NONLINEAR OPTIMIZATION WITH CONSTRAINTS:

A COOKBOOK APPROACH

by

L. O. Chua and G. N. Lin

Memorandum No. UCB/ERL M82/26

22 April 1982

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Nonlinear Optimization with Constraints: A Cookbook Approach*

*L. O. Chua and G. N. Lin***

ABSTRACT

Based on the stationary co-content theorem in nonlinear circuit theory and the penalty function approach in nonlinear programming theory, a canonical circuit for simulating general nonlinear programming problems with equality and/or inequality constraints has been developed. The task of solving a nonlinear optimization problem with constraints reduces to that of finding the solution of the associated canonical circuit using a circuit simulation program, such as SPICE.

A catalog of canonical circuits is given for each class of nonlinear programming problem. Using this catalog, an enginner can solve nonlinear optimization problem by a cookbook approach without learning any theory on nonlinear programming. Several examples are given which demonstrate how SPICE can be used, without modification, for solving *linear programming* problems, *quadratic programming* problems, and *polynomial programming* problems.

* Research sponsored in part by the Office of Naval Research Contract N00014-76-0572 and by the National Science Foundation under Grant ECS-8020640.

** L. O. Chua is with the Department of Electrical Engineering and Computer Sciences and the Electronic Research Laboratory, University of California, Berkeley, CA94720.

G. N. Lin is with the Shanghai Railroad Institute, Shanghai 201803, China. He is presently a visiting Research Fellow at the University of California, Berkeley.

1. Introduction

Nonlinear programming is widely encountered in engineering problems [1-4]. Indeed, almost any realistic design optimization problem subject to some practical constraints fall within the domain of nonlinear programming. The essence of a general nonlinear programming problem is to find the extremum of a nonlinear objective function subject to certain constraints. From nonlinear circuit theory we know that the operating point of a reciprocal circuit also corresponds to the extremum of some potential function under certain constraints (KCL,KVL and constitutive relations of circuit elements)[5]. Hence, if we can synthesize a *reciprocal* resistive nonlinear circuit whose potential function is identical to the given objective function being minimized, and whose element constitutive relations impose the same equality and inequality constraints, then the solution of this circuit is precisely the solution of the nonlinear programming problem. This observation was originally due Dennis [6] and subsequently extended by Stern [7]. However, their circuits contain circuit elements (e.g., dc multi-winding transformers and ideal diodes) which have been impractical to build until only recently. Moreover, their circuits are restricted to only two special classes of nonlinear programming problems; namely, linear programming and quadratic programming.

The above objections can now be overcome as follow. First, since general purpose circuit simulation programs [8], such as SPICE [9], are now widely available, the canonical " nonlinear programming " circuit can be simulated on a digital computer, thereby allowing the use of a much larger repertoire of circuit elements. Second, given *any* class of nonlinear programming problem, not just linear or quadratic problems, we give a canonical circuit in this paper which simulates the problem.

The main objective of this paper is to present a catalog of canonical nonlinear programming circuits in a strictly *cookbook fashion* so that anyone accessible to a circuit simulation program with a large enough repertoire of allowed circuit elements can solve a nonlinear programming problem without learning any theory on nonlinear programming. Indeed, since the circuit diagram remains unchanged for each class of nonlinear programming problems --hence the name "canonical circuit"-- it can be stored in the computer memory so that the user needs only supply the parameters associated with a particular problem. This catalog is given in *Section 2* along with an example and a step-by-step instruction on how to specify the parameters.

In *Section 3* we use the extremum property of potential functions from nonlinear circuit theory and the penalty function interpretation from nonlinear programming theory to justify the validity of our approach.

2. Catalog of Canonical Nonlinear Programming Circuits

In each of the following optimization program, we specify just the *objective function* to be minimized and the *equality and inequality constraints* to be satisfied. To avoid redundancy, all inequalities are assumed to be "greater then" ($\geq$) . There is no loss of generality in this assumption because if $f_i \leq 0$ or $x_i \leq 0$ for some i, we can rewrite it as $-f_i \geq 0$ or $-x_i \geq 0$, and then relabeling it by f_i and x_i , respectively. Actually, even the "equality" signs can be absorbed within the inequalities. However, consistent with the "cookbook" approach in this section, we will collect them separately before the inequality constraints. The portions of the canonical circuit for simulating the equality and inequality constraints will be identified so that if no equality (resp. inequality) constraints are present in a particular program, then the corresponding portion of the identified circuit can simply be deleted.

2.1. Linear Program

Minimizing the following objective function $\varphi(\mathbf{x})$ subject to the given constraints.

Objective function:

$$\varphi(\mathbf{x}) = a_1 x_1 + a_2 x_2 + \cdots + a_q x_q \tag{1a}$$

Equality constraints:

$$\left.\begin{aligned} f_1(\mathbf{x}) &= b_{11}x_1 + b_{12}x_2 + \cdots + b_{1q}x_q - c_1 = 0 \\ &\cdots\cdots \\ f_m(\mathbf{x}) &= b_{m1}x_1 + b_{m2}x_2 + \cdots + b_{mq}x_q - c_m = 0 \end{aligned}\right\} \tag{1b}$$

Inequality constraints:

$$\left.\begin{aligned} f_{m+1}(\mathbf{x}) &= b_{m+1,1}x_1 + b_{m+1,2}x_2 + \cdots + b_{m+1,q}x_q - c_{m+1} \geq 0 \\ &\cdots\cdots \\ f_p(\mathbf{x}) &= b_{p1}x_1 + b_{p2}x_2 + \cdots + b_{pq}x_q - c_p \geq 0 \end{aligned}\right\} \tag{1c}$$

$$\left.\begin{array}{c} x_1 - d_1 \geq 0 \\ \cdots\cdots \\ x_q - d_q \geq 0 \end{array}\right\} \qquad (1d)$$

The canonical linear programming circuit for solving this problem is given in Fig.1. Here, all diodes are pn-junction diodes whose $v_d - i_d$ curve is described by:

$$i_d = I_s(e^{v_d/V_T} - 1) \qquad (2)$$

where I_s and V_T denote the diode saturation current and thermal voltage, and can be assigned any convenient default value (usually stored in the simulation program library). The diamond-shape symbol denote a controlled voltage source if it encloses a plus-minus sign, or a controlled current source if it encloses an arrowhead.

The circuits for simulating the constraints (1b), (1c) and (1d) are enclosed within boxes N_b, N_c and N_d respectively. If there are no equality constraints, simply delete N_b. If there are no inequality constraints (1c), simply delete N_c and change subscript "p" to "m" in the controlled sources. If there are no inequality constraints (1d), simply delete the elements inside N_d.

In the special case where the constraints consists of only (1d), delete N_b, N_c, N_d, *as well as all controlled current sources*. In this case, the circuit reduces to only N_d and the dc current sources. Note that if Eq.(1d) had been deleted and embedded within Eq.(1c), the resulting circuit in this case (obtained by deleting N_b and N_d) would contain "2q" additional controlled sources. Hence, it is advantageous to consider (1d) separately.

If the computer simulation program does not allow controlled sources to depend on more then one variable, simply connect two or more controlled voltage (resp. current) sources in series (resp. in parallel).

Note that except for the two extra parameters R and A, all other parameters in Fig.1 are obtained directly from the specification Eqs.(1a)-(1d). For reasons that will be explained in *Section 3*, R should ideally be *zero* (short circuit) and A should ideally be *infinite* For circuit simulation programs, such as SPICE, which do not allow connecting a short circuit across an independent and/or controlled *voltage* source, R must be assigned a small value. Our experience with SPICE shows that too small a value for R could result in excessive computer time or non-convergence. Too large a value for R would result in an inaccurate solution. For SPICE, we recommend choosing $10^{-9} \le R \le 10^{-6}$.

Similarly, A should be specified by a large but finite number. Our experience with SPICE shows that too large a value for A could also result in excessive computer time or non-convergence. Similarly, too small a value for A would result in an inaccurate solution. For SPICE, we recommend choosing $10^{3} \le A \le 10^{5}$.

Solution Procedure:

Simulate the canonical linear programming circuit and solve for the ***node voltages*** $v_1, v_2, \cdots, v_q$. Then the solution of the linear programming problem is approximately equal to these node voltages; i.e., $x_1 = v_1$, $x_2 = v_2$, $x_q = v_q$. The solution becomes exact when $R = 0$ and $A = \infty$. Using SPICE and the above recommended values of R and A, the error is found to be less than 0.001% for most examples, which is clearly insignificant.

Example 1 ([10], p80)

minimize $\qquad \varphi = 50x_1 + 40x_2$

subject to:

inequality constraints:

$$f_1(\mathbf{x}) = 3x_1 + 2x_2 - 35 \ge 0$$

$$f_2(\mathbf{x}) = 5x_1 + 6x_2 - 60 \ge 0$$

$$f_3(\mathbf{x}) = 2x_1 + 3x_2 - 30 \ge 0$$

$$x_1, x_2 \ge 0$$

Since there are no equality constraints, the canonical linear programming circuit for solving Example 1 is shown in Fig.2. Note that since N_b is absent, only the extra parameter A is needed. The solution obtained from SPICE (with A= 10,000) is:

$$x_1 = v_1 = 8.999984, \quad x_2 = v_2 = 3.999987$$

For this simple circuit, we could obtain the solution by analyzing the circuit directly (without using a computer). In fact, since, (to be shown in *Section 3)*, the pn-junction diode-controlled source combination is used merely to approximate an ***ideal diode***, we can obtain the ***exact*** solution by solving the simplified circuit in Fig.3. Note the symbol of the ***ideal*** diode in Fig.3 differs from that of the *pn* junction diode in Fig.2, which is enclosed by a circle. Solving this circuit by inspection, we obtain

$$x_1 = v_1 = 9, \quad x_2 = v_2 = 4, \quad i_1 = 14, \quad i_2 = 0, \quad i_3 = 4$$

Remarks:

1. If some $f_j(\mathbf{x}) \le 0$ in Eq.(1c), rewrite it as

$$f_j(\mathbf{x}) = -b_{j1}x_1 - b_{j2}x_2 - \cdots - b_{jq}x_q - c_j \ge 0$$

2. If some $x_i - d_i \le 0$ in Eq.(1d), change the variable x_i in Eqs.(1a)-(1d) to $-x_i$.
3. If there are no equality constraints, set m=0 and delete Eq.(1b).
4. If there are no inequality constraints, delete Eq.(1c).

2.2. Quadratic Program

A programming problem is called a *quadratic program* if all constraints are linear (the same as Eqs.(1b)-(1d)) and the objective function is of the following form:

$$\varphi(\mathbf{x}) = \mathbf{A}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} \tag{3}$$

where **A** is a real q-vector and **G** is a $q \times q$ positive semi-definite symmetric real matrix. Quadratic programs are also a kind of convex program and have unique solutions.

The canonical quadratic programming circuit is shown in Fig.4. Note the only difference from Fig.1 is in Fig.4 where there is an additional set of multi-voltage controlled current sources $j_1, j_2, \ldots, j_q$ (on the far right of Fig.4). Their values are:

$$j_l = \sum_{j=1}^{q} g_{lj}x_j \tag{4}$$

This set of controlled current sources corresponds to the term $\frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x}$ in Eq.(3). Since the constraint functions are all linear, as in the case of the linear program, the remaining part of Fig.4 is identical to that of Fig.1.

If $\varphi(\mathbf{x})$ in Eq.(3) is given in the polynomial form instead of the matrix form, then $j_1, j_2, \ldots, j_q$ can also be specified by

$$j_l = \frac{\partial \varphi}{\partial x_l}, \qquad l = 1,2,\ldots,q \tag{5}$$

By expanding the matrix term $\mathbf{x}^T\mathbf{G}\mathbf{x}$ and differentiating it, we can see that Eq.(4) and Eq.(5) are equivalent.

Example 2 ([10], p324)

$$\text{minimize} \quad \varphi = 2x_1^2 - 6x_1x_2 + 9x_2^2 - 18x_1 + 9x_2 \tag{6}$$

$$\begin{aligned} \text{subject } to \quad f_1 &= -x_1 - 2x_2 + 12 \geq 0 \\ f_2 &= -4x_1 + 3x_2 + 20 \geq 0 \\ x_1, x_2 &\geq 0 \end{aligned} \tag{7}$$

Equation (6) can be rewritten as

$$\varphi = [-18 \;\; 9]\,\mathbf{x} + \frac{1}{2}\mathbf{x}^T\begin{bmatrix} 4 & -6 \\ -6 & 18 \end{bmatrix}\mathbf{x}$$

The simulating circuit is shown in Fig.5. The solutions of this circuit (with A=10,000) are

$$x_1 = v_1 = 6.30002, \quad x_2 = v_2 = 1.73333$$

They are in agreement with the solutions from [10].

2.3. Polynomial Program

SPICE can deal with nonlinear controlled sources with multidimensional polynomial nonlinearity. In principle, SPICE can precisely simulate those nonlinear programming problems whose objective and constraint functions are polynomial functions.

A general polynomial programming problem has the following form:

$$\text{minimize} \qquad \varphi(x_1,x_2,\ldots,x_q) \tag{8a}$$

subject to:

$$\text{equality constraints:} \quad \left.\begin{aligned} f_1(x_1,x_2,\ldots,x_q) &= 0 \\ \cdots\cdots \\ f_m(x_1,x_2,\ldots,x_q) &= 0 \end{aligned}\right\} \tag{8b}$$

$$\text{inequality constraints:} \quad \left.\begin{array}{l} f_{m+1}(x_1,x_2,\ldots,x_q) \geq 0 \\ \ldots\ldots \\ f_p(x_1,x_2,\ldots,x_q) \geq 0 \end{array}\right\} \tag{8c}$$

$$\left.\begin{array}{l} x_1-d_1 \geq 0 \\ \ldots\ldots \\ x_q-d_q \geq 0 \end{array}\right\} \tag{8d}$$

Here $\varphi, f_1, f_2, \ldots, f_p$ are multi-variable polynomial functions. The canonical simulating circuit is shown in Fig.6.

Again, if there are no equality constraints, simply let m=0 and delete N_b in Fig.6. If there are no inequality constraints, let p=m and delete N_c in Fig.6.

Note that the controlling variables of the "q" controlled current sources in Fig.6 include both voltages ($v_1,v_2, \ldots, v_q$) and currents ($i_1,i_2, \ldots, i_p$). Although some circuit simulation programs, such as SPICE, allow controlled sources controlled by multiple variables, the controlling variables must be of the same type (either all voltages or all currents). In this case, we have to convert the current variables ($i_1,i_2, \ldots, i_p$) to voltage variables. This can be done by introducing some additional controlled sources. For example, either Fig.7(a) or Fig.7(b) can convert a current variable i to a voltage variable v.

Example 3 ([11], p301)

$$\text{minimize} \quad \varphi = -x_1x_2x_3x_4$$

$$\text{subject to} \quad f_1 = x_1^3 + x_2^2 - 1 = 0$$

$$f_2 = x_1^2x_4 - x_3 = 0$$

$$f_3 = x_4^2 - x_2 = 0$$

$$x_1,\ x_2,\ x_3,\ x_4 \geq 0$$

The simulating circuit is shown in Fig.8. Taking

$$R = 1\times10^{-6}\,\Omega$$

the following solutions are obtained by SPICE:

$x_1 = v_1 = 0.793700, \quad x_2 = v_2 = 0.707106, \quad x_3 = v_3 = 0.529731, \quad x_4 = v_4 = 0.840896$

These solutions are in agreement with the solutions from [11].

Unlike linear and quadratic programs, the general polynomial programs do not necessarily have a unique solution: they may have no solution or have more than one solutions. Therefore, the corresponding simulating circuit may also have no operating point, or have more than one operating points. To which operating point the simulation program actually converges depends strongly on the initial conditions we give to the circuit. If we want to pick the global minimum, we must find all local minima. By comparing the values of φ at these extremum points, we can identify the global minimum. Unfortunately, there are currently no general circuit simulation programs capable of finding *all* operating points of a multivalued circuit.

2.4. Signomial Program

Another interesting class of nonlinear program is the *signomial program*. A function $g(\mathbf{x})$ is called a *signomial* if it has the following form[10]:

$$g(\mathbf{v}) = \sum_{i=1}^{n} c_i x_1^{a_{i1}} x_2^{a_{i2}} \cdots x_q^{a_{iq}}$$

where c_i and a_{ij} are real constants. An optimization problem is called a signomial program if its objective function and constraint functions are signomial functions. It can be stated as follows.

minimize $\qquad \varphi(x_1, x_2, \ldots, x_q)$

subject to $\qquad f_j(x_1, x_2, \ldots, x_q) \geq 0, \qquad j = 1,2,\ldots,p$

$$x_1, x_2, \ldots, x_q \geq 0$$

where φ and f_j are signomial functions of $(x_1, x_2, \cdots, x_q)$. Signomial programs are usually solved via a *geometric programming* approach. Obviously, polynomial program is a special case of signomial program. Since many programs, such as SPICE, can only deal with polynomial nonlinearities, we must convert a signomial program to a polynomial program. This can be done as follows. First, express each a_{ij} as

$$a_{ij} = \frac{m_{ij}}{n_{ij}}$$

where m_{il} and n_{il} are integers. For each variable x_l ($l = 1,2,\ldots,q$), choose the largest integer n_j from all n_{il}, and let $y_l = x_l^{\frac{1}{n_l}}$. Then,

$$\varphi'(y_1,y_2,\ldots,y_q) = \varphi(x_1,x_2,\ldots,x_q)$$

and

$$f_j'(y_1,y_2,\ldots,y_q) = f_j(x_1,x_2,\ldots,x_q)$$

are functions of $(y_1,y_2,\ldots,y_q)$ with only integer powers. If in $f_j'(y_1,y_2,\ldots,y_q)$ there are some terms including negative power, say y_l^{-m} (m is a positive integer), we can just multiply f_j' by y_l^m without changing the sign of the constraint f_j' since $y_l \geq 0$. By doing this, all constraint functions are converted to polynomial functions.

If in φ' there are some terms including negative powers, say y_l^{-m}, we introduce a new variable y_{q+1} and impose an additional equality constraint

$$y_l^{-m}\, y_{q+1}^{-1} = 1 \tag{9}$$

Substituting Eq.(9) into $\varphi'(y_1,y_2,\ldots,y_q)$, we get

$$\varphi''(y_1,y_2,\ldots,y_q,y_{q+1}) = \varphi'(y_1,y_2,\ldots,y_q)$$

Here $\varphi''(y_1,y_2,\ldots,y_q,y_{q+1})$ contains no negative power of y_l. Because we have imposed the new equality constraint (9), $\varphi''(y_1,y_2,\ldots,y_q,y_{q+1})$ and $\varphi'(y_1,y_2,\ldots,y_q)$ are equal. We can then solve it as a polynomial program.

Example 4 ([10], p376)

minimize $\qquad \varphi(x_1,x_2) = 7x_1^2 + 0.2x_1^{3.5}x_2^{2.5} + 15x_1^{-2}x_2^{-0.5}$

subject to $\qquad f_1(x_1,x_2) = 1 - 8x_1^{-2}x_2^{-1} \geq 0$

$$x_1,x_2 > 0$$

First, we introduce new variables

$$y_1 = x_1^{0.5}, \qquad y_2 = x_2^{0.5}$$

We then get a new programming problem:

minimize $\qquad \varphi'(y_1,y_2) = 7y_1^4 + 0.2y_1^7y_2^5 + 15y_1^{-4}y_2^{-1}$

subject to $\qquad f_1'(y_1,y_2) = 1 - 8y_1^{-4}y_2^{-2} \geq 0$

$$y_1, y_2 > 0$$

Next, multiply $f_1'(y_1, y_2)$ by $y_1^4 y_2^2$ and introduce a new variable y_3

$$y_3 \triangleq y_1^{-4} y_2^{-1}$$

we get the following equivalent polynomial programming problem:

minimize $\qquad \varphi''(y_1, y_2, y_3) = 7y_1^4 + 0.2y_1^7 y_2^5 + 15y_3$

subject to $\qquad f_1''(y_1, y_2, y_3) = y_1^4 y_2 y_3 - 1 = 0$

$$f_2''(y_1, y_2, y_3) = y_1^4 y_2^2 - 8 \geq 0$$

$$y_1, y_2, y_3 > 0$$

The simulating circuit is shown in Fig.9. The SPICE solution for this circuit with $R = 1\times10^{-7}$ and $A = 1{,}000$ is

$$y_1 = v_1 = 1.2335, \quad y_2 = v_2 = 1.8588, \quad y_3 = v_3 = 0.2324$$

These values are very close to the solution given in [10]. Again, since a signomial program does not necessarily have a unique solution, the corresponding simulating circuit does not necessarily have a unique operating point.

2.5. General nonlinear programs

A general nonlinear programming problem has the following form:

minimize $\qquad \varphi(x_1, x_2, \ldots, x_q) \qquad$ (8a)

subject to:

equality constraints:

$$\left.\begin{array}{l} f_1(x_1, x_2, \ldots, x_q) = 0 \\ \quad \cdots\cdots \\ f_m(x_1, x_2, \ldots, x_q) = 0 \end{array}\right\} \qquad (8b)$$

inequality constraints:

$$\left.\begin{array}{l} f_{m+1}(x_1, x_2, \ldots, x_q) \geq 0 \\ \quad \cdots\cdots \\ f_p(x_1, x_2, \ldots, x_q) \geq 0 \end{array}\right\} \qquad (8c)$$

$$\left.\begin{array}{c} x_1-d_1 \geq 0 \\ \cdots\cdots \\ x_q-d_q \geq 0 \end{array}\right\} \qquad (8d)$$

Here $\varphi, f_1, f_2, \ldots, f_p$ are arbitrary nonlinear functions. The canonical simulating circuit for this problem is the same as that shown in Fig.6. Unfortunately, since SPICE can only deal with nonlinear controlled sources with polynomial nonlinearities, this simulating circuit usually can not be implemented precisely using SPICE. To overcome this problem, we need some circuit simulation program which can deal with a broader category of nonlinear controlled sources. Or, if we are constrained to SPICE, we have to approximate nonlinear functions using polynomial functions and convert the general nonlinear program into an approximating polynomial program.

3. Theoretical Justification

A general nonlinear programming problem can be stated as follows:

Minimize a scalar function

$$\varphi(v_1, v_2, \ldots, v_q) \qquad (10)$$

subject to constraints:

$$\left.\begin{array}{c} f_1(v_1, v_2, \ldots, v_q) \geq 0 \\ \cdots \\ f_p(v_1, v_2, \ldots, v_q) \geq 0 \end{array}\right\} \qquad (11)$$

where q and p are two independent integers.

In the literature, there are a few different forms for stating the same problem. For example, instead of minimizing Eq.(10), we can also maximize its negative. Also, in Eq.(11) we can use $g(.)=-f(.)\leq 0$ instead of $f(.)\geq 0$. Besides, equality constraints are included in Eq.(11) already, since $f(.)=0$ is equivalent to $f(.)\geq 0$ and $-f(.)\geq 0$.

Therefore, to state the problem as in Eq.(10) subject to Eq.(11) does not lead to any loss of generality.

Corresponding to this problem, consider first the *ideal* simulating circuit in the canonical form shown in Fig.10. It is composed of *ideal* diodes and nonlinear controlled sources. If the original problem has a solution and if by any means we build this simulating circuit (either model it on computers or build it by using real devices in the laboratory), the node voltages $v_1, v_2, \ldots, v_q$ will be the solution minimizing (1) subject to (2).

To make the proof self-consistent, let us first introduce the concept of the potential function called *co-content* and the corresponding *stationary co-content theorem*. The potential function *co-content* $\bar{G}(\mathbf{v})$ of a reciprocal voltage-controlled n-port (Fig.11) described by $\mathbf{i}=\mathbf{i}(\mathbf{v})$ is defined by the following line integral [5]:

$$\bar{G}(\mathbf{v}) = \int_0^{\mathbf{v}} \mathbf{i}(\mathbf{v}) d\mathbf{v} \tag{12}$$

For a reciprocal voltage-controlled n-port, the Jacobian matrix of $\mathbf{i}(\mathbf{v})$ is *symmetric* and we have the following *stationary co-content theorem.*

Theorem 1: For a reciprocal voltage-controlled n-port described by $\mathbf{i}=\mathbf{i}(\mathbf{v})$, *the port voltage* $\mathbf{v}_Q$ *at the operating point is a stationary point of the co-content of the n-port, defined by (12); i. e., the gradient of* $\bar{G}(\mathbf{v})$ *vanishes at* $\mathbf{v}_Q$.

Proof: Let us denote the voltages and currents at the operating point by $\mathbf{v}_Q$ and $\mathbf{i}_Q$, respectively. Since the n-port in Fig.11 is open-circuited, obviously at the operating point we have

$$\mathbf{i}_Q = 0 \tag{13}$$

From Eq.(12) we have

$$i_j = \frac{\partial \bar{G}(\mathbf{v})}{\partial v_j}, \qquad (j = 1,2,\ldots,n) \tag{14}$$

Hence Eq.(13) implies

$$\nabla \bar{G}(\mathbf{v})|_{\mathbf{v}=\mathbf{v}_Q} = 0 \tag{15}$$

Equation (15) in turn implies $\bar{G}(\mathbf{v}_Q)$ is a stationary point of $\bar{G}(\mathbf{v})$.

Q.E.D.

Now we apply this theorem to the circuit of Fig.10. We divide the canonical simulating circuit into two parts. Suppose $\varphi(\mathbf{v})$ is continuously twice differentiable with respect to each v_i, then obviously the right hand part N_1 (Fig.12) is a voltage-controlled q-port and has a well-defined incremental conductance matrix $\mathbf{G}'$. The ij*th* element of $\mathbf{G}'$ is

$$g'_{ij} = \frac{\partial i'_i}{\partial v_j} = \frac{\partial^2 \varphi}{\partial v_i \partial v_j}, \qquad i,j=1,2,\ldots,q \tag{16}$$

Under the assumption that φ is twice differentiable, $\mathbf{G}'$ is symmetrical. Hence N_1 is reciprocal. Therefore, according to definition (12) the co-content $\overline{G}_1$ of N_1 is

$$\overline{G}_1(\mathbf{v}) = \int_0^{\mathbf{v}} \sum_{i=1}^{q} \left(\frac{\partial \varphi}{\partial v_i} dv_i \right) = \varphi(\mathbf{v}) \tag{17}$$

For the remaining part of Fig.10, we redraw it as Fig.13. It is a q-port seen from the right hand side. It can also be considered as a (p+q)-port with its p-ports on the left terminated by loads. This circuit has the following property: *If each $f_j(\mathbf{v})$ $(j=1,2,\ldots,p)$ is continuously twice differentiable with respect to v_i $(i=1,2,\ldots,q)$ and each load R_j is a voltage-controlled resistor, then the co-content $\overline{G}_2$ of the q-port defined by (12) equals the sum of co-contents of all resistors R_1 through R_p*. Namely,

$$\overline{G}_2(\mathbf{v}) = \sum_{j=1}^{p} \overline{G}_{R_j}(f_p(\mathbf{v})) \tag{18}$$

The proof of this important relationship is given in Appendix 1.

In the circuit of Fig.10, all loads R_1 through R_p are *ideal* diodes. The constitutive relation of an ideal diode (Fig.14) is

$$v \le 0,\ i=0; \qquad v=0,\ i \ge 0 \tag{19}$$

which is neither voltage-controlled nor current-controlled. But, an actual diode (as well as its model in a circuit-solving program, e.g. SPICE) is voltage-controlled and its dc characteristic can be modeled by an equation of the form (Fig.15):

$$i = I_s(e^{kv} - 1) \tag{20}$$

If I_s is chosen small enough and k is chosen large enough, Eq.(20) can approximate Eq.(19) as closely as we want. In a simulation program, such as SPICE, with a built-in pn-junction diode model, we can easily simulate this near-ideal characteristic by connecting the pn-junction diode in series with a voltage-controlled voltage source as shown in the preceding figures, when the controlling coefficient A is chosen to be a sufficiently large number.

The co-content of a diode with the v-i relation expressed by (20) is

$$\overline{G}_d(v) = \int_0^v i dv = I_s\left[\frac{e^{kv}-1}{k} - v\right] \tag{21}$$

Note that when $v<0$, $\overline{G}_d(v)$ is very small. On the other hand, when $v=0$, $\overline{G}_d(v)=0$ and when $v>0$, $\overline{G}_d(v)$ grows rapidly. For example, if we take

$$I_s = 1\times10^{-14}, \quad k = 1\times10^{6}$$

the co-content of a diode for some values of v will be as shown in Table 1.

Table 1
The co-content of a diode when
$I_s = 1\times10^{-14}$ and $k = 1\times10^{6}$

v	$\overline{G}_d(v)$
-100	1×10^{-12}
-10	1×10^{-13}
-1	1×10^{-14}
-0.1	1×10^{-15}
0	0
1×10^{-7}	5.17×10^{-23}
1×10^{-6}	7.18×10^{-21}
1×10^{-5}	2.20×10^{-6}
1×10^{-4}	2.69×10^{23}
2×10^{-4}	7.22×10^{68}

Since

$$v_{R_j} = f_j(\mathbf{v}), \qquad j = 1,2,\ldots,p,$$

in Fig.13, the co-content $\overline{G}_2$ of N_2 is given by:

$$\overline{G}_2(\mathbf{v}) = \sum_{j=1}^{p} \overline{G}_{d_j} = \overline{G}_D = \sum_{j=1}^{p} I_s\left[\frac{e^{-kf_j(\mathbf{v})}-1}{k} + f_j(\mathbf{v})\right] \tag{22}$$

where $\overline{G}_D$ denotes the *sum* of the co-content of the diodes.

Now, when we connect N_1 and N_2 together, according to the reciprocity closure theorem [6], the resulting circuit (Fig.10) is also a reciprocal voltage-controlled q-port. The co-content $\overline{G}$ of the whole circuit is

$$\overline{G}(\mathbf{v}) = \overline{G}_1 + \overline{G}_2 = \varphi(\mathbf{v}) + \sum_{j=1}^{p} I_s\left[\frac{e^{-kf_j(\mathbf{v})}-1}{k} + f_j(\mathbf{v})\right] \tag{23}$$

According to Theorem 1, the operating point of the circuit in Fig.10 is a stationary point of $\overline{G}$. We now have two situations:

(1) The operating point $\mathbf{v}_Q$ occurs *inside* the feasible region*. In such a situation, all constraints are *inactive* and all diodes work under negative or zero voltages. Consequently, $\overline{G}_D$ is almost zero over all feasible region and the second term in Eq.(23) is " swamped " by $\varphi(\mathbf{v})$. Therefore, as $I_s \to 0$ and $k \to \infty$, the minimum of $\overline{G}$ tends to the minimum of φ. Fig.16 shows the situation for the one-dimensional case. The operating point gives the solutions minimizing (1) subject to (2).

(2) The unrestricted minimum of φ occurs *outside* the feasible region. In such a situation, at least one of the diodes will work under forward voltage. The corresponding constraints are *active* now. As we can see from Fig.17, $\overline{G}_D$ grows very fast when the operating point leaves the boundary. The minimum of $\overline{G}$ is now located at a point outside the boundary but very close to the boundary. Note that although $\overline{G}$ appears to be discontinuous at this point in Fig.17(b), $\overline{G}$ is actually *differentiable* so long as $k < \infty$ in Eq.(23). Consequently the minimum point obtained by solving $\frac{d\overline{G}(\mathbf{v})}{dv} = 0$ will occur near the left boundary in Fig.17(b). Hence, $\overline{G}_D$ works as an *exterior penalty function* [3]: it forces the operating point to approach the boundary from the non-feasible region, as $I_s \to 0$ and $k \to \infty$.

Therefore, in both situations, the operating point gives the solution minimizing Eq.(10) subject to Eq.(11).

Remarks: *Two Special Constraints:*

(1) In practical programming problems it is frequent that some constraints have the following form:

$$v_l \geq d \tag{24}$$

* The set of all vectors $\mathbf{v}$ satisfying constraint (11) is called the feasible region.

where d is a constant including 0 as a special case. For such a constraint, we can simplify the canonical simulating circuit. Let us rewrite it as

$$f_j = v_l - d \geq 0$$

Obviously we have

$$\frac{\partial f_j}{\partial v_i} = \begin{cases} 1 & i = l \\ 0 & i \neq l \end{cases}$$

According to the ideal canonical simulating circuit in Fig.10, the circuit corresponding to this constraint is shown in Fig.18. Note that the controlled source in this simplified circuit depends only on the port voltage v_j and a constant d, which can be simulated by a battery. Hence, the circuit in Fig.18 can be replaced by the equivalent circuit in Fig.19.

When $d = 0$, the constraint circuit in Fig.19 simplifies further to the circuit in Fig.20. This explains why in Eqs.(1d) and (8d), as well as in Figs.1, 4, and 6, we specify Eq.(24) as a separate constraint, even though it is a special case of the other more general constraints.

(2) For an equality constraint

$$f_j(.) = 0$$

the corresponding part of the simulating circuit is shown in Fig.21. The two back-to-back ideal diodes in parallel form a short circuit. Since short-circuiting of a controlled voltage source is not allowed in many simulation programs, such as SPICE, we can use the circuit in Fig.22 to replace the circuit in Fig.21, where R is chosen to be a very small resistor. The co-content of a linear resistor R is

$$\bar{G}_R(v) = \frac{v^2}{2R}$$

If R is very small, $\bar{G}_R(v)$ will be very large when $v \neq 0$. Thus, choosing an adequately small value for R (for example, $R = 1\times10^{-6} \sim 1\times10^{-9}$), the circuit of Fig.22 will act as a penalty function for the corresponding equality constraints.

4. Concluding Remarks

(1) For a general nonlinear programming problem stated in Eqs.(10) and (11) we have developed a canonical simulating circuit shown in Fig.10. This circuit is canonical in the sense that its structure is invariant and can therefore be stored in the computer library. For a specific problem, the user need only specify the relevant parameters and functions.

(2) For certain classes of programming problems (linear program, quadratic program, polynomial program and signomial program) we have shown how to implement the canonical simulating circuit on SPICE in a cookbook fashion.

(3) For those programming problems which have more than one local minima, the SPICE solution in general gives only a local minimum. Which operating point SPICE actually converges to will depend largely on the initial conditions. Except for the piecewise-linear method described in [12], no general algorithm is currently available for finding the global minimum.

5. Appendix 1

Our task in this appendix is to prove:

(1) The q-port in Fig.13 has a well-defined co-content when R_1 through R_p are voltage-controlled resistors.

(2) The co-content of this q-port equals the sum of the co-contents of R_1 through R_p.

Assume each function $f_j(\mathbf{v})$ (j=1,2,...,p) is continuously twice differentiable and each resistor R_j (j=1,2,...,p) has a differentiable v-i relationship $i = R_j(v)$. Then the current through the mth port is

$$i_m^{**}(\mathbf{v}) = \sum_{j=1}^{p} R_j(f_j(\mathbf{v}))\frac{\partial f_j(\mathbf{v})}{\partial v_m}$$

The $mnth$ element of the conductance matrix **Y** is:

$$y_{mn} = \frac{\partial i_m^{**}}{\partial v_n} = \sum_{j=1}^{p}\left[\frac{\partial R_j}{\partial f_j}\frac{\partial f_j}{\partial v_n}\frac{\partial f_j}{\partial v_m} + R_j(f_j(\mathbf{v}))\frac{\partial^2 f_j}{\partial v_m \partial v_n}\right]$$

Under the assumptions mentioned above, we have

$$y_{mn} = y_{nm} \qquad m,n = 1,2,\ldots,q$$

Therefore the q-port is *reciprocal* and hence has a well-defined co-content; namely,

$$\overline{G} = \sum_{l=1}^{q} \int_0^{v_l} \left(\sum_{j=1}^{p} i_j \frac{\partial f_j}{\partial v_l} \right) dv_l = \sum_{j=1}^{p} \int_0^{f_j} -i_j \, df_j = \sum_{j=1}^{p} \overline{G}_{R_j} \tag{25}$$

Q.E.D.

Another interesting fact is that if all ports on the p-port side are open-circuited, the co-content of this (p+q)-port is zero. This is obvious from Eq.(17) if we notice the co-content of a linear resistor R is $\frac{v^2}{2R}$ and

$$\lim_{R \to \infty} \frac{v^2}{2R} = 0$$

Since this (p+q)-port has such a special property and since it reduces to a (p+q)-port transformer (apart from the addition of a set of dc voltage sources at the q-port side) when all $f_j(.)$ are linear functions, it can be considered as a nonlinear generalization of a (p+q)-port transformer.

6. References

[1] W. I. Zangwill, *Nonlinear Programming*, Prentice-Hall Inc., Englewood Cliffs, N. J., 1969.

[2] M. Avriel, *Nonlinear Programming* Prentice-Hall Inc., Englewood Cliffs, N. J., 1976.

[3] D. A. Wismer and R. Chattergy, *Introduction to Nonlinear Optimization*, Elsevier, 1978.

[4] R. J. Duffin, E. L. Peterson and C. Zener, *Geometric Programming-Theory and Application*, John Wiley & Sons, 1967.

[5] L. O. Chua, "Stationary Principles and Potential Functions for Nonlinear Networks", *Journal of the Franklin Institute*, Vol.296, No.2, pp.91-114, August 1973.

[6] J. B. Dennis, *Mathematical Programming and Electrical Networks*, Chapman & Hall, London, 1959.

[7] T. E. Stern, *Theory of Nonlinear Networks and Systems*, Addison-wesley,1965.

[8] L. O. Chua and P. M. Lin, " *Computer-aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*." Englewood Cliffs, N. J., Prentice-Hall, 1975.

[9] A. Vladimirescu, K. Zhang, A. R. Newton, D. O. Pederson and A. Sangiovanni-Vincentelli, "SPICE Version 2G User's Guide", Electronics Research Laboratory, University of California, Berkeley, Aug.10, 1981.

[10] C. S. Beightler, D. T. Phillips and J. W. Douglass, *Foundation of Optimization*, Prentice-Hall, 1979.

[11] G. P. Szego, (editor), *Minimization Algorithms–Mathematical theories and computer results*, Academic Press, 1972.

[12] L. O. Chua and Robin L. P. Ying, " Finding All Solutions of Piecewise-linear Equations ", *International Journal of Circuit Theory and Applications*, in press.

Figure captions

Fig. 1 The simulating circuit for a linear program.

Fig.2 The simulating circuit of Example 1.

Fig.3 An ideal simulating circuit of Example 1.

Fig.4 The simulating circuit for a quadratic program.

Fig.5 The simulating circuit of Example 2.

Fig.6 The simulating circuit for a polynomial program.

Fig.7 Two circuits for converting a current variable i to a voltage variable v.

Fig.8 The simulating circuit of Example 3.

Fig.9 The simulating circuit of Example 4.

Fig. 10 The *ideal* simulating circuit for a general nonlinear program.

Fig. 11 An n-port.

Fig. 12 The first part of the general simulating circuit.

Fig. 13 The second part of the general simulating circuit.

Fig. 14 An ideal diode.

Fig. 15 A pn-junction diode.

Fig. 16 The situation when $\min \varphi$ occurs inside the feasible region (one-dimensional case).
(a) The function $\varphi(v)$ and $\overline{G}_D(v)$.
(b) The fuction $\overline{G}(v) = \varphi(v) + \overline{G}_D(v)$. Note that $v(Q) \approx v(Q')$.

Fig. 17 The situation when $\min \varphi$ occurs outside the feasible region (one-dimensional case).
(a) The function $\varphi(v)$ and $\overline{G}_D(v)$.
(b) The fuction $\overline{G}(v) = \varphi(v) + \overline{G}_D(v)$ is differentiable at Q'.

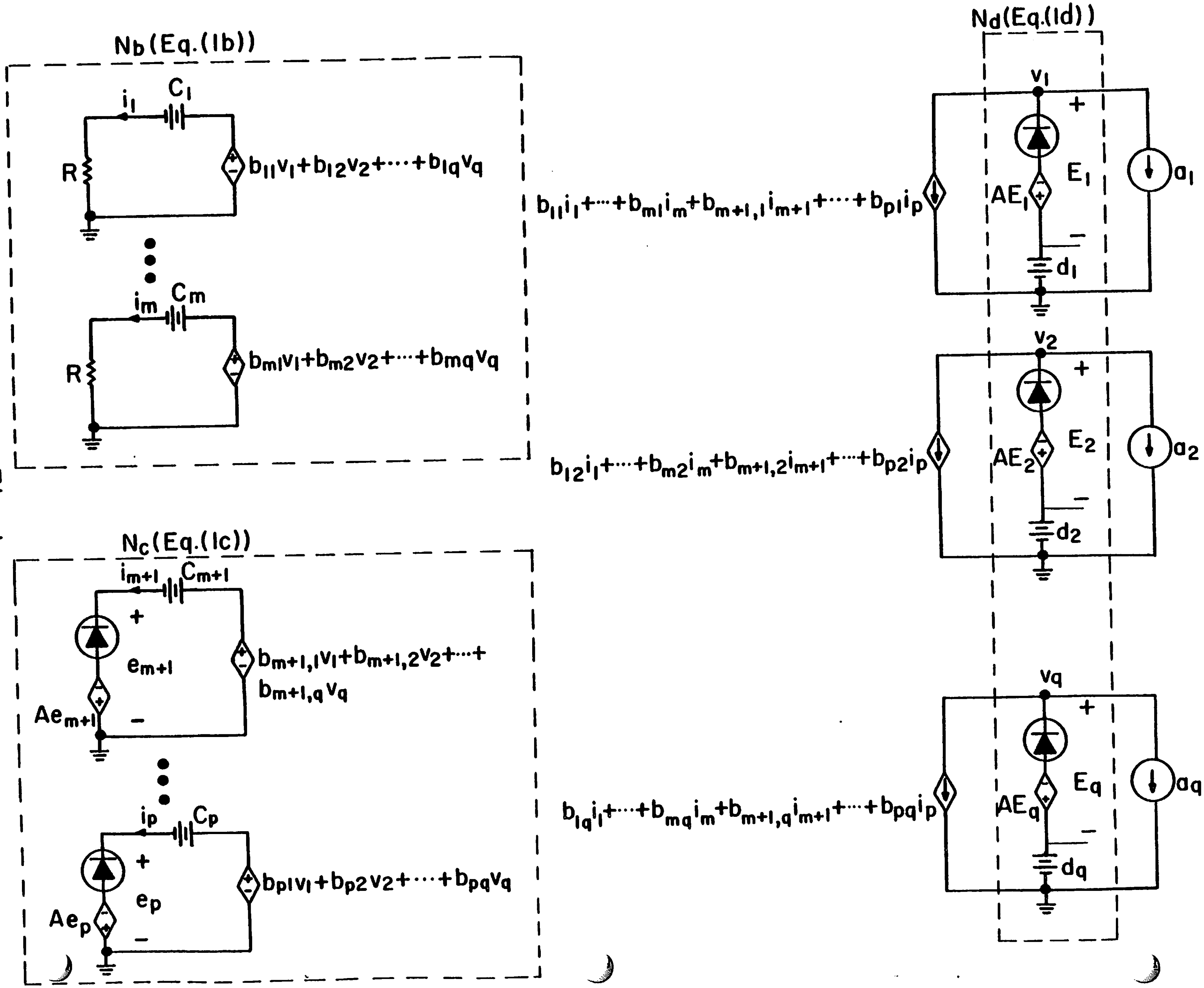
Fig. 18 The simulating circuit corresponding to the constraint $v_l \geq d$.

Fig. 19 The equivalent simulating circuit of Fig. 18.

Fig.20 The equivalent simulating circuit corresponding to the constraint $v_l \geq 0$.

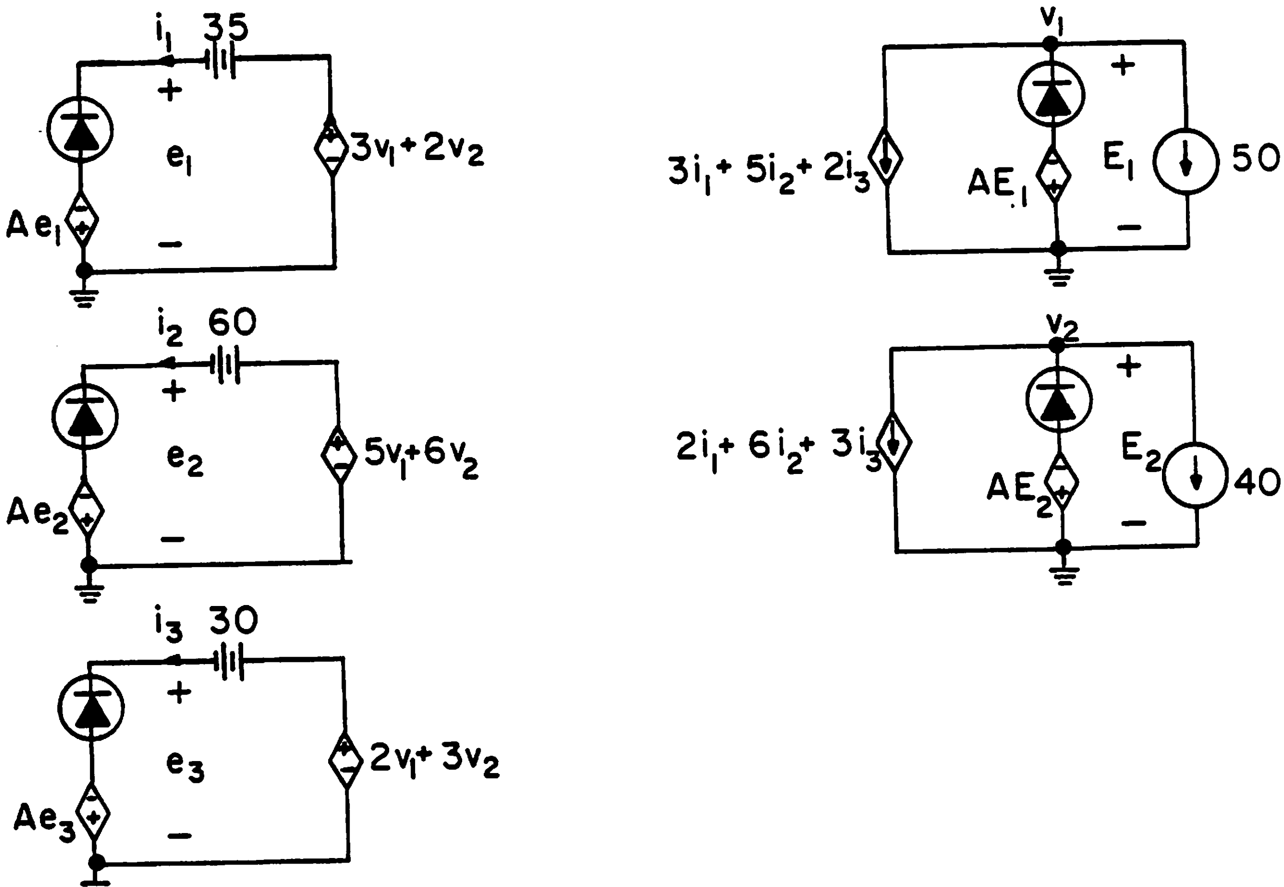
Fig.21 The circuit for simulating an equality constraint.

Fig.22 A practical method for simulating the equality constraint imposed by Fig.21. Here, R is chosen to be a very small but positive number.



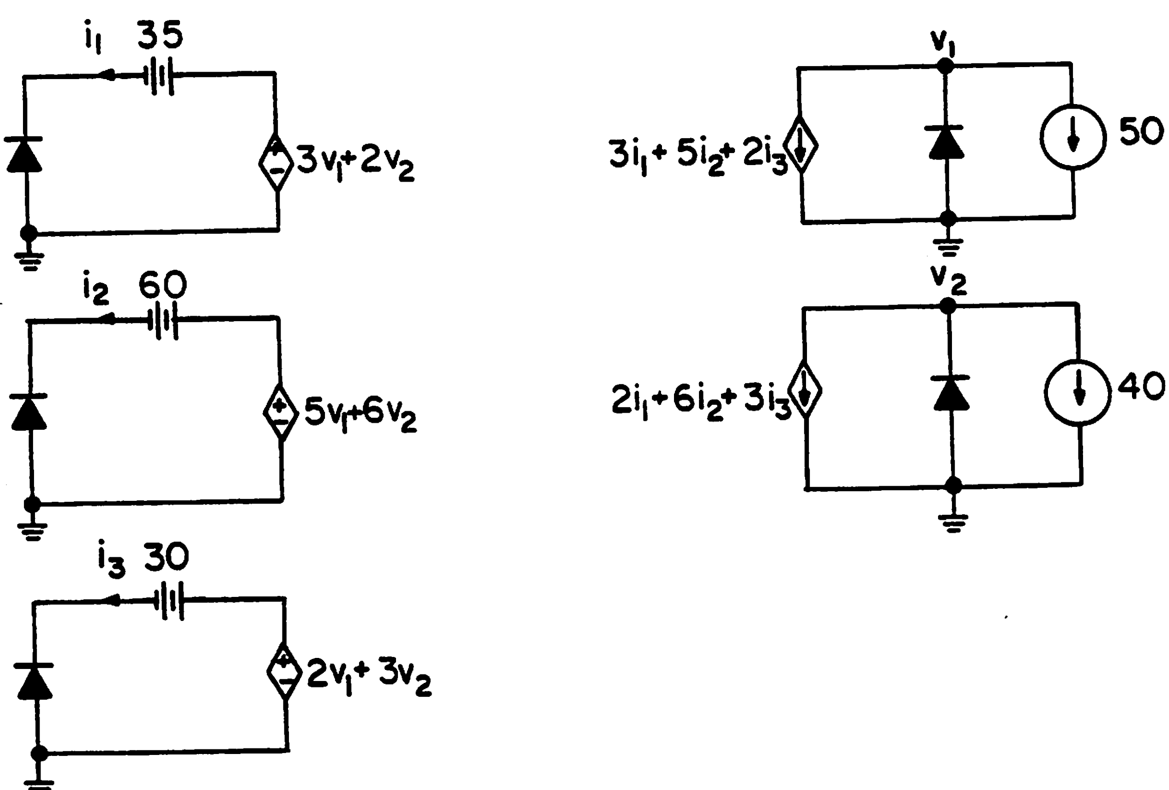
Nb(Eq.(1b))
i1
C1
R
b11v1+b12v2+····+b1qvq
im
Cm
R
bm1v1+bm2v2+····+bmqvq
Nc(Eq.(1c))
im+1
Cm+1
em+1
Aem+1
bm+1,1v1+bm+1,2v2+····+
bm+1,qvq
ip
Cp
ep
Aep
bp1v1+bp2v2+····+bpqvq
b11i1+····+bm1im+bm+1,1im+1+····+bp1ip
b12i1+····+bm2im+bm+1,2im+1+····+bp2ip
b1qi1+····+bmqim+bm+1,qim+1+····+bpqip
Nd(Eq.(1d))
v1
E1
AE1
d1
a1
v2
E2
AE2
d2
a2
vq
Eq
AEq
dq
aq

Fig. 1



i1 35
e1
Ae1
3v1+2v2
i2 60
e2
Ae2
5v1+6v2
i3 30
e3
Ae3
2v1+3v2
v1
3i1+5i2+2i3
AE1
E1
50
v2
2i1+6i2+3i3
AE2
E2
40

Fig. 2



i1 35
3v1+2v2
i2 60
5v1+6v2
i3 30
2v1+3v2
v1
3i1+5i2+2i3
50
v2
2i1+6i2+3i3
40

Fig. 3

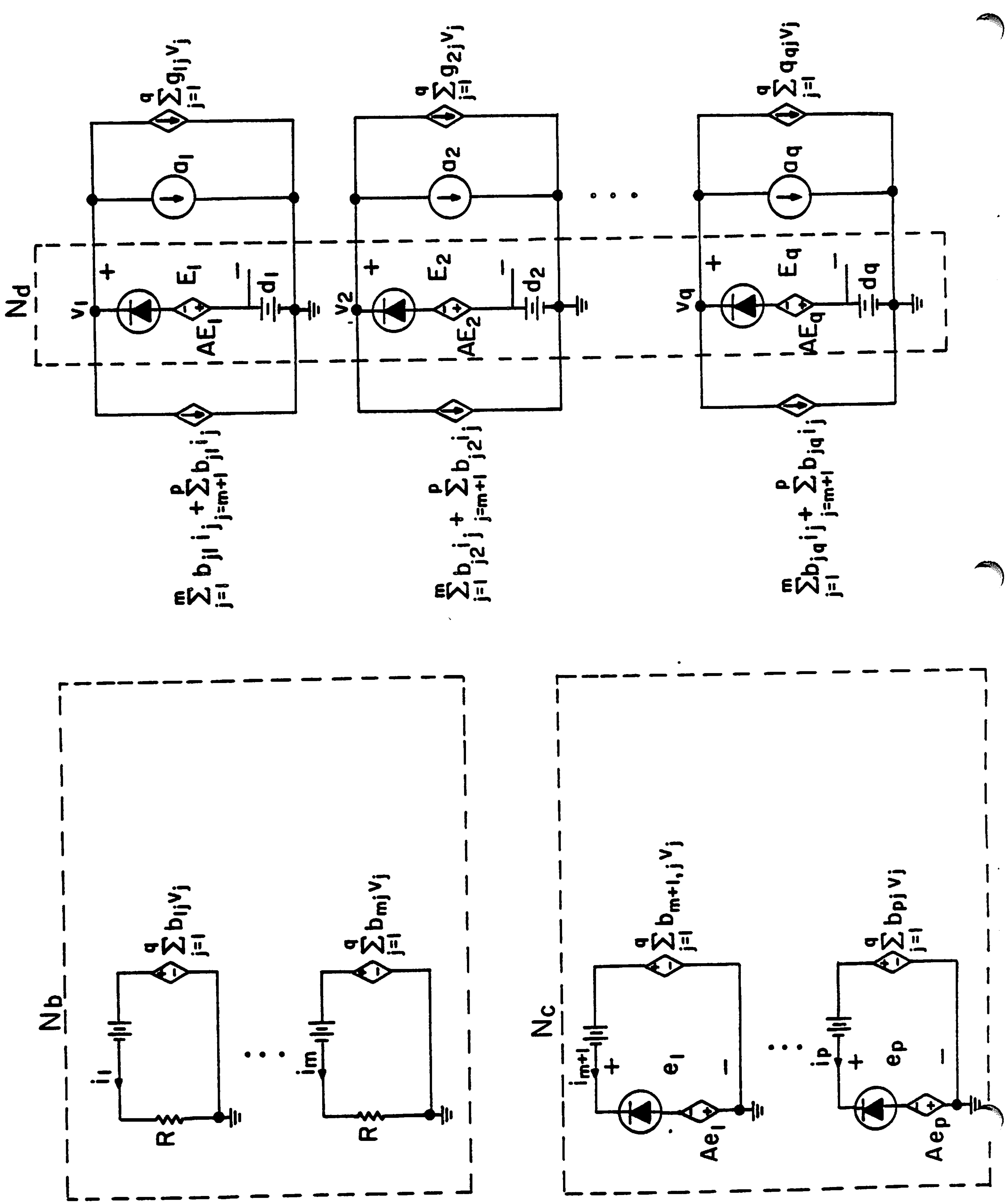
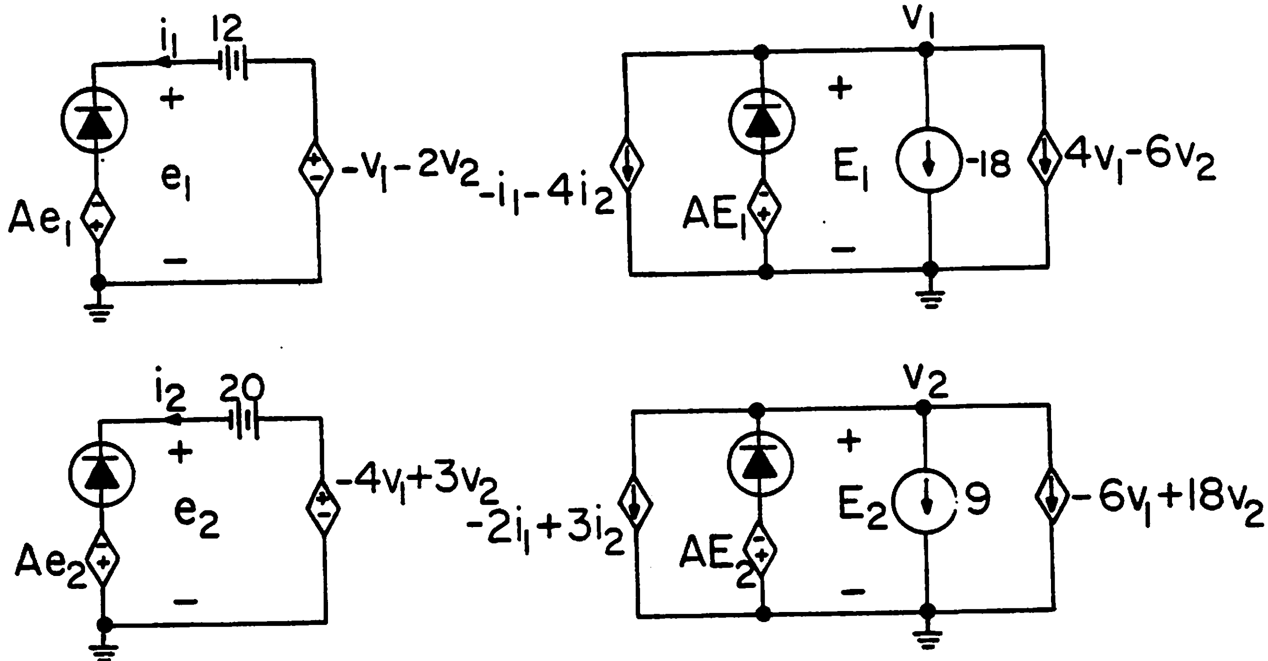
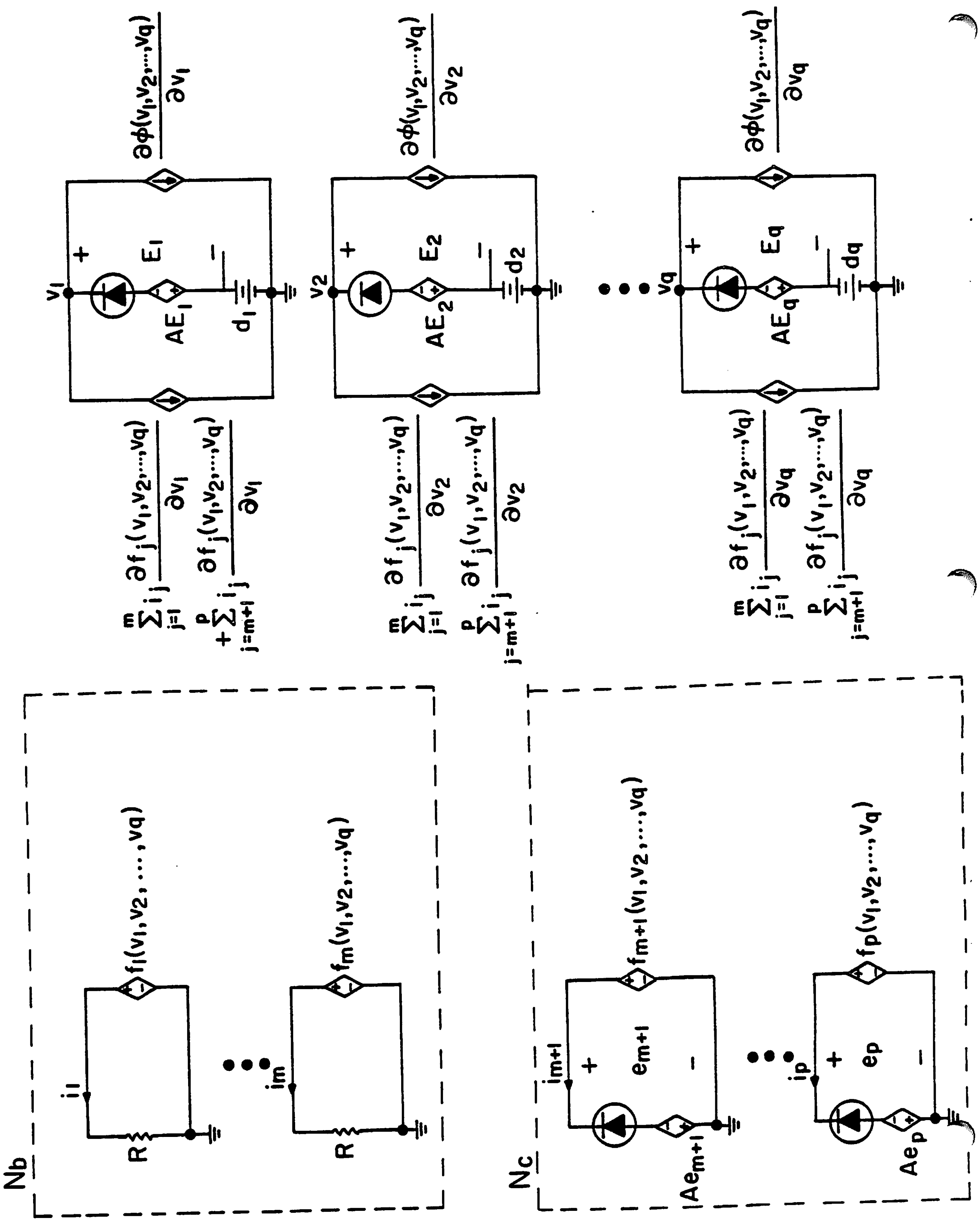


Fig. 4



i_1
12
+
e_1
Ae_1
−
$-v_1-2v_2$
$-i_1-4i_2$
v_1
+
AE_1
E_1
−
-18
$4v_1-6v_2$
i_2
20
+
e_2
Ae_2
−
$-4v_1+3v_2$
$-2i_1+3i_2$
v_2
+
AE_2
E_2
−
9
$-6v_1+18v_2$

Fig. 5



Nb
R
i1
im
f1(v1, v2, ..., vq)
fm(v1, v2, ..., vq)
Nc
im+1
em+1
Aem+1
fm+1(v1, v2, ..., vq)
ip
ep
Aep
fp(v1, v2, ..., vq)
v1
E1
AE1
d1
v2
E2
AE2
d2
vq
Eq
AEq
dq
∂φ(v1, v2, ..., vq)/∂v1
∂φ(v1, v2, ..., vq)/∂v2
∂φ(v1, v2, ..., vq)/∂vq

Fig. 6

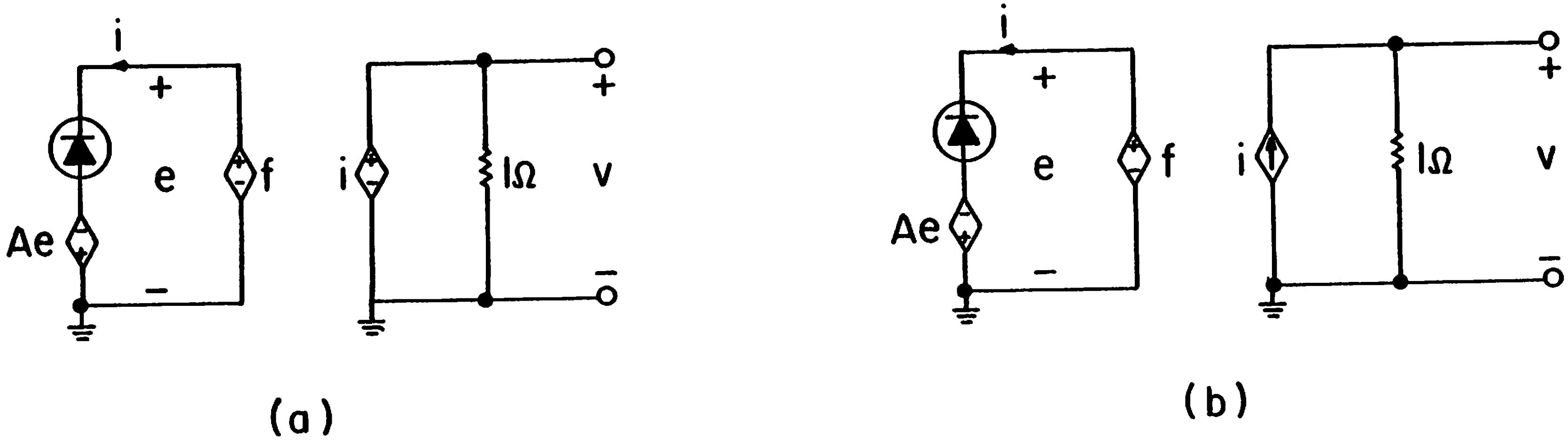
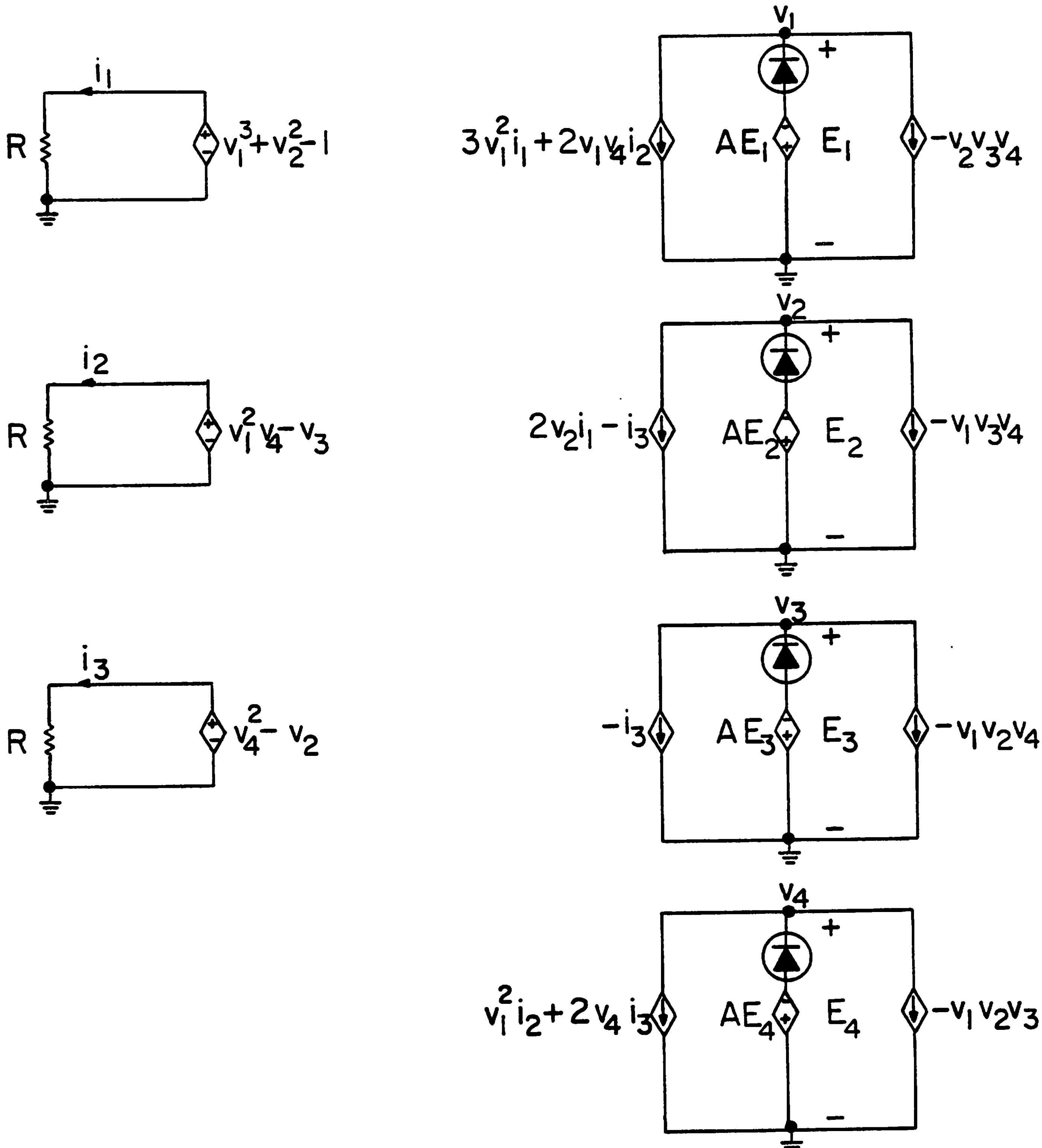
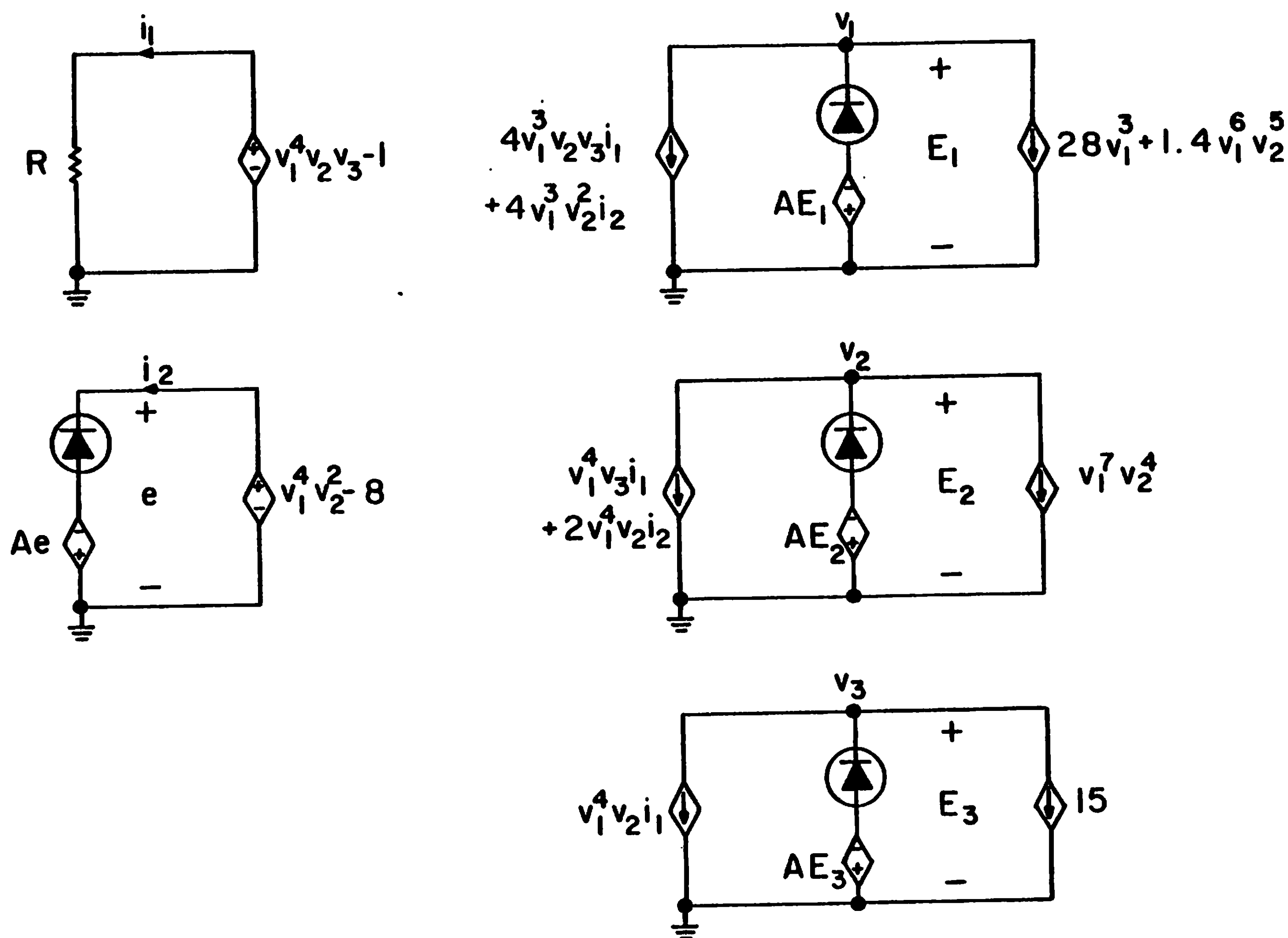


Fig. 7



i_1
R
$v_1^3+v_2^2-1$
i_2
R
$v_1^2v_4-v_3$
i_3
R
$v_4^2-v_2$
v_1
+
$3v_1^2i_1+2v_1v_4i_2$
AE_1
E_1
$-v_2v_3v_4$
−
v_2
+
$2v_2i_1-i_3$
AE_2
E_2
$-v_1v_3v_4$
−
v_3
+
$-i_3$
AE_3
E_3
$-v_1v_2v_4$
−
v_4
+
$v_1^2i_2+2v_4i_3$
AE_4
E_4
$-v_1v_2v_3$
−

Fig. 8



i1
R
v1^4 v2 v3 - 1
i2
+
e
Ae
v1^4 v2^2 - 8
v1
+
4v1^3 v2 v3 i1
+4 v1^3 v2^2 i2
E1
AE1
28v1^3 + 1.4 v1^6 v2^5
v2
v1^4 v3 i1
+2v1^4 v2 i2
E2
AE2
v1^7 v2^4
v3
v1^4 v2 i1
E3
AE3
15

Fig. 9

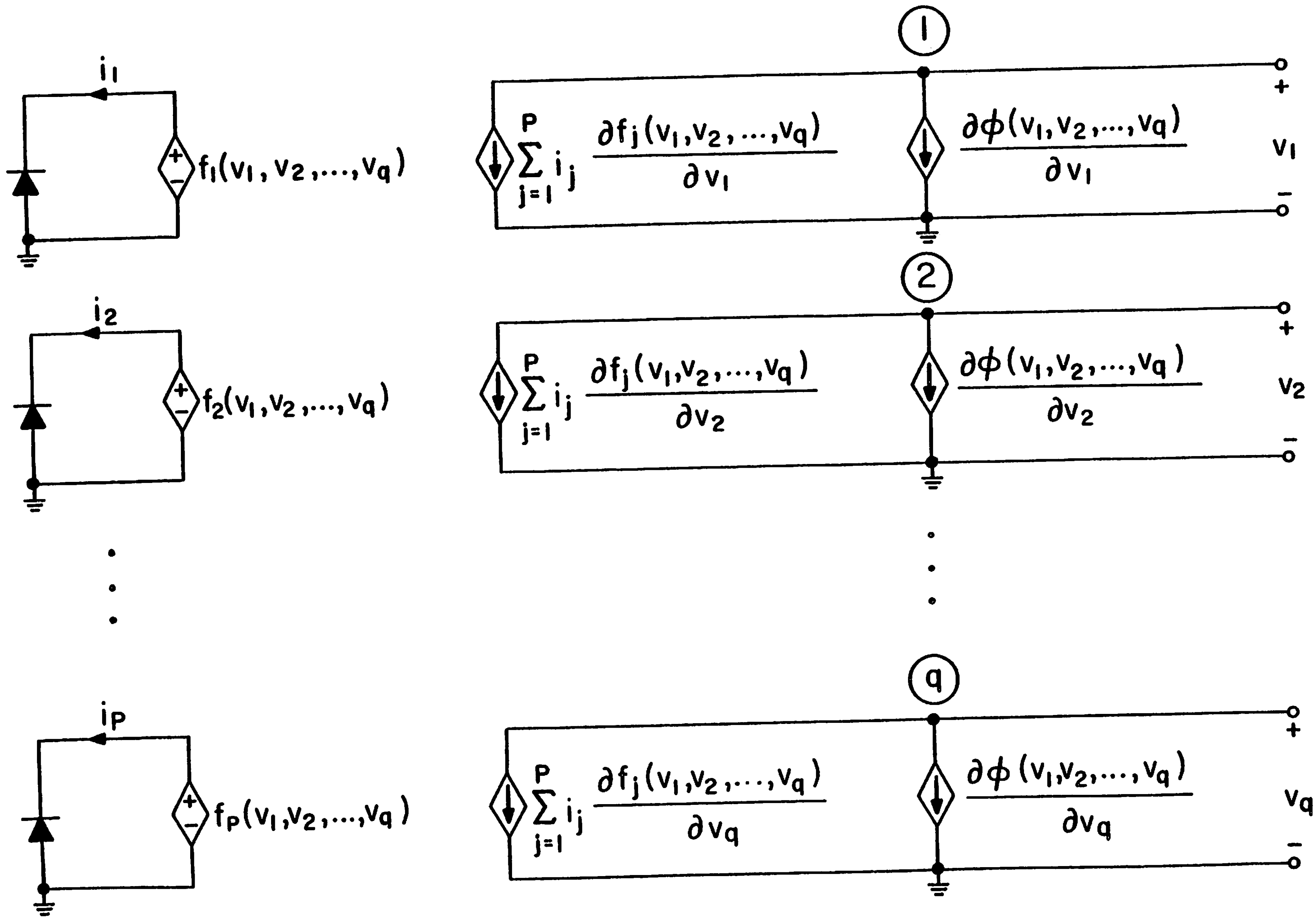

i1
f1(v1, v2,...,vq)
i2
f2(v1,v2,...,vq)
iP
fP(v1,v2,...,vq)
1
$\sum_{j=1}^{P} i_j \frac{\partial f_j(v_1,v_2,\ldots,v_q)}{\partial v_1}$
$\frac{\partial \phi(v_1,v_2,\ldots,v_q)}{\partial v_1}$
+
v1
−
2
$\sum_{j=1}^{P} i_j \frac{\partial f_j(v_1,v_2,\ldots,v_q)}{\partial v_2}$
$\frac{\partial \phi(v_1,v_2,\ldots,v_q)}{\partial v_2}$
+
v2
−
q
$\sum_{j=1}^{P} i_j \frac{\partial f_j(v_1,v_2,\ldots,v_q)}{\partial v_q}$
$\frac{\partial \phi(v_1,v_2,\ldots,v_q)}{\partial v_q}$
+
vq
−

Fig. 10

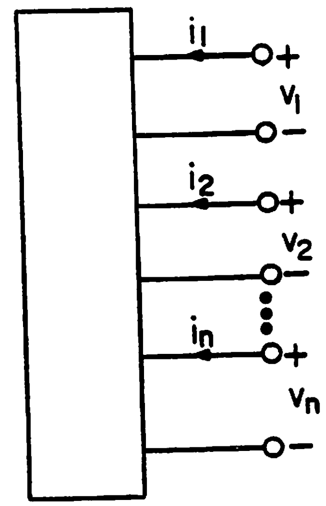


Fig. 11

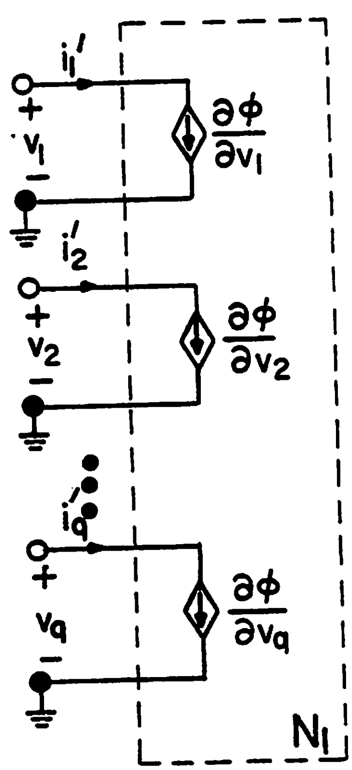


Fig. 12

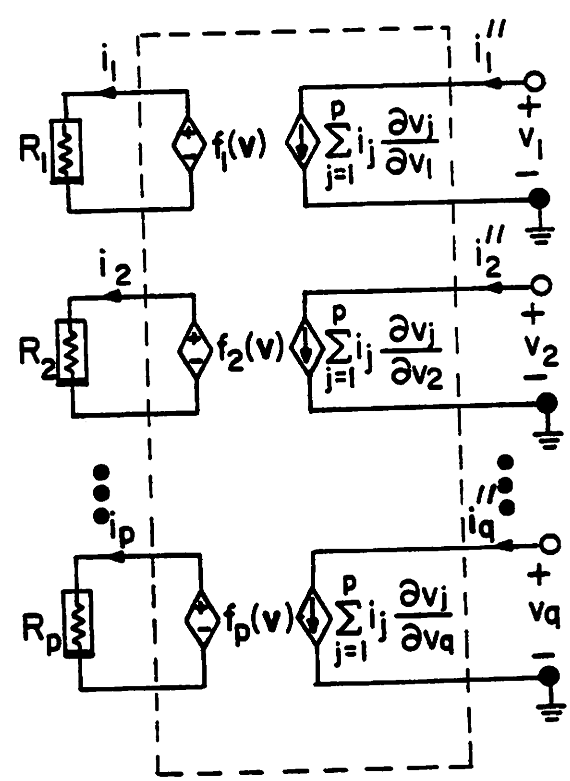


Fig. 13

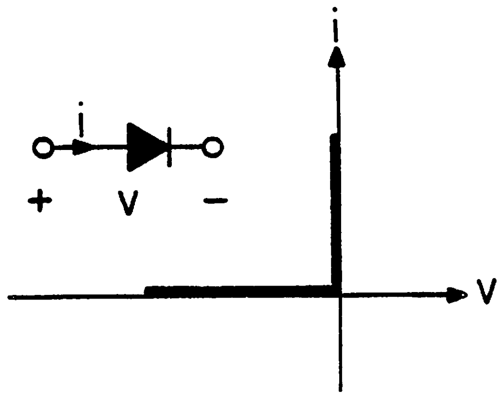


Fig. 14

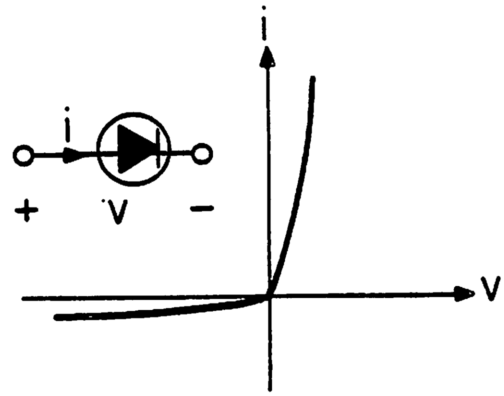


Fig. 15

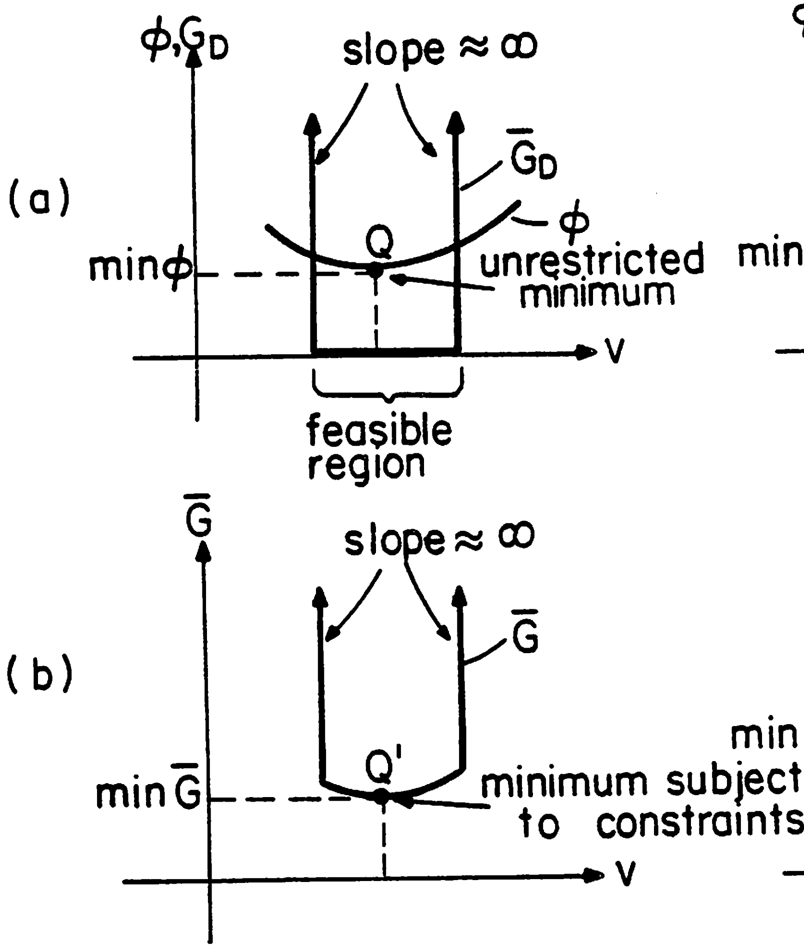


Fig. 16

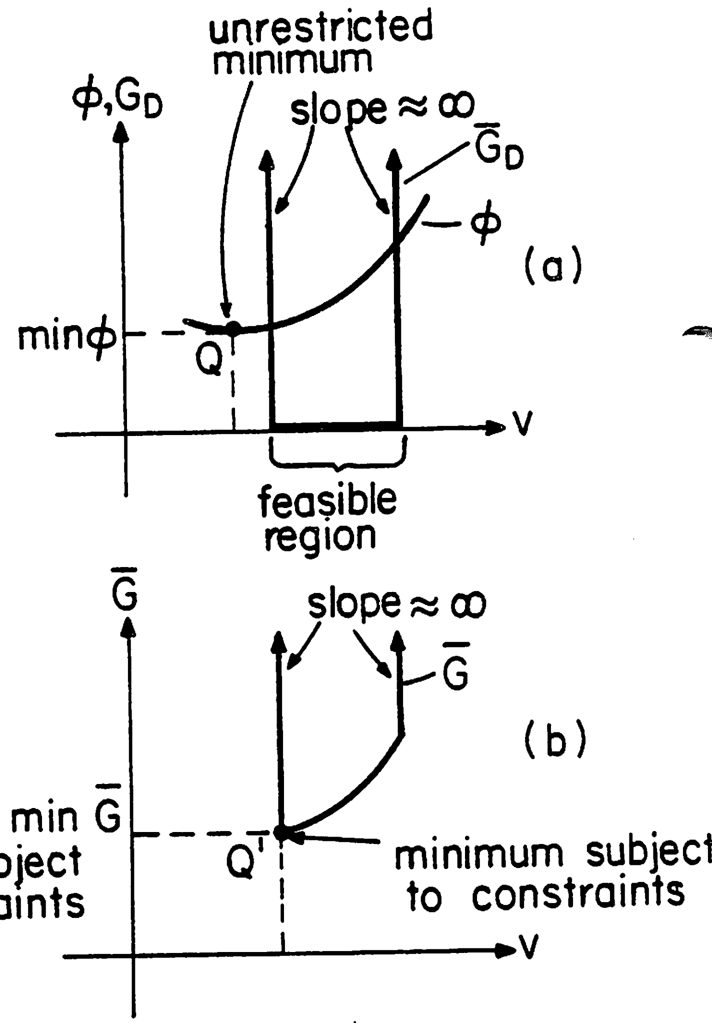


Fig. 17

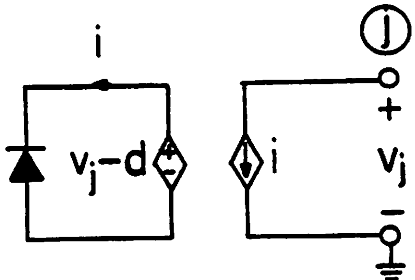


Fig. 18

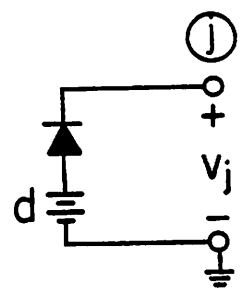


Fig. 19

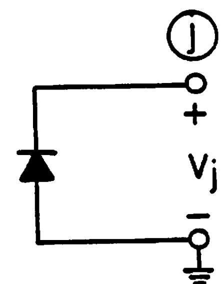


Fig. 20

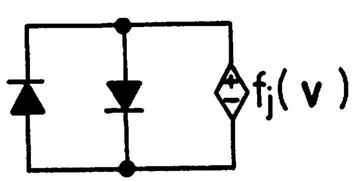


Fig. 21

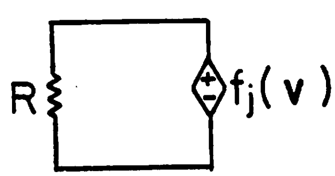


Fig. 22