by sinusoidal voltage driver

(6) Vacuum triode in a Colpitts oscillator circuit

(7) Restart of the solution of (6) at an arbitrary point in time by entering as initial conditions the capacitor voltages and inductor current of the solution that was in progress

(8) Vacuum triode performance with highly inductive plate-load circuit under a variety of excitations

(9) Transient analysis of series RLC circuit with initial capacitor charge. Comparison of response for saturating and non-saturating inductor

(10) Transient analysis of equivalent circuit of iron-core transformer with parallel linear capacitance, illustrating apparent negative-resistance region and discontinuous response resulting from transformer-core saturation

(11) Transient analysis of a two-mass mechanical system with masses coupled by nonlinear spring

(12) Transient analysis of full-wave semiconductor rectifier-filter

system under varying loads. Two-choke filter with input-saturating choke and output-nonsaturating choke

Typical data input, output, and plots for some of these examples are presented in Appendix F.

CHAPTER VI

SUMMARY

6.1 Conclusions. The results of this study have, in general, been quite gratifying. A considerable portion of the work was involved with problems directly associated with the computer programming. However, the performance of the formulation algorithm has been satisfactory in every respect. Within the limitations imposed by the algorithm as developed--that there can be no circuits of capacitors and/or voltage drivers and no junctions (nodes) with only inductors and/or current sources connected--the matrix coefficients of the equations of the systems have been properly generated for all examples. Based on the results from the DC analysis of several systems containing 30 to 40 elements, there is some problem of numerical accuracy, but eight digits floating-point precision yield results accurate to six digits. There is some improvement in accuracy if the smaller resistors are placed in the network tree, when this is possible. The exact amount of influence this will exert is not known--this is the result of observation of a limited number of cases and is supported by Reid at Michigan State (18).

In the development of the time-analysis programs and algorithms for the nonlinear elements, there have been many problems typical of nonlinear analysis. The nonlinear models developed specifically for the vacuum triode and iron-core inductor have proved to be excellent models. They are far from being complete models of the elements since they

ignore many effects such as magnetic hysteresis, thermal hysteresis, transit-time effects, and stray inductance and capacitances, which are known to be characteristic of the real devices. However, they are a significant improvement over previous models and, within their limitations, allow at least a beginning to be made in their study.

There has been no attempt to search out an iteration technique which will serve for the study of all nonlinear elements as it is felt that this is not feasible at the present time, with our limited experience in the computer analysis of nonlinear systems. This study, rather, has attempted to explore a number of techniques and their applicability, or failure, for certain model types. No definite conclusions have been reached in this area of great consequence, but the method of modified successive approximations has unique advantages for iteration processes where evaluation of the derivative is difficult or impossible. When evaluation of the derivative is straightforward, the Newton-Raphson method of iteration is a powerful technique.

The general concept of representation of nonlinear elements in a state-space formulation by the incorporation of dependent drivers has been successful. The results, for the specific elements attempted in this study, have been quite adequate. The implementation of a quite large class of nonlinear elements by this technique should be successful--within the ability to find a suitable equation of constraint for the nonlinear element. The generality of this approach is appealing when compared with the previous methods which involve more restrictive separation of the system into specific linear and nonlinear equation components.

While the over-all performance of the analysis process has been

good, it has failed completely in several instances. An interesting class of circuits which cause failure are multivibrators. The discontinuous nature of their performance as they switch rapidly from one state to another has caused failure of the iteration process on every trial. Reduction of the size of the integration interval step size affects some improvement but not a cure. The multivibrator is basically a very unstable circuit, and it appears that the circuit model of it, and its numerical solution, is just as unstable as the actual system. This is disappointing, and no solution has been found up to this time. However, it is felt that the trouble may lie in a circuit model which is too ideal and that the introduction of stray capacities and inductances as they appear in the actual circuit may reduce the discontinuities of the response to the point where the iteration processes can converge. The program is unable to handle enough elements to accomplish this type of analysis.

The program, in its present form, is not practical for everyday use as a circuit analysis tool for students and practicing engineers. It has been developed as a research tool with little attention directed to the details which simplify use. Many unnecessary features have been included for testing purposes, and many necessary features have been omitted; some deliberately, some of necessity, and some out of oversight.

6.2 Recommendations for Further Work. The performance of the analytic procedures in this study have been adequate to justify further work and development along the same lines. Implementing these procedures in a carefully planned and utilitarian program is recommended as a worth-while development. Such a program should prove useful to a

variety of people not only in electrical circuit analysis but in general systems analysis of other types.

It should be stressed that the techniques developed, while specifically applied to electrical networks in this study, are generally useful for any energy system. They are particularly powerful for systems of mixed components, containing electromechanical and mechanical-hydraulic transducers, where the elements can be represented by dependent drivers. An electric motor is an excellent example. Its output can be represented by a dependent torque (through) driver with the equation describing it having an electrical current as the controlling (input) variable. The possibilities of such analysis procedures and the utilization of an analysis program based on the techniques of this study are apparently limitless, requiring only further investigation of their use.

If the creation of such a utility program is undertaken, the following suggestions are offered for useful improvements:

(1) Addition of an automated tree-finding algorithm preceding the present program, which will find a suitable tree of the graph from a description of the element interconnection diagram. A useful algorithm for this purpose has been developed by Cummins and Thomason (16) from a method proposed by Minty (17). This will greatly reduce the work required of the user.

(2) Consideration should be given to removing the topological restrictions imposed by forbidding circuits of capacitors and/or voltage drivers or nodes connected only to inductors and/or current drivers. This is possible but requires elaboration of the model to include derivatives of the drivers in the state-space equations. This imposes the restriction that the

derivatives must be known and continuous.

(3) An iteration routine should be provided which will serve as many models as possible. It may be advisable to provide some composite routine of several methods which sequentially or selectively utilizes one or more methods.

(4) Implement the program on a larger computer which can handle most of the data in-core at all times with a very minimum of external storage on tape or disc. Reading and writing on external storage devices is far too time-consuming for an efficient utility program.

Several areas of modeling worthy of further study were encountered:

(1) Investigation of piecewise-linear models of devices difficult to model in other forms. The vacuum tube pentode provides a good example. Since the device, like the triode, is a two-variable element, investigation of piecewise-linear characteristics for the plate voltage variations and normal polynomial variation with grid voltage should be investigated. Such models could also be extremely useful in the representation of devices such as electric motors and other transducers.

(2) Investigation of the modeling of hysteresis elements. The normal-form differential equations of the state-space model make the first derivative of all variables available at every step in the iteration-integration process. With information on the variable and its derivative, it should be possible, with an adequate amount of storage of back-value history, to develop some sort of model for this type of device. This would be an exceedingly useful model in many areas.

In conclusion, I believe that the formulation techniques and solution methods presented in this study are powerful and useful techniques. They are worthy of much additional work and development. The analysis of large nonlinear systems is becoming of more importance every day, and the complexity and size of practical systems has reached the point where computerized analysis methods offer the only useful avenue of approach. The traditional closed-form linear solution, if possible, is of limited use.

It is imperative that engineers should not only change their methods of analysis but their entire reasoning processes to fully utilize the methods made possible by the general availability of large computers. But, one fundamental axiom due to Hamming should always be kept in mind--"THE PURPOSE OF COMPUTING IS INSIGHT, NOT NUMBERS."

A SELECTED BIBLIOGRAPHY

1. Brown, Buck F. "Sequential Solutions in Nonlinear Control Theory." Ph.D. Dissertation, Oklahoma State University, Stillwater, Oklahoma, May, 1964.

2. Hamming, R. W. Numerical Methods for Scientists and Engineers. New York: McGraw-Hill, 1962.

3. Ledley, R. S. Programming and Utilizing Digital Computers. New York: McGraw-Hill, 1962.

4. Calingaert, P. Principles of Computation. Reading, Massachusetts: Addison-Wesley, 1965.

5. McCracken, D. D. and W. S. Dorn. Numerical Methods and Fortran Programming. New York: John Wiley, 1964.

6. Pennington, R. H. Introductory Computer Methods and Numerical Analysis. New York: Macmillan, 1965.

7. Arden, B. W. An Introduction to Digital Computing. Reading, Massachusetts: Addison-Wesley, 1963.

8. Henrici, P. Discrete Variable Methods in Ordinary Differential Equations. New York: John Wiley, 1962.

9. Hildebrand, F. B. Introduction to Numerical Analysis. New York: McGraw-Hill, 1956.

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

11. Koenig, H. E., Y. Tokad, and H. K. Kesavan. Analysis of Discrete Physical Systems. New York: McGraw-Hill, pending publication.

12. _____, and W. A. Blackwell. Electromechanical Systems Theory. New York: McGraw-Hill, 1961.

13. Zadeh, L. A. and C. A. Desoer. Linear System Theory: The State Space Approach. New York: McGraw-Hill, 1963.

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

15. Callahan, D. A. "Linear Network Analysis and Realization Digital Computer Programs." University of Illinois Engineering Experiment Station Bulletin 472, Urbana, Illinois, 1965.

16. Cummins, R. L. and L. C. Thomason. "An Efficient Tree-Listing Program." presented at the Southwestern I.E.E.E. Conference, Dallas, Texas (April, 1964).

17. Minty, G. J. "A Simple Algorithm for Listing All the Trees of a Graph." I.E.E.E. Transactions on Circuit Theory, Volume CT-12, No. 1 (March, 1965).

18. Reid, R. J. "Computer Analysis of Linear Systems." presented at the Southwestern I.E.E.E. Conference, Dallas, Texas (April, 1964).

19. Hart, D. E., et al. "GMR DYANA Programming Manual." General Motors Corporation Research Laboratories, Warren, Michigan, 1959.

20. Branin, F. H., Jr., et al. "A Program for Computing the Transient Response of Transistor Switching Circuits--PETAP." I.B.M. Technical Report TR00.11000.700, Poughkeepsie, New York, 1959.

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

22. Branin, F. H., Jr. "The Relation Between Kron's Method and the Classical Methods of Network Analysis." I.B.M. Technical Report TR00.01000.686, Poughkeepsie, New York, 1959.

23. Searle, C. L., et al. Elementary Circuit Properties of Transistors. New York: John Wiley, 1964.

APPENDIX A

EXISTENCE OF THE INVERSE OF R_1

The elimination of the algebraic variables from the differential equations of the state-space model is necessary if the differential equations are to be put in normal form. This elimination depends upon the existence of the inverse of matrix $\underline{R}_1$, defined by

$$\underline{R}_1 = \underline{U} + \underline{R}_B \underline{S}_{22} \underline{G}_C \underline{S}_{22}^T \quad . \tag{A.1.1}$$

This matrix can be put into the form

$$\underline{R}_1 = \underline{R}_B [\underline{R}_B^{-1} + \underline{S}_{22} \underline{G}_C \underline{S}_{22}^T] \quad . \tag{A.1.2}$$

Since $\underline{R}_B$ is a diagonal matrix of real positive non-zero entries, the inverse of R_1 will exist if the inverse of C as defined by

$$\underline{C} = \underline{R}_B^{-1} + \underline{S}_{22} \underline{G}_C \underline{S}_{22}^T \tag{A.1.3}$$

exists. R_B^{-1} is a diagonal matrix of order n with real positive non-zero entries. G_C is a diagonal matrix of order m with real positive non-zero entries. S_{22} is a rectangular matrix of order n x m ($n \leq m$) and rank r ($r \leq n$).

Assertion 1. The matrix $S_{22}\ G_C\ S_{22}^T$ is either positive definite or positive semidefinite.

Proof: The assertion is established by the application of one of the following two theorems, from Reference 11, Appendix A:

Theorem A10.4. Let $\underline{B}$ be a n x m (n < m) real matrix of rank r with r = n, and let $\underline{D}$ be a diagonal matrix of order m with all positive entries. Then the matrix

$$\underline{A} = \underline{B}\ \underline{D}\ \underline{B}^T$$

is positive definite and nonsingular.

Theorem A10.5. Let $\underline{D}$ be a real symmetric and positive definite matrix of order m and $\underline{B}$ a real matrix of order n x m (n < m) and rank r with r < n. Then the matrix

$$\underline{A} = \underline{B}\ \underline{D}\ \underline{B}^T$$

is positive semidefinite.

Since the matrix G_C is diagonal with real positive non-zero entries, it is symmetric and positive definite and thus fulfills the hypothesis for matrix $\underline{D}$ of the theorems. The matrix $\underline{S}_{22}$ fulfills the hypothesis for matrix $\underline{B}$. Therefore, at least one of the theorems must apply to the product $S_{22}\ G_C\ S_{22}^T$. Theorem A10.4 is applicable if $\underline{S}_{22}$ is of maximum rank; Theorem A10.5 applies if it is not of maximum rank. Therefore, the product is either positive definite or positive semidefinite, as was asserted.

Assertion 2. The matrix $\underline{C}$ of Equation A.1.3 is positive definite.

Proof: The matrix $\underline{R}_B^{-1}$ is diagonal with real positive non-zero entries; therefore, it is positive definite. For any non-zero vector $\underline{X}$, consider the following product:

$$c = \underline{X}^T[\underline{A} + \underline{B}]\underline{X} \qquad (A.1.4)$$

where $\underline{A}$ is positive definite and $\underline{B}$ is positive semidefinite. This can be put into the form

$$c = \underline{X}^T \underline{A} \underline{X} + \underline{X}^T \underline{B} \underline{X} \quad , \tag{A.1.5}$$

or

$$c = a + b \quad . \tag{A.1.6}$$

Since, by hypothesis $\underline{A}$ is positive definite, $a > 0$. Also, Since $\underline{B}$ is positive semidefinite, $b \geq 0$. Therefore, $c > 0$; and, by definition, the matrix $\underline{C}$ is positive definite.

Assertion 3. The matrix $\underline{C}$ is nonsingular.

Proof: Since $\underline{C}$ is positive definite, by definition all leading principal minors are greater than zero. The determinant of $\underline{C}$ is a leading principal minor; and, therefore, det $\underline{C} > 0$, and $\underline{C}$ is nonsingular.

APPENDIX B

IRON-CORE INDUCTOR REPRESENTATION

Using least-squares error criteria, the function

$$\lambda = A \tan^{-1}\left(\frac{i}{C}\right) + B\left[\tan^{-1}\left(\frac{i}{C}\right)\right]^2 \qquad (B.1.1)$$

was fitted to the data represented by the solid curve of Figure B.1.1. The value of C was selected as 2.5, which is the value of current at a flux of approximately 1/2 the saturation flux. The curve is a plot of the B-H curve of a typical sample of high-quality transformer-core steel.

The dotted curve of Figure B.1.1 is a plot of the calculated values of flux for A = 55.8692 and B = 3.9182, as obtained from the curve-fitting process.

Figure B.1.2 is a plot of inductance L versus current with these coefficients inserted into the inductance equation

$$\frac{d\lambda}{di} = L = \frac{C}{i^2 + C^2}\left[A + 2B \tan^{-1}\left(\frac{i}{C}\right)\right] \qquad (B.1.2)$$

Comparison of this plot and the slope of the original flux-current curve reveals that this is a very realistic representation of the inductance that would result for a coil with this material as a core.

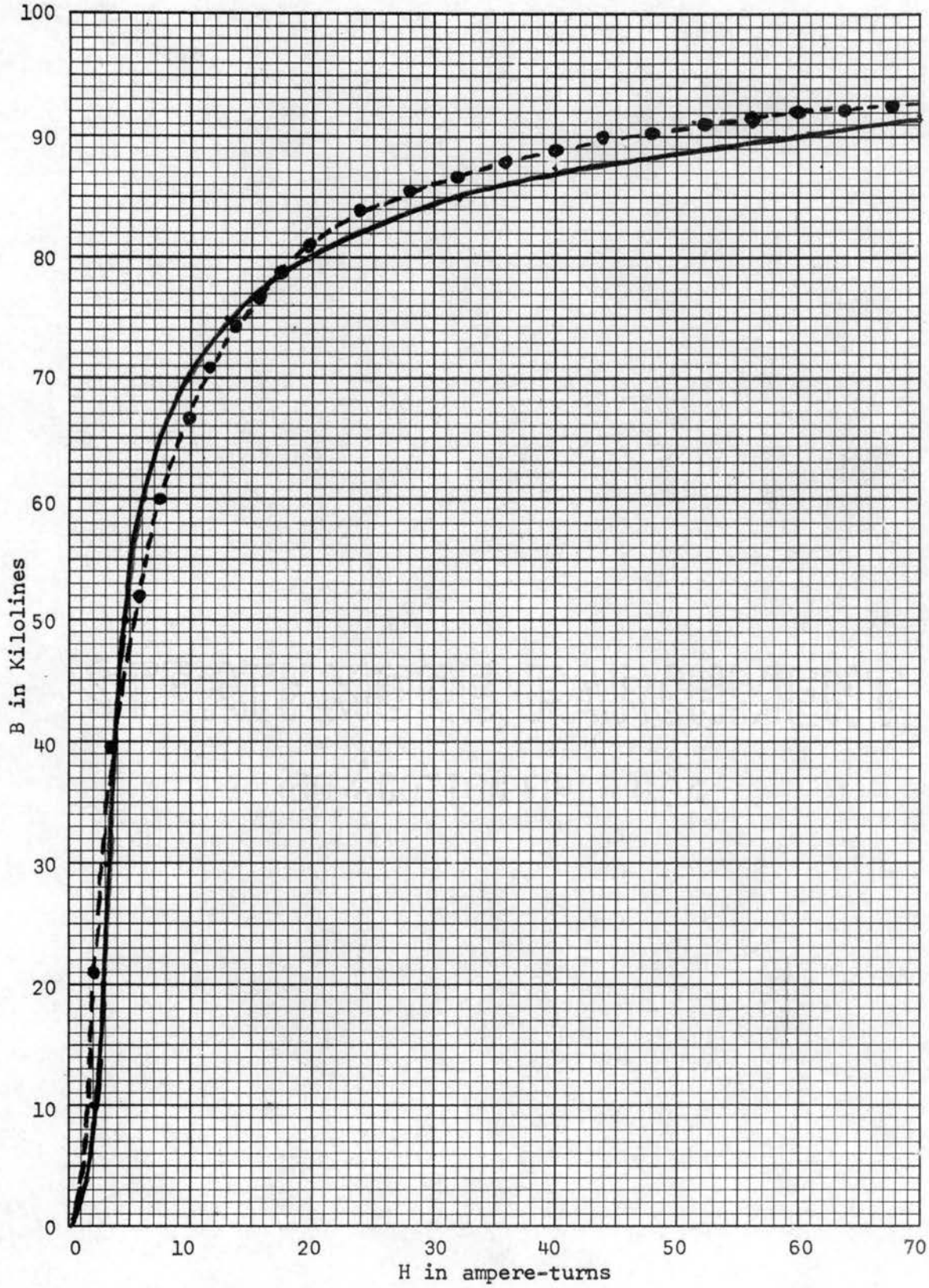


Figure B.1.1. B-H Curve of Transformer Steel

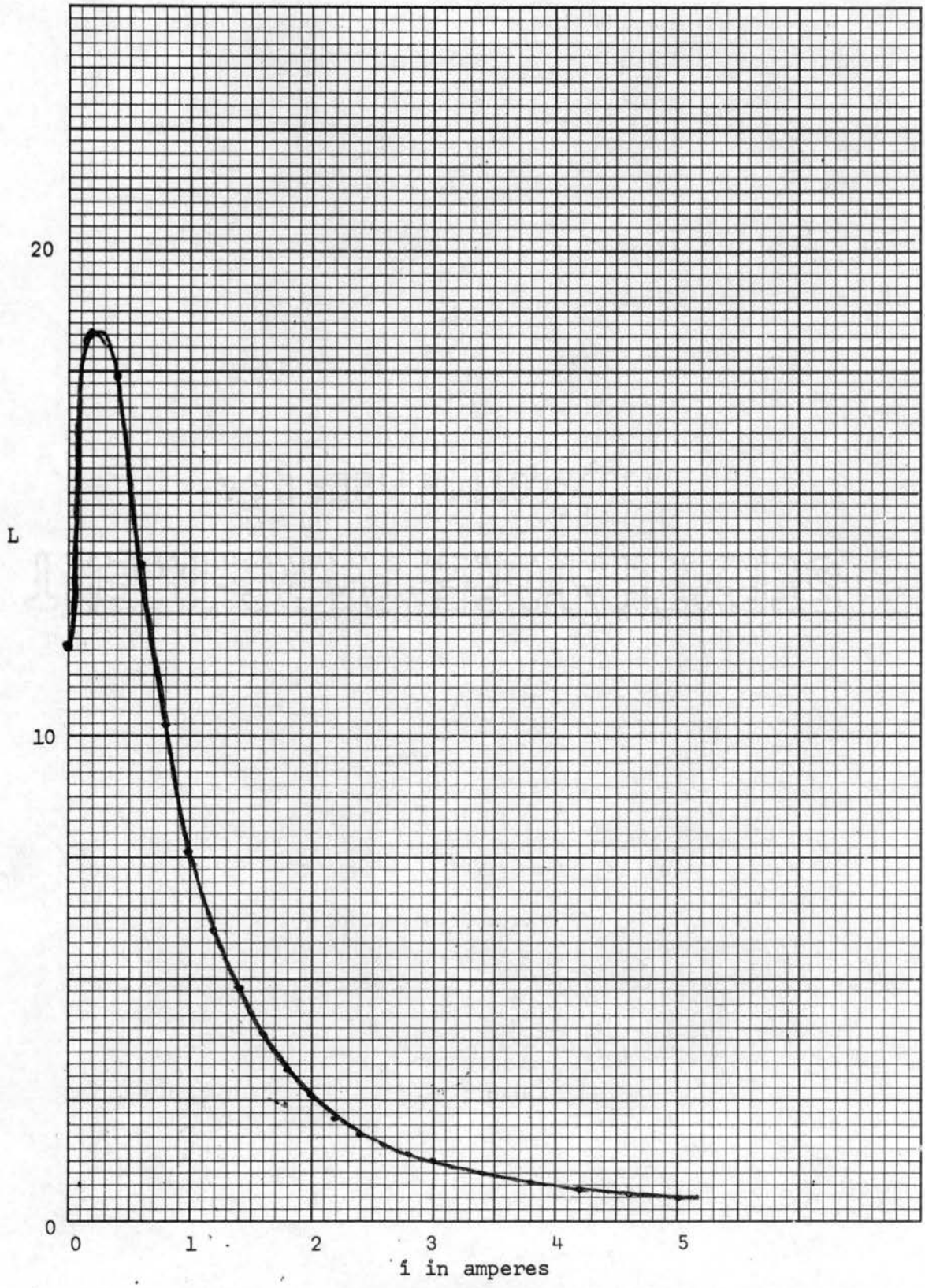


Figure B.1.2. L-i Curve for Saturable-Core Inductor

APPENDIX C

VACUUM TRIODE TWO-VARIABLE REPRESENTATION

The vacuum triode is represented by a power-series in two variables. A multiplying factor for the entire series is included to facilitate an adequate representation of the characteristics for the positive grid region. For this condition, the curvature of the constant-grid-voltage lines is downward, particularly in the region of low-plate voltage. For normal negative grid operation, the curvature is upward. The choice of the factor E in this correction factor must be made arbitrarily from inspection of the curves and is influenced somewhat by whether the most accurate representation is required in the regions of positive- or negative-grid voltage. For normal negative-grid operation, this factor is set to E = 0 so that the correction term becomes equal to 1 for all values of e_b.

The power series for this representation is of the form

$$i = \frac{e_b}{e_b + E}\left[A_1 e_b + A_2 e_b^2 + A_3 e_b^3 + A_4 e_c + A_5 e_c e_b + A_6 e_c e_b^2 + A_7 e_c^2 + A_8 e_c^2 e_b + A_9 e_c^3\right] . \qquad (C.1.1)$$

Figure C.1.1 illustrates the representation of the characteristics of a type 6J5 vacuum triode. The solid curves represent the normal

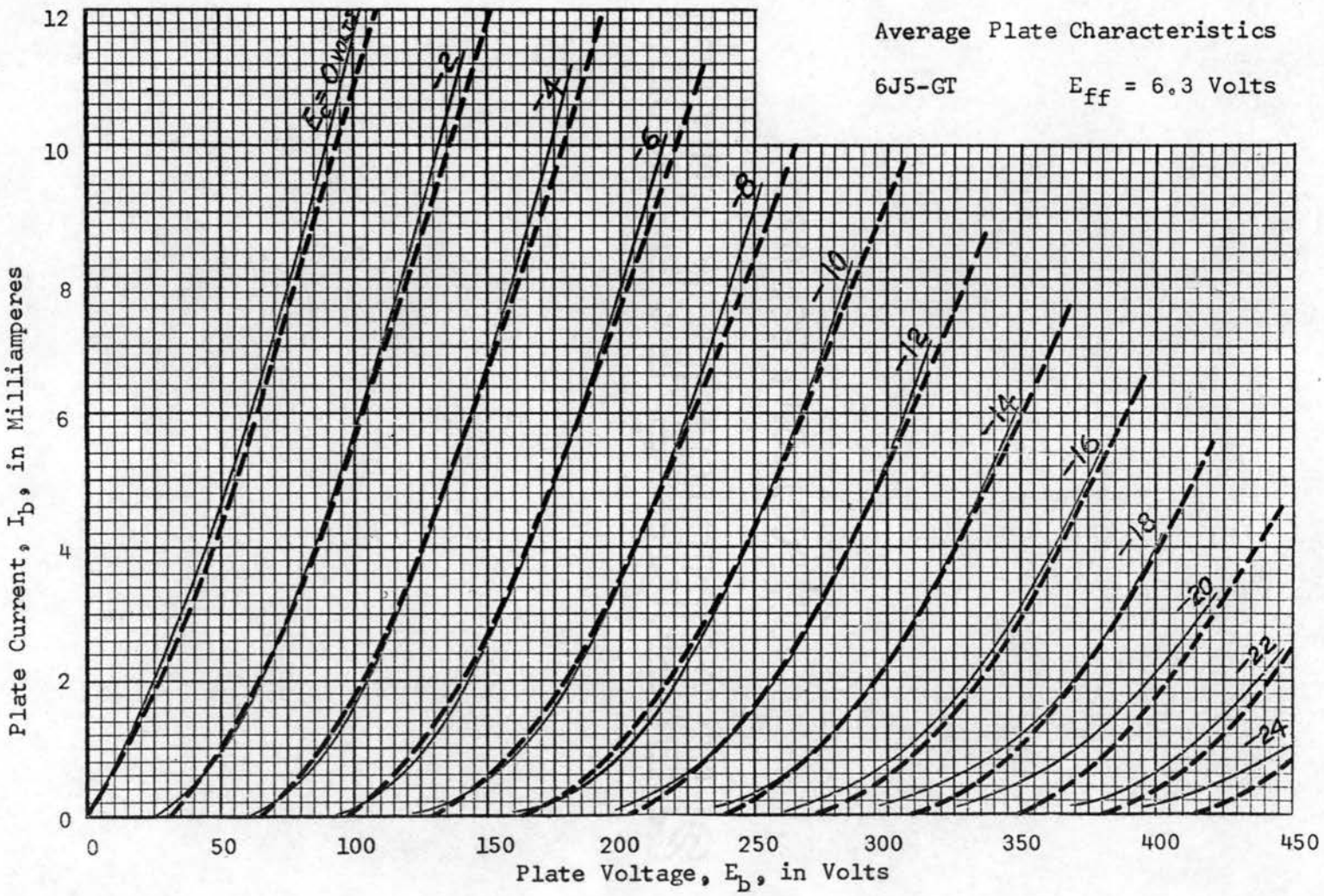


Figure C.1.1. Comparison of Graphical and Computed Characteristics for Vacuum Triode, Negative Grid Region

published characteristic curves of the device. The dotted curves illustrate the power-series calculated characteristics. The following values were obtained for the coefficients of the equation, fitted by a least-squares error criteria:

$E = 0.0$	$A_5 = 2.2686E-05$
$A_1 = 6.4023E-05$	$A_6 = -2.9570E-08$
$A_2 = 4.9485E-07$	$A_7 = 2.3818E-04$
$A_3 = -6.3400E-10$	$A_8 = -3.0620E-07$
$A_4 = 1.0467E-03$	$A_9 = 1.2100E-07$

Figure C.1.2 illustrates the characteristics for a type 6N7 vacuum triode. Normal operation of this tube is in the positive grid region. Again, the solid curve represents the published characteristics, the dotted curve, the characteristics calculated from the series representation.

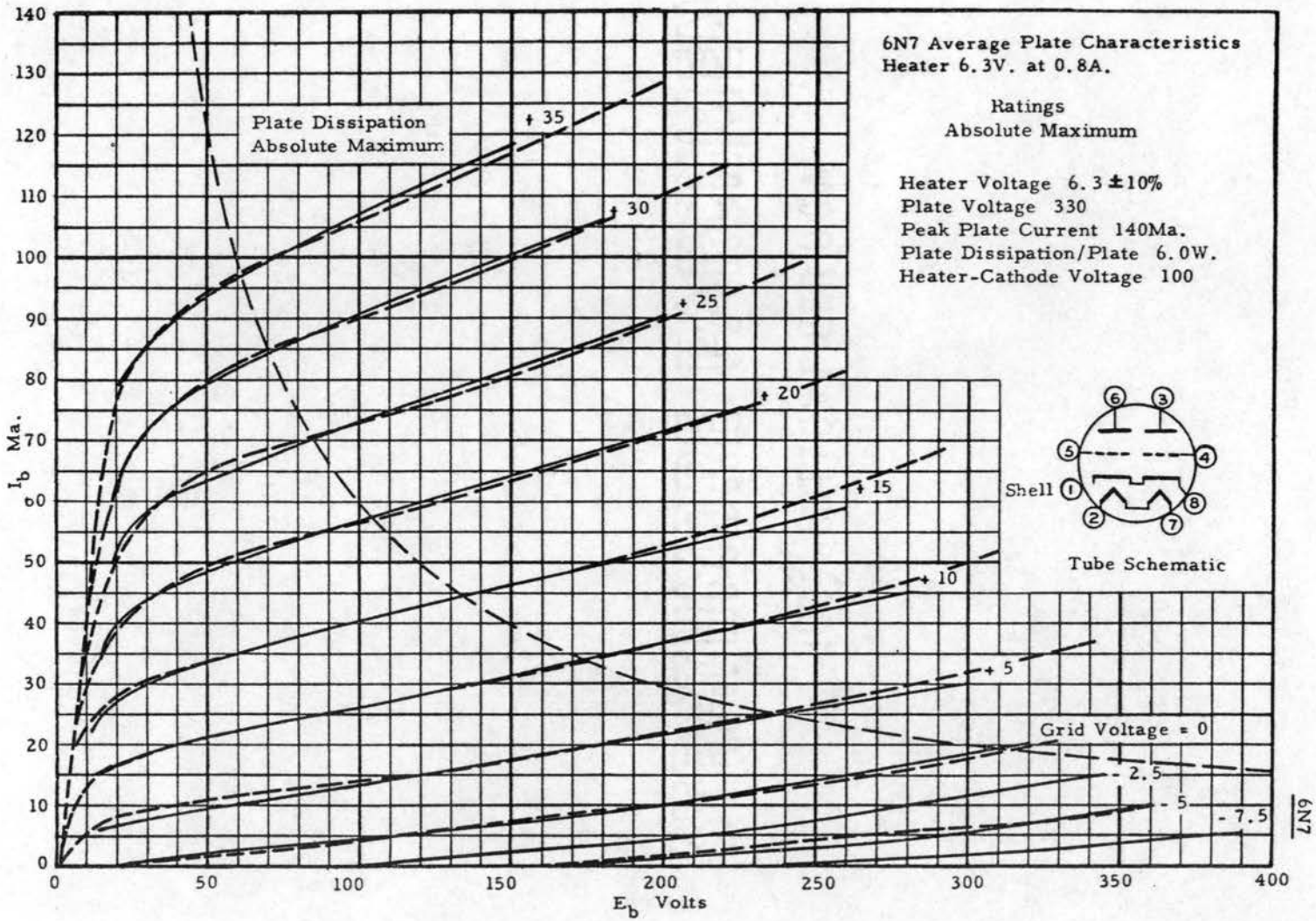


Figure C.1.2. Comparison of Graphical and Computed Characteristics for Positive Grid Region

APPENDIX D

FUNCTIONAL FLOW CHART OF FORMULATION PROGRAM

The conventional methods of illustrating computer program operations by flow charts do not lend themselves to a suitable description of the formulation program operations. Figures D.1.1 through D.1.4 present a nonstandard method of flow-charting which has been devised to better illustrate program operations.

The general sequence of operations depicted by these charts is from left to right and from top to bottom. The standard symbols for card and tape units are utilized. The appearance of a tape symbol on the left side of a page indicates that it is being read and is an input unit. The appearance of a tape symbol on the right side of a page indicates that it is being written onto. The number inside the tape unit symbol indicates the unit in use.

The square blocks symbolize matrices as they are being retrieved from temporary tape storage or as they are created and written onto temporary tape storage. The number or alphameric symbol inside these square blocks is the matrix identification as assigned in the algorithm. The sequence of these matrices, as read or written, is from left to right and top to bottom. The large circles indicate matrices held in temporary storage in core memory. The oval blocks indicate mathematical manipulations, which are illustrated in detail in Figure 5.1.1.

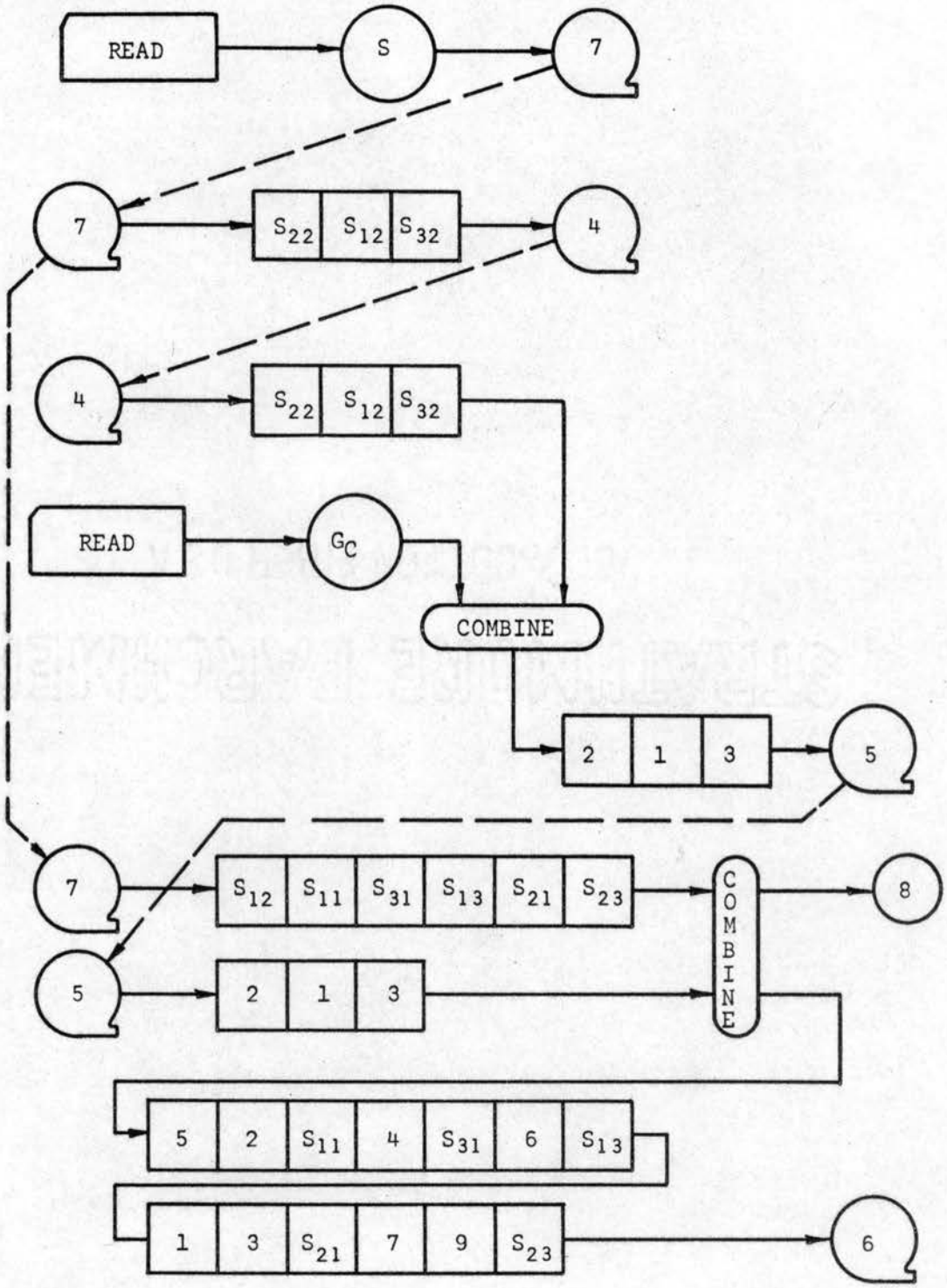


Figure D.1.1. Operational Flow Chart of Formulation Program, Part 1

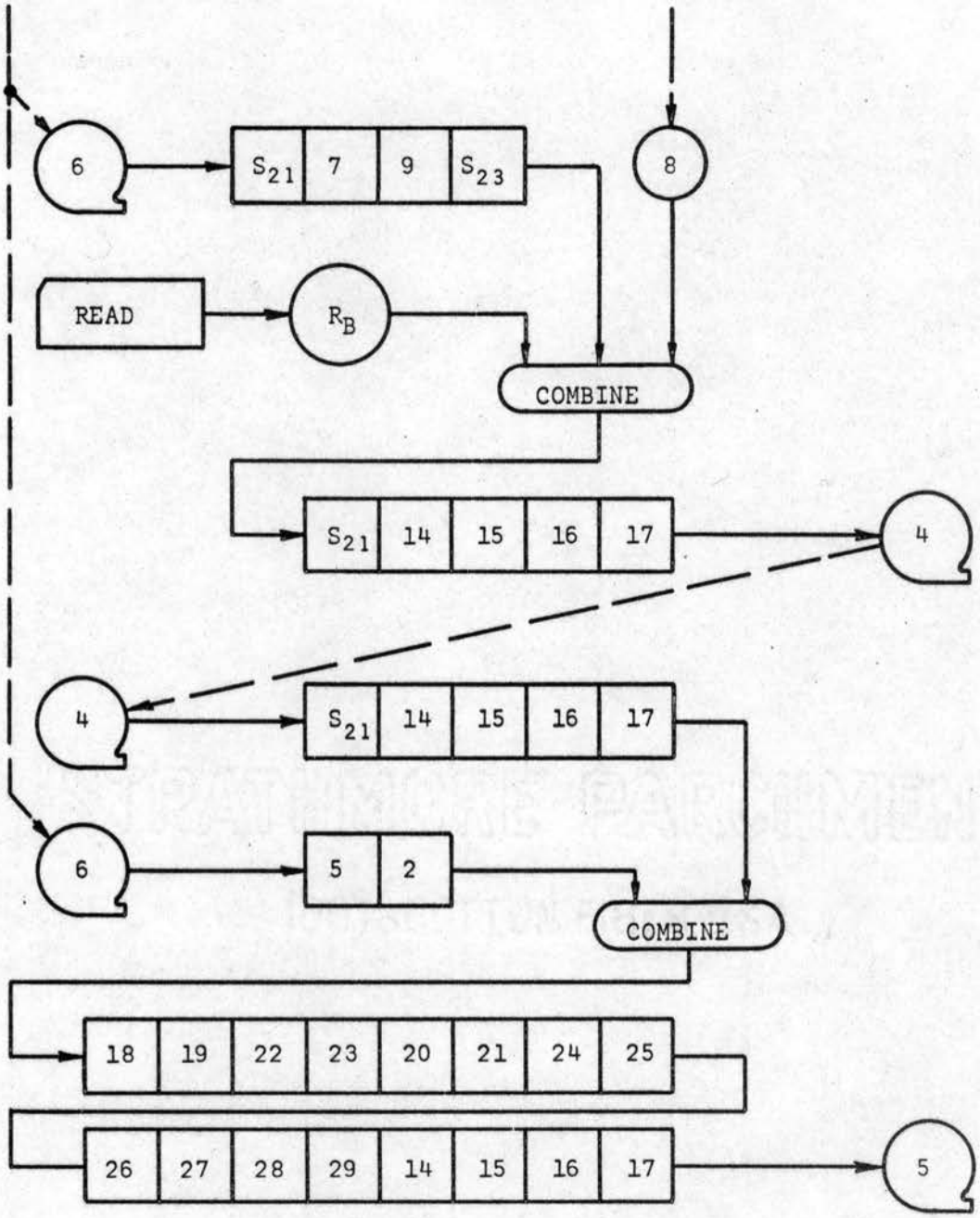


Figure D.1.2. Operational Flow Chart of Formulation Program, Part 2

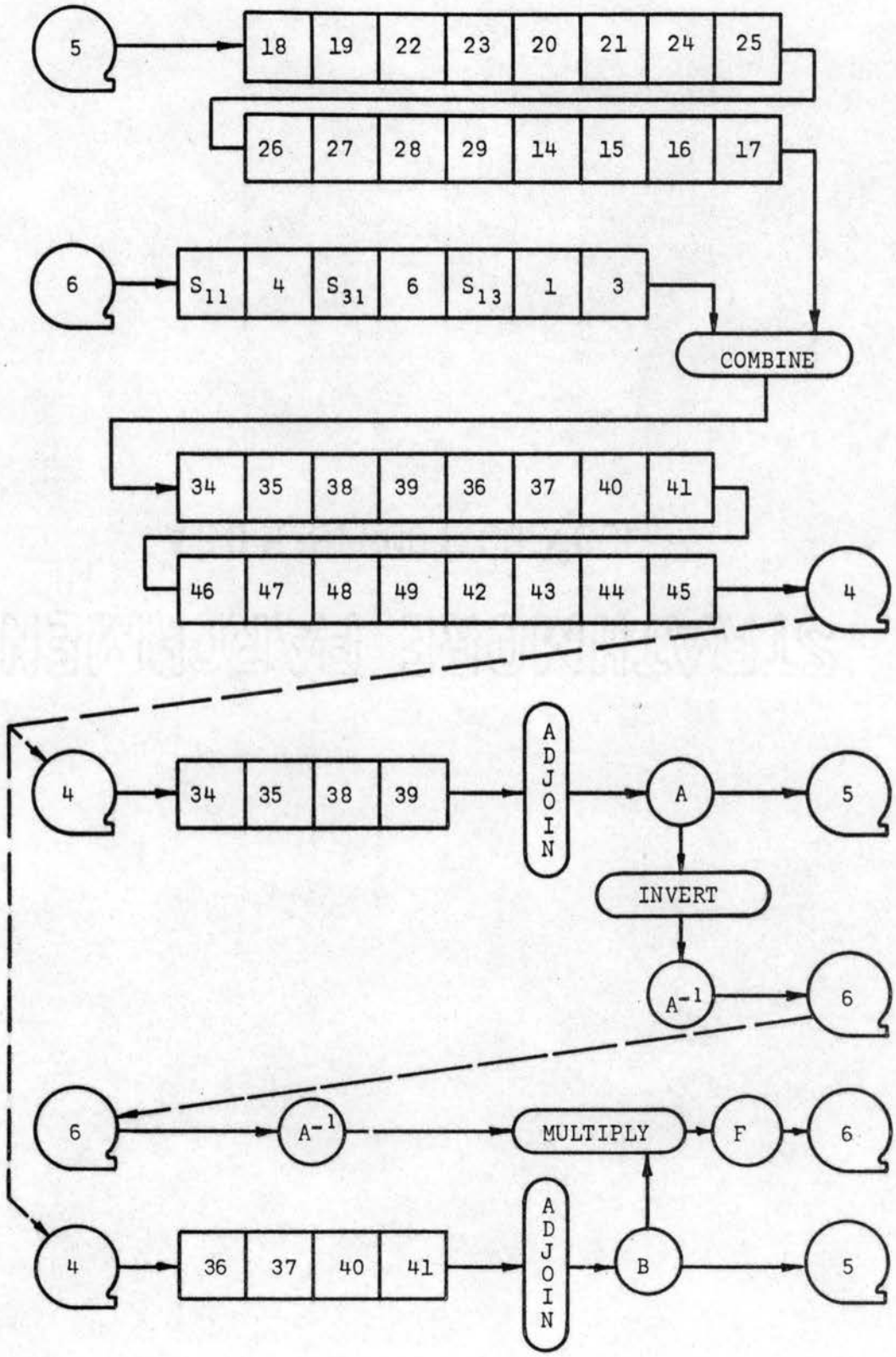


Figure D.1.3. Operational Flow Chart of Formulation Program, Part 3

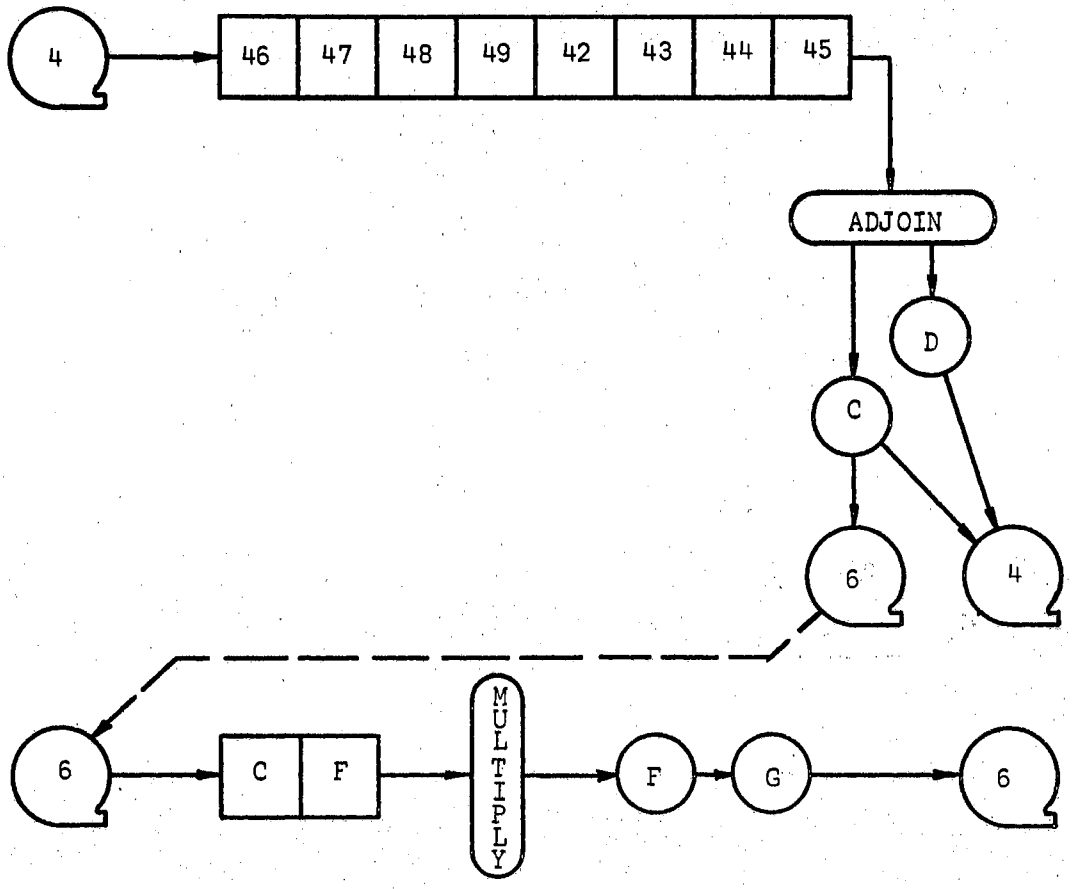


Figure D.1.4. Operational Flow Chart of Formulation Program, Part 4

The solid lines indicate the flow of information, and the dotted lines indicate the appropriate rewind or backspace operations for the tape unit with which they are associated.

For example: The first (top) line of Figure D.1.1 means that the $\underline{S}$ matrix is read from cards (card symbol on left side of page) into temporary core storage (the large circle) and then is written onto tape unit 7 (tape symbol on right side of page). This unit is then rewound or backspaced (dotted line down and to left), and the matrices $\underline{S}_{22}$, $\underline{S}_{12}$, and $\underline{S}_{32}$ are read in that order (square blocks) and written onto tape unit 4.

Comparison of Figures D.1.1 through D.1.4 with Figure 5.1.1 will reveal that the sequences of matrices written onto any particular tape are the same as the vertical rows of matrices shown in Figure 5.1.1. This correlation is maintained throughout the presentations so that it is possible to identify operations by both the physical manipulations and the accompanying mathematical operations. The actual mathematical operations at each step are presented in Equations 3.3.1 through 3.3.13.

APPENDIX E

DEVELOPMENT OF ITERATION EQUATIONS

The iteration process is based on the Newton-Raphson method. The iteration equation is easily developed and is used in several of the analysis programs.

However, for the functional relationship v = f (i) where f is logarithmic, the regular Newton-Raphson process diverges for v < 0. Since v, the terminal voltage of the device, can be negative, this is a very undesirable situation.

For this study a new iteration process (also based on Newton-Raphson) was developed for the situation where v < 0.

Consider first the case for v > 0 illustrated by Figure E.1.1.

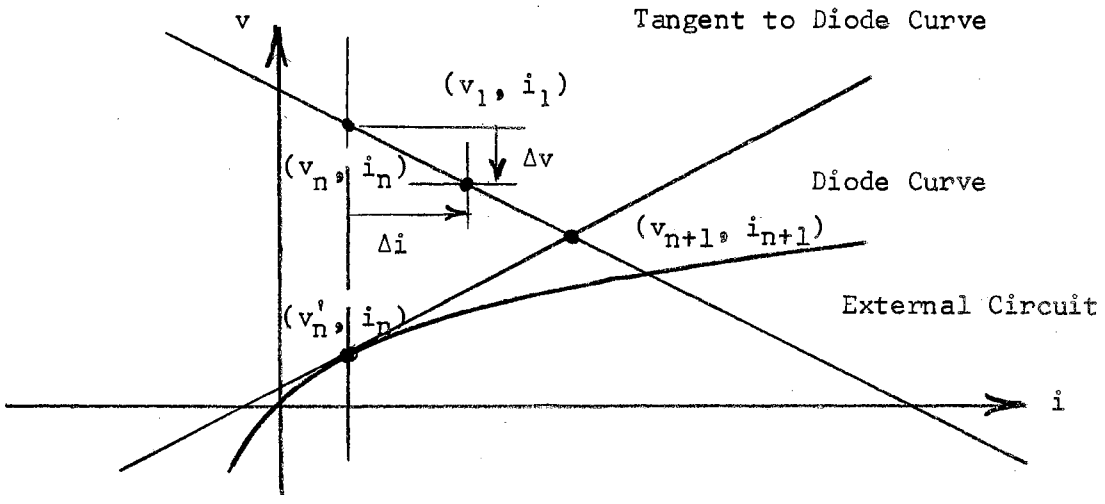


Figure E.1.1. Graph of Diode and External Circuit for v > 0

Introduce a change in current Δi and calculate the voltage for the circuit, v_1. From this

$$\Delta v = v_1 - v_n \quad . \tag{E.1.1}$$

Assuming a linear external circuit, the equation representing the load line is then

$$v = v_n - (i - i_n) \frac{\Delta v}{\Delta i} \quad . \tag{E.1.2}$$

The equation of the diode curve is

$$i = I_s \, (e^{bv} - 1) \tag{E.1.3}$$

which may be written in inverse form

$$v = \frac{1}{b} \ln \left(\frac{i + I_s}{I_s} \right) . \tag{E.1.4}$$

The tangent to the diode line has slope

$$\frac{dv}{di} = \frac{1}{b \, (i + I_s)} \quad , \tag{E.1.5}$$

so that the equation of the tangent line through (v_n', i_n) is

$$v = \frac{1}{b} \left[\frac{i - i_n}{i_n + I_s} + \ln \left(\frac{i_n + I_s}{I_s} \right) \right] . \tag{E.1.6}$$

The intersection of the load line and tangent line is at (v_{n+1}, i_{n+1}). Substituting these values into Equations E.1.2 and E.1.6, the right-hand sides can be equated to yield

$$v_n - (i_{n+1} - i_n) \frac{\Delta v}{\Delta i} = \frac{1}{b} \left[\frac{i_{n+1} - i_n}{i_n + I_s} + \ln \left(\frac{i_n + I_s}{I_s} \right) \right] . \tag{E.1.7}$$

Solving for i_{n+1} in Equation E.1.7 leads to

$$i_{n+1} = i_n + \frac{\Delta i}{\Delta v} \frac{(I_n + I_s)}{\frac{\Delta i}{\Delta v} + b\,(i_n + I_s)} \left[bv_n - \ell n \left(\frac{i_n + I_s}{I_s} \right) \right] . \quad (E.1.8)$$

Making substitutions

$$h = \frac{\Delta i}{\Delta v} \qquad \text{and} \qquad I = i_n + I_s$$

Equation E.1.8 can be written

$$i_{n+1} = i_n + \frac{hI}{h + bI} [bv_n - \ell n\,(I) + \ell n\,(I_s)] \quad . \qquad (E.1.9)$$

This is the iteration equation for i_{n+1} for the case $v > 0$.

Consider now the case for $v < 0$ illustrated in Figure E.1.2.

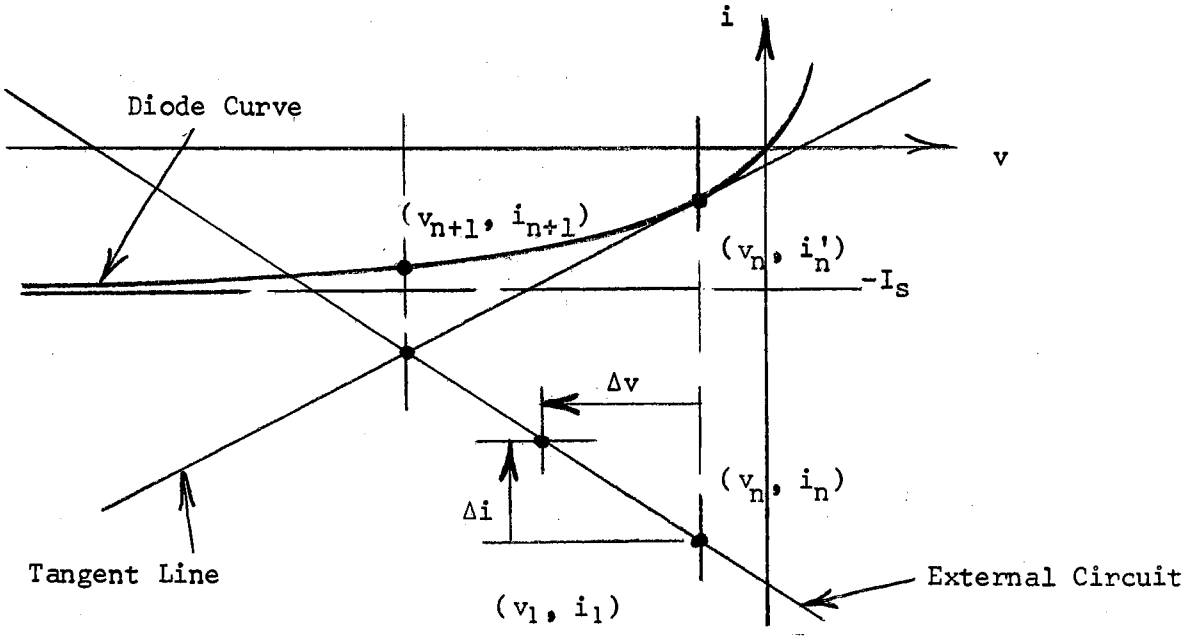


Figure E.1.2. Graph of Diode and External Circuit for $v < 0$

As before, a change in current, Δi, is introduced from which Δv may be obtained by calculation of the point (v_1, i_1).

The equation of the load line for the external circuit is

$$i = i_n - (v - v_n) \frac{\Delta i}{\Delta v} \quad . \qquad \text{(E.1.10)}$$

The equation of the diode curve is given by Equation E.1.3. The slope of the tangent to this curve is

$$\frac{di}{dv} = b\, I_s\, e^{bv} \quad . \qquad \text{(E.1.11)}$$

The equation of the tangent line through the point $(v_n, i_{n'})$ is

$$i = b\, I_s\, e^{bv_n} (v - v_n) + I_s (e^{bv_n} - 1) \quad . \qquad \text{(E.1.12)}$$

The load line and tangent line intersect at (v_{n+1}, i_{n+1}). Substituting these values into Equations E.1.10 and E.1.12, the right-hand sides can be equated to obtain

$$i_n - (v_{n+1} - v_n) \frac{\Delta i}{\Delta v} = b\, I_s\, e^{bv_n} (v_{n+1} - v_n) + I_s (e^{bv_n} - 1) \quad . \qquad \text{(E.1.13)}$$

This equation can be solved for v_{n+1} and yields

$$v_{n+1} = v_n + \frac{(i_n + I_s) - I_s\, e^{bv_n}}{\frac{\Delta i}{\Delta v} + b\, I_s\, e^{bv_n}} \quad . \qquad \text{(E.1.14)}$$

Substituting

$$h = \frac{\Delta i}{\Delta v} \quad , \quad I = i_n + I_s \quad , \quad \text{and} \quad I_1 = I_s\, e^{bv_n}$$

into Equation E.1.14 results in

$$v_{n+1} = v_n + \frac{I - I_1}{h + b\, I_1} \quad . \qquad \text{(E.1.15)}$$

To obtain the value of $i_{n'+1}$ required to set the current driver, v_{n+1} is used in the diode equation to obtain

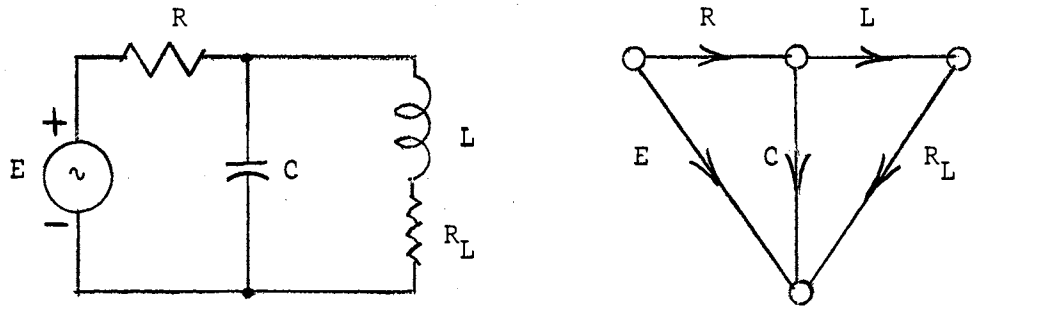
$$i_{n+1} = I_s \left(e^{bv_{n+1}} - 1\right) . \qquad \text{(E.1.16)}$$

Equations E.1.15 and E.1.16 provide the iteration method for the case $v < 0$.

APPENDIX F

EXAMPLES OF ANALYSIS

F.1 Transformer Equivalent Circuit with Shunt Capacity from Transmission Line. The circuit of Figure F.1.1(a) is used to represent the transformer and line capacity. The corresponding linear graph is shown in Figure F.1.1(b).



(a) (b)

Figure F.1.1. Schematic Diagram and Linear Graph of Transformer

The load and core losses for the transformer are represented by R_L. The inductance L represents the saturating iron-core model of the transformer and windings. The transmission line losses and capacity are represented by R and C. The exciting generator is sinusoidal,

represented by the voltage driver E.

The objective of the analysis is to investigate the effect of the nonlinear saturating inductance of the transformer under conditions of light loading, introduced by small values of R_L.

The linear graph is redrawn in Figure F.1.2 with the tree indicated by heavy lines.

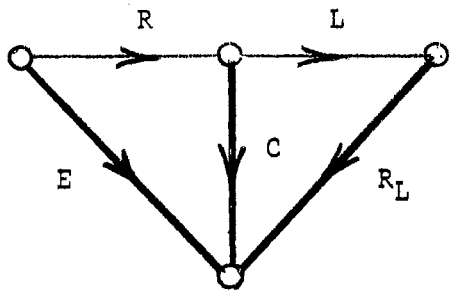


Figure F.1.2. Linear Graph with Tree Selected

From the graph the following cutset matrix can be generated

$$\begin{array}{c} \\ C \\ R_L \\ E \end{array} \begin{array}{c} \begin{array}{ccc|cc} C & R_L & E & L & R \end{array} \\ \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & -1 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{array}\right] \end{array} = \underline{P} \ . \qquad \text{(F.1.1)}$$

The rows and columns of this matrix are labeled to show their correspondence with edges of the graph. The matrix $\underline{P}$ is in the form

$$\underline{P} = [\ \underline{U} \mid \underline{S}\ ] \ , \qquad \text{(F.1.2)}$$

so that the matrix $\underline{S}$ takes the form

$$\underline{S} = \begin{bmatrix} 1 & -1 \\ -1 & 0 \\ 0 & 1 \end{bmatrix} . \qquad (F.1.3)$$

This matrix provides all the information required by the algorithm concerning the interconnections of the elements of the network.

The input-data cards required by the formulation program are

```
0                                          (Card 1)
1    1   1   1    1   0                    (Card 2)
1    1   1   1.0                           (Card 3)
1    1   2  -1.0                           (Card 4)
1    2   1  -1.0                           (Card 5)
1    3   2   1.0                           (Card 6)
0                                          (Card 7)
10.0                                       (Card 8)
1.0                                        (Card 9)
```

The 0 in card 1 indicates that the formulation program is to be executed.

The numbers in card 2 indicate, in order, the following items:

(a) The number of capacitors

(b) The number of resistors in the tree

(c) The number of voltage drivers

(d) The number of inductors

(e) The number of resistors in the co-tree

(f) The number of current drivers

Cards 3 through 6 are the non-zero entries in the $\underline{S}$ matrix. The

second and third numbers are the row and column index of the entry; the fourth number is the entry value. The first number in the card is for a control function.

The 0 in the seventh card signals the termination of the S matrix entries.

The numbers in cards 8 and 9 are the values of resistance for R and R_L, respectively.

The input-data cards required by the time solution program are

```
 0   0   0    0       0     1   0   1                    (Card 10)
 6   0   1   25.0    60.0                                (Card 11)
 8   1   1    1.30743       0.5942   1.5                 (Card 12)
 9   1.30743                                             (Card 13)
 9   0.00001                                             (Card 14)
 0                                                       (Card 15)
13   1   0    0                                          (Card 16)
14   0.025    0.0005                                     (Card 17)
15   0                                                   (Card 18)
 1                                                       (Card 19)
```

Card 10 specifies which nonlinear drivers and elements are present in the circuit.

Card 11 specifies the voltage and frequency of the sinusoidal voltage driver.

Card 12 specifies the parameters of the nonlinear inductance model.

Cards 13 and 14 specify the initial values of the inductance and capacitance. The inductance value is modified during solution by the

nonlinear model, but the capacitor value remains unchanged.

Card 15 indicates that there are no non-zero initial values for drivers.

Card 16 specifies several control parameters for the program to control test print-outs and other special program operations.

Card 17 specifies the total time and integration interval for the integration routine.

Card 18 indicates that there are no initial conditions to be entered for the capacitor voltage and inductor current.

Card 19 controls the print-interval for the final printed output.

Figures F.1.3 through F.1.8 are plots of typical voltage and current solutions obtained on several analysis runs with different values of applied voltage. The effect of the saturating inductor is clearly evident. The component values in this circuit are typical but not related to a particular transformer. The general form of the analysis results compare favorably with observed behavior of such devices.

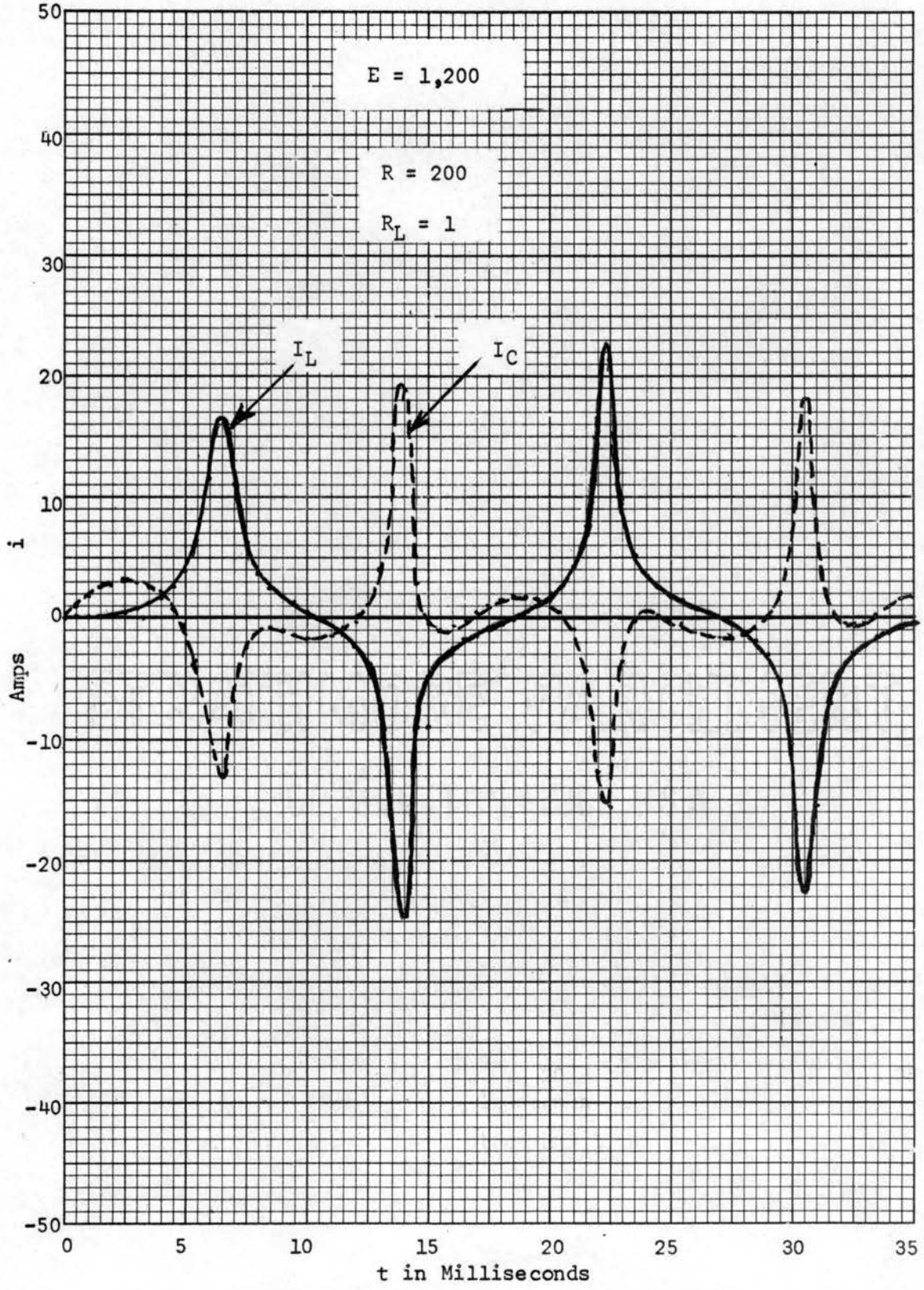


Figure F.1.3. Capacitor and Inductor Currents

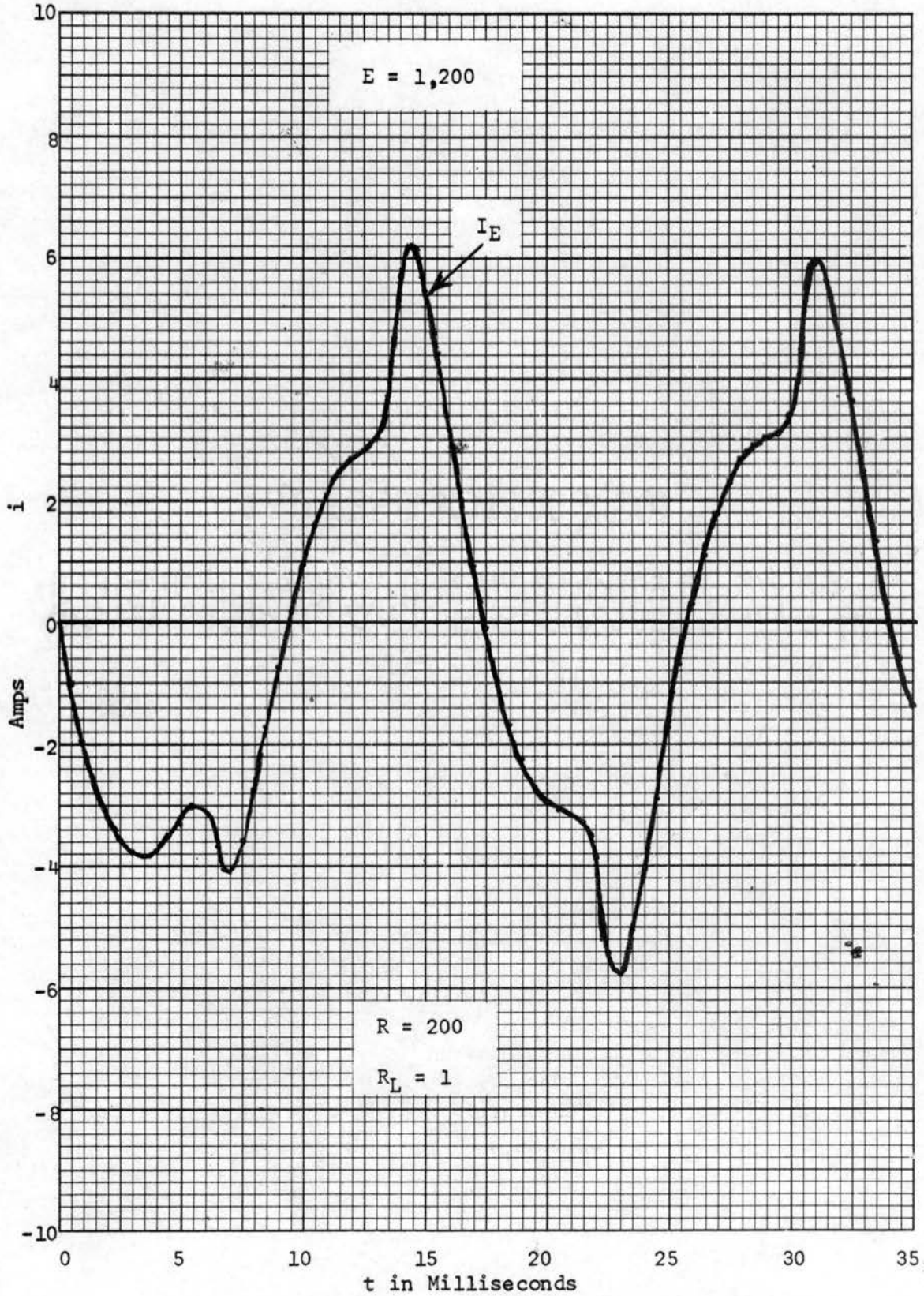


Figure F.1.4. Input Current from Driver

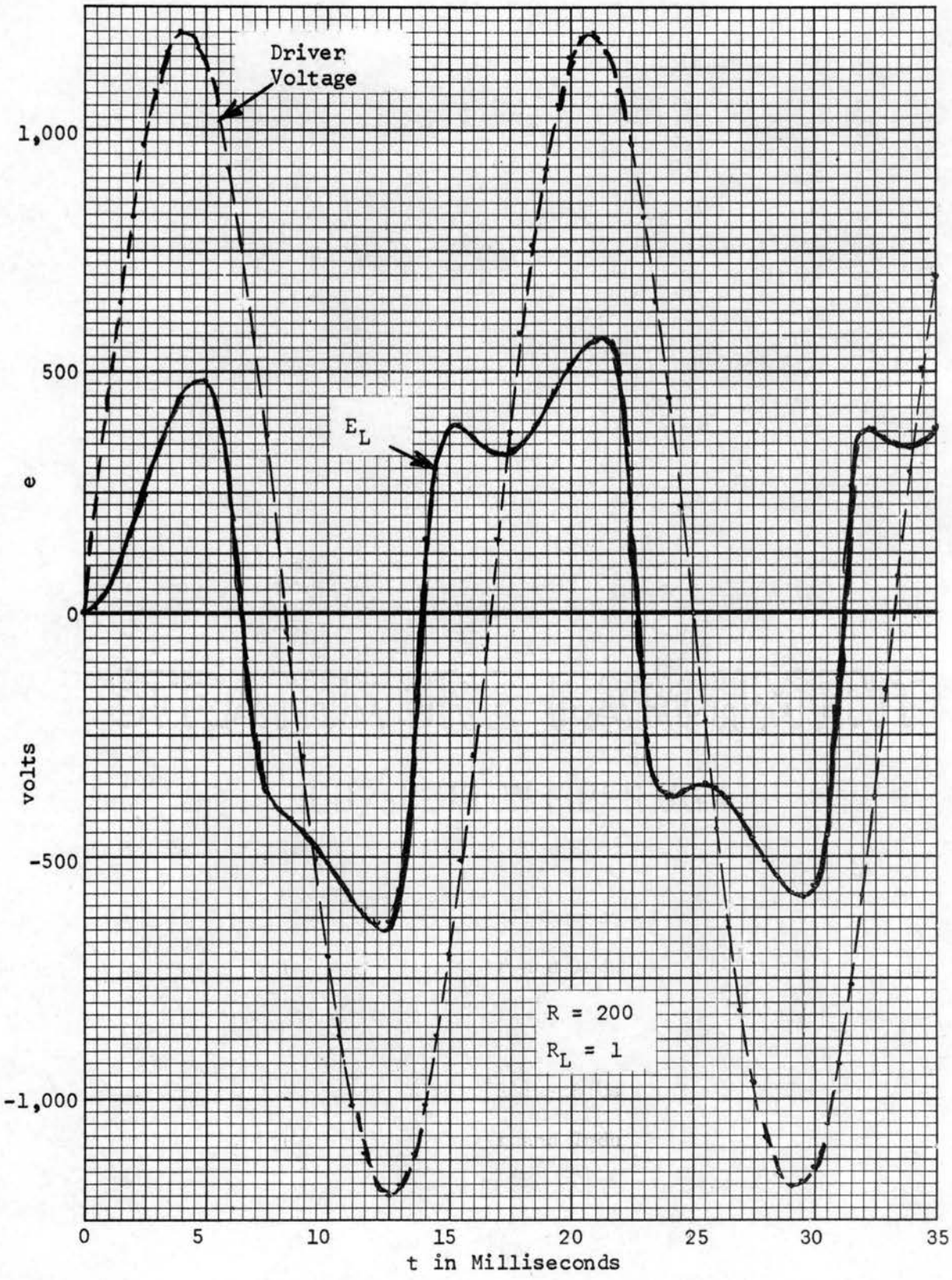


Figure F.1.5. Driver and Inductor Voltages

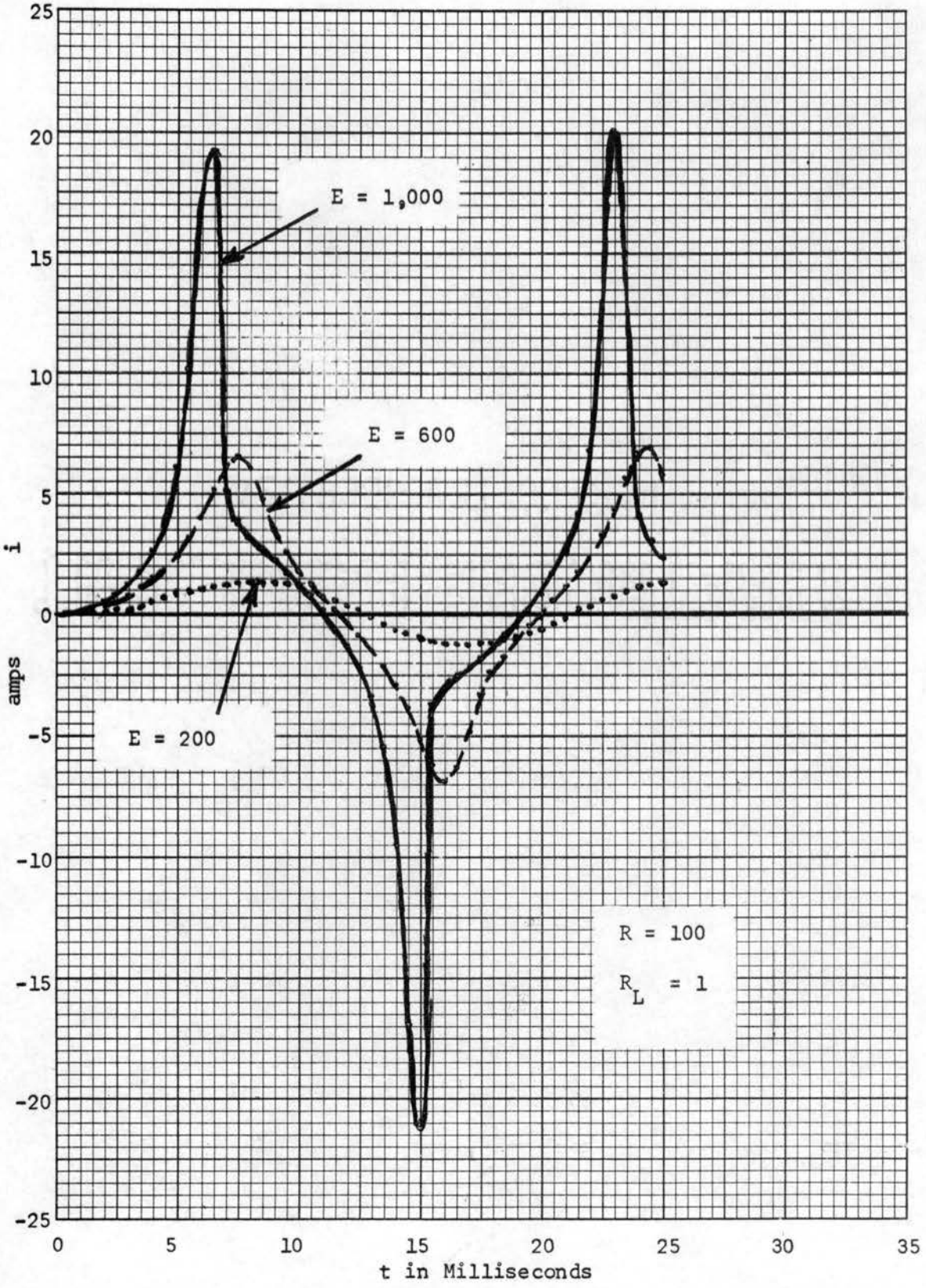


Figure F.1.6. Change in I_L for Increasing Driver Voltage

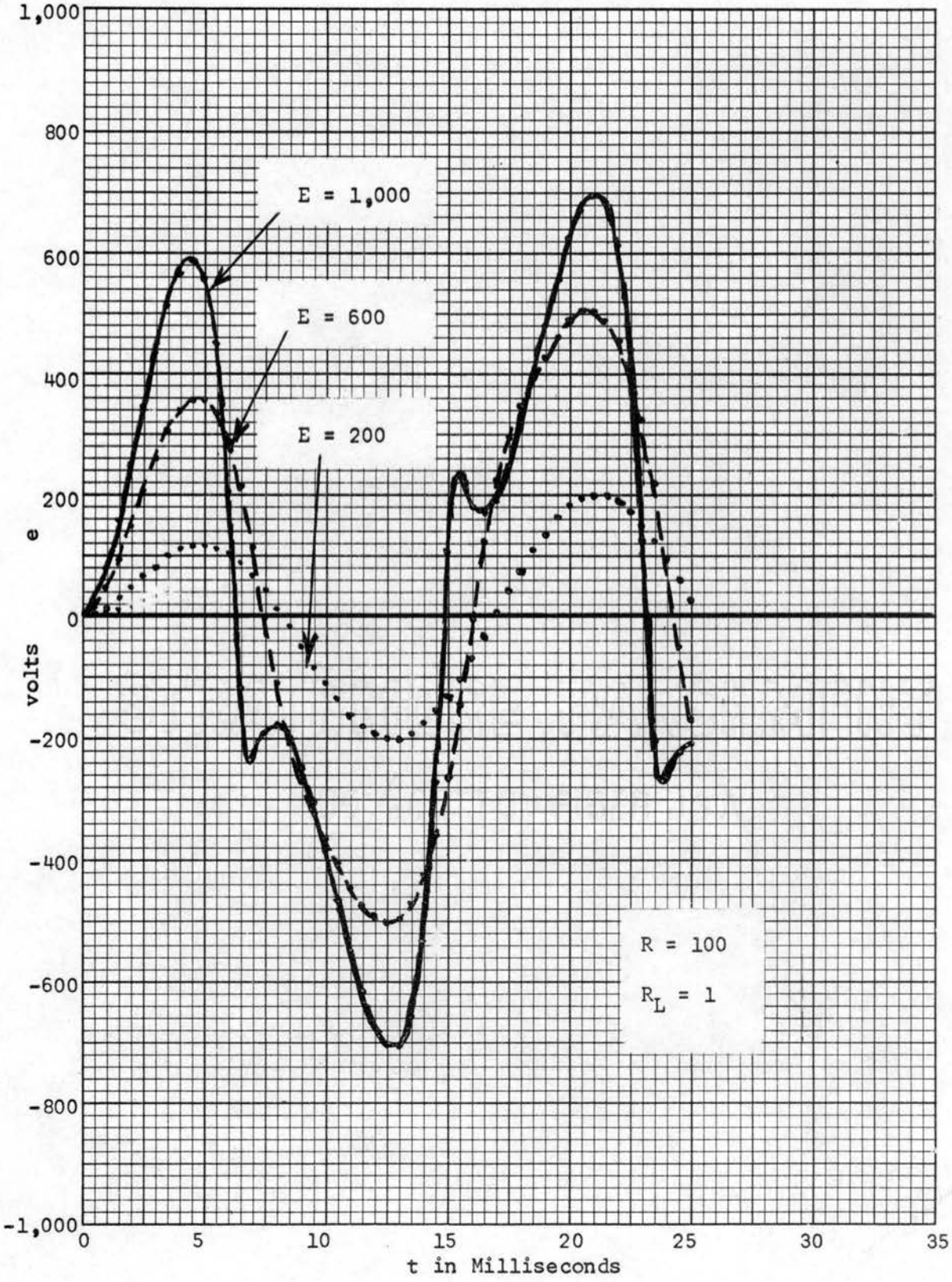


Figure F.1.7. Change in E_L for Increasing Driver Voltage

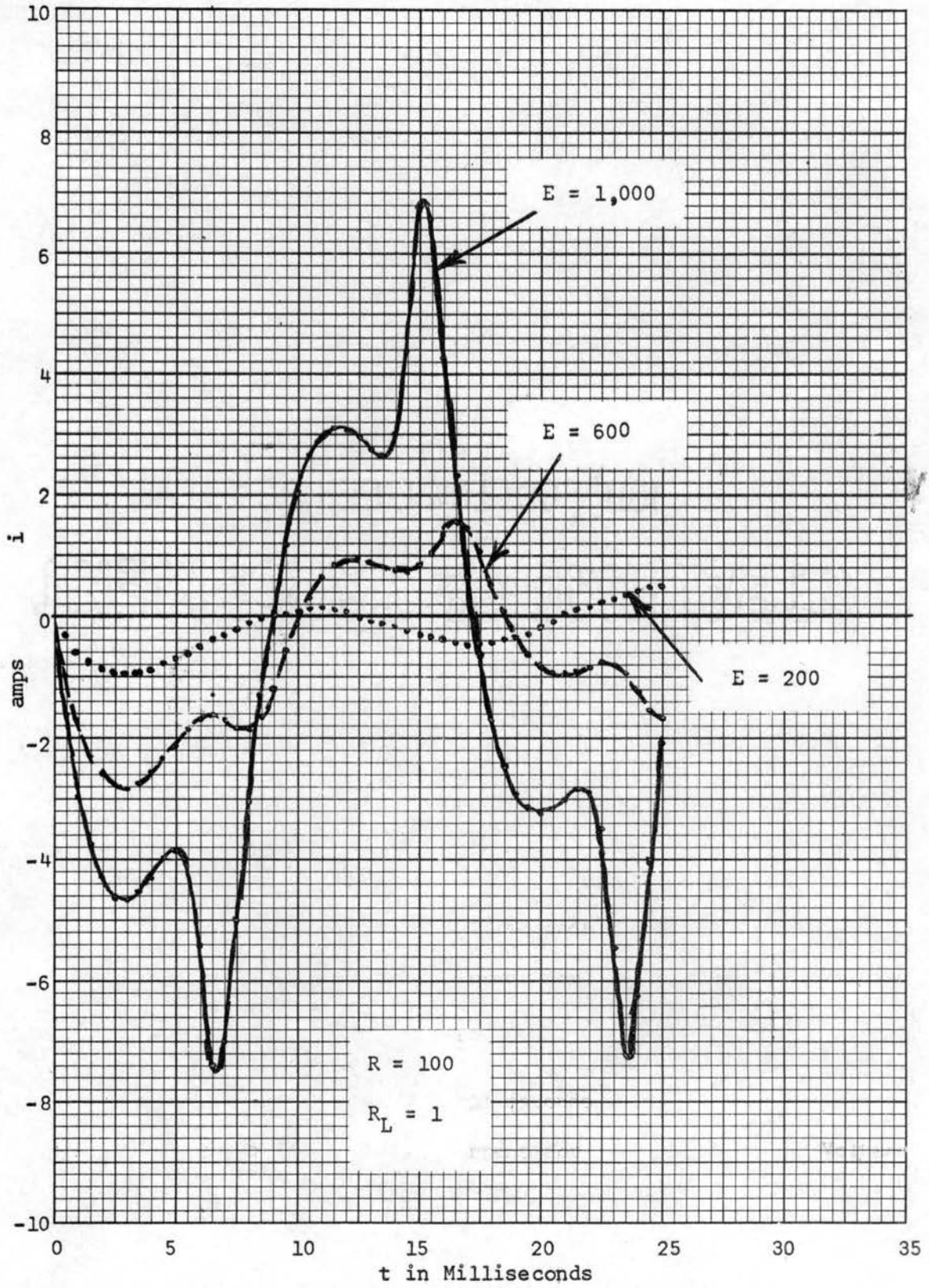


Figure F.1.8. Change in Input Current for Increasing Driver Voltage

F.2 Vacuum Triode in a Colpitts-Type Oscillator. The circuit to be studied is shown in Figure F.2.1.

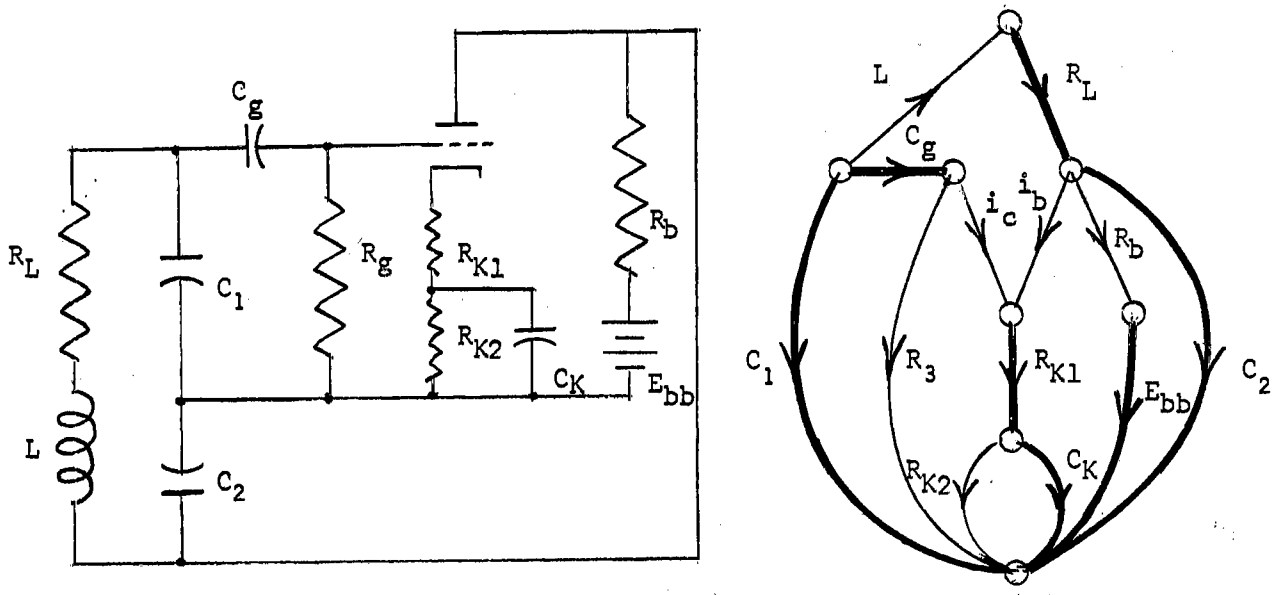


Figure F.2.1. Diagram and Linear Graph of Colpitts Oscillator

The S matrix for this graph is:

	L	R_g	R_{K2}	R_b	i_c	i_b
C_1	1	1	0	0	1	0
C_2	-1	0	0	1	0	1
C_g	0	-1	0	0	-1	0
C_k	0	0	1	0	-1	-1
R_{K1}	0	0	0	0	-1	-1
R_L	-1	0	0	0	0	0
E_{bb}	0	0	0	-1	0	0

The current driver i_b is specified as a nonlinear model of the two-variable vacuum tube triode. The two input variables to this model are specified as the voltages across i_c and i_b. Here is a case where it is necessary to have the total solution available, including the current-driver voltages.

Figure F.2.2 indicates a first-solution run on this circuit. Inadequate time was allowed for establishment of a limit cycle. The capacitor voltages and inductor current were entered as initial conditions, and the analysis was continued in a second run. The results from this second run are shown in Figure F.2.3.

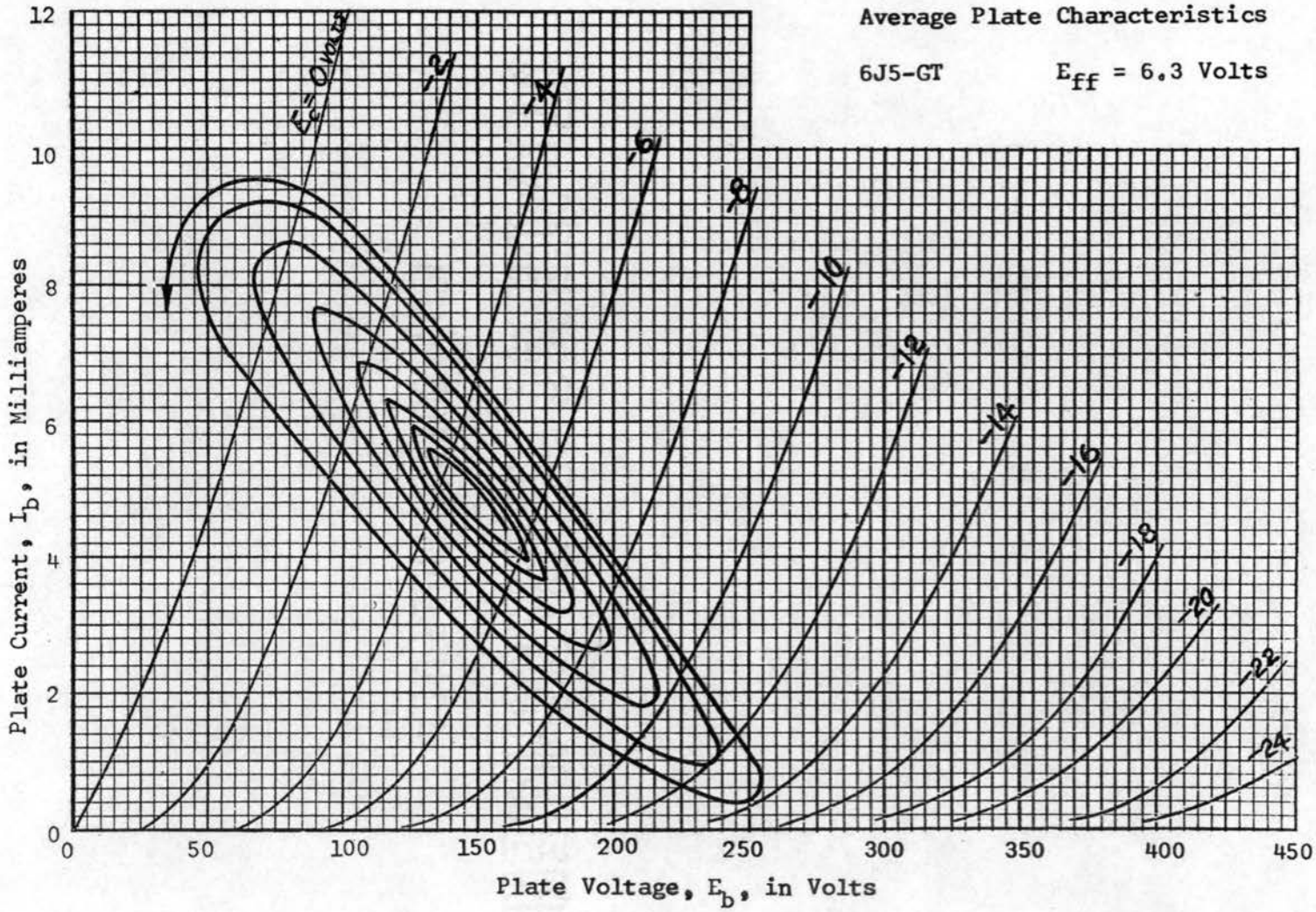


Figure F.2.2. Computed e_b - i_b Response for Vacuum Triode Colpitts Oscillator

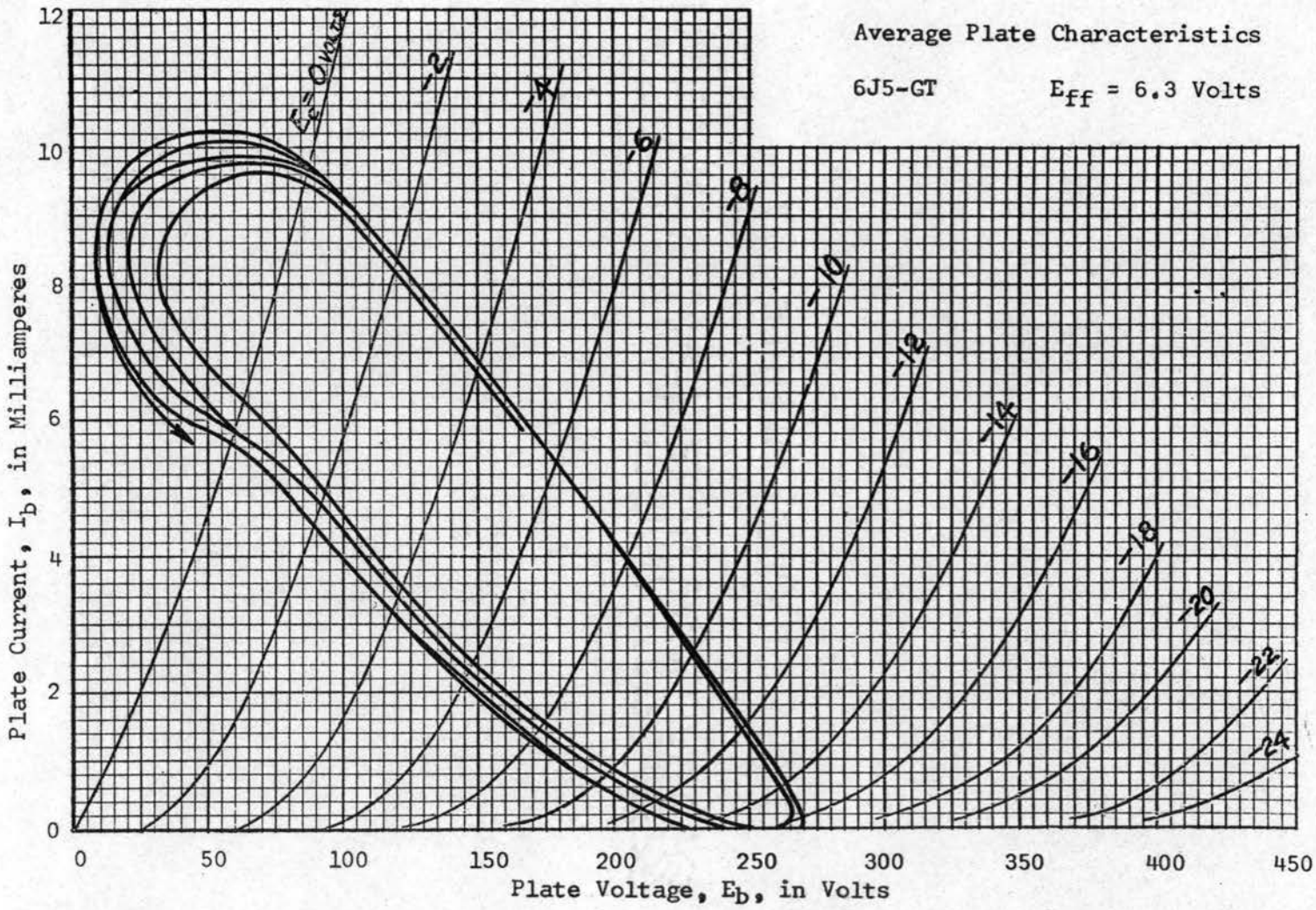


Figure F.2.3. Continuation of Colpitts Oscillator Solution by Imposing Initial Conditions on Circuit Components

F.3 Vacuum Triode with Inductive Plate Load. The circuit to be investigated is shown in Figure F.3.1 with the associated linear graph.

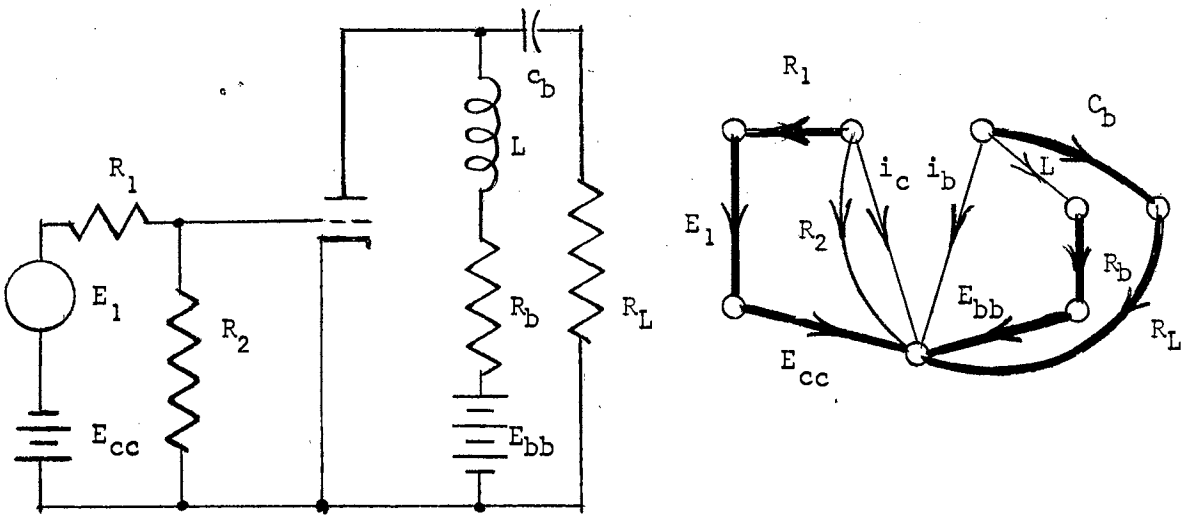


Figure F.3.1. Diagram and Linear Graph of Amplifier

Applied to the grid by means of E_i and E_{cc} are various conditions of fixed grid bias and input signals. The following signals were applied:

(1) A constant amplitude sine

(2) A sine wave of constantly increasing amplitude

(3) A rectangular pulse train with rise-and-fall times of length equal to five integration steps

(4) A single rectangular pulse of large negative amplitude and rise-and-fall time equal to one integration step

(5) A single rectangular pulse of large negative amplitude and

rise-and-fall time of several integration steps

Figures F.3.2 through F.3.5 illustrate the results of this analysis.

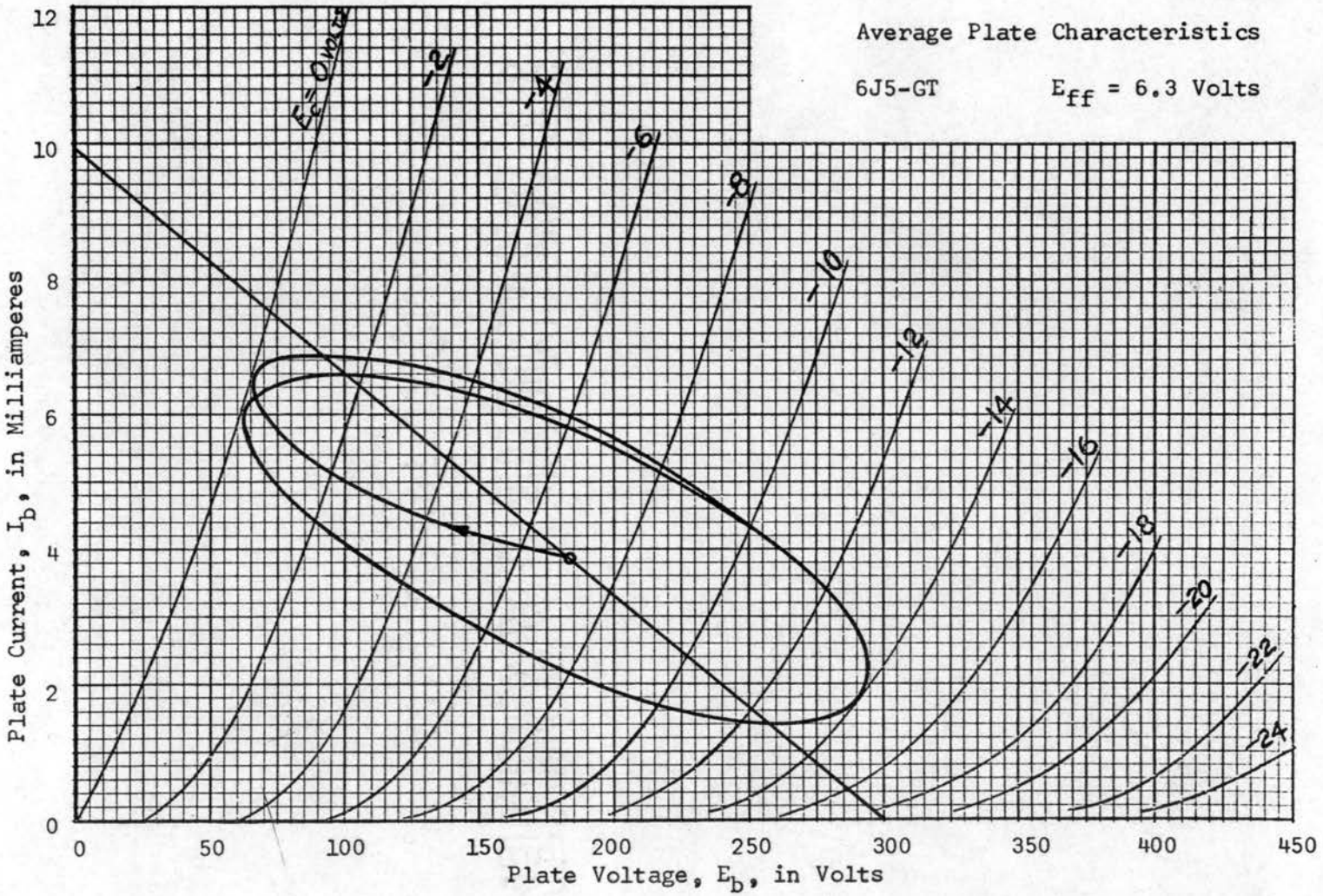


Figure F.3.2. Inductive Plate Load Response to Suddenly Applied Constant Sinusoid

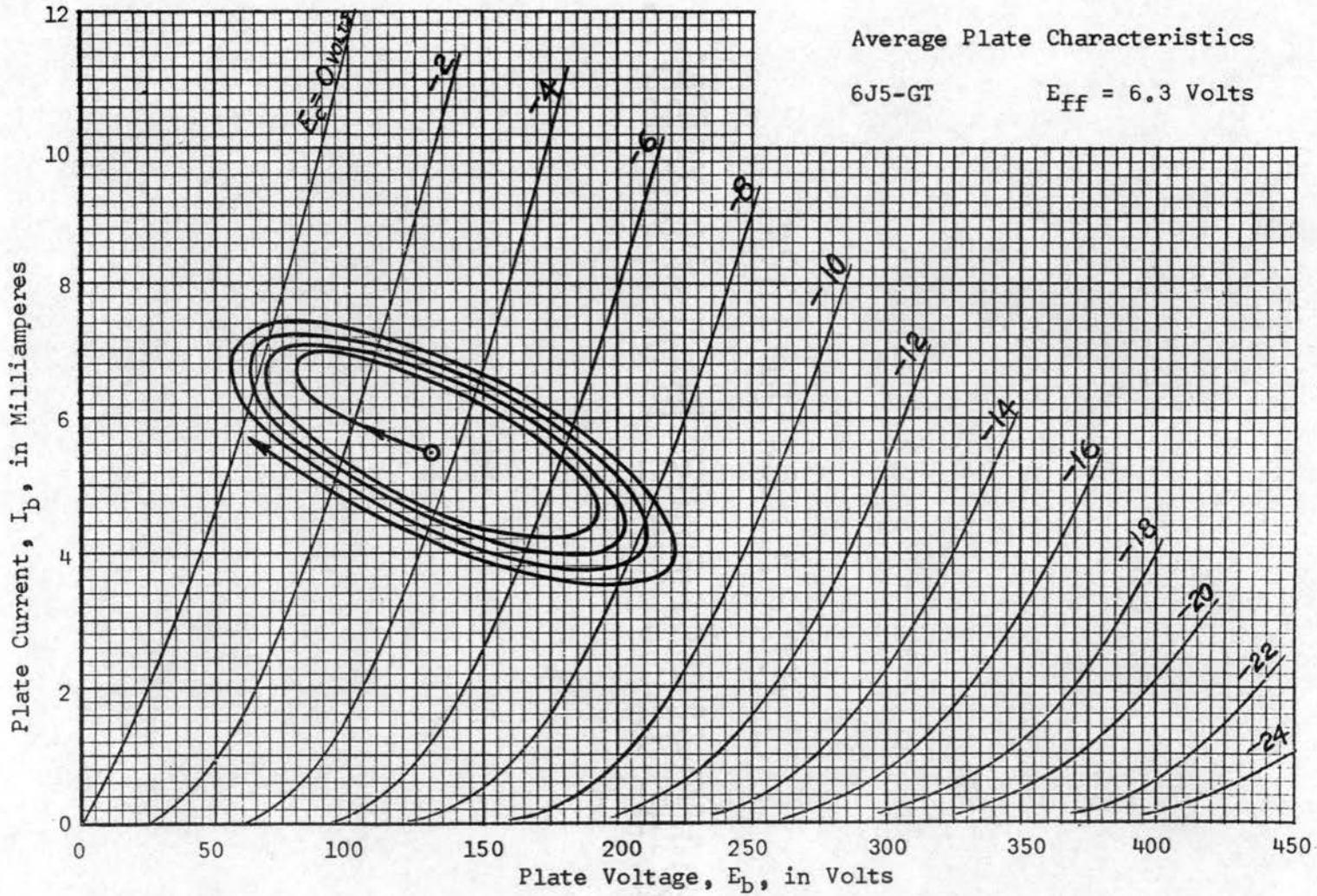


Figure F.3.3. Inductive Plate Load Response to Suddenly Applied Increasing Sinusoid

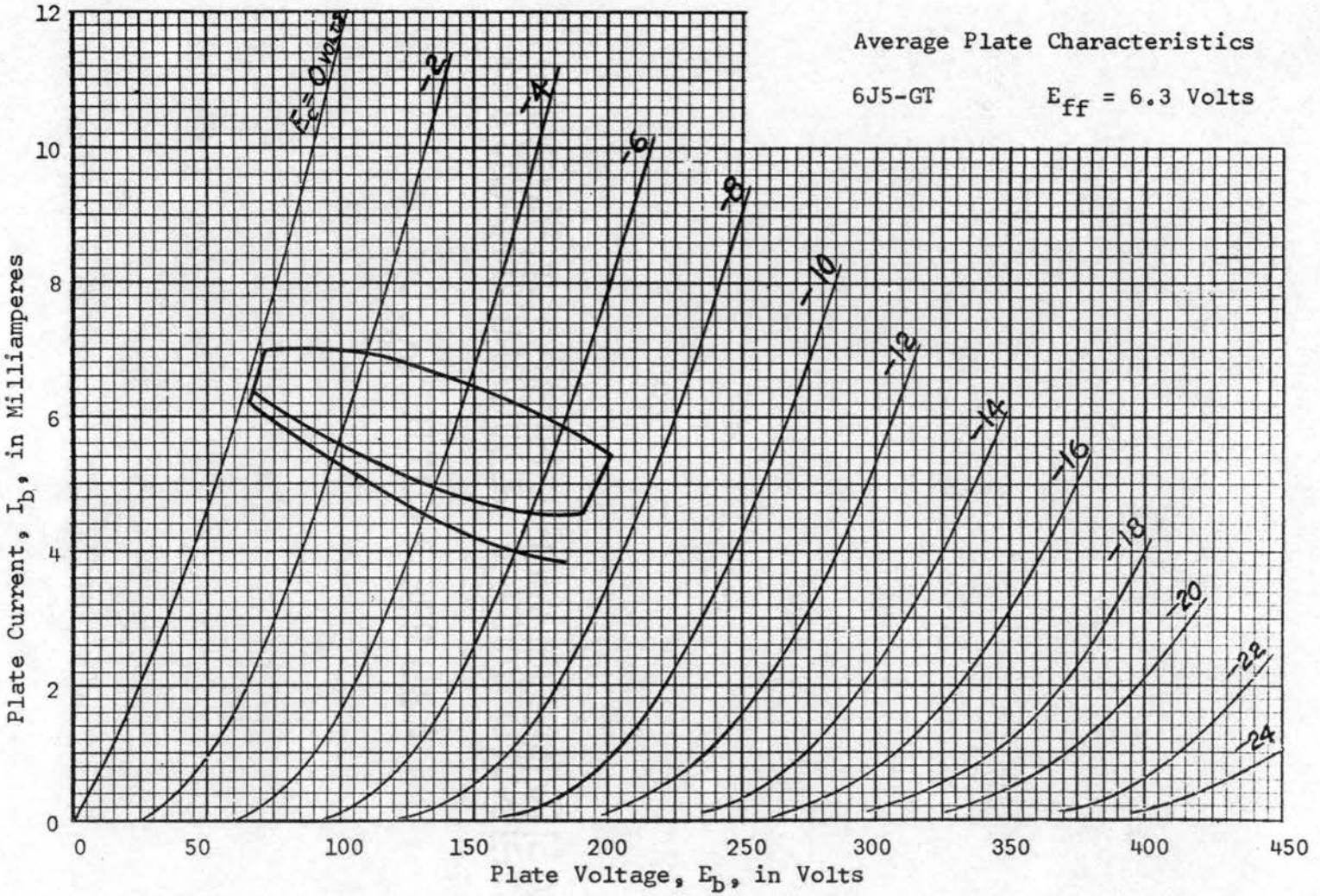


Figure F.3.4. Inductive Plate Load Response to Slow Rise-and-Fall Rectangular Pulse Train

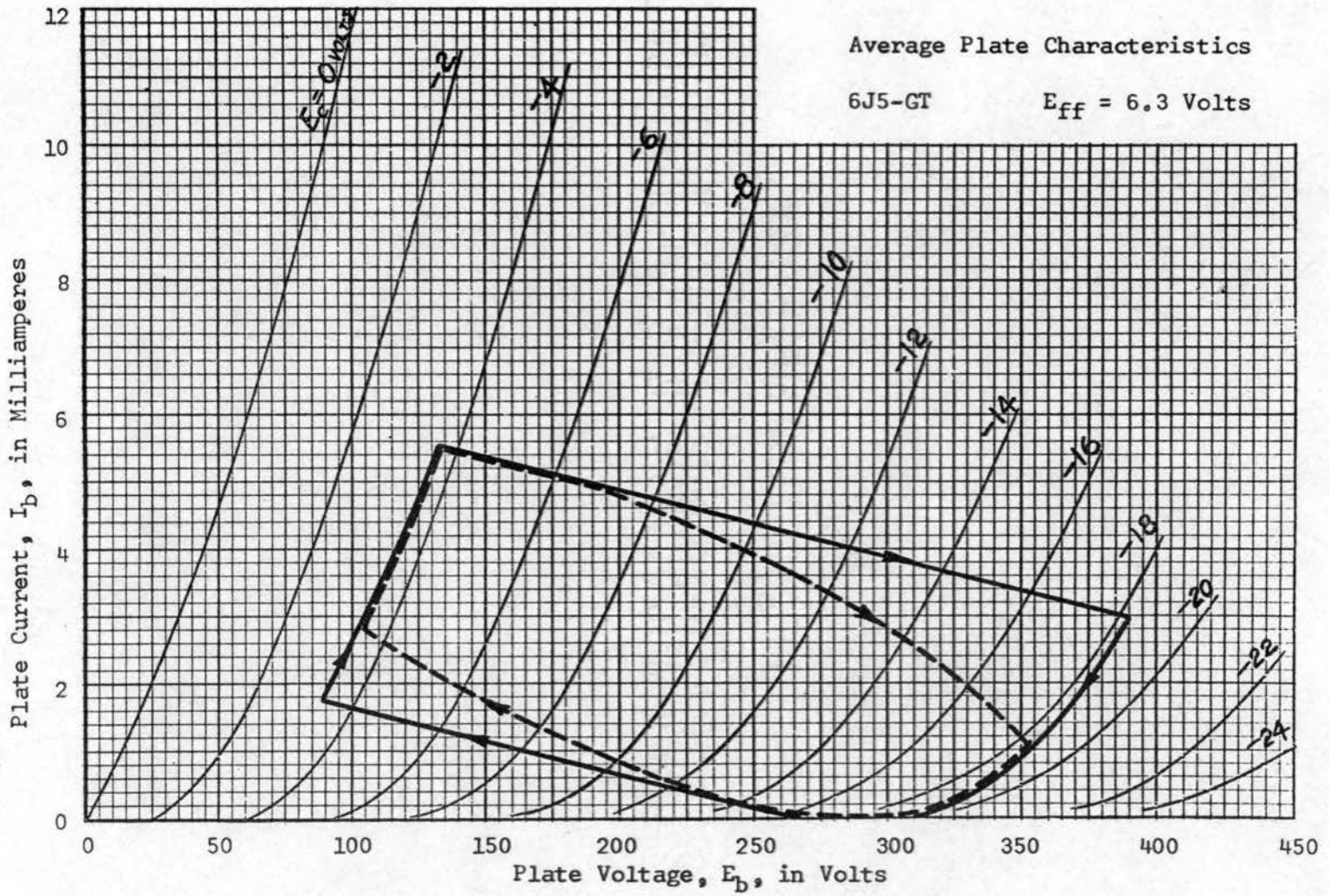


Figure F.3.5. Inductive Plate Load Response to Large Negative Rectangular Pulse

VITA

Jack Milfred Walden

Candidate for the Degree of

Doctor of Philosophy

Thesis: DIGITAL COMPUTER ANALYSIS OF SYSTEMS WITH NONLINEAR ELEMENTS

Major Field: Engineering

Biographical:

Personal Data: Born in Sheridan, Wyoming, July 1, 1922, the son of George M. and Mary R. Walden.

Education: Attended grade school and high school in Sheridan, Wyoming; graduated from Sheridan High School in June, 1940; received the Bachelor of Science Degree from South Dakota School of Mines and Technology, with major in Engineering Physics, in June, 1944; received the Master of Science Degree from Oklahoma State University, with major in Electrical Engineering, in August, 1962; completed requirements for the Doctor of Philosophy Degree at Oklahoma State University in May, 1966.

Professional Experience: Employed by Midnight Sun Broadcasting Company in Fairbanks and Anchorage, Alaska, as Chief Engineer from April, 1945, until August, 1952. Vice-President and Technical Director of Northern Television, Inc., in Anchorage and Fairbanks, Alaska, from August, 1952, to August, 1960. Served as Consultant or Supervising Engineer in the design, installation, and operation of three TV stations and seven AM radio stations during this period. Employed as an Instructor in the School of Electrical Engineering, Oklahoma State University, in September, 1960.

Professional Organizations: Senior Member of the Institute of Electrical and Electronics Engineers, Member of Association for Computing Machinery, Sigma Tau, Eta Kappa Nu, Phi Kappa Phi, Associate Member of Sigma Xi.