AD-A051 069 SOUTHERN METHODIST UNIV DALLAS TX DEPT OF OPERATIONS--ETC F/G 12/1
THE CONVEX COST OF NETWORK FLOW PROBLEM: A STATE-OF-THE ART SUR--ETC(U)
JAN 78 A I ALI, R V HELGASON, J L KENNINGTON AFOSR-77-3151

UNCLASSIFIED QREM-78001 AFOSR-TR-78-0332 NL

| OF |
ADA
051069

END
DATE
FILMED
4 -78
DDC

(2)

Technical Report OREM 78001

THE CONVEX COST NETWORK FLOW PROBLEM:

A STATE-OF-THE-ART SURVEY

A. I. Ali

R. V. Helgason

J. L. Kennington

Department of Operations Research
and
Engineering Management

January 1978

Comments and criticisms from interested readers are cordially invited.

See 1473 in back

ABSTRACT

This exposition presents a state-of-the-art survey of models and algorithms for the convex cost network flow problem.

ACKNOWLEDGEMENT

This research was supported in part by the Air Force Office of Scientific Research under Grant Number AFOSR 77-3151. This paper has benefited by a reading by Larry J. LeBlanc.

I. INTRODUCTION

The *convex cost network flow problem* may be easily described in terms of a distribution problem over a directed graph [V, E], where V is the node set and E is the arc set. The capacity of arc j is given by u_j, and the vector of all capacities is denoted by u. The decision variable x_j denotes the flow in arc j and the vector of all flows is denoted by x. The convex cost function associated with the flow in [V, E] is given by g(x). Mathematically, the convex cost network flow problem may be stated as follows:

$$\min \quad g(x) \tag{1}$$

$$\text{s.t.} \quad Ax = r \tag{2}$$

$$0 \leq x \leq u, \tag{3}$$

where A is a node-arc incidence matrix for [V, E], and r is the vector of requirements. If $r_i > 0$, then node i is a supply node with supply equal r_i. If $r_i < 0$, then node i is a demand node with demand equal $-r_i$. Nodes having $r_i = 0$ are transhipment nodes. We assume that $r_0 = \Sigma_i r_i$ is zero, in which case total supply equals total demand. If $r_0 < 0$, no feasible solution exists. If $r_0 > 0$, we may place the problem in the prescribed form by adding a dummy demand node having demand r_0 and extra arcs from each supply node to the dummy node. In this case g(x) remains a function of the original arc flows only.

The convex cost network flow problem is simply a specially structured nonlinear program and may be solved with any of a host of techniques. However, due to the underlying network structure, specializations of these approaches have been developed. This exposition presents a summary of the best known applications of convex cost network flow problems as well as a unification of the algorithmic approaches available for these problems.

Our objectives have been to (i) help researchers place their work in the proper context with the existing literature and (ii) to help practitioners in algorithm selection by presenting the algorithms available in a uniform notation. Furthermore, implementation has been the underlying theme which has guided our presentation of the algorithms.

II. APPLICATIONS

There are two basic types of problems which are modelled as convex cost network flow problems, *equilibrium problems* and *production-distribution problems*. Equilibrium models appear in studies involving urban transportation systems, pipe network systems, and electrical network systems. A summary of the various models which have appeared in the literature follows. Convex cost network flow problems may also arise as subproblems when using penalty or barrier techniques on nonlinear programs having network constraints as a proper subset of the constraint set (e.g. see 3, 11, 33, 41, 49).

2.1 Equilibrium Flow Models

The equilibrium problem most widely studied by operations researchers involves the distribution of traffic in urban transportation networks. For this type network the nodes represent zones or intersections of streets in a metropolitan area while the arcs represent streets, expressways, tollways, and so forth. The supplies and demands are given by the number of vehicles which travel between zones during some period of time (for example the morning rush hour). This model differs from other models presented in this exposition in that it has a multicommodity nature. The commodities may be viewed as either the flows between individual origin-destination pairs or the flows from each origin to all its destinations. To minimize the number of variables, the latter approach is adopted here. Let the sources (commodities) be indexed by $1,\ldots,p$ and let x_j^k denote the flow of commodity k on arc j. Let the function $f_j(z)$ denote the travel time for each vehicle on arc j when the total number of vehicles using arc j is z.

There are two basic models which have appeared in the transportation literature corresponding to Wardrop's [44] two principles of traffic flow. The first principle is stated as follows:

> *The journey times for all vehicles with the same orgin and destination are equal and no greater than the journey time which would be experienced by a single vehicle on any unused route.*

This is called the *principle of equal travel times* and is reflected in the following model:

$$\min \quad \Sigma_j \int_0^{\Sigma_k x_j^k} f_j(t)dt \qquad (4)$$

$$\text{s.t.} \quad Ax^k = r^k , \qquad \text{(all k)} \qquad (5)$$

$$x^k \geq 0 , \qquad \text{(all k)} \qquad (6)$$

where x^k is the vector of flows for commodity k and r^k is the vector of requirements for commodity k. It is assumed that $f_j(t)$ for each j is continuous and monotone increasing over $t \geq 0$. The continuity of $f_j(t)$ guarantees the existence of the integral while the assumption of a monotone increasing function guarantees convexity (see Roberts and Varberg [37]). The model (4) - (6) is generally called the *equilibrium traffic assignment problem* and its solution is often referred to as a *user optimized flow pattern.* Charnes and Cooper [8] in 1961 were among the first to present (4) - (6) as a model of Wardrop's first principle. Since then a series of studies directed toward developing efficient specialized algorithms for the equilibrium traffic assignment problem have appeared in the literature [17, 18, 24, 26, 30, 35, 36]. In addition Dafermos [13, 14] have extended this model for two-way streets and multiple classes of users.

Wardrop's second principle called the *principle of overall minimization* may be stated as follows:

> *Flows are distributed over the arcs of the network in such a manner that the sum of travel times for all users is minimized.*

The corresponding model is simply

$$\min \quad \Sigma_j f_j(\Sigma_k x_j^k) \cdot (\Sigma_k x_j^k)$$

$$\text{s.t.} \quad Ax^k = r^k , \qquad \text{(all k)}$$

$$x^k \geq 0 , \qquad \text{(all k).}$$

The solution to this model is often referred to as a *system optimized flow pattern.* Jorgensen [26] was among the first to investigate this model. Other equilibrium models related to traffic assignment may be found in [1, 18, 43].

The problem of finding steady-state flows and pressures in a pipe network has been of interest to civil engineers and city planners for more than forty years. In 1936 Hardy Cross [12] developed the first algorithm for solving such problems. His procedure is simply an iterative method for solving a system of nonlinear equations which describe the equilibrium behavior of pipe networks. Recently Hall [20] and Collins et. al. [9] discovered that this problem could be modelled as a convex cost network flow problem. For this type network the nodes represent reservoirs and pipe intersections while the arcs represent pumps, pipes, and valves of various types. The model is simply

$$\min \ \Sigma_j \int_0^{x_j} f_j(t)\, dt \tag{7}$$

$$\text{s.t.} \quad Ax = r \tag{8}$$

$$x \geq 0, \tag{9}$$

where for pipes, pumps, and valves, $f_j(t)$ denotes the head loss (pressure drop) in arc j as a function of flow in arc j and for artificial pipes connecting the ground to a reservoir, $f_j(t)$ is a constant. The decision variable x_j is the flow in arc j in units of volume per unit time. As before $f_j(t)$ is assumed to be continuous and monotone increasing.

The problem of finding the steady-state currents and voltages for a nonlinear resistive electrical network may also be modelled as a convex cost network flow problem [7, 10, 16, 23]. For this type network, nodes represent points of connection while arcs represent the various electrical components. The model is identical to (7) - (9) where for all components except voltage sources, $f_j(t)$ denotes the voltage drop across the element represented by arc j as a function of current flow in arc j. For voltage sources which are connected to the ground by arc j, $f_j(t)$ is a constant. The decision

variable x_j denotes the current in arc j. Each $f_j(t)$ is assumed to be both continuous and monotone increasing.

We note that both the pipe network model and the electrical network model each have special structure which can be exploited in developing solution techniques. Even though the particular cost functions used differ, both are separable and convex. Hence, we conjecture that an algorithm which proves to be effective for one of these models should also work well on the other model.

2.2 Production-Distribution Models

Distribution problems in which demands are given by random variables rather than by known constants may be modelled as convex cost network flow problems. Let $D \subset V$ denote the destinations whose demands are random variables. Then the requirements for $k \in D$ are random variables. Let $f_k(\cdot)$ for all $k \in D$ denote the density functions corresponding to these random variables. Further, let h_k denote the unit holding cost and s_k denote the unit shortage cost at destination k. Then the objective is to choose amounts to be shipped in order to minimize the shipping cost plus the expected holding and shortage costs. Mathematically this problem may be stated as follows:

$$\min \Sigma_{j\in E}\, c_j x_j + \Sigma_{k\in D}\, h_k \int_0^{|r_k|} (|r_k| - t) f_k(t)\,dt +$$

$$\Sigma_{k\in D}\, s_k \int_{|r_k|}^{\infty} (t - |r_k|)\, f_k(t)\,dt$$

$$\text{s.t.} \quad Ax = r$$

$$0 \leq x \leq u.$$

Note that for the demand nodes, the requirements are variables rather than constants. That is, $|r_k|$ for all $k \in D$ represents the total amount shipped into node k to meet its unknown demand. Problems of this class were first suggested by Dantzig [15]. Special procedures for models of this type may be found in [11, 32].

Distribution problems for which the demands are known constants but in which some of the supplies at the production facilities are unknown may also be modelled as convex cost network flow problems. The basic underlying idea is that production facilities can increase their capacities by any of a series increasingly expensive additions of equipment and services. For this type

model some of the requirements, r_k, are decision variables. Let $f_k(r_k)$ denote the total production cost at facility k for a production level of of r_k. Then we obtain the following model:

$$\begin{aligned} \min \quad & \Sigma_j c_j x_j + \Sigma_k f_k(r_k) \\ \text{s.t.} \quad & Ax = r \\ & 0 \leq x \leq u. \end{aligned}$$

Algorithms for models of this type have been developed by Sharp et. al. [40], and LeBlanc and Cooper [31].

A convex cost network flow model has recently been proposed by Rosenthal [39] to study multireservoir water release scheduling for the Tennessee Valley Authority. The objective is to maximize the benefit of hydroelectric power. This benefit function reflects the savings of thermal fuels that result from using hydroelectric power. For this model the nodes represent reservoirs at a given time period to a different reservoir at a later time period or (ii) a reservoir connected to itself at a later time period. One type of arc allows for the release of water while the other allows water to be stored.

III. ALGORITHMS

There are basically two types of algorithms which have been proposed for solving the convex cost network flow problem, *approximation methods* and *feasible direction methods*. Approximation methods use a piece-wise linear approximation for g(x) and solve the resulting linear network flow problem. Feasible direction methods, however, work directly with the nonlinear cost function g(x). Let y denote a feasible point for (1) - (3). We define a *direction* to be any nonzero vector. A direction d is said to be a *feasible direction* at y if $y + \lambda d$ is feasible for some $\lambda > 0$. A direction d is said to be an *improving feasible direction* if d is a feasible direction and the directional derivative of g in the direction d at the point y is negative. Given the above definitions, the feasible direction algorithm may be described as follows:

ALG-1: FEASIBLE DIRECTION ALGORITHM

0. *Initialization*

   Let y_0 be any feasible point, let $\varepsilon > 0$ be a termination tolerance, and set $k \leftarrow 0$.

1. *Find An Improving Feasible Direction*

   Let d be an improving feasible direction at y_k.

2. *Line Search*

   Find α^* such that

   $$g(y_k + \alpha^* d) = \min_{\alpha \geq 0} \{g(y_k + \alpha d): \; A(y_k + \alpha d) = r, \; 0 \leq y_k + \alpha d \leq u\}.$$

   Set $y_{k+1} \leftarrow y_k + \alpha^* d$ and $k \leftarrow k+1$. If $|g(y_{k+1}) - g(y_k)| < \varepsilon$ terminate; otherwise, go to 1.

The major difficulty with the above algorithm is the determination of an appropriate direction finding program for step 1. The various methods differ in the means for finding d. However, all the methods presented in this exposition make use of the directional derivative. Recall that the directional derivative of g(x) at y in the direction d with Euclidean norm $|d|$ is given by

$$D_d g(y) = \lim_{t \to 0} \frac{g\left[y + t\,\frac{d}{|d|}\right] - g(y)}{t}$$

providing the limit exists [28]. A more computationally useful form of the directional derivative is given by

$$D_d g(y) = \frac{\nabla g(y) d}{|d|}$$

where $\nabla g(y)$ is the gradient of g evaluated at y.

Since $D_d g(y) < 0$ whenever $\nabla g(y) d < 0$, $\nabla g(y) d$ is the expression used in all feasible direction methods for the selection of an improving direction.

The particular type of line search used in step 2 usually depends on whether or not g is differentiable. If g is differentiable and the derivatives are easily calculated, then a *bisection search* [3, 33, 49] is usually used; otherwise, either a *golden section* or *Fibonacci search* [3, 33, 41, 49] is used.

3.1 Piece-Wise Linear Approximation

In order to simplify the presentation we assume that $g(x)$ is separable, i.e. $g(x) = \Sigma_j g_j(x_j)$. Suppose K linear segments are to be used in the approximation. Then the interval $[0, u_j]$ is partitioned into K segments each of length u_j^k for $k = 1, \ldots, K$, such that $u_j = \Sigma_k u_j^k$. Let v_j^p denote the right end point for the pth segment. That is, $v_j^p = \Sigma_{k=1}^{k=p} u_j^k$. Let x_j^k denote the flow in segment k of arc j with unit cost of c_j^k. Then $x_j = \Sigma_k x_j^k$. Substituting $x_j = \Sigma_k x_j^k$ into (1) - (3) yields

$$\min \quad \Sigma_{j,k} \; c_j^k \, x_j^k$$

$$\text{st} \quad \Sigma_k \, Ax^k = r$$

$$0 \leq x^k \leq u^k \;, \text{ (all k)}$$

where x^k is a vector of all flows for the k^{th} segment and u^k is a vector of bounds for the k^{th} segment. There are numerous techniques [7, 9, 10] which may be applied to determine the unit costs, c_j^k. An extension of the idea of least squares has been used in [9, 10]. To apply this approach we define

$$h_j^k(x_j) = p_j^k + c_j^k \cdot (x_j - v_j^{k-1})$$

for the interval $v_j^{k-1} \leq x_j \leq v_j^k$, where $v_j^0 = 0$. Then c_j^k is selected as the value which minimizes

$$\int_{v_j^{k-1}}^{v_j^k} [g_j(t) - h_j^k(t)]^2 \, dt.$$

For the first segment, $p_j^1 = g_j(0)$ and for all other segments $p_j^k = h_j^{k-1}(v_j^{k-1})$. The formula which yields this minimization may be found in [10].

3.2 Frank-Wolfe Method

In 1956 Frank and Wolfe [19] proposed a method for solving nonlinear programs having a convex differentiable cost function and linear constraints. Given a feasible point at iteration k for (1) - (3), say y_k, one finds another feasible point, say z_k, by solving the linear network flow problem $\min \{\nabla g(y_k)x : Ax = r, 0 \leq x \leq u\}$. As a by-prodoct of the solution of the linear network problem, one gets a lower bound. Consider the following proposition:

Proposition 1.

Let $\hat{x}$ denote an optimum for (1) - (3) and let y be any feasible point for (1) - (3). Let z solve the linear program $\min \{\nabla g(y)x : Ax = r, 0 \leq x \leq u\}$. Then $g(\hat{x}) \geq g(y) + \nabla g(y)\cdot(z - y)$.

Proof. By Theorem 4 page 72 of [29] we have $g(\hat{x}) \geq g(y) + \nabla g(y)\cdot(\hat{x} - y)$. Then $g(\hat{x}) \geq g(y) + \nabla g(y)\hat{x} - \nabla g(y)y$. But $\nabla g(y)\hat{x} \geq \nabla g(y)z$. Therefore, $g(\hat{x}) \geq g(y) + \nabla g(y)\, z - \nabla g(y)\, y = g(y) + \nabla g(y)\cdot(z - y)$.

Using the lower bound provided by the above proposition, the Frank-Wolfe algorithm may be stated as follows:

ALG-2: FRANK-WOLFE ALGORITHM

0. *Initialization*

Let y_0 be any feasible solution. Set $k \leftarrow 0$, initialize the lower bound $\beta \leftarrow -\infty$, and choose the termination parameter $\varepsilon > 0$.

1. *Solve Network Subproblem, Update Bound, Check For Termination*

Let z denote the solution to $\min \{\nabla g(y_k)x : Ax = r, 0 \leq x \leq u\}$. Set $\beta \leftarrow \max \{\beta, g(y_k) + \nabla g(y_k)\cdot(z - y_k)\}$. If $g(y_k) - \beta \leq \varepsilon$, terminate with y_k as an ε-optimum; otherwise, $k \leftarrow k + 1$.

2. *Line Search*

Let y_k be the point on the line segment between y_{k-1} and z having smallest objective function value and go to 1.

It is well-known that the convergence rate of this procedure slows substantially as the optimum is approached [9, 17, 18, 32, 35]. Several modifications of the basic algorithm have been proposed to enhance the computational effectiveness of this procedure. These are designed to avoid the zigzag character frequently exhibited by this procedure. Figure 1 illustrates this phenomenon.

Figure 1 About Here

The method of parallel tangents (PARTAN) [33] integrates movements in the direction $(y_k - y_{k-2})$ along with the basic Frank-Wolfe steps. The algorithm is given below.

ALG-3: FRANK-WOLFE ALGORITHM WITH PARTAN

0. *Initialization*

(same as ALG-2)

1. *Solve Network Subproblem, Update Bound, Check For Termination*

(same as ALG-2)

2. *Line Search*

Let w be the point on the line segment between y_{k-1} and z having smallest objective function value. If k = 1, $y_k \leftarrow w$, and go to 1.

3. *PARTAN Step*

Let y_k be the feasible point having smallest objective function value on the half ray from y_{k-2} through w, and go to 1.

Figure 2 illustrates the Frank-Wolfe method using PARTAN.

Figure 2 About Here

In 1970 Wolfe [47] presented a modification of the Frank-Wolfe algorithm which incorporated what is called *away steps*. The modified algorithm is described below.

ALG-3: *FRANK-WOLFE ALGORITHM WITH AWAY STEPS*

0. *Initialization*

(same as ALG-2)

1. *Find Toward Step Direction, Update Bound, Check For Termination*

(same as ALG-2 with $k \leftarrow k + 1$ deleted)

2. *Away Step Direction*

Let w denote the solution to

$$\max \{\nabla g(y_k)x : Ax = r,\ 0 \leq x \leq u,\ x_j = 0 \text{ if } (y_k)_j = 0\}$$

Set $d^2 \leftarrow y_k - w$.

3. *Find Max Movement In Away Direction*

Let α^2 denote the solution to

$$\max_{\alpha \geq 0} \{\alpha : A(w + \alpha d^2) = r,\ 0 \leq w + \alpha d^2 \leq u\}.$$

If $\alpha^2 \leq 1$, go to 5.

4. *Select Direction*

If $|\nabla g(y_k)z| > |\nabla g(y_k)w|$, go to 5; otherwise, go to 6.

5. *Line Search With Toward Direction*

Let y_{k+1} be the point on the line segment between y_k and z having smallest objective value. Set $k \leftarrow k + 1$ and go to 1.

6. *Line Search With Away Direction*

Let α^* denote the solution to

$$\min \{g(y_k + \alpha d^2)\}. \quad \text{Set } k \leftarrow k + 1,\ y_k \leftarrow y_{k-1} + \alpha^* d^2,$$
$$1 \leq \alpha \leq \alpha^2$$

and go to 1.

The Frank-Wolfe method using away steps is illustrated in Figure 3.

Figure 3 About Here

Holloway [22] proposed a different extension of the Frank-Wolfe method. This extension requires that y_0 and each z produced in step 1 of ALG-2 be saved. Suppose that after k passes through step 1 that these solutions are denoted $z_0,\ldots,z_{k-1}$. Then Holloway suggests that the line search of step 2 ALG-2 be replaced by the following problem,

$$\left.\begin{array}{l} \min \quad g(y_0\lambda + \Sigma_{i=0}^{i=k-1} z_i\alpha_i) \\ \text{s.t. } \lambda + \Sigma_i \alpha_i = 1 \\ \qquad \lambda, \alpha_i \geq 0, \quad (\text{all } i). \end{array}\right\} \qquad (10)$$

Letting λ^*, α_i^* for i = 0,...,k-1, denote the solution to (10), then $y \leftarrow \lambda^* y_0 + \Sigma_i z_i \alpha_i^*$.

The computational experimentation as represented by [17, 22, 32] using both the away steps and Holloway's extension on different classes of problems is inconclusive. These extensions have been of benefit on some classes of problems but have slowed computational times for others.

3.3 Zoutendijk's Method of Feasible Directions

In this section we present a specialization of the work of Zoutendijk [50] to the convex cost network flow problem, (1) - (3). We assume that $g(x)$ is differentiable over $\{x : 0 \leq x \leq u\}$. Given a feasible point y, any feasible direction d must satisfy $A(y + \alpha d) = r$ for some $\alpha > 0$. Clearly, for (1) - (3) any feasible direction d must have $Ad = 0$. Furthermore,

$$d_j \geq 0, \text{ for all } j \text{ such that } y_j = 0 \text{ and};$$

$$d_j \leq 0, \text{ for all } j \text{ such that } y_j = u_j.$$

Then an appropriate direction finding program is the following:

$$\min \quad \nabla g(y)d \tag{11}$$

$$\text{s.t.} \quad Ad = 0 \tag{12}$$

$$d_j \geq 0, \text{ for all } j \text{ such that } y_j = 0 \; ; \tag{13}$$

$$d_j \leq 0, \text{ for all } j \text{ such that } y_j = u_j; \tag{14}$$

$$|d| = 1. \tag{15}$$

Due to the nonlinear constraint (15), (11) - (15) is difficult to solve. Hence a relaxation of (11) - (22) where (15) is replaced by

$$-\underline{1} \leq d \leq \underline{1},$$

is usually solved. The relaxed problem is simply a linear network flow problem. If an improving feasible direction for (11) - (15) exists, then an optimum solution to the relaxed problem will be an improving feasible direction although not necessarily the best local direction.

Beale [4] was the first to apply a feasible direction algorithm (similar to that described by Zoutendijk) to the convex cost network flow problem. However, this work was designed for a bipartite structure. Furthermore, the partial derivatives are approximated and the flows are always changed by a single unit. This approach does not require that $g(x)$ be differentiable, it avoids the need for a line search, and it is applicable to integer as well as continuous problems. Extensions of this algorithm to an arbitrary network have been given by Hu [23] and Klein [27]. Other similar approaches may be found in [34] and [45].

3.4 Rosen's Gradient Projection Method

The gradient projection method [38] is motivated by a desire to implement the feasible direction philosophy while not requiring the solution of a linear program at each iteration. The strategy employed is to move, if possible, in an improving feasible direction, derived from the negative gradient, so that all active (currently binding) constraints remain active. The negative gradient is projected onto a subspace of the feasible region which insures that all active constraints remain active. If the projection is a non-zero vector, it will lie along an improving feasible direction. If the projection is the zero vector, then either optimality can be verified or one attempts to find an improving feasible direction for which one or more binding constraints become non-binding. The procedure guarantees that either an improving feasible direction can be found or optimality can be verified. For the convex cost network flow problem, the equality constraints (2) are always active; whereas, the inequality constraints (3) may be either active or inactive at any given step.

Given a feasible point y, the subspace tangent to the active constraints is given by the set $M = \{d : Ad = 0,\ d_j = 0 \text{ if } y_j = 0 \text{ or } y_j = u_j\}$. It is well-known that one row of A is redundant [2], and that A has rank equal to the number of rows less one. Let $\hat{A}$ denote the A matrix with one row deleted. Let $\Gamma = \{j : y_j = 0 \text{ or } y_j = u_j\}$ and let k denote the order of Γ. Let

$$\bar{A} = \begin{bmatrix} \hat{A} \\ e'_{j_1} \\ e'_{j_2} \\ \vdots \\ e'_{j_k} \end{bmatrix},$$

where $j_i \in \Gamma$ for $i = 1,\ldots,k$, and e_{j_i} is a column vector with a one in position j_i and zeroes elsewhere. We assume that $\bar{A}$ has full row rank and we let $M = \{d: \bar{A}d=0\}$. A subspace orthogonal to M is defined by $N = \{h = \bar{A}'\alpha : \alpha \in E^p\}$ where p is the row dimension of $\bar{A}$ (i.e. the number of nodes + k minus 1). The following proposition proves that M and N are orthogonal.

Proposition 2.

M and N are orthogonal.

Proof. Let $d \in M$ and let $h \in N$. Then h can be represented as $h = \bar{A}'\alpha$ for some $\alpha \in E^p$. Then $d'h = d'\bar{A}'\alpha = (\bar{A}d)'\alpha = 0$.

Then any vector in E^n, can be represented by $d + h$ where $d \in M$ and $h \in N$. In particular, the negative of the gradient can be represented this way. That is $-\nabla g(y) = d + \bar{A}'\alpha$ for some $\alpha \in E^p$. But $\bar{A}d = -\bar{A}\nabla g(y) - \bar{A}\bar{A}'\alpha = 0$ implies $\alpha = -(\bar{A}\bar{A}')^{-1}\bar{A}\nabla g(y)$. Since $\bar{A}$ has full row rank the existence of $(\bar{A}\bar{A}')^{-1}$ is guaranteed [5].

Therefore $$d = -\nabla g(y) + \bar{A}'(\bar{A}\bar{A}')^{-1}\bar{A}\nabla g(y)$$
$$= -\{I - \bar{A}'(\bar{A}\bar{A}')^{-1}\bar{A}\}\nabla g(y).$$

Letting $P = \{I - \bar{A}'(\bar{A}\bar{A}')^{-1}\bar{A}\}$ we have
$$d = -P\nabla g(y),$$

where P is called the projection matrix. That is, Px for any vector x is the projection of x on the subspace M. The following proposition proves that $d = -P\nabla g(y)$ is an improving feasible direction, when $d \neq 0$.

Proposition 3.

$\nabla g(y)d < 0$, where $d = -P\nabla g(y) \neq 0$.

Proof. $\nabla g(y)d = [\nabla g(y) + d - d]d = [\nabla g(y) + d]d - dd$. But $\nabla g(y) + d$ is orthogonal to d. Hence $\nabla g(y)d = -dd < 0$ for $d \neq 0$.

If $-P\nabla g(y) = 0$, then either optimality may be verified, or one seeks an improving feasible direction by allowing movement of the variables corresponding to active constraints from (3). Verification of optimality requires straight-forward Kuhn-Tucker analysis, the details of which may be found in

[33]. When $-P\nabla g(y) = 0$ and a variable for which movement is to be allowed is identified, the corresponding row of (3) is removed from $\bar{A}$. This gives rise to a new projection matrix P and a newly projected vector $-P\nabla g(y)$. This process continues until either an improving feasible direction is obtained or optimality is verified.

A major part of the computational burden of the gradient projection method involves recomputing the projection matrix when $\bar{A}$ changes. However, $\bar{A}\,\bar{A}'$ is a symmetric matrix and specialized inversion techniques and data structures may be used to maintain $(\bar{A}\,\bar{A}')^{-1}$. Furthermore, all changes in $\bar{A}\,\bar{A}'$ may be accomplished by adding or deleting a single row from $\bar{A}$. Hence changes in $(\bar{A}\,\bar{A}')^{-1}$ may be obtained by a simple updating scheme which may be applied several times each iteration. These updating procedures are specializations of the following proposition.

Proposition 4.

Consider the partitioned matrices

$$A = \left[\begin{array}{c|c} A_1 & A_2 \\ \hline A_3 & A_4 \end{array}\right] \quad \text{and} \quad B = \left[\begin{array}{c|c} B_1 & B_2 \\ \hline B_3 & B_4 \end{array}\right],$$

where $B = A^{-1}$, and A_1^{-1} and B_4^{-1} are assumed to exist. Then

$$B_1 = A_1^{-1} + A_1^{-1} A_2 Q^{-1} A_3 A_1^{-1},$$

$$B_2 = -A_1^{-1} A_2 Q^{-1},$$

$$B_3 = -Q^{-1} A_3 A_1^{-1},$$

$$B_4 = Q^{-1},$$

where $Q = (A_4 - A_3 A_1^{-1} A_2)$; and

$$A_1^{-1} = B_1 - B_2 B_4^{-1} B_3.$$

A proof of Proposition 4 may be found in [6].

Suppose that a row, say e_s', is to be added to $\bar{A}$. Let $\bar{A}_N = \left[\begin{array}{c} \bar{A} \\ \hline e_s' \end{array}\right]$.

Then

$$\bar{A}_N \bar{A}_N' = \left[\begin{array}{c|c} \bar{A}\bar{A}' & \bar{A}e_s \\ \hline e_s'\bar{A}' & 1 \end{array}\right] .$$

Then by Proposition 4,

$$(\bar{A}_N \bar{A}_N')^{-1} = \left[\begin{array}{c|c} (\bar{A}\bar{A}')^{-1} + q(\bar{A}\bar{A}')^{-1}\bar{A}e_s e_s'\bar{A}'(\bar{A}\bar{A}')^{-1} & -q(\bar{A}\bar{A}')^{-1}\bar{A}e_s \\ \hline -q\, e_s'\bar{A}'(\bar{A}\bar{A}')^{-1} & q \end{array}\right] , \tag{16}$$

where $$q = \frac{1}{1 - e_s'\bar{A}'(\bar{A}\bar{A}')^{-1}\bar{A}e_s} .$$

Suppose that the last row, say e_s, is to be removed from $\bar{A}$. That is

$$\bar{A} = \left[\begin{array}{c} \bar{A}_N \\ \hline e_s' \end{array}\right] .$$

Let $[a' \mid q]$ denote the last row of $(\bar{A}\bar{A}')^{-1}$. Then

$$(\bar{A}\bar{A}')^{-1} = \left[\begin{array}{c|c} A^* & a \\ \hline a' & q \end{array}\right] .$$

Then by Proposition 4,

$$(\bar{A}_N \bar{A}_N')^{-1} = A^* - \left[\begin{array}{c} 1 \\ \hline q \end{array}\right] aa' . \tag{17}$$

Of course, if the row to be deleted is not the last, $(\bar{A}\,\bar{A}')^{-1}$ is transformed by interchanging both the last row with the one to be deleted and the last column with the one to be deleted, before applying the updating formula.

The general algorithm proceeds as follows:

ALG-4: *GRADIENT PROJECTION ALGORITHM*

0. *Initialization*

Let y_0 be any feasible solution and set $k \leftarrow 0$. Develop $\bar{A}$ and $(\bar{A}\,\bar{A}')^{-1}$. This inverse may be developed by beginning with $(\hat{A}\,\hat{A}')^{-1}$ and applying (16) successively.

1. *Calculate Projection*

Set $d \leftarrow -(I - \bar{A}'\,(\bar{A}\,\bar{A}')^{-1}\,\bar{A})\nabla g(y_k)$.

2. *Line Search*

If $d = 0$, go to 4, otherwise, let α^* denote the optimum of

$$\min_{\alpha \geq 0} \{g(y_k + \alpha d) : 0 \leq y_k + \alpha d \leq u\}.$$

Set $y_{k+1} \leftarrow y_k + \alpha^* d$, $k \leftarrow k + 1$

3. *Check Binding Constraints*

For each constraint from (3) which is binding and is not accounted for in $\bar{A}$, append a row to $\bar{A}$ and update $(\bar{A}\,\bar{A}')^{-1}$ using (16). Go to 1.

4. *Check for Termination*

Set $\delta \leftarrow -(\bar{A}\,\bar{A}')^{-1}\,\bar{A}\,\nabla g(y)$. Let $J_0 \leftarrow \{j : \delta_j > 0$ and row j of $\bar{A}$ corresponds to a variable at zero$\}$. Let $J_u \leftarrow \{j : \delta_j < 0$ and row j of $\bar{A}$ corresponds to a variable at upper bound$\}$. If $J_0 \cup J_u = \phi$, terminate with y_k optimal; otherwise, select any $k \in J_0 \cup J_u$, delete row k from $\bar{A}$, update $(\bar{A}\,\bar{A}')^{-1}$ using (17), and go to 1.

The major disadvantage of the above procedure for convex cost network flow problems is that the network structure cannot be easily exploited.

However, $\hat{A}\,\hat{A}'$ can easily be derived from the network structure. The element in the ith diagonal position is the number of arcs incident with the node represented by row i of $\hat{A}$ and for $i \neq j$, the (i,j)th element is the negative of the number of arcs connecting the nodes represented by rows i and j. However we know of no way to develop $(\hat{A}\,\hat{A}')^{-1}$ using only graphical operations. Also note that $\bar{A}e_s$ will always consist of the sth column of $\hat{A}$ with additional zeroes appended at the bottom.

3.5 Simplex Based Methods

There are two simplex type methods (the convex simplex method of Zangwill [48] and the reduced gradient method of Wolfe [46]) which may be used to solve convex cost network flow problems. Both techniques may be viewed as generalizations of the linear simplex method. They adhere to the simplex strategy of partitioning the variables into basic and nonbasic sets, but they differ from the linear simplex method in that the nonbasic variables are allowed to assume values other than their upper and lower bounds. As with the linear simplex method, the convex simplex method allows a single nonbasic to change at each iteration while any number of nonbasics may change with the reduced gradient method.

Suppose we partition $Ax = r$ into $[B \mid N]x = r$ where B is a basis. Likewise, partition x into $[x^B \mid x^N]$ and u into $[u^B \mid u^N]$. Then (1) - (3) may be stated as follows:

$$\begin{aligned} \min \quad & g([x^B \mid x^N]) \\ \text{s.t.} \quad & x^B = B^{-1}r - B^{-1}Nx^N \\ & 0 \leq x^B \leq u^B \\ & 0 \leq x^N \leq u^N. \end{aligned}$$

After substituting for x^B we obtain

$$\begin{aligned} \min \quad & f(x^N) \\ \text{s.t.} \quad & 0 \leq B^{-1}r - B^{-1}Nx^N \leq u^B \\ & 0 \leq x^N \leq u^N, \end{aligned}$$

where $f(x^N) = g([B^{-1}r - B^{-1}Nx^N \mid x^N])$. By making an analogous partitioning of ∇g, i.e. $\nabla g(x) = [\nabla g^B(x) \mid \nabla g^N(x)]$, we obtain

$$\nabla f(x^N) = \nabla g^N(x) - \nabla g^B(x)B^{-1}N. \tag{18}$$

Hence, $\nabla f(x^N)d/|d|$ is the directional derivative in nonbasic space. For the convex simplex method, d is always a vector with one nonzero entry which is either 1 or -1. The components of (18) play the role of the reduced costs in the linear simplex method. Given the direction of change (i.e. the nonbasic variable to change) a one dimensional search is required to determine the magnitude of the change. An actual simplex pivot is performed only if the resulting change forces one of the basic variables to zero or upper bound. An illustration of the convex simplex method is given in Figure 4.

Figure 4 About Here

Recall that the graph associated with (1) - (3) is defined by the node set V and the arc set E. Following the notation of Johnson [25] a *simple path* in [V,E] is a sequence of alternating nodes and connecting edges such that no node (or arc) is repeated. The direction of the arcs is unimportant in defining a simple path and both i, (i,j), (j,k), k and i, (j,i), j, (j,k), k are simple paths connecting nodes i and k. A *cycle* is a simple path together with an arc joining the beginning node and ending node of the path. A *connected graph* is a collection of nodes and arcs having at least one simple path between every pair of nodes, and a tree is connected graph with no cycles.

Since A has rank N - 1, we add an artificial variable to some row (node), say row ℓ, to obtain the following system,

$$Ax + e_\ell a = r, \tag{19}$$

where e_ℓ is a vector with a 1 in the ℓ^{th} position and zeroes elsewhere. The variable a does not appear in the objective function (1) and the upper bound

for a is set to zero. Let B be any basis of (19). Clearly a appears in every such basis. Then the subgraph generated by B, called F_B, consists of all vertices of V and arcs corresponding to columns of (19) in B. The artificial variable may be thought of as an arc incident to a single node, i.e., node ℓ. It is well-known that F_B is a tree, if and only if B is a basis [2, 25]. Node ℓ is called the root of the tree.

Let x_j denote the flow in arc j. Using this terminology the convex simplex method is now presented.

ALG-5: CONVEX SIMPLEX ALGORITHM

0. *Initialization*

   Let y be any feasible point and partition A into $[B \mid N]$.

1. *Pricing*

   Set $c^N \leftarrow \nabla g^N(y) - \nabla g^B(y)B^{-1}N$.

   Any non-basic arc j having $c_j^N > 0$ and $y_j > 0$ or $c_j^N < 0$ and $y_j < u_j$ is a candidate for flow change. If no such arc exists, terminate; otherwise, let ϕ denote the entering arc.

2. *Ratio Test*

   Suppose the entering arc has its tail at u and its head at v. The ratio test requires that one determine the orientation of the arcs on the simple path from u to v in F_B. Let R^+ denote the set of arcs in this path from u to v that are traversed in normal direction (i.e., tail to head) and let R^- denote the set of arcs in this path traversed in reverse direction. Then the maximum change in x_ϕ is given by

$$\Delta \leftarrow \begin{cases} \min \left\{ \min_{j \in R^+} x_j, \ \min_{j \in R^-} (u_j - x_j), \ u_\phi - x_\phi \right\}, & \text{if } c_\phi < 0 \\ \min \left\{ \min_{j \in R^-} x_j, \ \min_{j \in R^+} (u_j - x_j), \ x_\phi \right\}, & \text{if } c_\phi > 0. \end{cases}$$

If $\nabla = 0$, go to 5.

3. *Line Search*

Define the direction vector as follows:

$$d_j \leftarrow \begin{cases} -1, \text{ if arc } j \text{ is in } R^+, \\ 1, \text{ if arc } j \text{ is in } R^-, \\ 1, \text{ for } j = \phi, \\ 0, \text{ otherwise.} \end{cases}$$

Let α^* denote the solution to the following

$$\min_{0 \le \alpha \le \Delta} \left\{ g(y + \alpha d) \right\}, \text{ if } c_\phi < 0$$

$$\min_{-\Delta \le \alpha \le 0} \left\{ g(y + \alpha d) \right\}, \text{ if } c_\phi > 0.$$

4. *Update Flows*

$x \leftarrow x + \alpha^* d$

If $|a^*| < \Delta$, return to step 1.

If $c_\phi < 0$ and $\Delta = u_\phi - x_\phi$, or if $c_\phi > 0$ and $\Delta = x_\phi$, go to 1.

55 *Pivot Required?*

Some basic variable is either at 0 or upper bound. Perform a pivot entering x_ϕ in place of one of these basic variables and go to 1.

Specialization of the pricing and line search operations for the convex simplex method have been developed in [21]. Computational experience using this approach may be found in [9, 36, 42].

Using the same terminology the reduced gradient method is now presented.

ALG-6: REDUCED GRADIENT ALGORITHM

0. *Initialization*

Let y be any feasible point and partition A into $[B \mid N]$. Renumber the variables such that $y = [y^B \mid y^N]$ and $y^B = [y_1, \ldots, y_m]$ and $y^N = [y_{m+1}, \ldots, y_{m+n}]$.

Choose a termination parameter $\varepsilon > 0$.

1. *Calculate Reduced Gradient and Direction In Nonbasic Space*

$$c^N \leftarrow \nabla g^B(y)B^{-1}N - \nabla g^N(y).$$

$$d_{m+j} \leftarrow \left\{ \begin{array}{l} c_j^N \text{, if } 0 < y_j < u_j \\ c_j^N \text{, if } c_j^N < 0 \text{ and } y_j = u_j \\ c_j^N \text{, if } c_j^N > 0 \text{ and } y_j = 0 \\ 0 \text{, otherwise} \end{array} \right\} \quad j = 1, \ldots, n.$$

If $|d_{m+j}| < \varepsilon$, $j = 1, \ldots, n$, terminate.

2. *Calculate Direction In Basic Space*

Set $d_j \leftarrow 0$, $j = 1, \ldots, m$. Let arc j be denoted by (t_j, h_j). Let R_j^+ denote the set of arcs in the path from t_j to h_j in F_B that are traversed in normal direction and let R_j^- denote the remaining arcs in this path.

a. [Initialization] $k \leftarrow m + 1$.

b. [Cycle Trace Required?] If $d_k = 0$, go to d.

c. [Update Direction] $d_j \leftarrow d_j - c_k^N$ for all $j \in R_k^+$

$d_j \leftarrow d_j + c_k^N$ for all $j \in R_k^-$.

d. [Check For Termination] If $k = m + n$, go to 3; otherwise, $k \leftarrow k + 1$ and go to b.

3. *Ratio Test*

$$\alpha_1 \leftarrow \min_{j = 1,\ldots,m} \left\{ \min_{d_j>0} \left[\frac{u_j - y_j}{d_j} \right], \min_{d_j<0} \left[\frac{y_j}{-d_j} \right] \right\},$$

If $\alpha_1 = 0$, go to 6.

$$\alpha_2 \leftarrow \min_{j = m + 1,\ldots,m + n} \left\{ \min_{d_j>0} \left[\frac{u_j - y_j}{d_j} \right], \min_{d_j<0} \left[\frac{y_j}{-d_j} \right] \right\}$$

$\alpha_3 \leftarrow \min [\alpha_1, \alpha_2]$,

4. *Line Search*

Let α^* solve the problem

$$\min_{0 \leq \alpha \leq \alpha^3} g(y + \alpha d).$$

5. *Flow Update*

$y \leftarrow y + \alpha^* d$. If $\alpha^* < \alpha_1$, return to 1.

6. *Pivot*

Some basic variable is at zero or upper bound. Perform a pivot letting some nonbasic variable having $d_{m + k} \neq 0$ enter the basis in place of some basic variable at zero or upper bound and go to 1.

Computational experience with both the convex simplex method and reduced gradient method may be found in (42). In addition, mixed methods combining both techniques have also been suggested in (42).

REFERENCES

1. Abdulaal, M., and L. J. LeBlanc, "Multimodal Network Equilibrium", Technical Report 77013, Department of Industrial Engineering and Operations Research, Southern Methodist University, (1977).

2. Bazaraa, M. S., and J. J. Jarvis, Linear Programming and Network Flows, John Wiley and Sons, New York, New York, (1977).

3. Bazaraa, M. S., and C. M. Shetty, Nonlinear Programming, Published by Georgia Institute of Technology, Atlanta, Georgia, (1977).

4. Beale, E. M. L., "An Algorithm for Solving the Transportation Problem When the Shipping Cost Over Each Route is Convex", Naval Research Logistics Quarterly, 6, 1, 43-56, (1959).

5. Beltrami, E. J., An Algorithm Approach to Nonlinear Analysis and Optimization, page 134, Academic Press, New York, (1970).

6. Bisschop, J., and A. Meeraus, "Matrix Augmentation and Partitioning in the Updating of the Basis Inverse", Mathematical Programming, 13, 3, 241-254, (1977).

7. Charnes, A., and W. W. Cooper, "Nonlinear Network Flows and Convex Programming Over Incidence Matrices", Naval Research Logistics Quarterly, 5, 3, 231-240, (1958).

8. Charnes, A., and W. W. Cooper, "Multicopy Traffic Network Models", pages 85-96, Theory of Traffic Flow, Edited by R. Herman, Elsevier Publishing Co., (1961).

9. Collins, M., L. Cooper, R. Helgason, J. Kennington, and L. LeBlanc, "Solving the Pipe Network Analysis Problem Using Optimization Techniques", (forthcoming in Management Science).

10. Cooper, L., and J. Kennington, "Steady-State Analysis of Nonlinear Resistive Electrical Networks Using Optimization Techniques", Technical Report IEOR 77012, Department of Industrial Engineering and Operations Research, Southern Methodist University, (1977).

11. Cooper, L., and L. J. LeBlanc, "Stochastic Transportation Problems and Other Network Related Convex Problems", Naval Research Logistics Quarterly, 24, 2, 327-336, (1977).

12. Cross, H., "Analysis of Flow in Networks of Conduits or Conductors", Engineering Experiment Station Bulletin No. 286, The University of Illinois, (1936).

13. Dafermos, S. C., "An Extended Traffic Assignment Model With Applications to Two-Way Traffic", Transportation Science, 5, 4, 366-389, (1971).

14. Dafermos, S. C., "The Traffic Assignment Problem for Multiclass-User Transportation Networks", Transportation Science, 6, 1, 73-87, (1971).

15. Dantzig, G. B., "Linear Programming Under Uncertainty", Management Science, 1, 197-206, (1955).

16. Dennis, J. B., Mathematical Programming and Electrical Networks, John Wiley and Sons, Inc., New York, (1959).

17. Florian, M., "An Improved Linear Approximation Algorithm for the Network Equilibrium (Packet Switching) Problem", Technical Report, Centre de Recherche sur les Transports, Université de Montréal, (1977).

18. Florian, M., S. Nguyen, and F. Soumis, "Two Methods for Accelerating an Equilibrium Traffic Assignment Algorithm", Technical Report #251, Departement d'informatique, University of Montreal, Montreal, Quebec, (1977).

19. Frank, M., and P. Wolfe, "An Algorithm for Quadratic Programming", Naval Research Logistics Quarterly, 3, 2, 95-110, (1956).

20. Hall, M. A., "Hydraulic Network Analysis Using (Generalized) Geometric Programming", Networks, 6, 105-130, (1976).

21. Helgason, R. V., and J. L. Kennington, "An Efficient Specialization of the Convex Simplex Method for Nonlinear Network Flow Problems", Technical Report IEOR 77017, Department of Industrial Engineering and Operations Research, Southern Methodist University, (1977).

22. Holloway, C. A., "An Extension of the Frank and Wolfe Method of Feasible Directions", Mathematical Programming, 6, 14-27, (1974).

23. Hu, T. C., "Minimum-Cost Flows in Convex-Cost Networks", Naval Research Logistics Quarterly, 13, 1, 1-9, (1966).

24. Jewell, W. S., "Models for Traffic Assignment", Transportation Research, 1, 31-46, (1967).

25. Johnson, E. L., "Programming in Networks and Graphs", Operations Research Center Report No. 65-1, University of California, Berkeley, (1965).

26. Jorgensen, N. O., "Some Aspects of the Urban Traffic Assignment Problem", Institute of Transportation and Traffic Engineering Graduate Report, University of California, Berkeley, (1963).

27. Klein, M., "A Primal Method for Minimal Cost Flows With Applications to the Assignment and Transportation Problems", Management Science, 14, 3, 205-220, (1967).

28. Kreyszig, E., Advanced Engineering Mathematics, page 305, John Wiley, New York, (1964).

29. Lasdon, L. S., Optimization Theory for Large Systems, MacMillan Co., New York, (1972).

30. LeBlanc, L., E. Morlok, and W. Pierskalla, "An Efficient Approach to Solving the Road Network Equilibrium Traffic Assignment Problem", Transportation Research, 9, 309-318, (1975).

31. LeBlanc, L. J., and L. Cooper, "The Transportation-Production Problem", Transportation Science, 8, 4, 344-354, (1974).

32. LeBlanc, L. J., M. Abdulaal, and R. Helgason, "On the Improved Rate of Convergence of the Frank-Wolfe Algorithm for Large Scale Convex Network Problems", Presented at the National Meeting of ORSA/TIMS in Atlanta, (1977).

33. Luenberger, D. G., Introduction to Linear and Nonlinear Programming, Addison-Wesley, Reading, Mass., (1973).

34. Menon, V. V., "The Minimal Cost Flow Problem With Convex Costs", Naval Research Logistics Quarterly, 12, 2, 163-172, (1965).

35. Nguyen, S., "A Mathematical Programming Approach to Equilibrium Methods of Traffic Assignment With Fixed Demands", Technical Report, Centre de Recherche sur les Transports, Université de Montréal, (1973).

36. Nguyen, S., "An Algorithm for the Traffic Assignment Problem", Transportation Science, 8, 203-216, (1974).

37. Roberts, A. W., and D. E. Varberg, Convex Functions, Academic Press, New York, New York, (1973).

38. Rosen, J., "The Gradient Projection Method for Nonlinear Programming: I. Linear Constraints", J. Soc. Indust. Appl. Math., 8, 181-217, (1960).

39. Rosenthal, R. E., "Scheduling Reservoir Releases for Maximum Hydropower Benefit by Nonlinear Programming on a Network", Technical Report, Management Science Program, University of Tennessee, Knoxville, (1977).

40. Sharp, J. F., J. C. Snyder, and J. H. Greene, "A Decomposition Algorithm for Solving the Multifacility Production-Transportation Problem with Nonlinear Production Costs", Econometrica, 19, 490-506, (1970).

41. Simmons, D. M., Nonlinear Programming for Operations Research, Prentice-Hall, Englewood Cliffs, N. J., (1975).

42. Tjian, T. Y., and W. I. Zangwill, "Analysis and Comparison of the Reduced Gradient and the Convex Simplex Method for Convex Programming", Working Paper No. 273, Center for Research in Management Science, University of California, Berkeley, (1969).

43. Tomlin, J. A., "A Mathematical Programming Model for the Combined Distribution-Assignment of Traffic", Transportation Science, 5, 2, 122-140, (1971).

44. Wardrop, J. G., "Some Theoretical Aspects of Road Traffic Research", Proc. Inst. Civ. Eng., 1, Part III, 325-378, (1952).

45. Weintraub, A., "Primal Algorithm to Solve Network Flow Problems With Convex Costs", Management Science, 21, 1, 87-97, (1974).

46. Wolfe, P., "Methods of Nonlinear Programming", pages 99-130 of Nonlinear Programming, J. Abadie Editor, North Holland, Amsterdam, (1967).

47. Wolfe, P., "Convergence Theory in Nonlinear Programming", Integer and Nonlinear Programming, Edited by J. Abadie, North Holland, Amsterdam, (1970).

48. Zangwill, W. I., "The Convex Simplex Method", Management Science, 14, 3, 221-238, (1967).

49. Zangwill, W. I., Nonlinear Programming: A Unified Approach, Prentice-Hall, Inc., Englewood Cliffs, N. J., (1969).

50. Zoutendijk, G., Methods of Feasible Directions, Elsevier Publishing Company, Amsterdam, (1960).

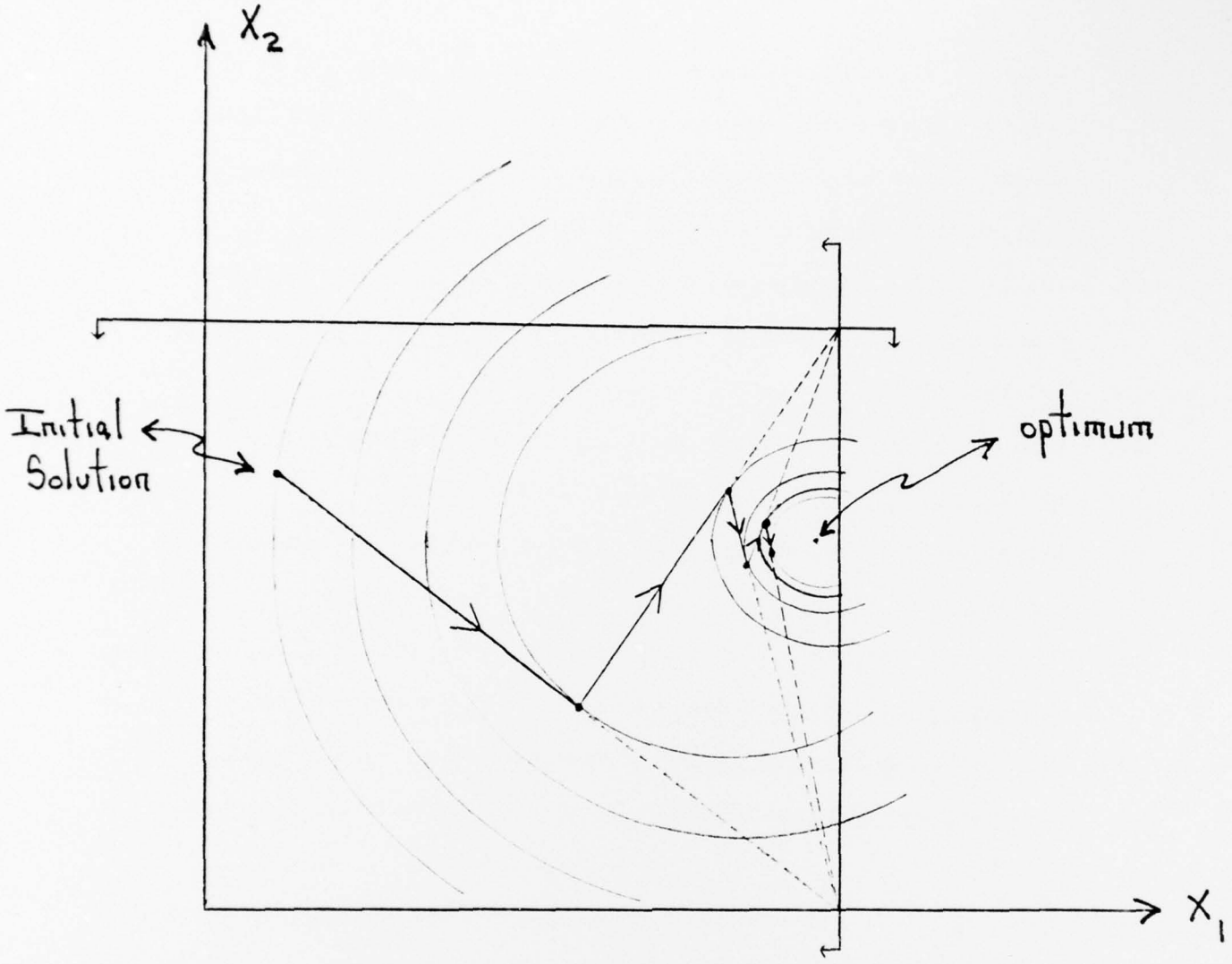

Figure 1. Illustration of Zigzagging Phenomenon of Frank-Wolfe Method.

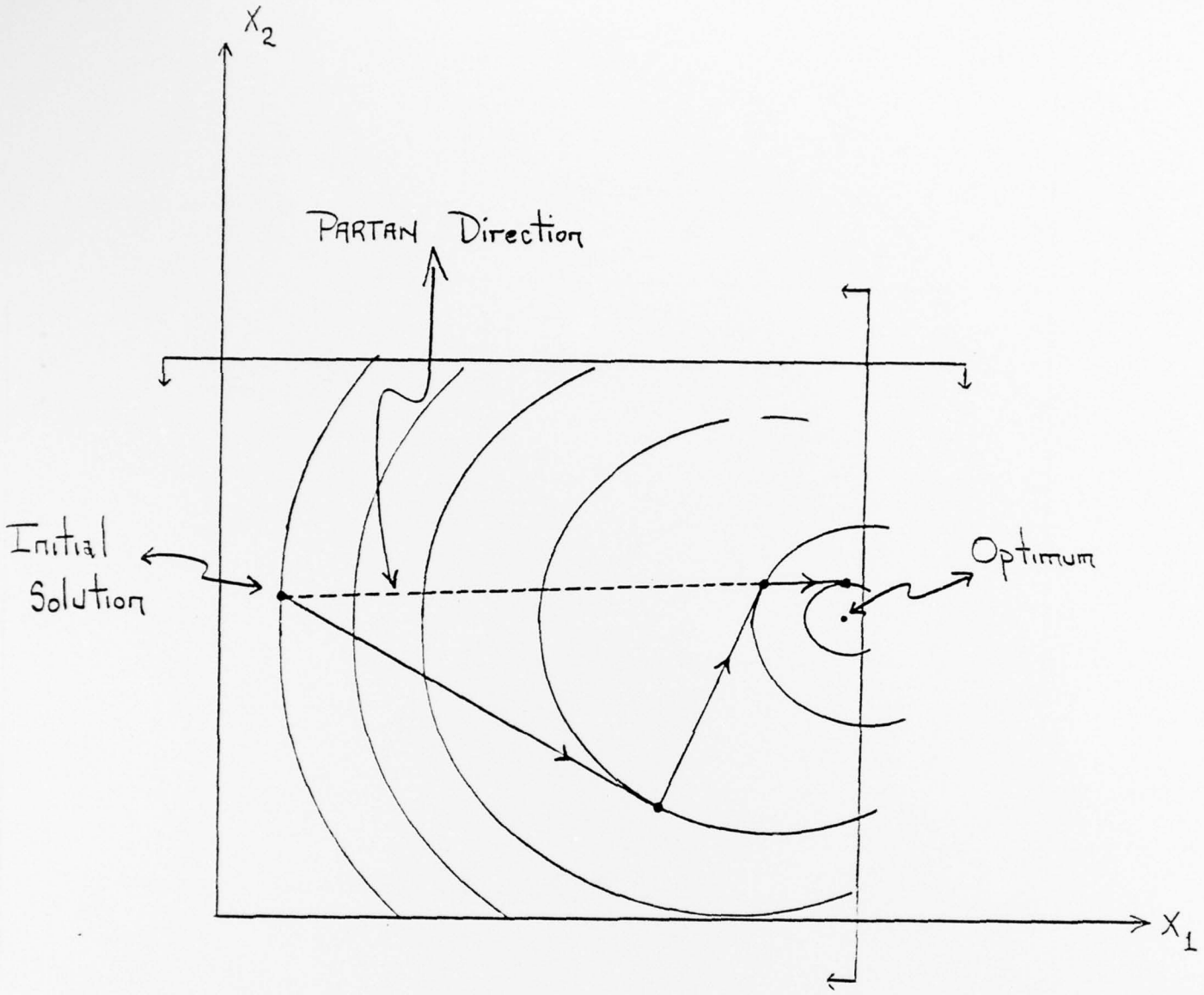

Figure 2. Illustration of Frank-Wolfe Method with PARTAN.

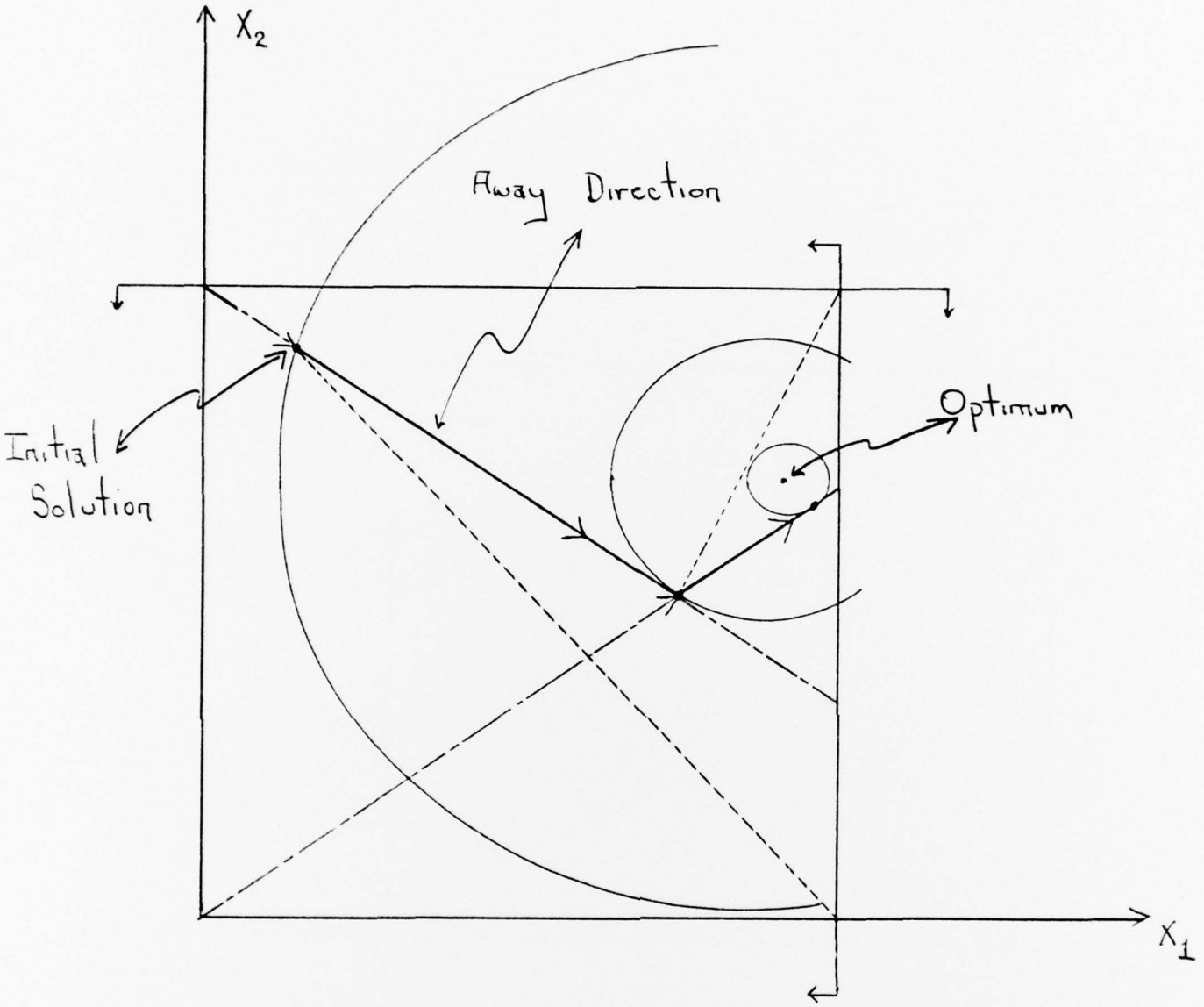

Figure 3. Illustration of Frank-Wolfe Method Using Away Steps. (----, toward step directon; — ——— - —,away step direction.)

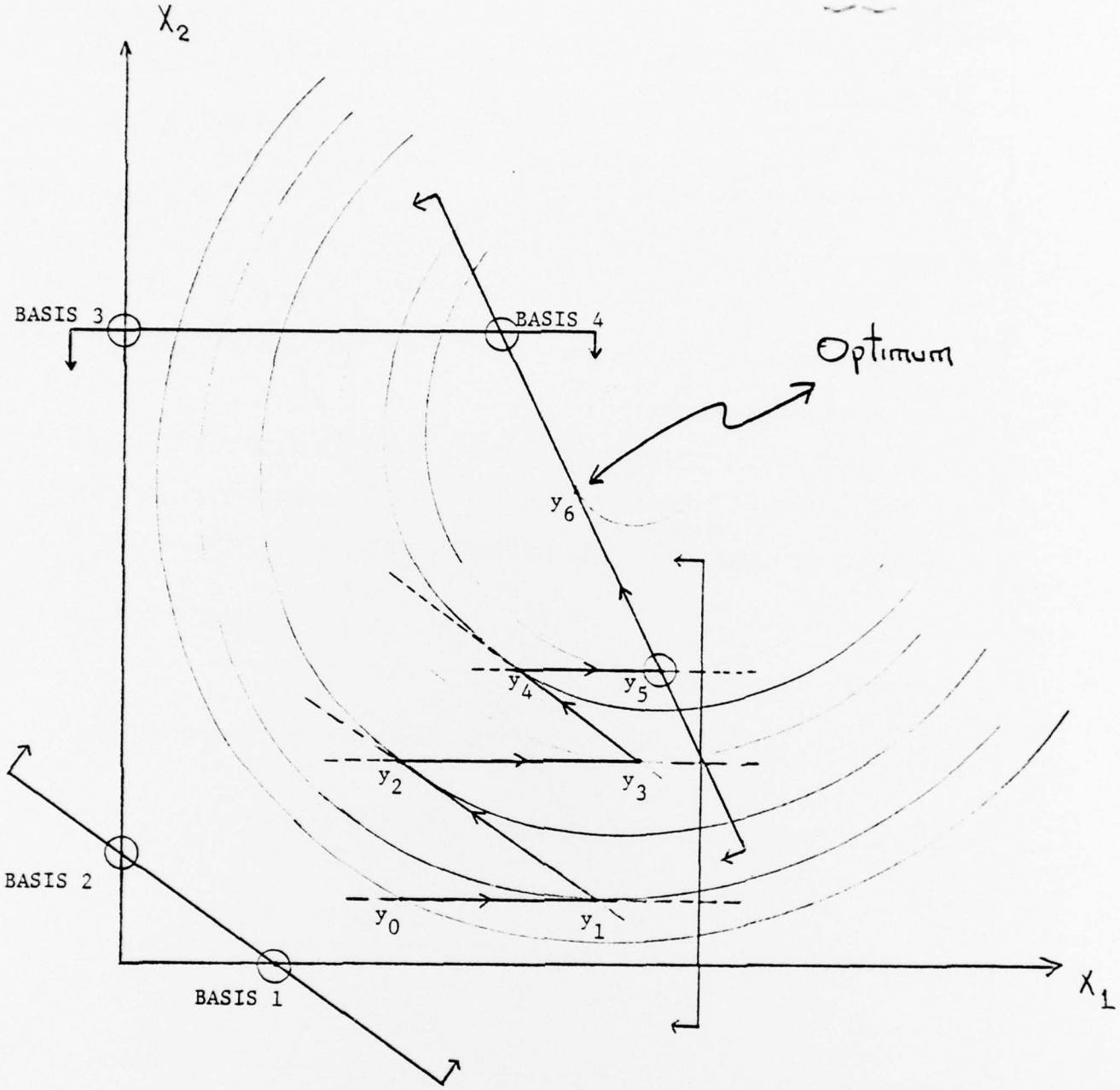

Figure 4. Illustration of Convex Simplex Algorithm. (Points y_0 through y_4 correspond to BASIS 1. There are two degenerate pivots at y_5 corresponding to bases 2 and 3 before reaching y_6).

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFOSR-TR-78-0332
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): THE CONVEX COST OF NETWORK FLOW PROBLEM: A STATE-OF-THE ART SURVEY.
5. TYPE OF REPORT & PERIOD COVERED: Interim rept.,
6. PERFORMING ORG. REPORT NUMBER: OREM-78112

7. AUTHOR(s): A. I. Ali, R. V. Helgason and J. L. Kennington
8. CONTRACT OR GRANT NUMBER(s): AFOSR-77-3151

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Southern Methodist University
Department of Operations Research
Dallas, TX 75275

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
61102F 2304/A6

11. CONTROLLING OFFICE NAME AND ADDRESS
Air Force Office of Scientific Research/NM
Bolling AFB, DC 20332

12. REPORT DATE: January 1978
13. NUMBER OF PAGES: 41

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
network, nonlinear programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This exposition presents a state-of-the-art survey of models and algorithms for the convex cost network flow problem.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

410 597 VL