An Algorithm for Solving Nonlinear Resistor Networks

By JACOB KATZENELSON

(Manuscript received May 17, 1965)

This article describes an algorithm for solving electrical networks which consist of linear and nonlinear resistors and independent sources, and where the characteristics of each of the resistors is described by a function $G_k(\cdot)$, $i = G_k(v)$, *where* $G_k(\cdot)$ *is continuous, monotonically increasing, piecewise linear, and one-to-one from* $(-\infty, \infty)$ *onto* $(-\infty, \infty)$, *and where* k *is an index which spans all the resistors in the network.*

The solution is found by solving successively equivalent linear networks which represent the nonlinear network locally and which correspond to a "solution curve." Essential to the efficiency of the computation process is the method of modifying matrices which enables the process to find the inverse of a conductance matrix by modifying another matrix rather than by matrix inversion.

The algorithm provides a fast computation method for both of the following two cases: (1.) the network contains both linear and nonlinear resistors and (2.) the sources are functions of time and the solution is required for successive values of time. In the latter case the algorithm computes each solution from the previous one rather than solving each case independently.

I. INTRODUCTION

This article considers an algorithm for solving electrical networks which consist of linear and nonlinear resistors and independent sources and where the current-voltage relation of each of the nonlinear resistors is described by a function $G_k(\cdot)$, $i = G_k(v)$, where $G_k(\cdot)$ is continuous, monotonically increasing, piecewise linear, and one-to-one from $(-\infty, \infty)$ onto $(-\infty, \infty)$, and where k is an index which spans all the resistors in the networks.

A piecewise linear network of this type can be considered to be an approximation to a more general nonlinear resistor network where the corresponding function $G_k(\cdot)$ is continuous, monotonically increasing, and one-to-one from $(\infty, -\infty)$ onto $(-\infty, \infty)$ but not necessarily piecewise linear.

Networks of the last type were discussed by various authors,[1,2,3,4,6,7] in particular Duffin, who has shown[1] that such networks have a solution which is unique. Various methods were proposed for finding the numerical value of the solution. Birkhoff and Diaz[2] gave an iterative method similar to Seidel's method[5] which is a form of relaxation procedure for solving linear equations. A direct iterative method[5] similar to the "standard" method of solving linear equations by iteration was described by Katzenelson and Seitelman.[6] An exact method (convergence in a finite number of steps) was described by Minty.[7] This latter method approximates a monotonic increasing characteristic by "stairs" and solves the approximating network by a search procedure.

The algorithm described here approximates the nonlinear resistors by piecewise linear resistors. The solution is found in a finite number of steps by successively solving linear networks, which locally represent the original network, along a path which is called the "solution curve" (Section III). Essential to the efficiency of the computation process is (6) by which the inverse of a conductance matrix is found by modifying another matrix rather than by matrix inversion.

In comparison with other solution methods, [2,6,7] the advantage of the algorithm is in providing a fast computation method for the cases where (a) the network contains both linear and nonlinear resistors and where (b) the sources are time dependent and the solution is required for all time t in some interval $[t_0, t_\alpha]$. The iteration methods, [2,6] and Minty's method,[7] do not take direct advantage of the occurrence of linear resistors in the network. In this algorithm, however, these resistors simplify the computation considerably. Case (b) is solved by sampling the time interval and for each instant of time solving the networks with constant sources whose value is equal to the value of the time-varying sources at that instant. The algorithm computes each solution from the previous one in a rather simple manner resulting in a significant reduction of computation time. Our interest in a fast calculation method for resistive networks with time dependent sources is related to the problem of solving nonlinear *RLC* networks. It was shown[4] that these networks can be viewed as a combination of three one-element-kind networks: a capacitive network, a resistive network and an inductive network. Generally, finding the time response of the *RLC* network involves solving, for each instant of time t, the three one-element-kind networks. Each of these networks is solved as a purely resistive network with the currents or voltages of the other networks playing the role of sources. An algorithm of the type described here can be used to obtain efficiently the solution of each one-element-kind network at time t from its solution at $t - \Delta t$.

Properties of piecewise linear network which are relevent to the algorithm are discussed in Section II. Section III contains a general description of the algorithm. Section IV contains a convergence proof. Section V considers the computation time which is required for solving nonlinear resistor networks. The various methods of solution are compared from this point of view. Appendix A describes a modification which reduces the computation time used by the algorithm. A step-by-step description of the algorithm is given in Appendix B.

II. PROPERTIES OF PIECEWISE LINEAR NETWORKS

This section describes the type of networks which are solved by the algorithm. A network η can be solved by the algorithm if it satisfies the following conditions:

(1.) η consists of a finite number of branches. Each branch is either a resistor (linear or nonlinear) or an independent source. Without loss of generality it can be assumed that η is connected, is nonseparable and does not contain cut sets of current sources only or loops of voltage sources only. It follows from the above that η contains a tree τ such that all voltage sources are tree branches and all current sources are links. Without loss of generality it can be further assumed that each current source appears in parallel with a resistive tree branch, and that each resistive tree branch has only one current source connected in parallel with it.[4]

(2.) The characteristics of each of the resistors of η satisfy the following conditions:

(a.) Let i and v be the current and the voltage of the resistor. The sign convention of i and v is shown in Fig. 1. The characteristics of the resistor can be represented by a function $G(\cdot)$, $i = G(v)$, where $G(\cdot)$ is a continuous, *piecewise linear*, monotonic increasing function which is one-to-one from $(-\infty, \infty)$ onto $(-\infty, \infty)$. (Fig. 2).

(b.) Each characteristic has a finite number of breakpoints in any finite interval.

It follows from (a.) and (b.) that in each finite interval the slope of $G(\cdot)$ is bounded from above and below. The lower bound is positive.

Let us discuss the properties of networks which satisfy (1.) and (2.). It follows from Duffin's work[1] that networks of this type have a unique solution. Thus, to any value of the sources corresponds one and only

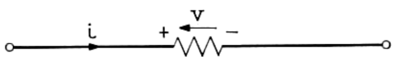

Fig. 1 — Sign convention for a branch of the network.

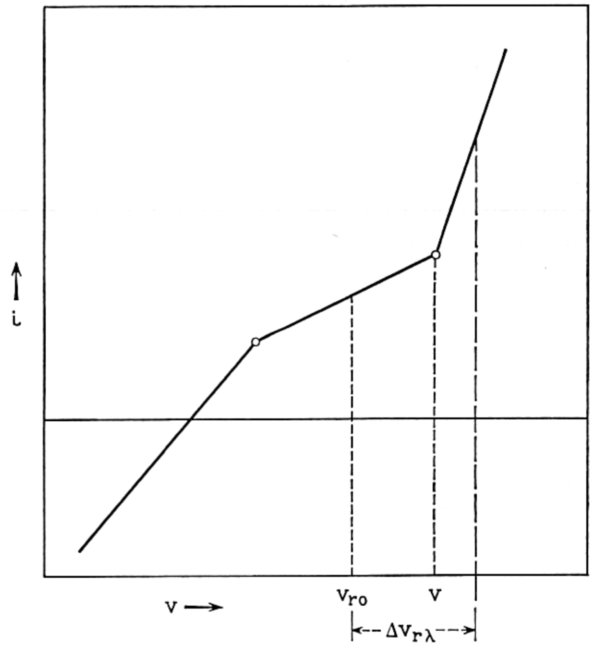

Fig. 2 — A monotonically increasing piecewise linear resistor characteristic.

one set of voltages and currents of the resistors which satisfies the Kirchhoff's laws and the branch characteristics. The purpose of our algorithm is to evaluate these voltages and currents for a given value of the sources. Before proceeding let us make a few notations. We shall choose a tree of the network and refer to branches which are in the tree as tree branches and branches which are not in the tree as links. Let τ be a tree of η such that all voltage sources are tree branches and all current sources are links of τ. Let $\mathbf{v}_r(\mathbf{i}_r)$ be a vector whose components are the voltages (the currents) of the resistive branches. Let $\mathbf{e}_r$ denote the vector whose components are the voltages of the resistive tree branches. Similarly, **E** and **J** denote the voltage and current sources of the network.

From the fact that η has a unique solution for any value of $(\mathbf{E},\mathbf{J})$, it follows that there exists a (single valued) function which maps $(\mathbf{E},\mathbf{J})$ into $\mathbf{e}_r$. The domain of this function is $\mathcal{E} \times \mathcal{J}$ where $\mathcal{E}$ and $\mathcal{J}$ denote the vector spaces corresponding to the domains of **E** and **J**, respectively.

Similarly, it follows from conditions (1.) and (2.) that **E** and $\mathbf{e}_r$ determine uniquely the currents in all the resistors and since the components of **J** are the fundamental cut set current sources it follows that there exists a (single valued) function which maps $(\mathbf{E},\mathbf{e}_r)$ into **J**. The domain of this function is $\mathcal{E}_r \times \mathcal{E}$ where $\mathcal{E}_r$ is the vector space corresponding to the domain of $\mathbf{e}_r$.

Let us fix **E**. It follows from the above that the mapping $\mathbf{J} \rightarrow \mathbf{e}_r$, for a given **E**, is one-to-one from $\mathcal{J}$ onto $\mathcal{E}_r$. In addition, it was proved[4] that for networks satisfying conditions (1.) and (2.) the mappings $\mathbf{J} \rightarrow \mathbf{e}_r$ and $\mathbf{e}_r \rightarrow \mathbf{J}$ are continuous and satisfy Lipschitz conditions on any bounded set of their respective domains.

The network η of Fig. 3 will be used for illustrating the next property of interest. Let η satisfy conditions (1.) and (2.). The tree τ of η consist of three branches 1, 2, and 3. Since 3 is a voltage source $\mathbf{e}_r$ and **J** are two dimensional vectors and the spaces $\mathcal{E}_r$ and $\mathcal{J}$ are planes.

Denote the voltage corresponding to the kth breakpoint of the first resistor by e_1^k. Consider the space $\mathcal{E}_r$ and the locus of all points $\mathbf{e}_r$ such that the voltage on the first resistor, e_{r1}, is equal to the voltage corresponding to one of the breakpoints of the resistor characteristics, e_1^k, $k = 1, 2, \cdots, n$. This locus is a set of straight lines parallel to the e_{r2} axes. Similarly the locus corresponding to the breakpoints of the second resistor are lines parallel to the e_{r1} axes. The lines corresponding to the third resistor satisfy $E + e_{r1} - e_{r2} = e_4^k$ which are lines which form a 45° angle with the axes. These lines partition the plane $\mathcal{E}_r$ into regions as shown in Fig. 4. In the case of larger networks these loci are suitable hyperplanes which partition the space $\mathcal{E}_r$ into similar regions.

The interesting fact about the regions is that inside each, the mapping $\mathbf{e}_r \rightarrow \mathbf{J}$ is linear. This property is heavily used by the algorithm.

With each region we associate a linear network called the linear

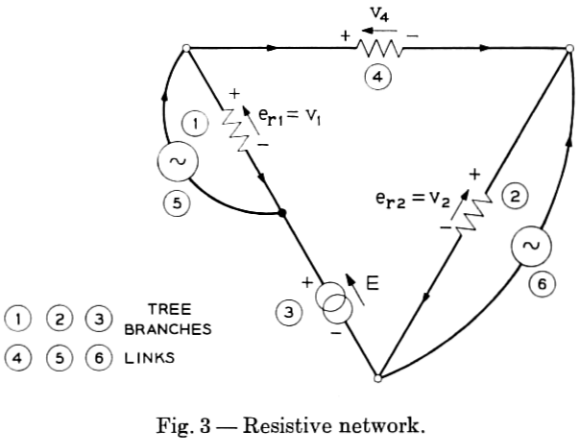

Fig. 3 — Resistive network.

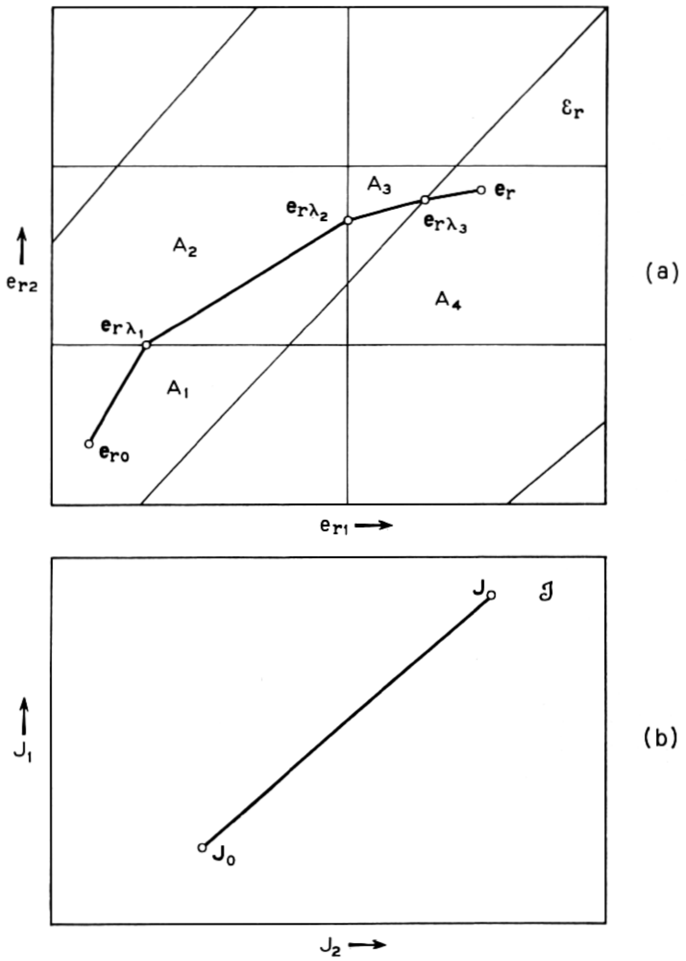

Fig. 4 — The partition of $\mathcal{E}_r$ into linear regions and the solution curve.

equivalent network of the region. It has the same topology as the original network with each nonlinear resistor replaced by a linear resistor with conductance equal to the incremental conductance of the nonlinear resistance at the region.

Before proceeding with the algorithm let us describe a convenient notation. Consider the network of Fig. 3. The cut set equation could be written as

$$g_1(e_{r1}) + g_4(e_{r1} + E - e_{r2}) = J_5$$
$$g_2(e_{r2}) - g_4(e_{r1} + E - e_{r2}) = J_6$$

where g_1, g_2, and g_4 are the functions which correspond to the characteristics of the resistors. This can be written symbolically as

$$\mathbf{G}^* \begin{pmatrix} \mathbf{e}_r \\ \mathbf{E} \end{pmatrix} = \mathbf{J};$$

where $\mathbf{G}$ is a mapping from $\mathcal{E} \times \mathcal{E}_r$ to $\mathcal{J}$ and $\begin{pmatrix} \mathbf{e}_r \\ \mathbf{E} \end{pmatrix}$ denotes a vector whose components are the components of $\mathbf{e}_r$ and $\mathbf{E}$ arranged as the notation implies, namely components of $\mathbf{e}_r$ first and components of $\mathbf{E}$ second. This notation is used as a shorthand notation for the network equations. (Somewhat more elaborate notation for equations of this type is discussed in Ref. 4).

This concludes the discussion of the properties of piecewise linear resistor networks. The algorithm itself will be discussed next.

III. A GENERAL DESCRIPTION OF THE ALGORITHM

The following contains a general description of the algorithm. A detailed, step-by-step description will appear in Appendix B.

The solution of the network problem requires the evaluation of $\mathbf{e}_r$ from

$$\mathbf{G}^* \begin{pmatrix} \mathbf{e}_r \\ \mathbf{E} \end{pmatrix} = \mathbf{J}. \tag{1}$$

Consider the spaces $\mathcal{E}_r$ and $\mathcal{J}$. Let us chose a point in $\mathcal{E}_r$ and denote it by $\mathbf{e}_{r0}$. The corresponding point in $\mathcal{J}$ is given by

$$\mathbf{G}^* \begin{pmatrix} \mathbf{e}_{r0} \\ \mathbf{E} \end{pmatrix} = \mathbf{J}_0 . \tag{2}$$

Consider the points $\mathbf{e}_{r\lambda}$ which are a solution to

$$\mathbf{G}^* \begin{pmatrix} \mathbf{e}_{r\lambda} \\ \mathbf{E} \end{pmatrix} = \mathbf{J}_0 + \lambda(\mathbf{J} - \mathbf{J}_0) \tag{3}$$

where λ attains all values between 0 and 1.

For $0 \leqq \lambda \leqq 1$, the right hand side of (3) describes a straight line in $\mathcal{J}$ which connects the point $\mathbf{J}_0$ with $\mathbf{J}$. For a given $\mathbf{E}$, (3) describes a mapping of this line from $\mathcal{J}$ to $\mathcal{E}_r$. The image of the line $(\mathbf{J}_0, \mathbf{J})$ in $\mathcal{E}_r$ space has the following properties: for $\lambda = 0$, $\mathbf{e}_{r\lambda} = \mathbf{e}_{r0}$; for $\lambda = 1$, $\mathbf{e}_{r\lambda} = \mathbf{e}_r$; which is the solution of (1). As the mapping is continuous and one-to-one onto *the line* will be mapped to a path from $\mathbf{e}_{r0}$ to $\mathbf{e}_r$. As within each region the mapping from $\mathcal{J}$ to $\mathcal{E}_r$ is linear, inside each region of $\mathcal{E}_r$ the path consists of a linear segment. An example of such a path is given in Fig. 4. This path is called *the solution curve.*

Generally speaking the algorithm calculates the solution of the nonlinear network by tracing the solution curve from its beginning at the point chosen arbitrarily $\mathbf{e}_{r\lambda} = \mathbf{e}_{r0}$, $\lambda = 0$, to its end, $\mathbf{e}_{r\lambda} = \mathbf{e}_r$, $\lambda = 1$ which is the solution of (1). This operation will be explained in the following.

The solution curve is traced by taking advantage of the fact that inside each region the mapping is linear. Consider the example of Fig. 3 and its corresponding $\mathcal{E}_r$ and $\mathcal{J}$ spaces of Fig. 4. In Fig. 4(a), the regions A_1, A_2, etc., are the regions through which the solution curve passes and the points $\mathbf{e}_{r\lambda_1}$, $\mathbf{e}_{r\lambda_2}$ etc., are the intersections of the solution curve with the boundaries. The point $\mathbf{e}_{r\lambda_1}$ is on the boundary of region A_1 which includes the initial point $\mathbf{e}_{r0}$. The point $\mathbf{e}_{r\lambda_1}$ can be calculated from $\mathbf{e}_{r0}$ as follows. Let λ_1 be the value of λ corresponding to $\mathbf{e}_{r\lambda_1}$. Let $\mathbf{G}_{A_1}$ denote the conductance matrix of the linear equivalent network for region A_1.

$$\mathbf{e}_{r\lambda_1} - \mathbf{e}_{r0} = \lambda_1 \mathbf{G}_1^{-1}(\mathbf{J} - \mathbf{J}_0). \tag{4}$$

To find $\mathbf{e}_{r\lambda_1}$, $\boldsymbol{\Delta} e_r$ is first evaluated from

$$\Delta\mathbf{e}_r = \mathbf{G}_1^{-1}(\mathbf{J} - \mathbf{J}_0). \tag{5}$$

Next, we find the largest M, $0 \leqq M \leqq 1$ such that $\mathbf{e}_{r0} + M\Delta\mathbf{e}_r$ is in region A_1.
Thus,

$$\mathbf{e}_{r\lambda_1} = \mathbf{e}_{r0} + M\Delta\mathbf{e}_{r0}.$$

The actual computation of M is described in detail in Appendix B, steps 5 and 6. Now $\mathbf{e}_{r\lambda_1}$ is considered to be a part of A_2 and $\mathbf{e}_{r\lambda_2}$ is calculated from $\mathbf{e}_{r\lambda_1}$ in the same way as $\mathbf{e}_{r\lambda_1}$ was calculated from $\mathbf{e}_{r0}$. The process continues in this way and proceeds along the solution curve until $M = 1$ indicates that $\lambda = 1$ and that the solution is found.

At this point let us consider the computational aspects of the algorithm. The network of Fig. 3 and the corresponding Fig. 4 will be used again for illustration.

At each region, (5) is used to find the intersection of the solution curve with the region boundary. The use of (5) involves the inverse of the conductance matrix of the corresponding region. In our example, the $\mathbf{e}_{r\lambda_1}$ is obtained from $\mathbf{e}_{r\lambda_0}$ and $\mathbf{e}_{r\lambda_2}$ from $\mathbf{e}_{r\lambda_1}$ by using the matrices $\mathbf{G}_{A_1}^{-1}$ and $\mathbf{G}_{A_2}^{-1}$ which correspond to the conductance matrices of the equivalent linear network in regions A_1 and A_2. The main point is that the process of obtaining the conductance matrix and inverting it directly is slow in comparison with all other operations in the algorithm and the required time increases rapidly with the size of the network. The algorithm circumvents this difficulty as follows: The inverse of the matrix of the new region is obtained by modifying the inverse of the matrix of the previous region. Consider again our example:

The algorithm computes $\mathbf{G}_{A_1}{}^{-1}$ in the first step. However, once a boundary is crossed $\mathbf{G}_{A_2}{}^{-1}$ is obtained from $\mathbf{G}_{A_1}{}^{-1}$ by the method of modifying matrices.[8,9,10] This is a method of inverting a matrix which is a modification of another matrix whose inverse is known. The formula involved is

$$(\mathbf{F} + \mathbf{IHK})^{-1} = \mathbf{F}^{-1} - \mathbf{F}^{-1}\mathbf{I}(\mathbf{KF}^{-1}\mathbf{I} + \mathbf{H}^{-1})^{-1}\mathbf{KF}^{-1} \tag{6}$$

where **F**, **I**, **H** and **K** are matrices of suitable dimensions and the inverse of **H** is assumed to exist. Note that if **I** is a column vector, **K** a row vector, **H** a 1×1 matrix and $\mathbf{F}^{-1}$ known then the calculation of the left hand side requires the inversion of a 1×1 matrix only. The application of (6) to *linear* networks is quite well known.[8] It is related to the Kron method and is used for finding the inverse of conductance matrices, adding resistors and nodes to a network, etc. The application of (6) to nonlinear networks is believed to be new.

In the example of Fig. 3, assume that while moving from A_1 to A_2 only one boundary was crossed which implies that the two equivalent linear networks differ in the conductance of one resistor only. Let this be the second resistor and let this difference be ΔG. Thus,

$$\mathbf{G}_1 = \mathbf{Q}\begin{bmatrix} G_1 & 0 & 0 \\ 0 & G_2 & 0 \\ 0 & 0 & G_3 \end{bmatrix}\mathbf{Q}_T$$

where **Q** is the fundamental cut set matrix, after voltage sources are replaced by short-circuit and current sources removed from the network. The subscript T denotes transposition. Then

$$\mathbf{G}_2 = \mathbf{Q}\begin{bmatrix} G_1 & 0 & 0 \\ 0 & G_2 + \Delta G & 0 \\ 0 & 0 & G_3 \end{bmatrix}\mathbf{Q}_T = \mathbf{G}_1 + \mathbf{Q}\begin{bmatrix} 0 & 0 & 0 \\ 0 & \Delta G & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{Q}_T$$

$$= \mathbf{G}_1 + \mathbf{Q}_{c2} \cdot \Delta\mathbf{G} \cdot [\mathbf{Q}_{c2}]_T$$

where $\mathbf{Q}_{c2}$ is a column vector equal to the second column of Q. Thus to find $\mathbf{G}_{A_2}{}^{-1}$ from $\mathbf{G}_{A_1}{}^{-1}$, (6) can be used with

$$\mathbf{I} = \mathbf{Q}_{c2}, \qquad \mathbf{K} = [\mathbf{Q}_{c2}]_T, \qquad \mathbf{F}^{-1} = \mathbf{G}_{A_1}{}^{-1} \quad \text{and} \quad \mathbf{H} = \Delta\mathbf{G}.$$

It is to be noted that when one boundary is crossed, the use of (6) requires the inversion of a 1×1 matrix only since both $\mathbf{H}$ and $\mathbf{KF}^{-1}\mathbf{I}$ are 1×1.

Thus, a difficult matrix inversion is performed only once for the initial region in which the computation starts. Inverses of matrices corresponding to other regions along the solution curve are computed successively by modifying the matrix corresponding to the previous region. This process takes only a small fraction of the time required for a direct inversion and essentially makes the algorithm as described above a usable computation process.

When the solution curve intersects a boundary it always continues to the adjacent region. The identity of the adjacent region is quite clear in Fig. 4 where the solution curve crosses one boundary at a time. Fig. 5 illustrates the case where the solution curve passes through the intersection of two boundaries and there are three adjacent regions. In order to apply (6) to this case it is necessary to find the region in which the solution curve continues behind the double boundary point. The region can be found by a search procedure which selects a region and attempts to continue the curve in it. If the attempt fails the next region is selected.

The occurrence of a solution curve intersecting a multi-boundary point is admittedly rare. However, when it occurs the search procedure can be quite lengthy for large networks. Large networks can have points in which a large number of boundaries intersect forming a large number of adjacent regions namely $2^n - 1$ where n is the number of boundaries which intersect in one point. Appendix A describes a method to overcome this difficulty.

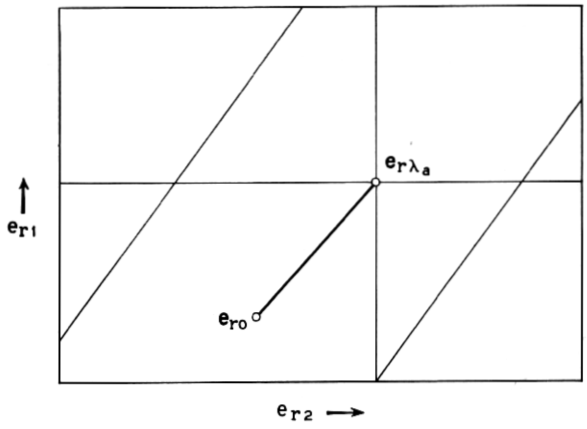

Fig. 5 — Crossing of a boundry intersection.

IV. CONVERGENCE OF THE ALGORITHM

In this section, it is proved that the algorithm converges in a finite number of steps. It will be proved that the solution curve (Fig. 4) crosses a finite number of boundaries and therefore this algorithm converges in a finite number of steps.

For simplicity, consider the network of Fig. 3. In $\mathcal{J}$ space the image of the solution curve is a straight line joining $\mathbf{J}_0$ and $\mathbf{J}$. Consider the boundaries of a region, say A_1, (Fig. 4(a)) in $\mathcal{E}_r$. These boundaries are linear segments and since the mapping $\mathcal{E}_r \rightarrow \mathcal{J}$ is continuous and linear inside each region the image of each segment is a linear segment. Now, if a segment and the solution curve have some points in common, one of two possibilities exists: (1.) they have one common point or (2.) they have a common finite interval. The first case corresponds to one boundary crossing. When a segment of this type is crossed the solution line never crosses it again. In the second case, the solution curve remains in one region since each region includes its boundaries.

Let $\| \mathbf{A} \|$ denote the square norm of $\mathbf{A}$ and for a given $\mathbf{J}$, $\mathbf{J}_0$ let S_1 be the set

$$\{\mathbf{J}_1 \mid \| \mathbf{J}_1 - \mathbf{J}_0 \| < \| \mathbf{J} - \mathbf{J}_0 \| + \varepsilon, \qquad \varepsilon > 0\}.$$

Let S_2 be the image of S_1 in $\mathcal{E}_r$. S_2 is bounded since S_1 is bounded and the mapping is continuous. It follows from the last part of condition (2.), Section II, that S_2 contains a finite number of regions. Therefore, S_1 contains a finite number of segments. Now the image in $\mathcal{J}_r$ of the solution curve is the linear segment $(\mathbf{J}, \mathbf{J}_0)$. As it is linear and included in S_1 it can cross each boundary segment only once; it therefore has a finite number of crossing points. Therefore the solution terminates in a finite number of steps.

V. COMPUTATION TIME

This section is concerned with the computation time required by the algorithm and with the way this time is used by various parts of the algorithm. The times quoted corresponds to a FORTRAN program written by the author and run on the IBM 7094.

From the nature of the algorithm, it is clear that the computer time is a function of the size of the network and the distance of the initial point from the solution. The size of a network, whose resistors are all nonlinear, was defined as the number of nonlinear resistors. As the distance between initial point and result changes from problem to problem, the following parameters rather than the total solution time were used to investigate the dependency of the computation time on the size of the network: (i) set up time — most of it is the time to invert the conductance matrix, (ii) the time used by the solution curve to cross a region from one boundary to the other, and (iii) the time used by the solution curve to reach from a boundary of a region which contains the solution to the solution itself. While (i), (ii), and (iii) depend on the size of the network, they do not depend on the initial point or the particular values of the resistors. The results for (ii) are given in Table I. Parameters (ii) and (iii) are approximately equal, and if (6) is used for setting and inverting the (i) is approximately equal to (ii) times the number of resistors in the network.

VI. CONCLUSION

We shall conclude the article by considering again the properties of the algorithm and by comparing the algorithm with the iteration method of Ref. 6.

Let us again consider the solution of the nonlinear resistor network with time-varying sources where the solution is required on some time interval $[t_0, \alpha t]$. Once the solution is found for t_0, the solution for $t_0 + \Delta t$ can be found by modifying the conductance matrix which has been used to get the solution for t_0. Thus, the solution at each instant of time is obtained by modifying the results of previous calculation rather than setting up an independent calculation and a significant reduction of computation time is obtained.

Another problem in which the algorithm can be used advantageously is finding a solution of a network which was obtained from another network by adding or subtracting branches. This use is similar to the use of (6) in linear network problems.

If some of the network resistors are linear the time required for solution is reduced. This is a result of the fact that neither boundary crossing nor checking for boundary crossing is done for the linear resistors in the network.

We have compared the computation time required to solve identical

TABLE I

Number of Resistors (All Nonlinear)	Time for Crossing a Region (msec)
2	2
5	8
9	20
15	36
24 (11 nodes)	78

problems by this algorithm and the direct iteration method[6] which is given by (11) of Appendix B.

All the networks solved for comparison met the following conditions: (1.) The initial point was a few regions away from the result, (2.) Let ϵ be the number which is compared with the norm of the error after each iteration step to determine termination. In all the tested cases ϵ defined a region which was much smaller than the "typical" regions defined by the characteristics breakpoints. It was found that under these conditions the algorithm performed much better than the iteration method and terminated in a considerably shorter time.

ACKNOWLEDGMENT

The author thanks Drs. J. L. Kelly, Jr., A. J. Goldstein and J. F. Kaiser for reading the manuscript and commenting on it.

APPENDIX A

Crossing of a Boundary Intersection

This appendix considers the case where the solution curve passes through the intersection of two or more boundaries. This case is illustrated in Fig. 5. The problem of finding the region in which the solution curve continues behind the double boundary point arises. Section III suggested finding the region by means of a search procedure. This procedure was judged inefficient since it is lengthy, especially for large networks. The purpose of this appendix is to describe the method by which the algorithm overcomes this difficulty.

Let the boundary point be $\mathbf{e}_{r\lambda_a}$ (Fig. 5). When the solution curve meets a multi-boundary point the algorithm makes a small "jump" across the boundary. The "jump" is a choice of a new point $\mathbf{e}_{rj}$ which is (1) near $\mathbf{e}_{r\lambda_a}$ and (2) nearer (in norm) to the solution than $\mathbf{e}_{r\lambda_a}$ is. The next step is to consider $\mathbf{e}_{rj}$ to be a new initial point and to continue the solution curve from it. The new initial point does not require a new inversion of the conductance matrix but might require one or more successive uses of (6) for obtaining the corresponding matrix inverse.

The point $\mathbf{e}_{rj}$ is found as follows:

$$\mathbf{e}_{rj} = \mathbf{e}_{r\lambda_a} + k(\mathbf{J} - \mathbf{J}_{\lambda a}) \tag{7}$$

where k is given by

$$k = \frac{Min(k_i)}{\| \mathbf{G} \|} \tag{8}$$

where k_i is the smallest slope of the ith resistor and the minimization is carried out on all the resistive tree branches. $\mathbf{G}$ is the conductance matrix of a linear network obtained from the original network by replacing each nonlinear resistor by a linear resistor with a conductance equal to the largest slope of the replaced resistor. $\| \mathbf{G} \|$ denotes the square norm of the matrix.[5] It was proved[6] that (7), written in the form

$$\mathbf{e}_{n+1} = \mathbf{e}_n + k(\mathbf{J} - \mathbf{J}_n) \tag{9}$$

with k given by (8), can be used for solving the nonlinear resistor network problem by iteration. This interation converges in a way which decreases the error in each step and, therefore, the use of (7) implies

$$\| \mathbf{J} - \mathbf{J}_j \| < \| \mathbf{J} - \mathbf{J}_{\lambda_a} \|$$

Thus, the new starting point $\mathbf{e}_{rj}$ is nearer in norm to the solution $\mathbf{e}_r$ than $\mathbf{e}_{r\lambda_a}$; hence the multiple boundary point is passed.

Consider the convergence of a modified algorithm which contains "jumps" whenever the solution curve meets a double boundary point. It will be proved that the modified algorithm converges in a finite number of steps.

Since (7) defines a convergent iteration process it follows, first, that the modified algorithm does converge. The fact that the solution is attained in a finite number of steps is proved in the following way. Let the solution $\mathbf{e}_r$ lie inside some region, say region A. Since the algorithm converges, it will be inside any containing $\mathbf{e}_r$ in a finite number of steps. Once the solution curve is in A, the solution is attained in one step. Thus, the algorithm terminates in a finite number of steps.

In case $\mathbf{e}_r$ is on the boundary, A is considered to consist of all regions which share this boundary. The proof proceeds as above.

APPENDIX B

A Step-By-Step Description of the Algorithm

This section gives a detailed description of the algorithm. It is assumed that the network satisfies conditions (1.) and (2.) of Section II and that a tree τ was chosen such that all voltage sources are tree branches and all current sources are links. The algorithm proceeds as follows:

(1.) Arbitrary values are chosen for the resistor tree branch voltages $\mathbf{e}_{r0}$. Let the region of $\mathcal{E}_r$ in which the point $\mathbf{e}_{r0}$ is located be called region 'a'.

(2.) Form the conductance matrix $\mathbf{G}_a$ of the linear equivalent network of region 'a' and calculate its inverse $\mathbf{G}_a^{-1}$.

(3.) Calculate the change in the tree voltages

$$\Delta\mathbf{e}_r = \mathbf{G}_a^{-1}(\lambda(\mathbf{J} - \mathbf{J}_0)) \text{ for } \lambda = 1. \tag{10}$$

(4.) Calculate the branch voltages $\mathbf{v}_r$ for all resistors and check if the new voltage value requires the crossing of a boundary.

(5.) Set $\lambda_i = 1$ if no boundary crossing is needed for the ith resistor. If a boundary crossing did occur, set

$$\lambda_i = \frac{v^i - v_{r0}{}^i}{\Delta v_r{}^i}$$

where v^i is the breakpoint voltage of the first boundary that was crossed (Fig. 2), and $\Delta v_{r\lambda}{}^i$ is the change in the ith resistor voltage which corresponds to a change $\Delta\mathbf{e}_r$ in the tree voltages. $v_{r0}{}^i$ is the ith branch voltage which corresponds to $\mathbf{e}_{r0}$.

(6.) Set

$$\lambda_a = \min \lambda_i.$$

This is the largest value of λ for which the point $\mathbf{e}_{r\lambda} = \mathbf{e}_{r0} + \lambda\Delta\mathbf{e}_r$ is in region 'a'. Thus the boundary point of region 'a' is given by

$$\mathbf{e}_{r\lambda_a} = \mathbf{e}_{r0} + \lambda_a \Delta\mathbf{e}_{r\lambda} \tag{11}$$

$$\mathbf{J}_{\lambda_a} = \mathbf{J}_0 + \lambda_a(\mathbf{J} - \mathbf{J}_0). \tag{12}$$

Note that λ_a is the M of Section III.

(7.) If $\lambda_a = 1$, the solution of the problem is in region 'a' and its value is given by (11) and (12). If $\lambda < 1$, the process continues as follows. $\mathbf{e}_{r\lambda_a}$ is on a boundary of region 'a'. This point is either on a boundary between only two regions, as point $\mathbf{e}_{r\lambda_1}$ in Fig. 4, or an intersection of two or more boundaries (Fig. 5). In the former case, the algorithm proceed to Step 8. In the later case the algorithm proceed to Step 9.

(8.) The boundary point $\mathbf{e}_{r\lambda a}$ is on the boundary between regions 'a' and 'b'. The point is considered now as part of the region 'b' and is taken to be the initial point in this region. Thus, $\mathbf{e}_{r0}$ and $\mathbf{J}_0$ are made equal to $\mathbf{e}_{r\lambda a}$ and $\mathbf{J}_{\lambda a}$ respectively. The inverse of the conductance matrix for region 'b' is calculated by modifying the inverse of the conductance matrix for region 'a' and the algorithm return to Step 2 to perform with respect to region 'b' the same operation as was performed with respect to region 'a'.

(9.) Choose a new point $\mathbf{e}_{rj}$ such that

$$\mathbf{e}_{rj} = \mathbf{e}_{r\lambda_a} + k(\mathrm{J} - \mathrm{J}_{\lambda_a})$$

where k is a constant having the dimension of resistance and given by a bound on (8). Consider $\mathbf{e}_{rj}$ to be a new initial point, set the suitable $\mathbf{G}^{-1}$ matrix (by successive use of (6)) and return to Step 3.

REFERENCES

1. Duffin, R. J., Nonlinear Networks IIa, Bull. Am. Math. Soc., *53*, Oct., 1947, pp. 963–971.
2. Birkhoff, G., and Diaz, J. B., Nonlinear Network Problems, Quart. Appl. Math. *13*, Jan., 1956, pp. 431–443.
3. Minty, G. T., Monotone Networks, Proc. Roy. Soc. (London), A, *257*, Sept., 1960, pp. 194–212.
4. Desoer, C. A., and Katzenelson, J., Nonlinear *RLC* Networks, B.S.T.J., *44*, Jan. 1965, pp. 161–198.
5. Faddeeva, V. N., *Computational Methods of Linear Algebra*, Translated by Curtis D. Benster, Chapt. II, Dover Publications, Inc., 1959.
6. Katzenelson, J., and Seitelman, L. H., An Iterative Method for Solving Networks of Nonlinear Resistors, to be published.
7. Minty, G. T., Solving Steady State Nonlinear Networks of 'Monotone' Elements, Trans. IRE. P. G. on Circuit Theory, *CT-8*, 2, June, 1961.
8. Branin, F. H., The Relation between Kron's Method and the Classical Methods of Network Analysis, IRE Wescon Convention Record, Part 2, August, 1959, pp. 3–28.
9. Householder, A. S., A Class of Methods for Inverting Matrices, Trans. Soc. Ind. and Appl. Math. *6*, 1958, pp. 189–195.
10. Householder, A. S., *Principles of Numerical Analysis*, McGraw-Hill, Inc., 1953, p. 79.