NASA CR-

NUMERICAL INTEGRATION AND OTHER TECHNIQUES FOR COMPUTER AIDED NETWORK DESIGN PROGRAMMING (Final Technical Report)

By

Charlie H. Cooke
Eddie H. Young

~~FOR NASA INTERNAL USE ONLY~~

Prepared Under Grant No. NGR 47-003-026

By

OLD DOMINION UNIVERSITY
RESEARCH FOUNDATION
NORFOLK, VIRGINIA 23508

For

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

08

Annual Report
to
National Aeronautics and Space Administration
Hampton, Virginia

Numerical Integration and Other Techniques For Computer Aided Network Design Programming

Covering Work Performed Under NGR-47-003-026
1 January 1970 to 1 January 1971

FROM

Charlie H. Cooke, Principal Investigator

Eddie H. Young, Principal Investigator*

SCHOOL OF ENGINEERING

Old Dominion University

Norfolk, Virginia 23508

*Employed by Radiation, Incorporated, Melbourne, Florida, 32901, as of July 1, 1970.

Table of Contents

I. Abstract

A limitation to the capability of first generation computer aided network design programs occurs for the case of networks characterized by so-called stiff systems of first order differential equations. A study of methods for strengthening this capability is reported here. A matrix method for a limited class of networks is developed, which avoids numerical integration, the usual source of difficulty. The performance of Gear type implicit linear multistep methods of numerical integration is investigated, and the mathematical structure essential in the construction of this class of algorithms is obtained.

II. Introduction and Summary

In the last several years a number of computer aided circuit analysis programs have been developed [1,2]. Such programs have proven invaluable for research and design work in both industry and the academic community. However, many of the first generation circuit analysis programs share a common shortcoming; severe numerical problems occur in the approximate integration techniques employed, when applied to networks characterized by large time constant spreads. In order to obtain a reliable design analysis in this ill-conditioned case, extremely small time steps must be taken, with the step-size usually inversely proportional to the largest time constant present. Such a measure proves highly uneconomical, since quite often the high frequency eigen-modes are parasitic effects, having slight dominance in the circuit except at very high frequencies, or during the very early initial periods in transient analysis. Thus, it would appear that the step size could be increased when observing the more dominant long time effects in the circuit. However, for ordinary integration techniques such as Adam's or the Runge-Kutta methods, increasing the step-size concomitantly produces severe numerical problems, even when the high frequency modes have decayed.

The present study aims at the alleviation of these numerical problems, through search of existing techniques as well as research on new algorithms. As it turns out, the approaches can be divided into two categories: a direct matrix approach avoiding numerical integration, and investigation of the implicit linear multistep numerical integration techniques first discovered by Gear, purposely designed for stiff systems of differential equations.

The first method is concerned with the application of matrix theories in solving a specialized class of linear network problems frequently encountered in a large percentage of design problems; namely, linear time invariant networks forced by sinusoidal, co-sinusoidal, or step inputs. This method employs a spectral decomposition of the matrix exponential exp(At), in terms of the eigenvalues of the system matrix of a linear network forced by one of the previously categorized inputs. This allows one to obtain a closed form solution, which avoids numerical integration and has the advantage that output time points may be arbitrarily selected without effecting program efficiency. The theoretical basis of the matrix method is reported in Section III[1], and computational results appear in Section V.

The second area of investigation concerns the use of implicit numerical integration techniques specifically designed for systems of first order differential equations with wide-spread eigenvalues, the so-called stiffly stable implicit linear multistep algorithms of Gear. A program obtained from Gear has been used to process typical networks which ordinarily experience numerical difficulty. Computational results are reported in Section V.

---

[1]Sections III and IV consist of exact reproductions of papers currently submitted for publication in technical journals.

A survey of the literature concerning implicit integration techniques has been made, and an editing of the principal results appears in a previous report [3].

Although the stiffly stable algorithms of Gear appear extremely effective, as regards the purposes for which designed, the present literature survey has not indicated any results concerning the general properties of this class of algorithms, or of methods for constructing members of the class other than the specific algorithms detailed by Gear [4]. Thus, we have devoted some effort to the isolation of the essential mathematical structure of the stiffly stable algorithms. Necessary conditions for constructing the general member of the class have been discovered, and are reported in Section IV.

Section III

The Matrix Method

NUMERICAL SOLUTION OF LINEAR STATE

EQUATIONS WITH LARGE EIGENVALUE SPREADS

SUMMARY

An algorithm is presented for solving linear state equations excited by sinusoidal and rectangular waveforms without numerical integration. This method is valid in the entire λ plane and is especially advantageous when a system has large eigenvalue spreads.

In the last several years, a number of computer aided circuit analysis programs (1,2,3) have been developed, and are now widely in use as an aid to the electrical engineer. However, they share one common shortcoming in that they cannot solve problems with large time constant spreads. The difficulty lies in the fact that ordinary integration techniques will most likely become unstable or in some cases become nonconvergent in this ill-conditioned case. As a result when large spreads of eigenvalues exist, extremely small time steps have to be taken, with some criteria of the form $|h\lambda_{max}| \leq C$. This is certainly inconvenient and uneconomical since quite often the high frequency eigen modes are parasitic effects in the system and are not dominant except at very high frequencies or during the very initial periods in transient analysis. Since these modes decay rapidly in time then it seems reasonable that the step size can be increased to observe the more prominent long time effects of the circuit. This is exactly how ordinary integration techniques such as Adam's or the Runge Rutta methods fail. The step size cannot be increased even when the high frequency modes have decayed. Gear (4,5) has written a very powerful algorithm to solve a set of linear and nonlinear differential equations using multistep methods. In the linear case, however, certain matrix manipulations may actually bring about advantages that are not found in the implicit methods. Hence, this prompted the present investigation.

The set of linear differential equations describing a network can be cast in the state variable form

$$\dot{\underline{x}} = \underset{\sim}{A}x + \underset{\sim}{B}\underline{u} = \underset{\sim}{A}\underline{x} + \underline{u}' \tag{1}$$

and the solution of Equation (1) is

$$\underline{x} = \exp(\underset{\sim}{A}(t-t_o))\underline{x}(t_o) + \int_{t_o}^{t} \exp(A(t-\tau))\underline{u}'(\tau)d\tau \tag{2}$$

When $\underset{\sim}{A}$ is ill-conditioned with large spreads in eigenvalues then the integral solution given by Equation (2) experiences the difficulty mentioned above. One way to eliminate the problem is to integrate exactly for certain waveforms such as sinusoids and block pulses, which probably constitute many of the excitations encountered in electronic circuits. The explicit integration can be carried out if the transition matrix $\exp(\underset{\sim}{A}(t-t_o))$ can be expressed in terms of the eigen modes. The eigenvalues of the $\underset{\sim}{A}$ matrix are solved using the QR transformation (6,7), which is the best method available. The particular expansion of the transition matrix, in terms of the eigen modes, was studied by Kirchner (8) and is essential for this algorithm. At the present moment the discussion is limited to nondegenerate modes; the case for repeated eigenvalues is somewhat complex but still solvable and the computational aspects of the problem have not been examined. For the case with distinct eigenvalues the transition matrix is expressed as

$$\exp(\underset{\sim}{\Lambda}t) = \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij}\Lambda^{j-1} \exp(\lambda_i t) \tag{3}$$

The coefficients C_{ij} form a matrix which turns out to be the inverse of the Vandermonde matrix

$$\underset{\sim}{C} = [C_{ij}] = \begin{bmatrix} 1 & 1 & \cdot & \cdot & \cdot & 1 \\ \lambda_1 & \lambda_2 & \cdot & \cdot & \cdot & \lambda_n \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \lambda_1^{n-1} & \lambda_2^{n-1} & \cdot & \cdot & \cdot & \lambda_n^{n-1} \end{bmatrix}^{-1} \tag{4}$$

The Vandermonde inverse can be computed efficiently through the method developed by Kaufman (9),

$$C_{ij} = \frac{\sum_{k=0}^{n-j} a_k \lambda_i^{n-j-k}}{\prod_{\substack{k=1 \\ k\neq i}}^{n} (\lambda_i - \lambda_k)} \tag{5}$$

where the a_k values are the coefficients of the characteristic equation

$$\underline{P}(\lambda) = a_o\lambda^n + a_1\lambda^{n-1} + \ . \ . \ . + a_{n-1}\lambda + a_n \tag{6}$$

with $a_o = 1$

The a_k coefficients can be formed from the combination of the traces of $\underset{\sim}{A}^k$, $k = 1, \ldots, n$

$$a_k = -\frac{1}{k} \sum_{m=1}^{k} a_{k-m} T_m \tag{7}$$

$$T_m = \text{tr}\underset{\sim}{A}^m = \sum_{\ell=1}^{n} (\lambda_\ell^{\,m}) \tag{8}$$

The total number of operations involved in forming the $\underset{\sim}{C}$ matrix is about 4.5 N^2 operations, which is considerably faster than Gauss elimination, for large N. It is very cumbersome to store the transition matrix as expressed by Equation (3). Since the matrix is always multiplied into a vector such as $\exp(\underset{\sim}{A}(t-\tau))\underline{u}'$, then it would be convenient to consider the vectors

$$\exp(\underset{\sim}{A}(t-\tau))\underline{u}' = \underline{p}_1 \exp(\lambda_1(t-\tau)) + \ldots + \underline{p}_n \exp(\lambda_n(t-\tau)) \tag{9}$$

where

$$\begin{aligned} \underline{p}_j &= (C_{j1}\underset{\sim}{I} + C_{j2}\underset{\sim}{A} + \ldots + C_{jn}\underset{\sim}{A}^{n-1})\underline{u}' \\ &= (C_{j1}\underline{Y}_1 + C_{j2}\underline{Y}_2 + \ldots + C_{jn}\underline{Y}_n) \end{aligned} \tag{10}$$

with the $\underline{Y}_{m+1}$ vector defined by

$$\underline{Y}_1 = \underline{u}'$$

$$\underline{Y}_{m+1} = \underset{\sim}{A}\, \underline{Y}_{m-1} \tag{11}$$

This can be easily set up as a recursive process and use is made to take advantage of the sparseness of the $\underset{\sim}{\Lambda}$ matrix so that the programming is more efficient.

The integral in Equation (2) can now be expressed as

$$\int_{t_o}^{t} \exp(\underset{\sim}{\Lambda}(t-\tau))\ \underline{u}'\ (\tau)\ d\tau = \sum_{j=1}^{n} \int_{t_o}^{t} \underline{p}_j \exp(\lambda_j(t-\tau))\ d\tau \tag{12}$$

For a finite number of excitation sources $\underline{p}_j$ can be written as

$$\underline{p}_j = \sum_{k=1}^{k=M} \underline{r}_k s_k(\tau) \tag{13}$$

where $\underline{r}_k$ is a constant column vector, and Equation (12) becomes

$$\sum_{j=1}^{n} \sum_{k=1}^{M} \underline{r}_k \int_{t_o}^{t} \exp(\lambda_j(t-\tau)) s_k(\tau)\ d\tau \tag{14}$$

Notice the integrals in Equation (14) are now scalar quantities and these are nothing more than convolution integrals. Since the integral

$$\int_{t_o}^{t} f\ d\tau = \int_{o}^{t} f\ d\tau\ - \int_{o}^{t_o} f\ d\tau \tag{15}$$

it is sufficient to examine the integral

$$\int_{o}^{t} \exp(\lambda_j(t-\tau)) s_k(\tau)\ d\tau \tag{16}$$

from which Equation (15) can be computed. The Laplace transform of the above integral is

$$\frac{1}{p-\lambda_j}\ S(p) = \frac{1}{p-\lambda_j}\ \frac{n(p)}{d(p)} = \frac{a}{p-\lambda_j} + \frac{b(p)}{d(p)} \tag{17}$$

The first term in the above equation is the natural mode in the circuit while the remainder term is the response due to the forcing function. If λ_j and the poles of the driving function are situated widely apart, then the ill-conditioned case comes up. In the present algorithm, an exact explicit integration will be performed for Equation (16) for certain excitation waveforms, thereby eliminating the difficulty. In theory, λ_j can be anywhere in the complex plane, and it seems that this method may also be extended to quasilinear cases. When there are complex eigenvalues, then the following integral pair is considered

$$\left\{\int_0^t \exp(\lambda_i(t-\tau))rs(\tau)\ d\tau + \int_0^t \exp(\lambda_i^*(t-\tau))r^*s(\tau)\ d\tau\right\} \qquad (18)$$

where r is an element in the column vector of Equation (14). Equation (18) is valid if the original state equations consist of real quantities only.

As an example the explicit integration of the rectangular waveform is given. The integral

$$\int_0^t \exp(\lambda(t-\tau))E(\tau)\ d\tau \qquad (19)$$

is a solution of

$$\frac{dx}{dt} - \lambda x = E(t) \qquad (20)$$

with $x(0) = 0$,

$$E(t) = E_o \qquad 0<t<T_1$$
$$= -E_1 \qquad T_1<t<T$$
$$= \frac{E_o-E_1}{2} \qquad t = T_1, T$$
$$= E(t + nT) \qquad n = 0, 1, 2, \ldots \tag{21}$$

If the problem was solved by the Laplace transform method, the solution would be in the form of an infinite series and this would be undesirable for computational purposes. It is much simpler to replace the differential equation by a sequence of differential equations

$$\frac{dx_{n,1}}{dt} - \lambda x_{n,1} = E_o \qquad nT<t<nT + T_1 \tag{22}$$

$$\frac{dx_{n,2}}{dt} - \lambda x_{n,2} = -E_1 \qquad nT + T_1<t<(n+1)T \tag{23}$$

where $n = 0, 1, 2, \ldots$

Since x is to be continuous, the beginning of $x_{n,2}$ and the end points of $x_{n,1}$ must coincide with each other. Thus, a sequence of points are defined at 0, T_1, T, T + T_1, 2T, ... This sequence can be solved from the solutions of Equations (22 and 23) with the aid of z transform theory. For the case Re(λ) is negative then the natural mode will be a decaying mode, and is expressed as

$$x_{n,1}(t = t' + nT) = -\frac{\exp(\lambda(t' + nT))}{(1-\exp(\lambda T))} \{\frac{E_1}{\lambda}(1-\exp(\lambda(T-T_1)))$$
$$+\frac{E_o}{\lambda} \exp(\lambda T)(1-\exp(-\lambda T_1))\} \quad 0\leq t'\leq T_1 \tag{24}$$

$$x_{n,2}(t = t' + nT + T_1) = -\exp\,(\lambda(t' + nT + T_1))\frac{1}{(1 - \exp(\lambda T))}$$

$$\cdot\{\frac{E_1}{\lambda}\ (1 - \exp(\lambda(T - T_1)) + \frac{E_o}{\lambda}\ \exp(\lambda T)(1 - \exp(\lambda T_1))\}$$

$$T_1 \leq t' \leq T \qquad (25)$$

When $n \rightarrow \infty$ the two above expressions will tend to zero. The terms due to the forcing function are

$$x_{n,1}(t = t' + nT) = -\frac{E_o}{\lambda}\ (1 - \exp(\lambda t')) + \exp(\lambda t')\frac{1}{(1-\exp(\lambda T))}$$

$$\cdot\{\frac{E_1}{\lambda}\ (1 - \exp\ (\lambda(T-T_1))) + \frac{E_o}{\lambda}\ \exp(\lambda T)(1 - \exp(-\lambda T_1))\}$$

$$0 \leq t' \leq T_1 \qquad (26)$$

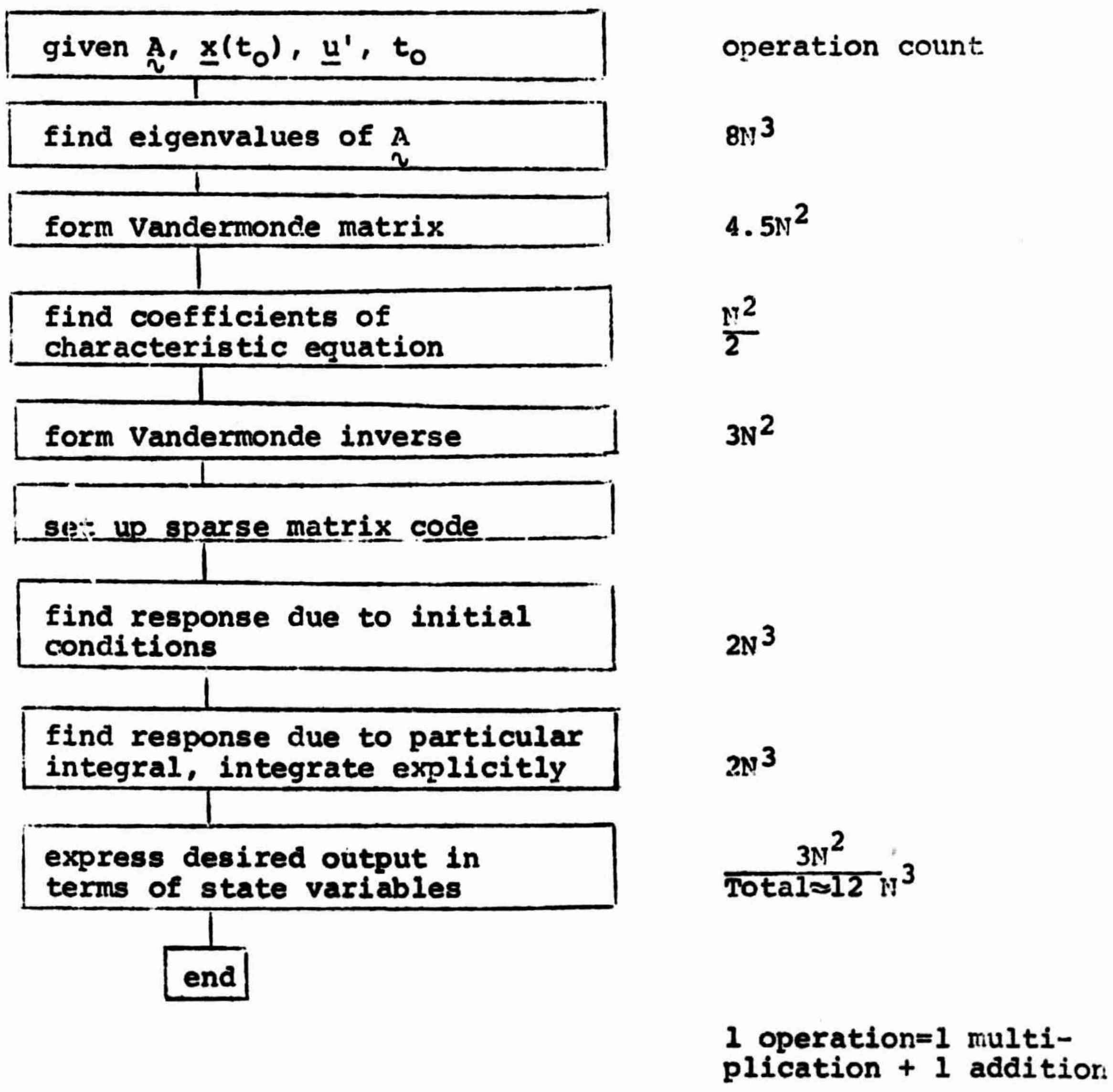
$$x_{n,2}(t = t' + nT_1) = \frac{E_1}{\lambda}(1 - \exp(\lambda t')) - \frac{E_o}{\lambda}\ (1-\exp(\lambda T_1))\exp(\lambda t$$

$$+\frac{\exp(\lambda(t'+T_1))}{1-\exp(\lambda T)}\{\ \frac{E_1}{\lambda}\ (1-\exp(\lambda(T-T_1)) + \frac{E_o}{\lambda}\ \exp(\lambda T)(1-\exp(-\lambda T_1))\}$$

$$T_1 \leq t' \leq T \qquad (27)$$

Notice the factors $\exp(\lambda T)$, $(1-\exp(\lambda T))$ and others will always have their conjugate pairs when complex eigenvalues are considered. The complex terms can be combined to form real terms only, thus the result can be applied to all eigenvalues. The separation of the transient term from the forcing function is a real asset in this algorithm. It may take many periods to reach a steady state, however, in many circumstances only the steady state solution is of interest, which can be obtained directly from Equations (26 and 27). The numerical integration techniques do not have this convenience.

When a finite pulse train is considered, the excitation can be realized by a superposition of two infinite rectangular pulses, and the analysis is again the same. The development for sinusoidal waveform is simpler in theory but involves some tedious manipulation and will not be covered here. A flow chart of the complete algorith is given below



given $\underset{\sim}{A}$, $\underline{x}(t_o)$, $\underline{u}'$, t_o
operation count
find eigenvalues of $\underset{\sim}{A}$
$8N^3$
form Vandermonde matrix
$4.5N^2$
find coefficients of characteristic equation
$\frac{N^2}{2}$
form Vandermonde inverse
$3N^2$
set up sparse matrix code
find response due to initial conditions
$2N^3$
find response due to particular integral, integrate explicitly
$2N^3$
express desired output in terms of state variables
$3N^2$
Total$\approx$12 N^3
end
1 operation=1 multiplication + 1 addition

FLOW CHART AND OPERATION COUNT OF ALGORITHM TO SOLVE LINEAR STATE EQUATIONS

The total operation count is of the order of 12 N^3. This is about a factor of N times less operations compared to the eigen vector method (10). The method used in Pottle's program (3) is about 16 N^3 for a single output. For multiple outputs, Pottle's program also approaches N^4 units of operation. The present scheme has been found to work well for several trial problems. Some further work is being carried out in examining the case for repeated roots, the processing of the convolution integral for arbitrary waveforms, and the search for a sparse technique method in obtaining the eigenvalues, which is the most time consuming part of the program.

The authors wish to acknowledge the support by NASA under grant #NGR47003026 for this work.

July 20, 1970

E. H. YOUNG, JR.
RADIATION INC.
MELBOURNE, FLA., 32901, USA

J. H. HEINBOCKEL

M. N. RANSOM

OLD DOMINION U.
NORFOLK, VIRGINIA 23508, USA

REFERENCES

(1) "1620 Electronic Circuit Analysis Program [ECAP] [1620-EE-02X] Users Manual," IBM Application Program File, H20-0170-1, 1965.

(2) Mathers, H. W., Sedore, S. R. and Sents, J. R., "Automated Digital Computer Program for Determining Responses of Electronic Circuits to Transient Nuclear Radiation (SCEPTRE)," Vol. 1, IBM Space Guidance Center, Owego, N. Y., IBM FILE 66-928-611, February 1967.

(3) Pottle, C., "A 'Textbook' Computerized State-Space Network Analysis Algorithm", IEEE International Symposium, December 1968.

(4) Gear, C., "Numerical Integration of Stiff Ordinary Differential Equations," 1967, Dept. of Computer Science, U. of Illinois, report #221.

(5) Dill, C., Ellis, C., Gear, C., Ratliff, K. "ODESSY--- Ordinary Differential Equation Solver System," 1969, Dept. of Computer Science, U. of Illinois, Urbana, Illinois, FILE No. 779.

(6) Wilkinson, J. H., "The Algebraic Eigenvalue Problem," 1965, Oxford.

(7) Parlett, B. N., "The LU and QR Algorithms," Chapter 5 of Mathematical Methods for Digital Computers, Vol. 2, 1967, Wiley.

(8) Kirchner, R. B., "An Explicit Formula for e^{At}," AMM, 1967, pp. 1200-1204.

(9) Kaufman, I., "Evaluation of an Analytical Function of a Companion Matrix with Distinct Eigenvalues," 1969, Proceedings of IEEE Letters, pp. 1180-1181.

(10) Branin, Jr., F. H., "Computer Methods of Network Analysis," Proceedings IEEE, 1967, pp. 1787-1801.

Section IV

Mathematical Structure of the Stiffly Stable Algorithms

ON THE CLASS OF STIFFLY STABLE IMPLICIT LINEAR MULTI-STEP NUMERICAL INTEGRATION ALGORITHMS

Charlie H. Cooke [+]

Abstract

An analysis of the mathematical structure of the class of stiffly stable implicit linear multi-step numerical integration algorithms is presented. Conditions sufficient for construction of typical members of the class are obtained. These conditions in general consist of one-to-one-ness of the stability mapping on the unit circle, together with analyticity on its exterior.

1. Introduction

The usual analysis of the properties of numerical integration algorithms for the solution of the initial value problem in ordinary differential equations entails a study of the convergence of the numerical solution to the exact solution as the stepsize h approaches zero, the so-called asymptotic theory [1, 2, 3]. However, in most instances the prime concern is numerical accuracy and stability in tandem with maximum stepsize. This is particularly exemplified by the case of stiff systems of ordinary differential equations, such as occur in electrical networks

[+]Old Dominion University, Norfolk, Virginia, 23508. This investigation sponsored by NASA Grant NGR-47-003-026, Langley Research Center, Hampton, Virginia.

characterized by large spreads in the time constants of the various components [5]. Such systems exhibit some rapidly decaying components which quickly become of no interest. However, the maximum step size allowable in the numerical integration of such systems is usually of the order of magnitude of the smallest time constant present. Severe numerical problems occur, for most general purpose integration techniques, if a larger step size is attempted.

The motivation to increase the step size with no degradation of numerical accuracy and stability has led to the discovery of particular members of the class of so-called stiffly stable implicit linear multistep algorithms. Although specific algorithms have been developed, it is not known precisely how to construct the general member of the class, except "by using common sense and investigating the properties of the guess" [6]. In this paper an investigation of the mathematical structure of this class of algorithms is presented. Conditions sufficient for construction of typical members of the class are obtained.

2. Stiffly Stable Methods

The general implicit linear k-step method for the numerical solution of the initial value problem

$$\frac{d\bar{x}}{dt} = \bar{f}(\bar{x},t),\ \bar{x}(o) = \bar{x}_o,\ (\bar{x}\epsilon R^s,\ t\geq o) \qquad (2.1)$$

is defined by the relation

$$\alpha_k \bar{x}_{n+k} + \ldots + \alpha_o \bar{x}_n = h[\beta_k \bar{f}_{n+k} + \ldots + \beta_o \bar{f}_n], \qquad (2.2)$$

with α_k, $\beta_k \neq 0$. In what follows it is assumed that the α_i, β_i, $i = 0, 1, 2, \ldots, k$, are real constants; $h>0$ is the step size; $t_m = mh$; and $\bar{f}_m = \bar{f}(\bar{x}_m, t_m)$. Given vectors $\bar{x}_o$, $\bar{x}_1$, $\ldots$, $\bar{x}_{k-1}$, then $\bar{x}_k$, $\bar{x}_{k+1}$, $\ldots$, are computed recursively from (2.2) by predictor-corrector methods [6]. It is further assumed that the convergence of the predictor is such that it has no influence on the computed solution of (2.2). If the Jacobian matrix $\frac{\partial \bar{f}}{\partial \bar{x}}$ of system (2.1) is slowly varying, the numerical stability of eqn. (2.2) is then determined by the location of the roots of the associated polynomial equations [1,4]

$$\rho(z) - h\lambda_i \sigma(z) = 0. \qquad (2.3)$$

Here

$$\rho(z) = \sum_{j=0}^{k} \alpha_j z^j, \quad \sigma(z) = \sum_{j=0}^{k} \beta_j z^j; \qquad (2.4)$$

the λ_i, $i = 1, 2, \ldots, S$ are the eigen-values of the Jacobian matrix; and ρ, σ are assumed to have no common factors. System (2.2) is said to be stable (absolute stability) provided the roots of (2.3), for each λ_i, lie within the unit circle, with the possible exception of at most one simple root occurring on the unit circle.

Recall that (2.1) is a stiff system if the eigen-values λ_i are widely spread, presumably over the left half plane in the case of a physically stable system. The requirements [6] that (2.2) be a stiffly stable algorithm are summarized by Fig. (1). Let λ_i be an eigen-value of $\frac{\partial \bar{f}}{\partial \bar{x}}$. For $h\lambda_i$ having a real part less than the parameter $D \leq 0$, or for $h\lambda_i$ within the rectangle bounded by the lines $x=D$, $x=\alpha$, and $y=\pm\theta$, it is required that the corresponding roots of (2.3) be inside the unit circle in the Z-plane, with the possible exception of at most one simple root on the unit circle. If, in addition, for $h\lambda_i$ within the rectangle the one step truncation error is of the order h^{p+1} then (2.2) provides a stiffly stable method of order p with respect to the parameters D,θ,α. Hence stiff stability requires an algorithm which is numerically stable in the region of $h\lambda$ corresponding to the rapidly decaying system components of little significance, and which is both stable and accurate in the region corresponding to reasonably large step sizes and less slowly decaying system components.

The usual definition of accuracy is that of <u>order</u>. The order of the method (2.2) is determined by the operator [4]

$$L = \rho(E) - hD\sigma(E),$$

where $D = \frac{d}{dt}$ and E is the shift operator,

$$E(g(t)) = g(t+h), \text{ or } E(g_n) = g_{n+1}.$$

The order of the method is the largest integer p such that $L(g(t)) = 0$ for all polynomials $g(t)$ of degree p. The definition of order and alternate formulations of it are thoroughly discussed by Henrici.

A method is said to be consistent if $p \geq 1$. The condition of consistency may be expressed by the conditions [4].

$\rho(1) = 0, \; \rho'(1) = \sigma(1).$

The property of consistency assures that (for an infinite precision machine) the solution of the difference equation (2.2) will converge to that of the differential equation (2.1) as h approaches zero.

3. Regions of Stability and Instability

Consider the stability mapping

$$H = \rho(z)/\sigma(z), \qquad (3.1)$$

implicitly defined by eqn. (2.3). H is a rational function mapping the extended z-plane k-to-one and onto the $h\lambda$ plane, and there is a one-to-one correspondence between the class of rational functions with real coefficients and the class of algorithms of the form (2.1). Hence the problem of determining stiffly stable algorithms is essentially that of analyzing the properties of a certain subclass of rational functions. For instance, in practice it is too cumbersome to apply the definition of absolute stability to each individual $h\lambda$ value in order to determine the H-plane stability characteristics of a method (2.2). One thus needs global criteria for determining the subset U of the $h\lambda$ plane which is the region of instability, and whose complement S is the region of stability. By determining conditions under which S contains the "stable" plus "stable and accurate" regions of

Fig. (1) we thus obtain stiffly stable algorithms.

We assert that the region U is composed of the set of $h\lambda$ values which are either the images under H of points z which are exterior to the unit circle in the z-plane, or else points which have more than one pre-image on the unit circle. In the next section we present conditions on H which allow a ready determination of U and S.

4. Complex Mappings which Preserve Boundary Points

Consider the problem of determining the image under the complex transformation $w = f(z)$ of a region R in the extended z-plane, which is the interior or exterior of a bounded simple closed curve C. In what follows theorems are stated which allow one to do so by merely determining the image of C. The essential property required of the function $f(z)$ is that the boundary of the image of R be the image of the boundary of R.

A simple closed curve C: $z = z(t)$, $a \leq t \leq b$, $z(a) = z(b)$, and $z(t_1) \neq z(t_2)$ otherwise, is <u>positively oriented</u> if as t varies between a and b the point $z(t)$ traverses C counter-clockwise. If C': $w = f(z)$, $z = z(t)$ is the one-to-one image of C under $w = f(z)$, then C' takes an orientation with respect to that of C which is positive if as the point z traverses C counter-clockwise, $w = f(z)$ traces C' in the same manner.

Let $E(C)$ denote the exterior of a simple closed curve C, $I(C)$ the interior, and $\bar{A}$ the closure in the extended plane of a subset A. If $f(z)$ is one-to-one on C, then variants of the following theorem give conditions under which $f(z)$ is boundary preserving on $I(C)$ [7,8]:

THEOREM 1:

Given a closed rectifiable positively oriented Jordan curve C, suppose $f(z)$ is analytic on $\overline{I(C)}$ and one-to-one on C. Then

a) $C' = f(C)$ is positively oriented, and

b) $f(z)$ maps $\overline{I(C)}$ one-to-one and onto $\overline{I(C')}$.

By applying Theorem 1 to the function $f(1/z)$ one can prove the following:

Corollary 1 - Let C be a closed rectifiable Jordan curve on which $f(z)$ is one-to-one. If $f(z)$ is analytic on $\overline{E(C)}$, and $C' = f(C)$, then $f(z)$ maps $\overline{E(C)}$ one-to-one and onto $\overline{I(C')}$.

The hypothesis of analyticity of $f(z)$ in Theorem 1 may be weakened slightly by addition of another requirement [7]:

THEOREM 2:

Given a closed rectifiable Jordan curve C, suppose $f(z)$ is analytic on $\overline{I(C)}$ except for a pole on C, and one-to-one on C. If there is a point W_o which is not the image of any point of $\overline{I(C)}$, then $f(z)$ maps $I(C)$ one-to-one and onto one of the domains with boundary $f(C) = C'$; i.e., the domain on the left of an observer moving with the point $w = f(z)$ as z traverses C counter-clockwise.

5. A Class of Stiffly Stable Algorithms

The central result of this paper is the following:

THEOREM 3

Let G_p be the class of consistent implicit linear multistep algorithms of order p whose stability mapping (3.1) is a rational function H_p which is one-to-one on the unit circle C, and analytic on $\overline{E(C)}$. Then each member M_p of G_p is stiffly stable of order p with respect to suitable parameters D_p, θ_p, α_p, with D_p, $\alpha_p \leq 0$, and $\theta_p > 0$.

Proof

Let M_p be a typical member of G_p with stability mapping H_p. Applying Corollary 1, H_p maps $\overline{E(C)}$ one-to-one and onto $\overline{I(C')}$, $C' = f(C)$. Then $I(C')$ is the region of instability, and $\overline{E(C')}$ the region of stability (see Fig. 2). By consistency of M_p, and the one-to-oneness of H_p, C' is a bounded simple closed curve which passes through the origin. Further, since the coefficients of H_p are real, C' is symmetric with respect to the real axis, and intersects it a second time at $(X_p, 0)$. It may be inferred from the consistency conditions and the curve orientation characteristics of Theorem 1 that X_p is positive.

The boundedness of C' implies the existence of a finite $D_p \leq 0$ such that the vertical line $\mathrm{Re}(h\lambda) = D_p$ is tangent to C', and M_p is stable for $\mathrm{Re}(h\lambda) \leq D_p$. Furthermore, the lines $y = \pm \tan\beta x$, $x \leq 0$, are tangent to but do not cross C', for some β, $0<\beta\leq\pi/2$. Let θ_p, α_p be defined by

$$0 \leq \theta_p \leq |D_p \tan\beta|, \quad \alpha_p = \theta_p/\tan\beta.$$

Then M_p is stiffly stable with respect to the parameters D_p, θ_p, α_p. Theorem 3 follows.

Theorem 3 provides sufficient conditions for constructing a class of stiffly stable algorithms. It is not known whether the one-to-oneness condition of this theorem can be relaxed ; however, the requirement of analyticity on E(C) cannot. For, if H_p has a pole in E(C) the region of instability necessarily contains a neighborhood of infinity; hence, no appropriate D_p exists. It is now shown that analyticity on the unit circle is not required at all points:

THEOREM 4

Let G_p be the class of consistent algorithms of order p referred to in Theorem 3, but whose stability mapping H_p satisfies the conditions.

(1) H_p is one-to-one on C, and analytic on $\overline{E(C)}$, except for a simple pole on C;

(2) H_p satisfies the boundedness condition

$$\mathrm{Inf}\,[\mathrm{Re}\, H_p(z)] = \bar{D}_p, \; \bar{D}_p \leq 0 \text{ but finite,}$$

for z restricted to C; and

(3) H_p does not map the domain $\overline{E(C)}$ onto all of the $h\lambda$ plane.

Then the corresponding algorithm M_p is stiffly stable of order p, with respect to $\overline{D_p}$ and suitable parameters θ_p, α_p .

The proof of Theorem 4 is analagous to that of Theorem 3. We apply Theorem 2 to the function H_p (1/z) and assert that the $h\lambda$ plane image of the unit circle C is an unbounded simple curve C' symmetric with respect to the real axis, which intersects it once, passing through the origin and the point at infinity. By considerations similar to the previous, the region of instability lies to the right of C', for consistent M_p, and the existence of appropriate parameters θ_p, α_p, $D_p = \overline{D_p}$ is readily verified (see Fig. 3).

5. Concluding Remarks

It is known that the class G_p of Theorem 3 is not vacuous, $2 \leq p \leq 6$, as it may be verified that the algorithms of Gear [6] satisfy the hypothesis of Theorem 3. Indeed, these algorithms have the very desirable property that α_p is very nearly zero. For $p = 2$, D_p is zero; a circumstance whose occurrence for $p > 2$ is excluded, for all consistent methods, by a negative result of Dahlquist [4].

The central problem in constructing stiffly stable algorithms using the results presented here is that of finding rational functions which are one-to-one on the unit circle. A procedure for doing so has been obtained for the cases $k = p = 2, 3$; the general case is still under investigation and will be reported on in the

7. Acknowledgments

The author is grateful to L. W. Wilson, of the Old Dominion University Mathematics Department, for helpful discussions concerning the truths of Corollary 1 and Theorem 2, prior to a discovery of these published results in [7].

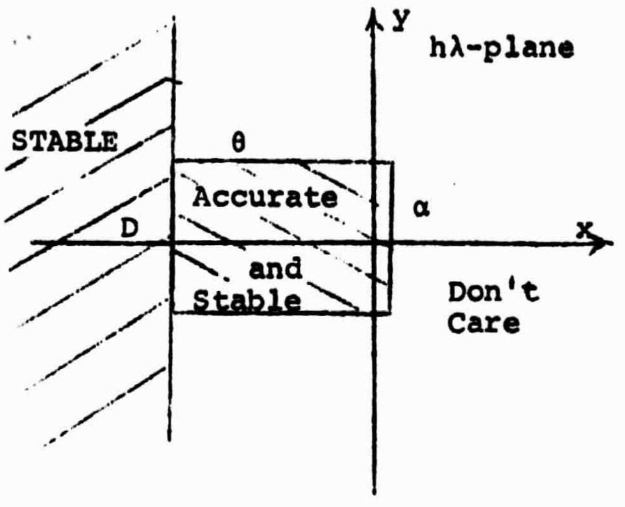


Figure 1. Stiff Stability Requirements

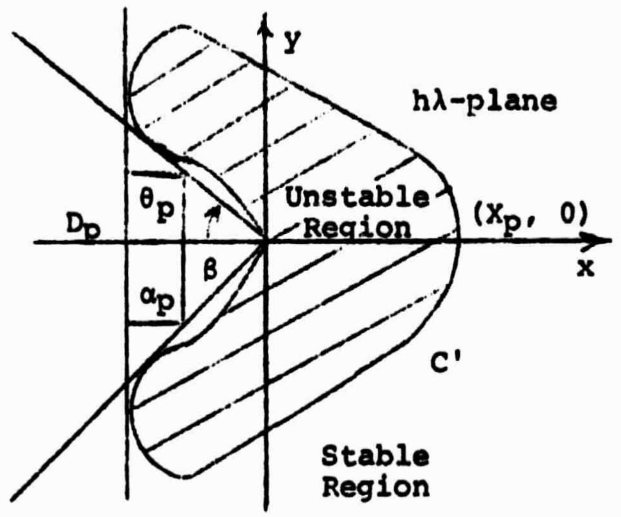


Figure 2. Bounded Image Curve

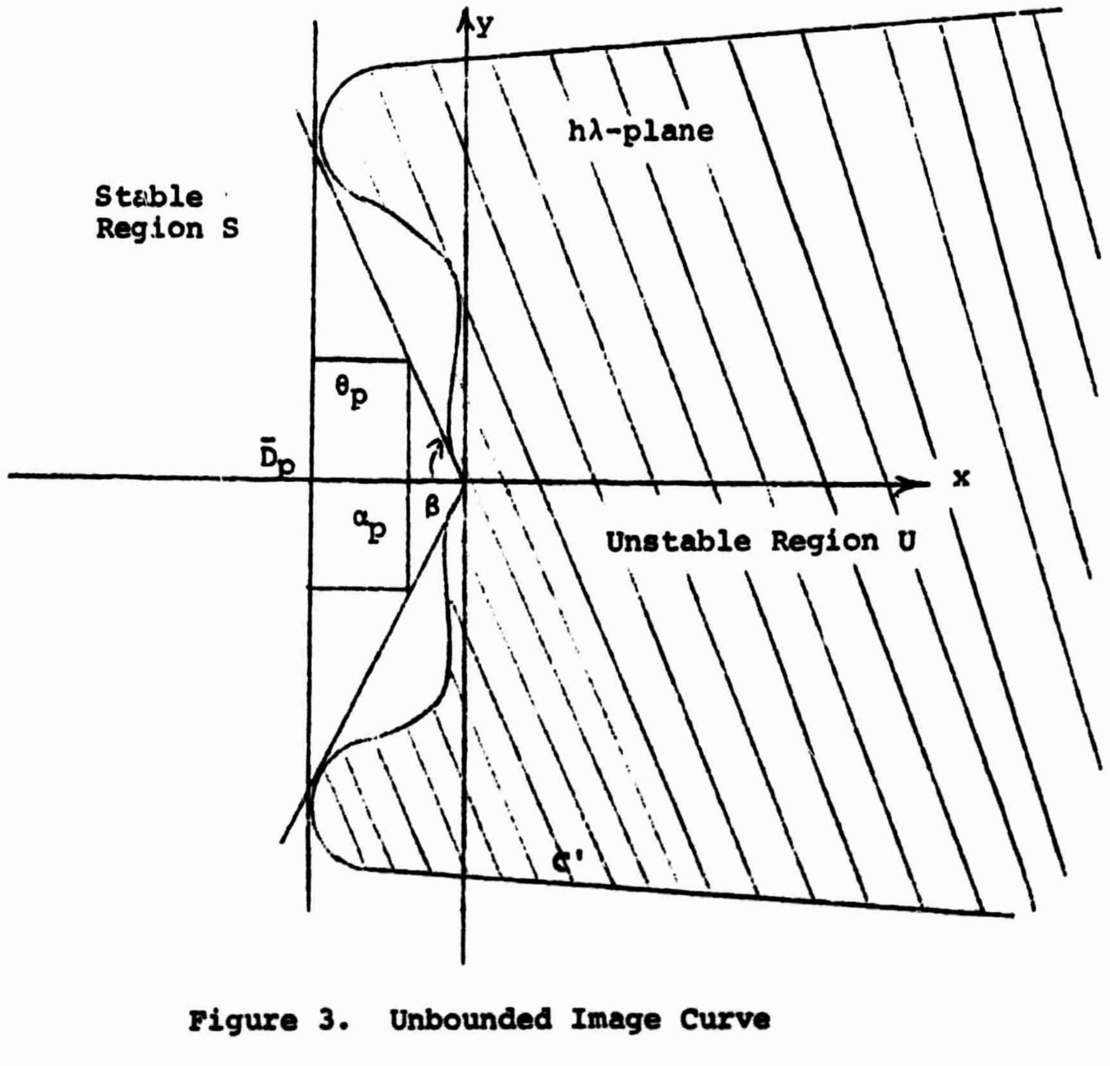


Figure 3. Unbounded Image Curve

REFERENCES

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

2. M. Urabe, "Theory of Errors in Numerical Integration of Ordinary Differential Equations," J. Science, Hiroshima University, Ser. A-I, V. 25, 1961.

3. G. Dahlquist, "Convergence and Stability in the Numerical Integration of Ordinary Differential Equations," Math. Scand., V. 4, 1956, pp. 33-53, MR 18, 338.

4. G. Dahlquist, "A Special Stability Problem for Linear Multi-step Methods," BIT 3 (1963), pp. 27-43.

5. E. Young, "Numerical Integration and Other Techniques for Computer Aided Network Design Programming," Semi-Annual Report, NASA Grant NGR-47-003-026, School of Engineering, Old Dominion University, June 30, 1970.

6. C. W. Gear, "Numerical Integration of Stiff Ordinary Differential Equations," Dept. of Computer Science, University of Illinois, Report #221, January, 1967.

7. A. I. Markushevich, Theory of Functions of A Complex Variable, Vol. II, Prentice-Hall, Englewood Cliffs, N. J., 1965.

8. H. Cohn, Conformal Mapping on Riemann Surfaces, McGraw-Hill, New York, 1967.

V. Computational Results

A typical network whose state variable representation is characterized by a system of simultaneous "stiff" first order differential equations may be represented by the circuit diagram of Figure 1. This network consists of two series resonant circuits loosely coupled by a small resistor Rc. The circuit whose elements are R_1, C_1, and L_1 is tuned to a higher frequency than the circuit composed of elements R_2, C_2, and L_2, which is overdamped. The eigen-modes present in the circuit are plotted in Figure 2. In this section results obtained when processing the circuit using SCEPTRE, Gear's program, and the matrix method program are presented.

SCEPTRE Solution

In processing the circuit using SCEPTRE, the numerical integrations were accomplished using each of the three schemes it provides: the trapezoidal rule, the Runge-Kutta Method, and an exponential method. The results appear in Figures 3 through 8. Figures 3, 4 were obtained using trapezoidal integration; Figures 5, 6 by Runge-Kutta methods; and Figures 7, 8 by the exponential scheme. The results obtained from the trapezoidal rule are generally accurate; however, the computation of the component appearing in Figure 4 is decidedly unstable. The Runge-Kutta results appear superior to the previous; nevertheless, numerical inaccuracies are seen to occur, in Figure 6. The graphs of two separate components obtained using the exponential method appear in Figures 7 and 8. Although it is claimed [1] that this method is superior for the case of stiff equations, the results for this particular example are to the contrary, the exponential method giving poorest overall results.

The total central processing time for the three methods was:

Trapezoidal Rule	100 sec.
Runge-Kutta Method	75 sec.
Exponential Method	90 sec.

Gear's Method

The state variable equations for this circuit, manually obtained, are:

$$\frac{d\bar{x}}{dt} = A\bar{x} + \bar{R}\sin 2t$$

where

$$\bar{x} = \begin{bmatrix} V_{C_1} \\ V_{C_2} \\ I_{L_1} \\ I_{L_2} \end{bmatrix} \qquad \bar{R} = \begin{bmatrix} 0 \\ 0 \\ 1.0 \\ .01 \end{bmatrix}$$

and

$$A = \begin{bmatrix} 0 & 0 & 10^5 & 0 \\ 0 & 0 & 0 & 100 \\ -1 & 0 & -1 & -1 \\ 0 & -.01 & -.01 & -10.01 \end{bmatrix} .$$

The results of applying Gear's program to this system of equations were found to be very good; two components of the computation are shown in Figures 9 and 10. Here central processing time was

21 seconds; however, this figure does not include circuit translation time, in the neighborhood of an additional 4 seconds maximum.

Matrix Method

These state variable equations were solved using the matrix method program. The closed form solution obtained was*

$$v_{C_1} = -.045e^{-10t} + .10e^{-.1t} + .02e^{-.5t}\sin(100t - .01) + \sin(2t-.1)$$

$$v_{C_2} = -.002e^{-10t} + .05e^{-.1t} + .2e^{-.5t}\sin(100t - .01) + .05\sin(2t + 1.5)$$

$$i_{L_1} = -1.5\times10^{-5}e^{-10t} + 5\times10^{-4}e^{-.1t} + 2\times10^{-8}e^{-.5t}\sin(100t + 1.5) + 3\times10^{-4}\sin(2t -.15)$$

$$i_{L_2} = 2\times10^{-4}e^{-10t} - 5\times10^{-5}e^{-.1t} + 2\times10^{-8}e^{-.5t}\sin(100t + .1) + 10^{-3}\sin(2t - .15)$$

This solution agreed with that of Gear's program within 5 significant figures and thus gives the same plots as in Figures 9 and 10.

The central processing time for the matrix program was 3.4 sec. Here, again, no circuit translation time (time required to obtain the circuit differential equations from an input circuit description) is included.

The matrix theory underlying the matrix method technique

*All numbers shown were rounded to two significant figures for illustrative purposes.

applies only to the case of a system matrix with distinct eigen-values. Hence, as should be expected, the numerical performance of the algorithm deteriorates as the eigen-values become close together, and was found to be rather poor for a pathological network characterized by small eigen-value spreads.

VI. Recommendations for Future Work

Consider the essential features of an effective network analysis program. Such a program needs:

(1) The capability of translating the circuit description of a network from the language of the electrical engineer into a mathematical description involving an initial value problem concerning simultaneous differential equations for the state variables of the network,

(2) the capability of obtaining a numerically accurate solution of this initial value problem over desired ranges of the independent variables, and

(3) the capability of representing this solution data and/or solution data on any dependent non-state variable circuit parameters in a form amenable to analysis by the circuit designer.

The present research has focused on the investigation of methods for strengthening capability (2) above, for programs such as the SCEPTRE program, and the matrix method program as well as Gear's program contain only this capability. Each program starts with a state variable initial value problem and produces a state variable solution over a desired time interval. Here Gear's program has a non-linear capability, and the limitations of the matrix method are as previously described in Section I.

In view of this aspect of the present results, future work might very usefully be pursued along the following lines: a strengthening of existing circuit design capabilities might be accomplished by means of a merging of the matrix method and/or Gear's method with a program which incorporates capabilities (1) and (3) as well. This might be accomplished by merging the matrix method and/or Gear's method limited to linear networks with SCEPTRE, or with some program such as Pottle's [5], which contains capability (1) (for linear systems). Of the two, a merger of Gear's program with SCEPTRE as the base program is clearly the most advantageous, from the point of view of a broader general capability, since SCEPTRE accepts circuits with certain types of non-linearities.

However, the installation of Gear's method with its full non-linear capability in SCEPTRE represents a systems programming task of no small magnitude, from two aspects: First, Gear's program requires the requires the Jacobian matrix of the right members of the first order state variable differential equations for a network; this matrix is at present not obtainable within the SCEPTRE program, in the non-linear case. To obtain it would require extensive revisions and/or additions to phase I of SCEPTRE. Secondly, phase I of SCEPTRE is a large systems program which, for each individual input network, obtains its state variable differential equations and then manufactures another program which contains them. This program is compiled and executed in phase II, to yield the desired numerical integrations and outputting of results. Hence, a great deal of knowledge of the structure of SCEPTRE is essential; a merging of these two programs could most efficiently be obtained by means of a subcontract to a programming outfit such as the source[2] of the SCEPTRE program.

A preliminary study of the structure of the program CORNAP indicates that the programming skills and manpower resources required for a merger of this program with Gear's program and/or the matrix method appear within university capability. However, a drawback of this approach is the limitation to the processing only of linear, time invariant systems; stiff or otherwise.

An inquiry into the state of the art of various second generation nonlinear analysis type computer aided circuit design programs indicates development by the following sources: ASTAP and ECAP-II, by IBM Corporation; TRAC, by Harry Diamond Laboratories; and CIRCUS (or CIRCAL), by Boeing Scientific Laboratories. Such programs usually employ sparse matrix techniques, to cut circuit translation time; Gear type integration techniques, for stiff-system capability, and have some type of non-linear capability. The strong points and limitations of these currently being developed programs have not been investigated. They are not readily available, their reasons for development being chiefly commercial distribution through sales or rentals.

---

[2]Developed by IBM, Owego, N. Y.

VII. <u>References</u> (Sections II, V, VI)

1. H. W. Mathers, et al, "Automated Digital Computer Program for Determining Responses of Electronic Circuits to Transient Nuclear Radiation, (SCEPTRE)," IBM Corporation, Owego, N. Y., developed under contract AFWL-TR-66-126, Feb., 1967

2. L. D. Millman, et al, "CIRCUS, A Digital Computer Program for Transient Analysis of Electronic Circuits," CIRCUS Program Manual, Boeing Company, Seattle, Washington, under Contract DA-49-186-AMC-346 (X), January, 1967

3. E. H. Young, et al, "Numerical Integration and Other Techniques for Computer Aided Network Design Programming," Semi-Annual Report, NGR 47 003 026, School of Engineering, Old Dominion University, Norfolk, Virginia

4. C. W. Gear, "Numerical Integration of Stiff Ordinary Differential Equations," Department of Computer Science, University of Illinois, Urbana, Illinois, Rept. #221, January, 1967

5. C. Pottle, "A Textbook Computerized State-Space Network Analysis Algorithm," Proceedings, IEEE International Circuit Theory Symposium, Miami Beach, Florida, December 4-6, 1968

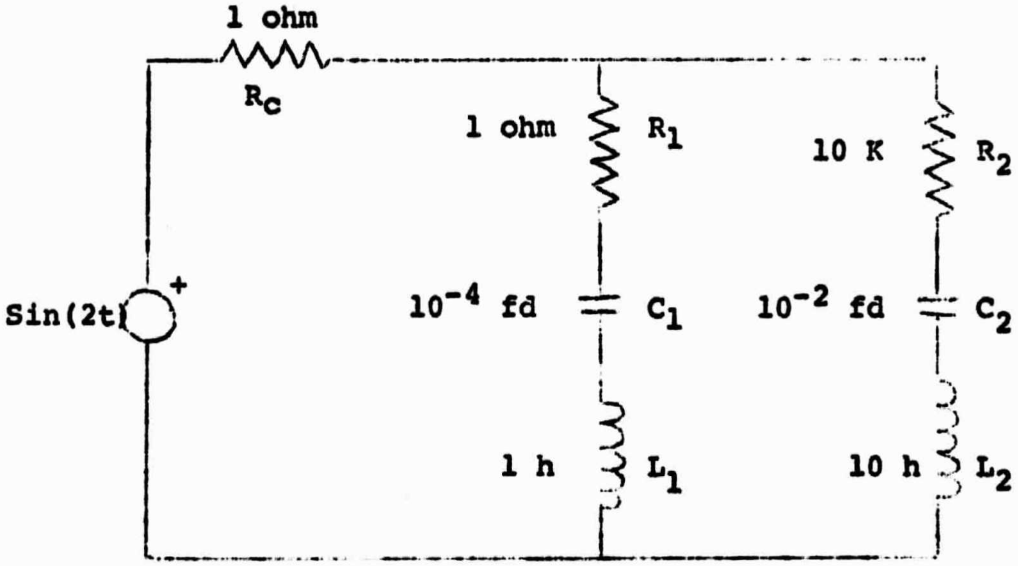


Figure 1. Typical Stiff Circuit

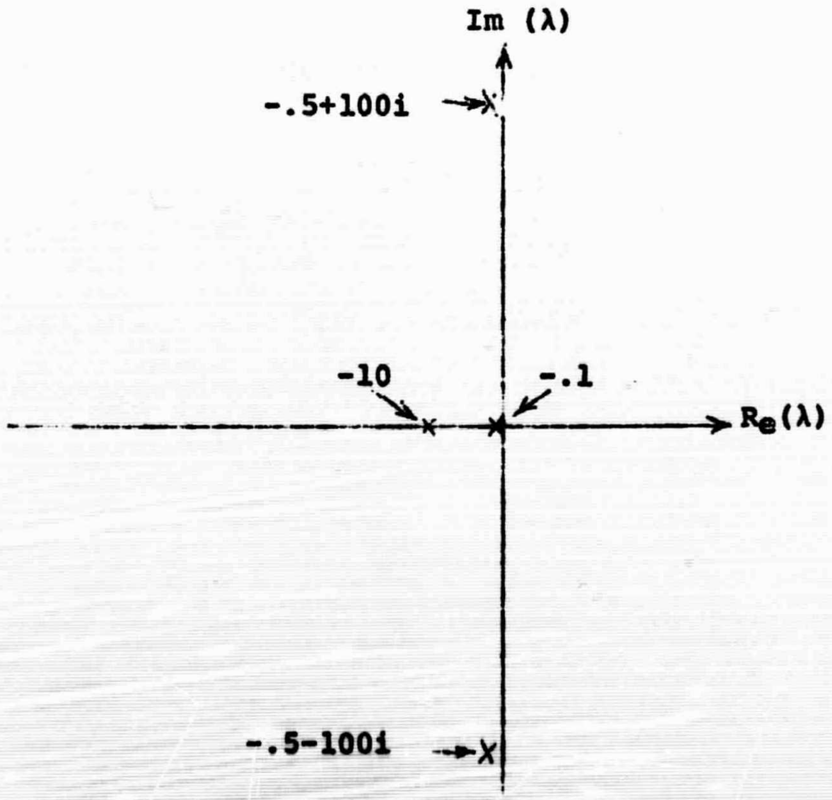
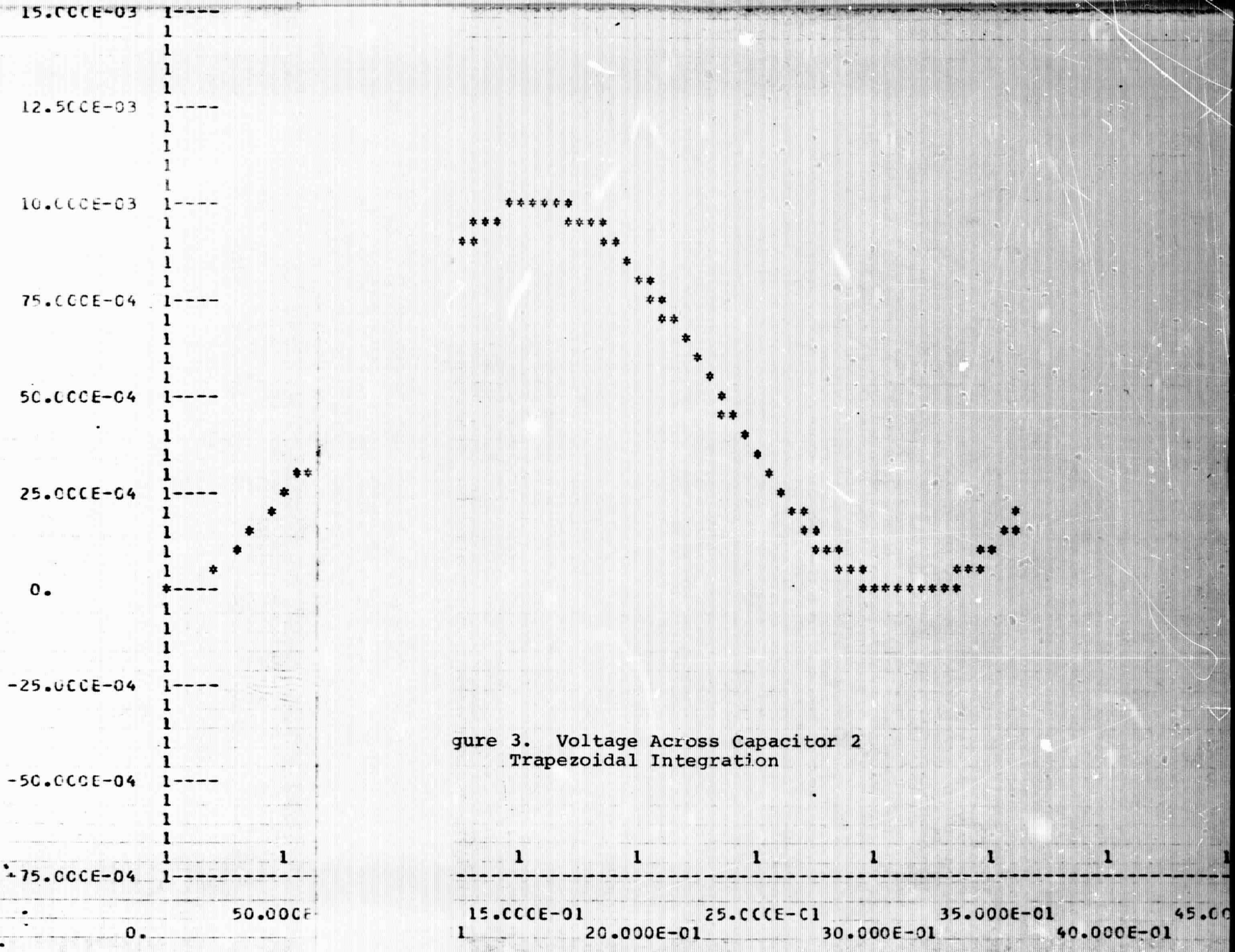


Figure 2. Eigen-Value Configuration



gure 3. Voltage Across Capacitor 2
Trapezoidal Integration

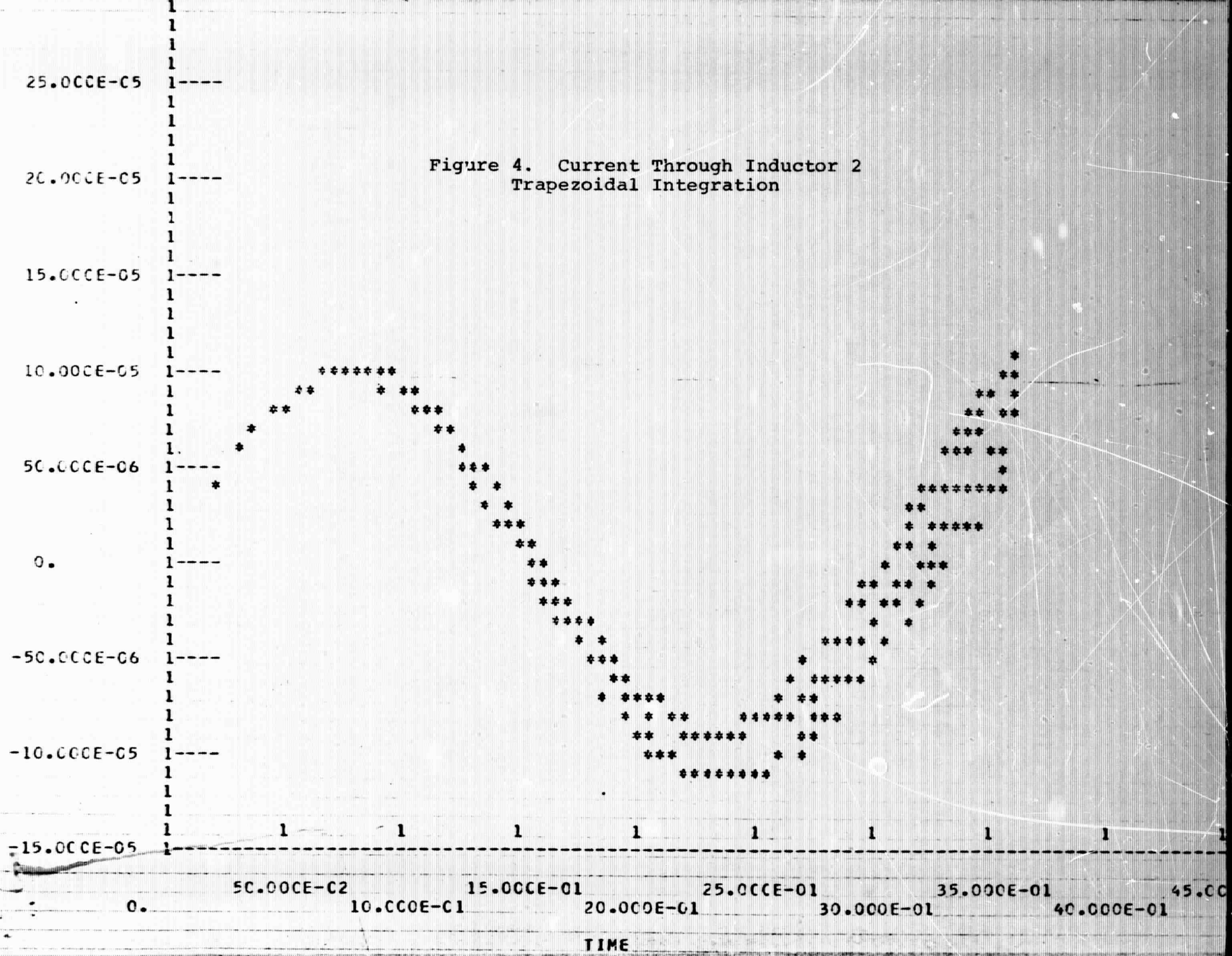


Figure 4. Current Through Inductor 2
Trapezoidal Integration

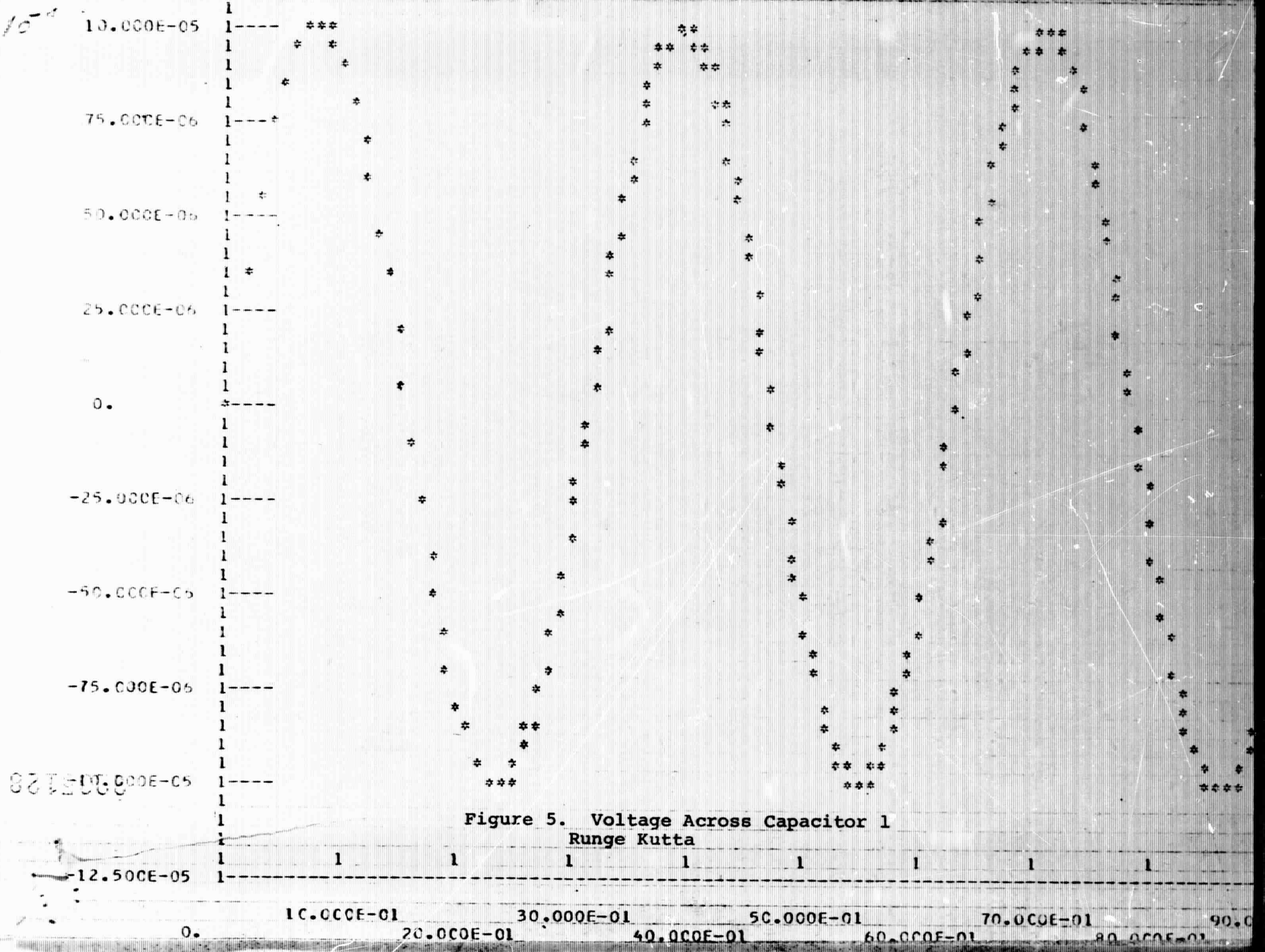
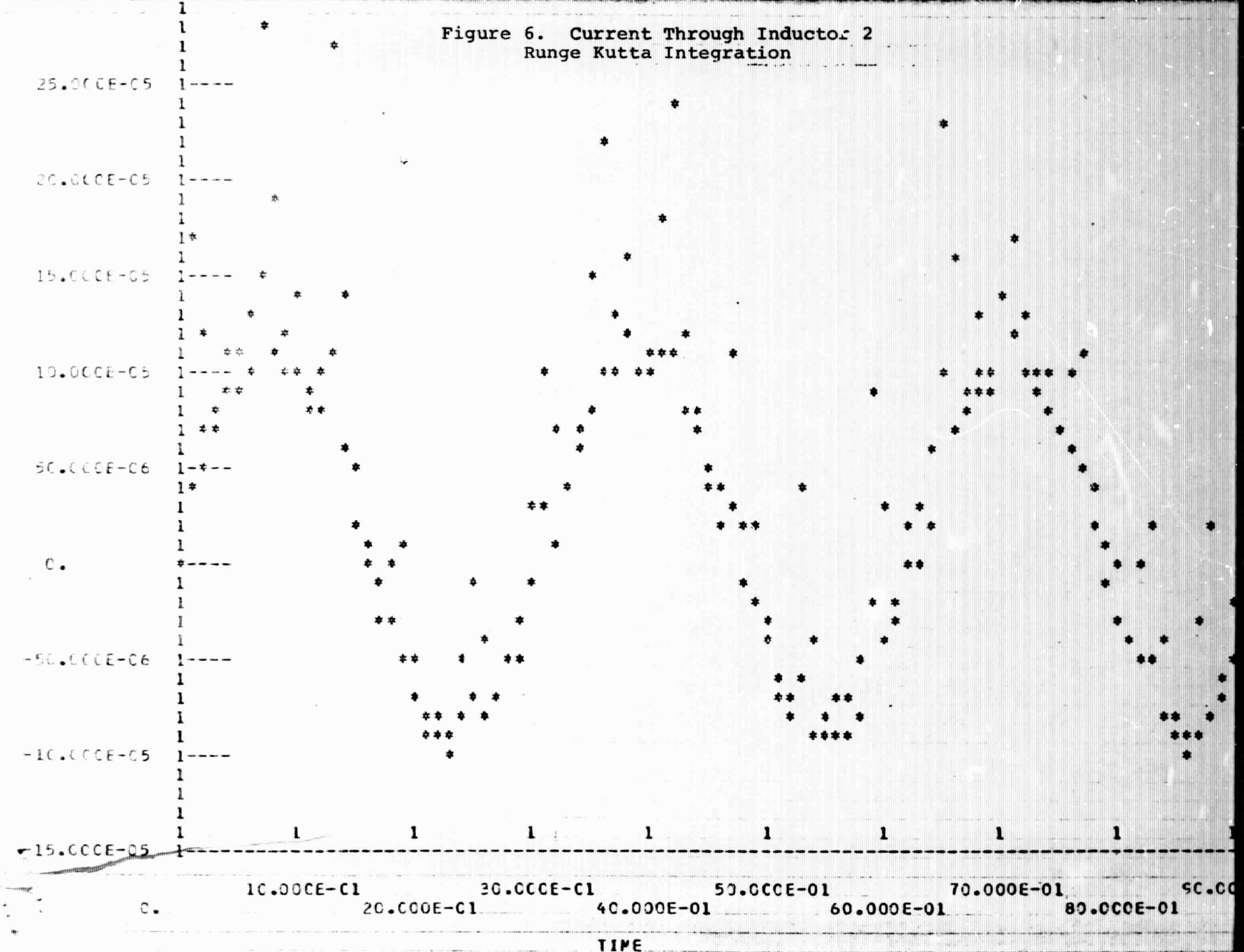


Figure 5. Voltage Across Capacitor 1
Runge Kutta

Figure 6. Current Through Inductor 2
Runge Kutta Integration



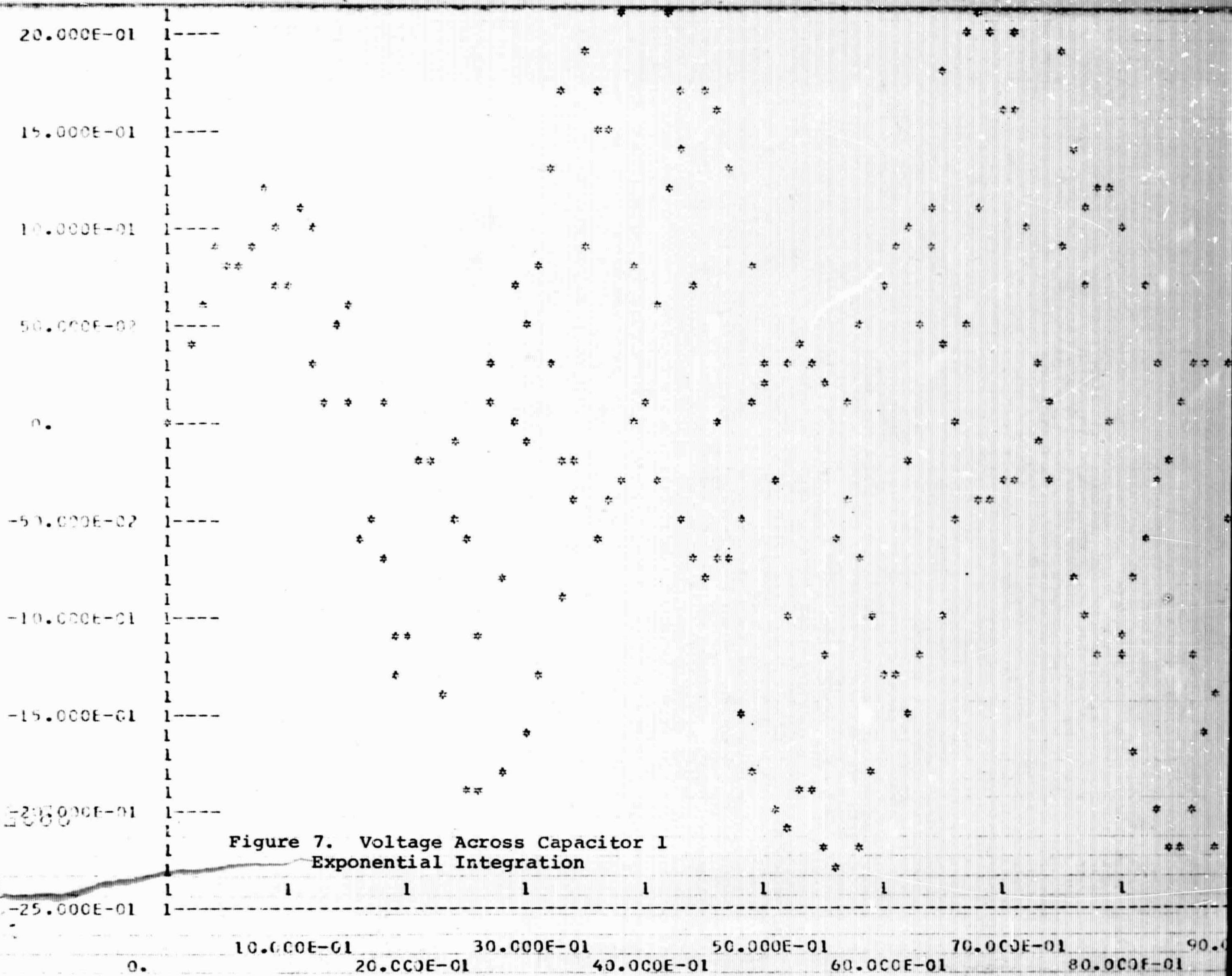


Figure 7. Voltage Across Capacitor 1
Exponential Integration

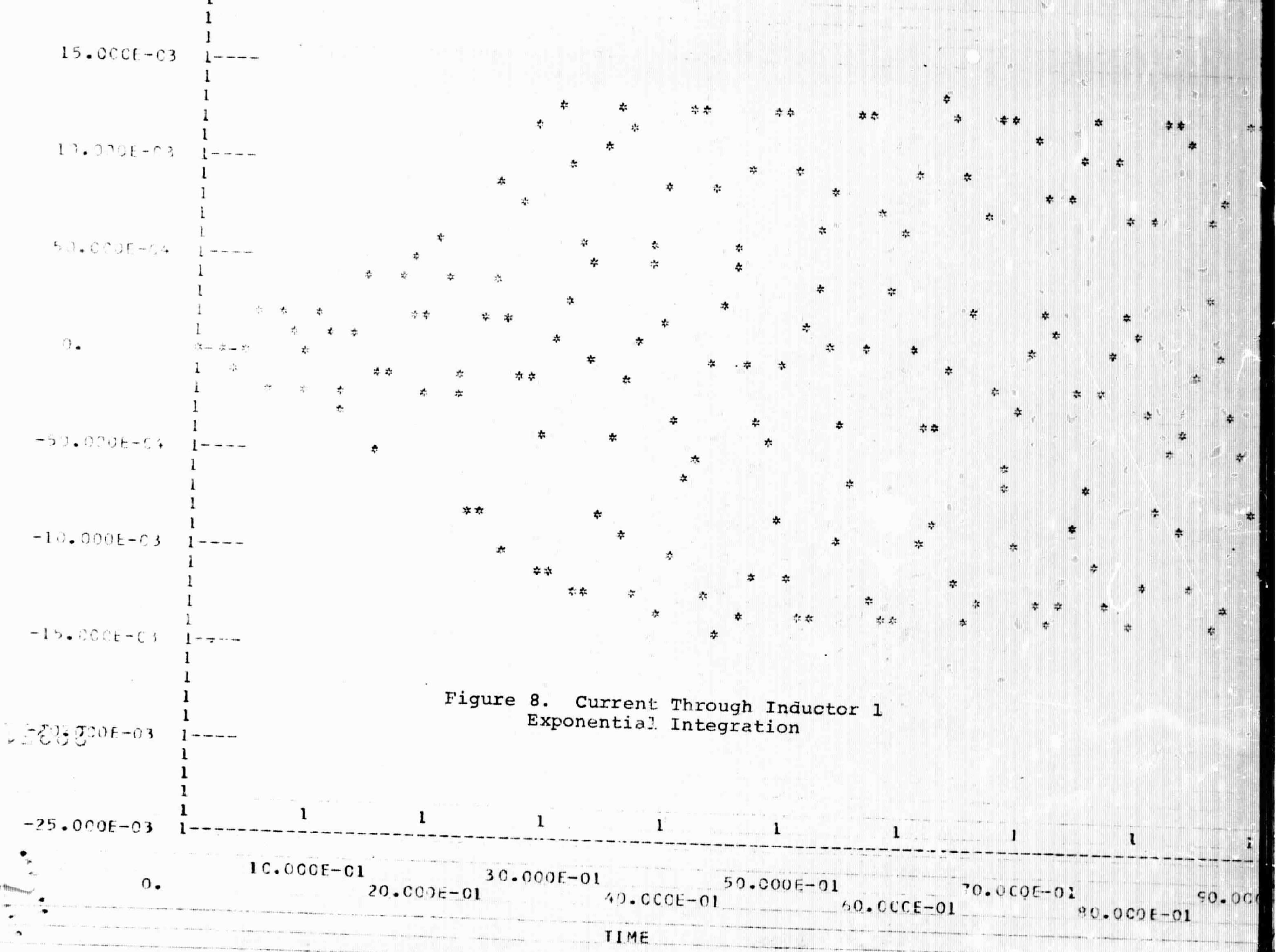
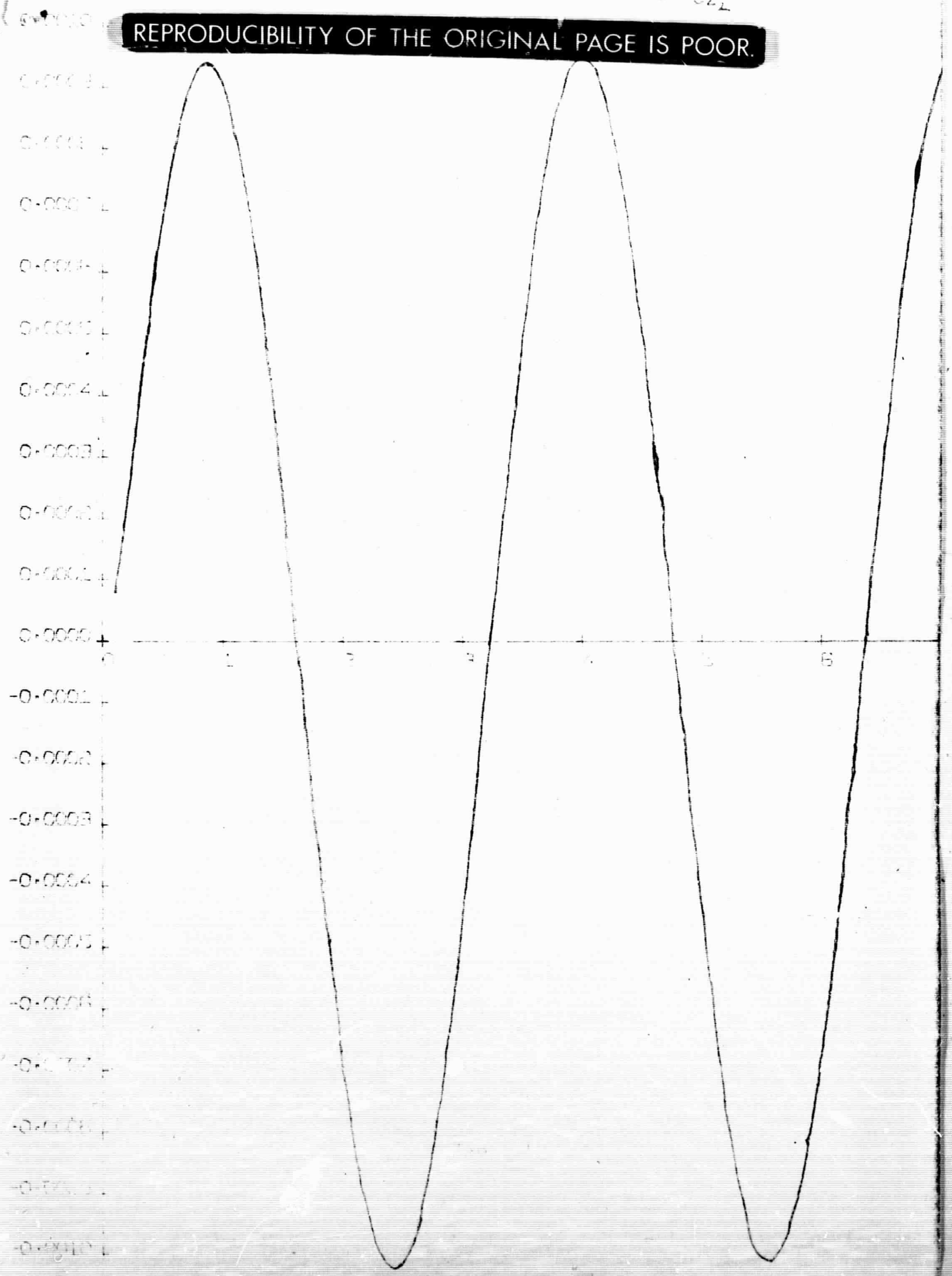


Figure 8. Current Through Inductor 1
Exponential Integration

Figure 9. Current Through Inductor 2
Gears Method

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR.



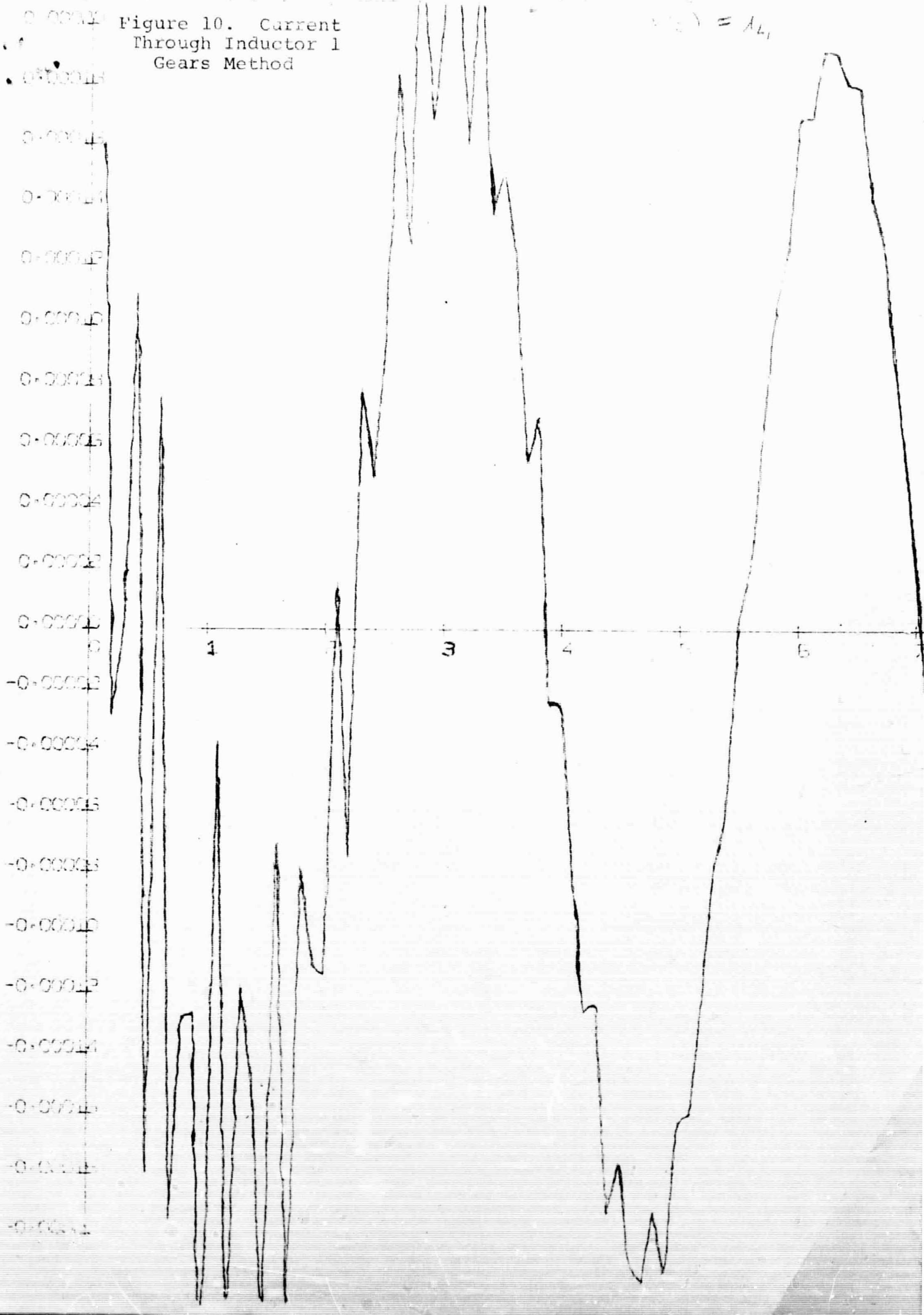


Figure 10. Current Through Inductor 1 Gears Method