2011 International Symposium on Nonlinear Theory and its Applications
NOLTA2011, Kobe, Japan, September 4-7, 2011

DC Analysis of Piecewise-Linear Circuits Using Separable Programming

Hiroyuki Kato and Kiyotaka Yamamura

Faculty of Science and Engineering, Chuo University
Tokyo, 112-8551 Japan
Email: {katou, yamamura}@yamamura.elect.chuo-u.ac.jp

Abstract—A new algorithm is proposed for finding DC solutions of piecewise-linear circuits using separable programming. In this algorithm, we formulate the problem of finding DC solutions by a separable programming problem, and solve it by the modified simplex method using the restricted-basis entry rule. The proposed algorithm is globally convergent and can find all solutions of piecewise-linear resistive circuits.

1. Introduction

DC analysis of nonlinear circuits is one of the most central tasks in circuit simulation. In this paper, we discuss the problem of finding DC solutions of piecewise-linear (PWL) circuits that are obtained by PWL approximation of nonlinear functions.

In the DC analysis of PWL circuits, systems of PWL equations are solved by some numerical method. Several methods have been proposed for solving PWL equations; among them, the PWL Newton-Raphson method and the Katzenelson method are well-known [1],[2]. However, the Newton-Raphson method is not globally convergent and often fails to converge unless the initial point is sufficiently close to the solution. On the other hand, the convergence of the Katzenelson method (more properly, the *generalized* Katzenelson method [3]) is guaranteed under suitable conditions. Furthermore, as more extended methods, the algorithms using simplicial subdivision [4] and rectangular subdivision [5] have also been proposed.

In this paper, we propose a new algorithm using a completely different approach. In this algorithm, we formulate the problem of finding DC solutions by a constrained optimization problem involving only linear functions, and solve it by the modified simplex method of separable programming. This algorithm is not only globally convergent but also can find all solutions of PWL resistive circuits. Moreover, the proposed algorithm can be easily implemented by modifying conventional linear programming (LP) codes a little.

2. Proposed Algorithm

2.1. A System of PWL Equations to be Solved

Consider a PWL resistive circuit containing n PWL resistors, linear resistors, linear controlled sources, and independent sources. Such a circuit can be described by a system of n PWL equations [1]

$$f(x) \stackrel{\triangle}{=} Pg(x) + Qx - r = 0 \tag{1}$$

where $x = (x_1, x_2, \cdots, x_n)^T \in \mathbb{R}^n$ is a variable vector, $s = (s_1, s_2, \cdots, s_n)^T \in \mathbb{R}^n$ is a constant vector, P and Q are $n \times n$ constant matrices, $g(x) = [g_1(x_1), g_2(x_2), \cdots, g_n(x_n)]^T$ is a continuous PWL function with component functions $g_i(x_i) : \mathbb{R}^1 \to \mathbb{R}^1$ $(i = 1, 2, \cdots, n)$, and $f(x) = [f_1(x), f_2(x), \cdots, f_n(x)]^T$ is a continuous PWL function from $\mathbb{R}^n$ to $\mathbb{R}^n$. In this paper, we discuss the problem of finding solutions of (1) contained in an n-dimensional box $D = ([l_1, u_1], \cdots, [l_n, u_n])^T \subset \mathbb{R}^n$. Notice that $f(x)$ is a separable function, namely, it can be expressed as:

$$f_i(x) = \sum_{j=1}^{n} f_{ij}(x_j), \qquad i = 1, 2, \cdots, n. \tag{2}$$

Assume that we have chosen a partitioning

$$l_j = x_j^{(0)} < x_j^{(1)} < x_j^{(2)} < \cdots < x_j^{(K)} = u_j \tag{3}$$

of the interval $[l_j, u_j]$ and have defined a PWL function $f_{ij}(x_j)$ that is linear over the interval $[x_j^{(k-1)}, x_j^{(k)}]$ for $j = 1, 2, \cdots, n$ and $k = 1, 2, \cdots, K$ (see Figure 1). For the simplicity of notation, we have assumed that the number of partitioning is the same for all x_j-directions.

For any point $x_j^{(k)} \le x_j \le x_j^{(k+1)}$, $f_{ij}(x_j)$ can be expressed as

$$f_{ij}(x_j) = \lambda_j^{(k)} f_{ij}(x_j^{(k)}) + \lambda_j^{(k+1)} f_{ij}(x_j^{(k+1)}) \tag{4}$$

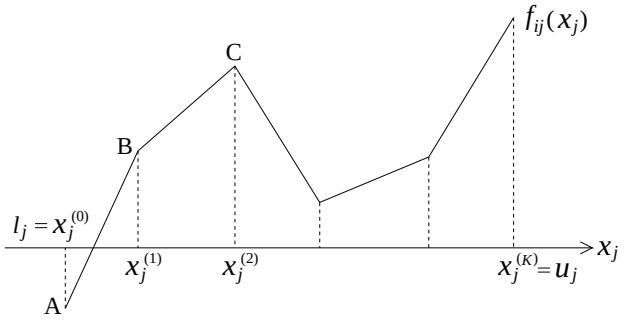

Figure 1: PWL function $f_{ij}(x_j)$.

with

$$x_j = \lambda_j^{(k)} x_j^{(k)} + \lambda_j^{(k+1)} x_j^{(k+1)} \tag{5}$$

and

$$\lambda_j^{(k)} + \lambda_j^{(k+1)} = 1, \quad \lambda_j^{(k)} \geq 0, \quad \lambda_j^{(k+1)} \geq 0. \tag{6}$$

Generalizing this representation, $f_{ij}(x_j)$ can be expressed as

$$\begin{aligned} & f_{ij}(x_j) = \sum_{k=0}^{K} \lambda_j^{(k)} f_{ij}(x_j^{(k)}) \\ & \text{where} \quad x_j = \sum_{k=0}^{K} \lambda_j^{(k)} x_j^{(k)}, \quad \sum_{k=0}^{K} \lambda_j^{(k)} = 1, \quad \lambda_j^{(k)} \geq 0. \end{aligned} \tag{7}$$

Hence, (1) is reformulated as follows:

$$\sum_{j=1}^{n} \sum_{k=0}^{K} \lambda_j^{(k)} f_{ij}(x_j^{(k)}) = 0, \qquad i = 1, 2, \cdots, n \tag{8a}$$

$$x_j = \sum_{k=0}^{K} \lambda_j^{(k)} x_j^{(k)}, \qquad j = 1, 2, \cdots, n \tag{8b}$$

$$\sum_{k=0}^{K} \lambda_j^{(k)} = 1 \tag{8c}$$

$$\lambda_j^{(k)} \geq 0, \qquad j = 1, 2, \cdots, n; \;\; k = 0, 1, \cdots, K \tag{8d}$$

$$\lambda_j^{(k)} \lambda_j^{(k')} = 0 \quad \text{if} \;\; k' > k + 1; \;\; k = 0, 1, \cdots, K - 1. \tag{8e}$$

Condition (8e) merely requires that no more than two of the $\lambda_j^{(k)}$'s be positive and if two are positive, say $\lambda_j^{(k)}$ and $\lambda_j^{(k')}$ with $k' > k$, then it must be true that $k' = k + 1$, that is, the λ_j's must be adjacent. This restriction ensures that only points lying on the PWL segments are considered part of the PWL function. To see this, note from Figure 1 that if $\lambda_j^{(0)}$ and $\lambda_j^{(1)}$ were positive and $\lambda_j^{(2)}$ to $\lambda_j^{(K)}$ were zero, then the resulting point would lie on the line segment joining points A and B, a portion of the PWL function. However, if, say $\lambda_j^{(0)}$ and $\lambda_j^{(2)}$ were allowed to be positive and the other λ_j's were all equal to zero, then a point on the line connecting A and C, which is not part of the PWL function, would be generated. Moreover if, say, $\lambda_j^{(2)}$ alone were positive, then from condition (8c) it would have to be equal to 1, and the point C lying on the PWL function would be generated.

2.2. DC Analysis Using Separable Programming

Separable programming is a technique first proposed by C. E. Miller [6] in 1963 by means of which certain types of nonlinear constrained optimization problems can be reformulated into equivalent problems involving only linear functions. The resulting approximating problems are then solved using a specially modified simplex method. In this paper, we use this idea for solving (1).

Consider the separable programming problem:

max (arbitraly constant)

subject to

$$\begin{aligned} & \sum_{j=1}^{n} \sum_{k=0}^{K} \lambda_j^{(k)} f_{ij}(x_j^{(k)}) = 0, \qquad i = 1, 2, \cdots, n \\ & \sum_{k=0}^{K} \lambda_j^{(k)} = 1 \\ & \lambda_j^{(k)} \geq 0, \qquad j = 1, 2, \cdots, n; \;\; k = 0, 1, \cdots, K \\ & \lambda_j^{(k)} \lambda_j^{(k')} = 0 \quad \text{if} \;\; k' > k + 1; \;\; k = 0, 1, \cdots, K - 1, \end{aligned} \tag{9}$$

where $\lambda_j^{(k)}$ $(j = 1, 2, \cdots, n; \; k = 0, 1, \cdots, K)$ are the variables. Note that the constraints of (9) is equivalent to (1) and $x \in D$. Hence, the solution of (1) that lies in D can be obtained by solving (9) and calculating x by (8b).

The separable programming problem (9) can be solved by conventional LP codes. The only feature requiring special attention is condition (8e). This restriction can, however, be readily accommodated, since in the ordinally simplex method the basic variables are the only ones that can be nonzero. Thus, prior to entering one of the λ_j's into the basis (which will make it nonzero), a check is made to ensure that no more than one other λ_j associated with the corresponding variable x_j is in the basis (is nonzero) and, if it is, that the λ_j's are adjacent. If these checks are not satisfied, then the λ_j to be entered into the basis is rejected and another is selected. This modification to the normal simplex rules required to ensure that condition (8e) is always met is known as *restricted-basis entry rule* [7]. Recall that the reason for imposing condition (8e) and for restricted basis entry is to ensure that the points generated lie on the PWL segments rather than between them.

In the proposed algorithm, we solve (9) by the modified simplex method mentioned above. It has been shown that the modified simplex method using the restricted-basis entry rule will yield local maximal of the separable programming (9) [7]. Since the constraints of (9) is equivalent to (1) and $x \in D$, the feasible region of (9) is a set of points that satisfies $f(x) = 0$. Hence, we can obtain the solution of (1) at the optimal point of (9)[1].

Since the modified simplex method using the restricted-basis entry rule always converges to a local maximal [7], the global convergence of the proposed algorithm is guaranteed. Moreover, the proposed algorithm can be easily implemented by modifying conventional LP codes a little

[1] The simplex method consists of Phase I and Phase II. In Phase I, we find a basic feasible solution using artificial variables. In Phase II, we optimize the objective function starting with the basic feasible solution obtained by Phase I. If there is no feasible solution, then Phase I terminates with that information [7]. Notice that, since the objective function of (9) is a constant, the simplex method for (9) consists of Phase I only.

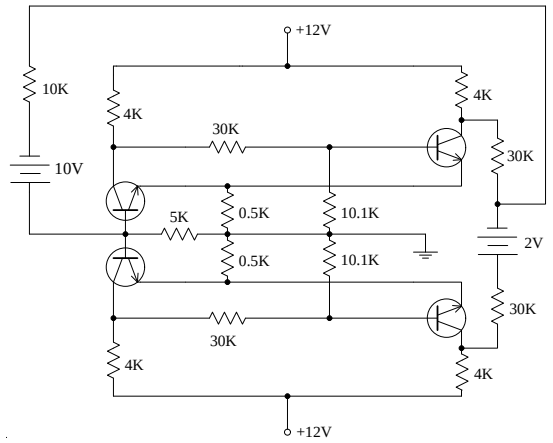

Figure 2: Transistor circuit 1.

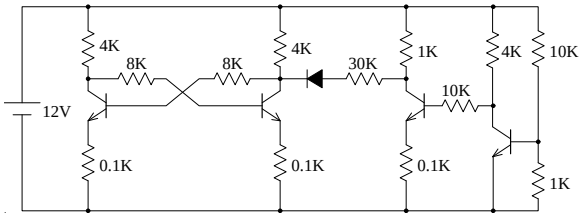

Figure 3: Transistor circuit 2.

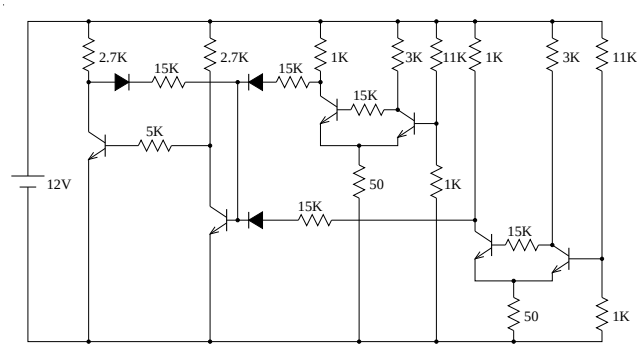

Figure 4: Transistor circuit 3.

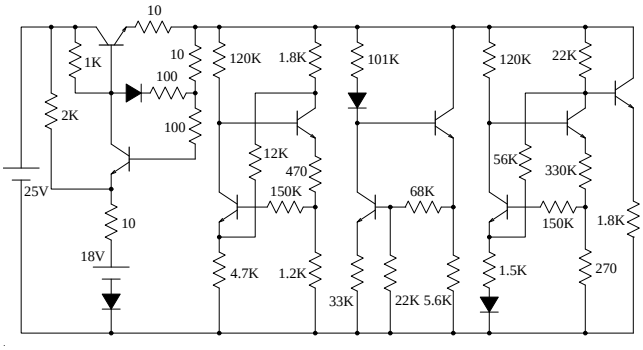

Figure 5: Transistor circuit 4.

3. Finding All Solutions of PWL Resistive Circuits

The proposed algorithm can find all solutions of PWL resistive circuits by introducing a simple technique. Suppose that we have applied the proposed algorithm to a multistate circuit and have obtained a solution $\alpha^1 \in \mathbb{R}^n$ by solving (9). Let $([x_1^{(k_1)}, x_1^{(k_1+1)}], \cdots, [x_n^{(k_n)}, x_n^{(k_n+1)}])^T$ be the linear region containing α^1, where $k_j \in \{0, 1, \cdots, K-1\}$ and there is a one-to-one correspondence between $(k_1, k_2, \cdots, k_n)^T$ and a linear region. In this case, $\lambda_j^{(k_j)} + \lambda_j^{(k_j+1)} = 1$ $(j = 1, 2, \cdots, n)$ hold. Then, it is clear that α^1 is the only solution that satisfies

$$\sum_{j=1}^{n} (\lambda_j^{(k_j)} + \lambda_j^{(k_j+1)}) = n, \tag{10}$$

and for other solutions,

$$\sum_{j=1}^{n} (\lambda_j^{(k_j)} + \lambda_j^{(k_j+1)}) < n \tag{11}$$

holds. Hence, by adding the constraint (11) to the constraints of (9), and by solving the new separable programming problem (9) with (11), we can find the second solution α^2 that is different from α^1. Repeating the same procedure, we can find the third solution α^3. Thus, we can obtain all solutions of (1) in D by solving (9) with the new constraints $N+1$ times, where N denoted the number of solutions. If the algorithm terminates with the information that the feasible region is empty, then we have obtained all solutions.

4. Numerical Examples

We implemented the proposed algorithm using the programming language C on a Dell Precision T7500 (CPU: Intel Xeon 3.33GHz). In this section, we show some numerical examples. In the implementation, we used the FORTRAN program of separable programming written in [8] as a reference.

Example 1: We first consider the transistor circuits shown in Figures 2–4 [9]. We applied the proposed algorithm to these circuits. We considered the initial box $D = ([-20, 0.5], \cdots, [-20, 0.45])^T$ and used the PWL function with ten segments that are linear on $[-20, 0]$, $[0, 0.05]$, $[0.05, 0.1], \ldots, [0.4, 0.45]$. Table 1 shows the solution of the example circuit in Figure 2 obtained by the proposed algorithm. Table 2 shows the number of iterations (pivotings) of the modified simplex method and the CPU time. The CPU time was too small and unmeasurable.

Example 2: We next consider the circuit containing n tunnel diodes discussed in [9]. The initial box is $D = ([-1, 4], \cdots, [-1, 4])^T$, and the characteristic of tunnel diodes is represented by a PWL function with ten segments. We applied the proposed algorithm to this circuit for various n. Table 3 shows the number of iterations (pivotings) of the modified simplex method and the CPU time.

Table 1: Solution of the example circuit 1.

Variable	Solution
v_{be1}	0.308463
v_{bc1}	−7.886730
v_{be2}	0.358434
v_{bc2}	0.263821
v_{be3}	0.305233
v_{bc3}	−8.121885
v_{be4}	0.358913
v_{bc4}	0.266193

Table 2: Number of iterations and CPU time (Example 1).

Circuit	n	Iterations	CPU time (s)
Figure 2	8	31	< 0.01
Figure 3	9	42	< 0.01
Figure 4	15	52	< 0.01
Figure 5	22	77	< 0.01

Table 3: Number of iterations and CPU time (Example 2).

n	Iterations	CPU time (s)
10	42	< 0.01
100	490	0.22
1000	4 851	259
2000	10 609	2 255
3000	14 176	6 277

It is seen that the proposed algorithm can solve large-scale circuits in practical computation time. It is also seen that the number of iterations is almost proportional to n. Since the computational complexity of pivot operation is $O(n^2)$, the CPU time is almost proportional to n^3.

5. Conclusion

In this paper, a new algorithm has been proposed for finding DC solutions of piecewise-linear circuits using separable programming. In the proposed algorithm, we formulate the problem of finding DC solutions by a separable programming problem, and solve it by the modified simplex method using the restricted-basis entry rule. The proposed algorithm is globally convergent and can find all solutions. Moreover, the proposed algorithm can be easily implemented by modifying the existing programs of the simplex method a little.

The simplex method has several additional techniques such as the sensitivity analysis. It is left as a future problem to apply these techniques to the proposed algorithm.

Acknowledgments

This work was supported in part by the Grants-in-Aid for Scientific Research No.20560369 of the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

[1] L.O. Chua and P. M. Lin, Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques, Prentice-Hall, Englewood Cliffs, New Jersey, 1975.

[2] J. Katzenelson, “An algorithm for solving nonlinear resistor networks,” Bell Syst. Tech. J., vol.44, no.8, pp.1605–1620, Oct. 1965.

[3] T. Ohtsuki, T. Fujisawa, and S. Kumagai, “Existence theorems and a solution algorithm for piecewise-linear resistor networks,” SIAM J. Math. Anal., vol.8, no.1, pp.69–99, Feb. 1977.

[4] J. Roos and M. Valtonen, “An efficient piecewise-linear DC analysis method for general non-linear circuits,” Int. J. Circuit Theory Appl., vol.27, no.3, pp.311–330, May. 1999.

[5] K. Yamamura and K. Horiuchi, “A globally and quadratically convergent algorithm for solving nonlinear resistive networks,” IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.9, no.5, pp.487–499, May 1990.

[6] C. E. Miller, “The simplex method for local separable programming,” in Recent Advances in Mathematical Programming, R. L. Gravesand P. Wolfe (Eds.), McGRAW-Hill Book Company, New York, 1963.

[7] S. I. Gass, “Linear Programming: Methods and Applications, Fifth Edition,” Dover Publication Inc., New York, 2003.

[8] C. McMillan, Mathematical Programming: An Introduction to the Design and Application of Optimal Decision Machines, John Wiley & Sons, 1970.

[9] K. Yamamura and K. Yomogita, “Finding all solutions of piecewise-linear resistive circuits using an LP test,” IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol.47, no.7, pp.1115–1120, July 2000.