A Second-Order Method for Solving the Continuous Multifacility Location Problem[+]

P.H. Calamai
Department of Systems Design

and

A.R. Conn
Department of Computer Science

University of Waterloo
Waterloo, Ontario, Canada

Research Report CS-81-26
August 1981

Faculty

of

Mathematics

University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

A SECOND-ORDER METHOD FOR SOLVING THE CONTINUOUS MULTIFACILITY LOCATION PROBLEM†

P.H. CALAMAI and A.R. CONN

ABSTRACT

A unified and numerically stable second-order approach to the continuous multifacility location problem is presented. Although details are initially given for only the unconstrained Euclidean norm problem, we show how the framework can be readily extended to l_p norm and mixed norm problems as well as to constrained problems.

Since the objective function being considered is not everywhere differentiable the straightforward application of classical solution procedures is infeasible. The method presented is an extension of an earlier first-order technique of the authors and is based on certain non-orthogonal projections. For efficiency the linear substructures that are inherent in the problem are exploited in the implementation of the basic algorithm and in the manner of handling degeneracies and near-degeneracies. The line search we describe also makes use of the structure and properties of the problem. Moreover, the advantages that we derive from the linear substructures are equally applicable to small-scale and large-scale problems.

Some preliminary numerical results and comparisons are included.

1. INTRODUCTION

Since the 17th century, when Fermat first posed a single facility location problem involving Euclidean distances, the issue of locating an object according to some set of rules and criteria has recieved a great deal of attention in the literature. The bibliographies of Lea [11] and Francis and Goldstein [7] together represent well over a thousand references to these problems.

In general, location problems ask where some object or objects should be placed to improve a measure of the performance of the system in which they interact. Here, we consider a prototype location problem: the static and deterministic formulation of the minisum multifacility location problem involving l_p distances. The objective involves locating a set of new facilities (objects) in a system of existing facilities to minimize the sum of weighted l_p distances between the new and existing facilities and among the new facilities.

One difficulty with the l_p distance problem is that the objective function is not everywhere differentiable. In fact, nondifferentiability occurs whenever any two facilities coincide. The straightforward use of gradient reducing procedures to solve this problem is therefore inapplicable. However, various methods that circumvent this nondifferentiability have been used. For example, linear programming methods (see [15] and [21]) and gradient reduction methods on approximating functions (see [20]) have been used to solve the rectilinear distance (l_1) location problems. For Euclidean distance (l_2) problems, modified gradient reducing methods have also been used (see [6]

† This work was supported in part by Natural Science and Engineering Research Council of Canada Grant No. A8639.

and [12]), as have subgradient methods, pseudo-gradient methods and heuristic methods (see [3], [1] and [19]).

In this paper we present a projected Newton method for solving the l_p distance location problem and we describe an implementation of this method that takes full advantage of the structure of the problem and its graphic interpretation. This second-order technique is a natural extension of the first-order projected steepest descent algorithm reported in [2]. A similar extension, developed independently by Overton, is presented in [17]. In his paper, the quadratic convergence of a projected Newton method is proved and a special line search is described.

2. PROBLEM STATEMENT

The multifacility minisum problem involving costs associated with Euclidean distances between facilities in $\mathbf{R}^q$ can be stated as: Find the point $x^{*T} = \{x_1^{*T}, \ldots, x_n^{*T}\}$ in $\mathbf{R}^{qn}$ to minimize

$$f(x) = \sum_{1 \leqslant j < k \leqslant n} v_{jk} \| x_j - x_k \| + \sum_{j=1}^{n} \sum_{i=1}^{m} w_{ji} \| x_j - p_i \| , \tag{2.1}$$

where

$n \underline{\Delta}$ number of new facilities (NF's) to be located.

$m \underline{\Delta}$ number of existing facilities (EF's).

$x_j^T = (x_{j1} \cdots x_{jq}) \underline{\Delta}$ vector location of NF_j in $\mathbf{R}^q$, $j = 1, \ldots, n$.

$p_i^T = (p_{i1} \cdots p_{iq}) \underline{\Delta}$ vector location of EF_i in $\mathbf{R}^q$, $i = 1, \ldots, m$.

$v_{jk} \underline{\Delta}$ nonnegative constant of proportionality relating the l_2 distance between NF_j and NF_k to the cost incurred, $1 \leqslant j < k \leqslant n$.

$w_{ji} \underline{\Delta}$ nonnegative constant of proportionality relating the l_2 distance between NF_j and EF_i to the cost incurred, $1 \leqslant j \leqslant n$, $1 \leqslant i \leqslant m$.

$\| x_j - x_k \| = (\sum_{c=1}^{q} | x_{jc} - x_{kc} |^2)^{1/2} \underline{\Delta}\, l_2$ distance between NF_j and NF_k, $1 \leqslant j < k \leqslant n$.

$\| x_j - p_i \| = (\sum_{c=1}^{q} | x_{jc} - p_{ic} |^2)^{1/2} \underline{\Delta}\, l_2$ distance between NF_j and EF_i, $1 \leqslant j \leqslant n$, $1 \leqslant i \leqslant m$.

If we randomly collect all the nonzero v_{jk} constants into an ordered set where the first member is called α_1 and the last member is called α_η and then randomly collect all the nonzero w_{ji} constants into an ordered set where the first member is called $\alpha_{\eta+1}$ and the last member is called α_τ, then, if we define the index set $M = \{1, \ldots, \tau\}$, problem (2.1) can be restated more conveniently as

$$\text{minimize} \qquad f(x) = \sum_{i \in M} \| A_i^T x - b_i \| \tag{2.2}$$

where the $q \times qn$ matrix A_i^T and the $q \times 1$ vector b_i are defined by

$$A_i^T = [0_1 \quad \alpha_i I \quad 0_2 \quad \overline{\alpha}_i I \quad 0_3] \qquad i = 1, \ldots, \tau$$

$$b_i = (\alpha_i + \bar{\alpha}_i) p_{i^*} \qquad i = 1, \ldots, \tau$$

and

$$\bar{\alpha}_i = \begin{cases} -\alpha_i & i = 1, \ldots, \eta \\ 0 & i = \eta+1, \ldots, \tau \end{cases}$$

I is a $q \times q$ identity matrix

0_1 is a zero matrix of dimension $q \times q\ (j^* - 1)$

0_2 is a zero matrix of dimension $q \times q\ (k^* - j^* - 1)$

0_3 is a zero matrix of dimension $q \times q\ (n - k^*)$

$j^* = j$ where $\alpha_i \equiv v_{jk}$ or $\alpha_i \equiv w_{ji}$

$$k^* = \begin{cases} k & \text{where } \alpha_i \equiv v_{jk} \\ j+1 & \text{where } \alpha_i \equiv w_{ji} \end{cases}$$

$$i^* = \begin{cases} \text{an arbitrary constant} & \text{where } \alpha_i \equiv v_{jk} \\ i & \text{where } \alpha_i \equiv w_{ji} \end{cases}$$

3. ANALYSIS

3.1 Introduction

One difficulty in solving problem (2.2) arises because the convex objective function $f(x)$ is not everywhere differentiable. If we let

$$r_i(x) = A_i^T x - b_i \qquad \forall i \in M$$

then the objective function can be written as

$$f(x) = \sum_{i \in M \backslash I_\epsilon(x)} \| r_i(x) \| + \sum_{i \in I_\epsilon(x)} \| r_i(x) \| \qquad (3.1.1)$$

$$= \tilde{f}(x) + \sum_{i \in I_\epsilon(x)} \| r_i(x) \| .$$

With a proper choice of the index set $I_\epsilon(x)$ we can guarantee that the function $\tilde{f}(x)$ is clearly differentiable in the neighbourhood of x and that the remaining expression contains all the nondifferentiable (and near-nondifferentiable) terms.

3.2 A First-Order Method

Suppose we wish to minimize the first-order change in the objective function $f(x)$ by moving in some direction h. This can often be accomplished by minimizing the unit first-order change in the function $\tilde{f}(x)$ subject to the condition that the first-order change in the remaining terms $\| r_i(x) \|$, $i \in I_\epsilon(x)$ remain zero. That is, we solve

$$\begin{aligned} &\underset{h}{\text{minimize}} && h^T\nabla\tilde{f}(x) \\ &\text{subject to} && A_i^T h = 0 \qquad i \in I_\epsilon(x) \end{aligned} \tag{3.2.1}$$

(Notice that we are able to take advantage of the linear substructure of this problem in the constraint terms.)

The solution to this problem yields the direction

$$h = -\gamma P \nabla\tilde{f}(x) \tag{3.2.2}$$

where P is the orthogonal projector onto $S^\perp$, S is the space spanned by the columns of A_i $i \in I_\epsilon(x)$ and $\gamma > 0$ is chosen to satisfy the bound on the norm of h. Thus, for $h = -\gamma P \nabla\tilde{f}(x) \neq 0$, $g = \nabla f$ and λ sufficiently small, we have

$$\begin{aligned} f(x+\lambda h) - f(x) &= \lambda g^T h + 0(\lambda^2) \\ &= -\lambda\gamma \| P\nabla\tilde{f}(x) \| + 0(\lambda^2) \\ &< 0 \ . \end{aligned} \tag{3.2.3}$$

If $P\nabla\tilde{f}(x) = 0$ then $\nabla\tilde{f}(x)$ must lie entirely in S (in this case we call the point x a dead point; see § 3.4). Letting $\mathbf{A} = [\bar{A}_{i_1} \cdots \bar{A}_{i_t}]$ where $I_\epsilon(x) = \{i_1 \cdots i_t\}$ and $\alpha_i \bar{A}_i = A_i$ $i \in I_\epsilon(x)$ then, assuming $\mathbf{A}$ is full rank, $\nabla\tilde{f}(x)$ can be uniquely expressed as

$$\begin{aligned} \nabla\tilde{f}(x) &= \mathbf{A}u \qquad u^T = [u_{i_1}{}^T \cdots u_{i_t}{}^T] \\ &= \sum_{i \in I_\epsilon(x)} \bar{A}_i u_i . \end{aligned} \tag{3.2.4}$$

(The vector u is called the Lagrange or dual vector; see § 3.4.) Then, for any choice of h, we have

$$\begin{aligned} f(x+\lambda h) - f(x) &= \lambda \left[h^T\nabla\tilde{f}(x) + \sum_{i \in I_0} \| A_i^T h \| \right. \\ &\quad \left. + \sum_{i \in I_1} h^T \nabla(\| r_i(x) \|) \right] + 0(\lambda^2) \\ &= \lambda \left[\sum_{i \in I_0} \left(u_i^T \bar{A}_i^T h + \alpha_i \| \bar{A}_i^T h \| \right) \right. \\ &\quad \left. + \sum_{i \in I_1} \left(u_i^T \bar{A}_i^T h + \alpha_i \frac{r_i(x)^T \bar{A}_i^T h}{\| r_i(x) \|} \right) \right] + 0(\lambda^2) \end{aligned} \tag{3.2.5}$$

where $I_0 = \{ i \in I_\epsilon(x) \mid \| r_i(x) \| = 0 \}$ and $I_1 = I_\epsilon(x) \backslash I_0$. If, under these circumstances, there exists an index $l \in I_\epsilon(x)$ such that $\| u_l \| > \alpha_l$ we take as our descent direction

$$\tilde{h}_l = -\gamma P_l \bar{A}_l u_l \tag{3.2.6}$$

where P_l is the orthogonal projector onto $S_l^\perp$, S_l is the space spanned by the columns of $\mathbf{A}$ with columns $\bar{A}_l$ deleted and $\bar{A}_l^T \tilde{h}_l = -\rho u_l$ where $\rho > 0$ (see [2]). For this choice of direction (i.e. $h = \tilde{h}_l$) and for sufficiently small $\lambda > 0$ we will have

$$f(x+\lambda h)-f(x) = \lambda g^T h + 0(\lambda^2)$$

$$= \begin{cases} -\lambda\rho\left[\| u_l\|^2 - \alpha_l \| u_l \|\right] + 0(\lambda^2) & l \in I_0 \\ -\lambda\rho\left[\| u_l\|^2 + \alpha_l \dfrac{r_l(x)^T u_l}{\| r_l(x)\|}\right] + 0(\lambda^2) & l \in I_1 \end{cases} \qquad (3.2.7)$$

$$< 0$$

where

$$g = \begin{cases} \nabla\tilde{f}(x) - \dfrac{A_l u_l}{\| u_l\|} & l \in I_0 \\ \nabla\tilde{f}(x) + \dfrac{A_l r_l(x)}{\| r_l(x)\|} & l \in I_1 \ . \end{cases}$$

For detail of this first-order method see [2].

3.3 A Second-Order Method

We wish now to find a direction h which minimizes the second-order change in the objective function. This can be accomplished by minimizing the change in the function $f(x)$ up to second-order terms subject to the condition that the change in the remaining expressions remain zero up to second-order terms. Thus, we solve the following problem

$$\underset{h}{\text{minimize}} \ \ h^T\nabla\tilde{f}(x) + \tfrac{1}{2} h^T\nabla^2\tilde{f}h \qquad (3.3.1)$$

$$\text{subject to} \quad A_i^T h = 0 \qquad i \in I_\epsilon(x).$$

(The quadratic programming problem with quadratic constraints that ensues in [4] when both function and constraint curvature are included has been simplified because of the linear substructure of this problem's constraints.)

Now, define a $qn \times q(n-t)$ matrix Z satisfying

$$A_i^T Z = 0 \qquad i \in I_\epsilon(x) \qquad (3.3.2)$$

$$Z^T Z = I_{q(n-t)} \qquad (3.3.3)$$

and the transformation $h = Zw$ so that problem (3.3.1) becomes

$$\min_{\mathrm{w}} \ w^T Z^T \nabla\tilde{f} + 1/2\, w^T Z^T \nabla^2 \tilde{f} Z w \qquad (3.3.4)$$

The solution to this problem can be obtained (assuming $\nabla^2\tilde{f}$ is positive definite; see § 4.7) by finding the vector $w = w^*$ that satisfies

$$Z^T \nabla^2 \tilde{f} Z w = -Z^T \nabla \tilde{f} \qquad (3.3.5)$$

Problem (3.3.1) can then be solved by setting

$$h = Zw^*. \tag{3.3.6}$$

Assuming this solution is different from zero, the direction h is a second-order descent direction for f in the neighbourhood of the point x.

3.4 Dead Points, Dual Estimates and Dropping

Assume we are at some point $\bar{x}$ where the solution to (3.3.5) is zero. Under this condition, $\bar{x}$ is called a dead point for this problem. As in the first-order case, if **A** is full rank, $\nabla f(x)$ can be expressed uniquely as

$$\nabla \tilde{f}(\bar{x}) = \mathbf{A}\bar{u} \tag{3.4.1}$$

What if we are at a point x^k in the neighbourhood of $\bar{x}$ where $I_\epsilon(x^k) \equiv I_\epsilon(\bar{x})$ and the solution to (3.3.5) is "small"? If we define the dual estimate u^k as the least-squares solution to

$$\mathbf{A}u^k = \nabla \tilde{f}(x^k) \tag{3.4.2}$$

then this dual estimate will usually give a reasonable approximation to the dual $\bar{u}$ found in (3.4.1) [see [4], § 2b].

If we are at this dead point $\bar{x}$ and there exists an index $l \in I_\epsilon(\bar{x})$ such that $\| \bar{u}_l \| > \alpha_l$, where $\bar{u}$ is the solution to (3.4.1), then the direction $\bar{h}_l = -\gamma P_l \bar{A}_l \bar{u}_l$ is a descent direction for f at $\bar{x}$ as shown in § 3.2.

If x^k is "sufficiently close" to $\bar{x}$ then $\| u_l^k \| > \alpha_l$ where u^k is the least-squares solution obtained in (3.4.2) when $x = x^k$. Thus there is a neighbourhood of $\bar{x}$ for which

$$\tilde{h}_l = -\gamma P_l \bar{A}_l u_l^k \tag{3.4.3}$$

is a descent direction for f at x^k. (When we use this direction we "drop" $\bar{A}_l$ from **A**.)

3.5 Optimality Conditions and the Linear Refinement

Suppose we are at a point x^* which is a dead point for our function f. In addition, suppose $I_\epsilon(x^*) = \{i_1, \ldots, i_t\}$, $\tilde{\mathbf{A}} = [\bar{A}_{i_1}, \ldots, \bar{A}_{i_t}]$ is full rank and $\| u^*_i \| \leqslant \alpha_i \;\; \forall i \in I_\epsilon(x^*)$ where $u = u^*$ is a solution to $Au = \nabla \tilde{f}(x^*)$. Then, as long as $I_1(x^*) = \emptyset$, we have (from (3.2.5))

$$f(x + \lambda h) - f(x) = \lambda \left[\sum_{i \in I_\epsilon(x^*)} u^{*T}_i \bar{A}_i^T h + \alpha_i \| \bar{A}_i^T h \| \right] + 0(\lambda^2) \tag{3.5.1}$$

for any choice of direction h. For sufficiently small $\lambda > 0$ this expression must be nonnegative. Since our objective function f is convex, the point x^* must therefore solve our original problem.

However, suppose we are at a point x^k in the neighbourhood of some dead point $\bar{x}$. In addition, assume $\| u_i^k \| \leqslant \alpha_i \;\; \forall i \in I_\epsilon(x^k)$ where u^k is the least-squares solution to (3.4.2) at this point x^k. As long as x^k was "sufficiently close" to $\bar{x}$ we would then expect $\| \bar{u}_i \| \leqslant \alpha_i \;\; \forall i \in I_\epsilon(\bar{x})$ where $\bar{u}$ is the solution to (3.4.1). If this were true and $I_1(\bar{x}) = \emptyset$ then $\bar{x}$ would be a solution to our problem! How then should we proceed from the point x^k?

The direction h^k obtained by solving problem (3.3.4) with $x = x^k$ would certainly be a local descent direction for f at the point x^k. It would therefore make sense to take the step $x^k + \lambda_k h^k$ (where λ_k is some computed stepsize) to reduce f. It would also seen appropriate to take some action to force the condition $I_1(x^k + \lambda_k h^k) = \emptyset$. We therefore define the refinement step v^k as the solution to the linear system

$$\begin{bmatrix} \mathbf{A}^T \\ ----- \\ Z^T \end{bmatrix} v^k = \begin{bmatrix} -\bar{r}(x^k+\lambda_k h^k) \\ --------------- \\ 0 \end{bmatrix} \qquad (3.5.2)$$

where $\bar{r}$ is the ordered vector of residuals $\bar{r}_i$ (where $\alpha_i \bar{r}_i = r_i$) corresponding to the matrices A_i in $\mathbf{A}$. (The reader should note that the solution to (3.5.2) is the least-squares solution of minimal norm to $\mathbf{A}^T v = -\bar{r}(x^k+\lambda_k h^k)$ since $Z^T v^k = 0$).

With this choice of refinement step we force $I_1(x^k+\lambda_k h^k) = \emptyset$ since

$$\begin{aligned} \bar{r}(x^k+\lambda_k h^k + v^k) &= \bar{r}(x^k+\lambda_k h^k) + \mathbf{A}^T v_k \\ &= 0. \end{aligned}$$

(We try to take $\lambda_k = 1$ in this instance since, for second-order methods, a stepsize of one will asymptotically be optimal; see § 3.7.).

3.6 Degeneracy and Perturbations

Whenever the solution to (3.3.4) is "small" the dual estimate, obtained by solving (3.4.2) in the least-squares sense, becomes important in finding a descent direction or in determining optimality. The uniqueness of this estimate is based on the assumption that the matrix $\mathbf{A} = [\bar{A}_{i_1} \cdots \bar{A}_{i_t}]$, where $I_\epsilon = \{i_1 \cdots i_t\}$, is full rank. This uniqueness is surrendered whenever $\mathbf{A}$ is rank deficient. Fortunately this difficulty can be resolved.

Assume, for the moment, that we are at a point x^k where $\mathbf{A}$ is rank deficient. If we redefine $\mathbf{A} = [\bar{A}_{j_1} \cdots \bar{A}_{j_s}]$, where $J_\epsilon(x^k) = \{j_1 \cdots j_s\} \subset I_\epsilon(x^k)$ and $\mathbf{A}$ forms a basis for the column space of $A_i \;\; i \in I_\epsilon(x^k)$, then we can find the least-squares solution to (3.4.2) and obtain the dual estimate u^k which is uniquely defined by this choice of basis. If, after proceeding in this fashion, we find that $\| u_j^k \| \leqslant \alpha_j$ for all $j \in J_\epsilon(x^k)$ then we can safely continue as outlined in § 3.5 (since $A_i^T h = 0 \;\; \forall i \in I_\epsilon(x^k)$). If, however, there exists an index $l \in J_\epsilon(x^k)$ such that $\| u_l^k \| > \alpha_l$, can we then take the direction $h_l = -\gamma P_l \bar{A}_l u_l^k$ as our descent direction under the assumption that (3.2.7) still holds?

The answer to this question is, in general, no. This is because (3.2.7) is based, in part, on the result that, when $\mathbf{A} = [\bar{A}_{i_1} \cdots \bar{A}_{i_t}]$ $\;\bar{A}_i^T \tilde{h}_l = 0$ for all $i \in I_\epsilon(x^k) - \{l\}$. When $\mathbf{A} = [\bar{A}_{j_1} \cdots \bar{A}_{j_s}]$ we can only guarantee that $\bar{A}_j^T \tilde{h}_l = 0$ for all $j \in J_\epsilon(x^k) - \{l\}$.

Determining an optimal strategy under these circumstances is not a trivial exercise. Here we suggest an approach that is both simple and effective. After taking a refinement step and setting $x^k \leftarrow x^k + v^k$ we randomly perturb the values b_i of all the residuals in the set $I_\epsilon(x^k) \backslash J_\epsilon(x^k)$ so that the gradients $\nabla (\| r_i(x^k) \|)$ are well-defined. This allows these perturbed terms to join the function f when we proceed with our minimization method. Using this approach we either leave this degenerate neighbourhood or identify a solution in this degenerate neighbourhood. In the latter case the point x^k is our solution.

3.7 Minimization Strategy

In order to decrease the objective function at each stage in the minimization process, a decision must be made as to which direction to use. The strategy we suggest here is based, in part, on the analysis presented in [4] and [5].

We consider the following three cases (in all three cases $h = -Z(Z^T \nabla^2 \tilde{f} Z)^{-1} Z^T \nabla \tilde{f}$):

CASE 1: $\| h \| > \beta$

The fact that $\| h \|$ is "large" suggests we are outside the neighbourhood of any dead points. Under this condition we use this direction h to decrease f by setting

$$x \leftarrow x + \lambda h$$

where the stepsize λ is determined via the line search described in § 3.8.

CASE 2: $\| h \| \leqslant \beta$ and $\| u_i \| \leqslant \alpha_i \quad \forall i \in I_\epsilon$

The assumption here is that we are in the neighbourhood of some dead point (which may be optimal). We therefore set

$$\tilde{x} \leftarrow x + h + v$$

where v is the solution to

$$\left[\begin{array}{c} A^T \\ \hline Z^T \end{array} \right] v = \left[\begin{array}{c} -\bar{r}(x+h) \\ \hline 0 \end{array} \right].$$

If $f(\tilde{x}) - f(x) < -\delta_0 \; (\| h \|^2 + r^\infty)$ where δ_0 is some positive constant and $r^\infty = \max\{\| r_i(x) \| \, , \, i \in I_\epsilon(x)\})$ then we accept this as being a "sufficient" decrease and take

$$x \leftarrow \tilde{x};$$

otherwise, we set

$$\epsilon \leftarrow \epsilon/2$$

$$\beta \leftarrow \beta/2$$

and

$$x \leftarrow x + \lambda h$$

where the stepsize λ is determined via the line search described in § 3.8.

CASE 3: $\| h \| \leqslant \beta$ and there exists at least one index $l \in I_\epsilon$ such that $\| u_l \| > \alpha_l$

In this case we define the direction

$$\tilde{h}_l = -\gamma P_l \bar{A}_l u_l.$$

If $g^T \tilde{h}_l < -\delta$, where $\delta = \delta_0 * f(x)/\tau$ and g is defined in § 3.2, then a sufficient decrease along this direction is expected and we set

$$x \leftarrow x + \lambda \tilde{h}_l$$

where the stepsize λ is determined via the line search described in § 3.8. If, on the other hand, $g^T \tilde{h}_l \geqslant -\delta$ for all $l \in I_\epsilon$ with $\| u_l \| > \alpha_l$, we remain at the same point x but set

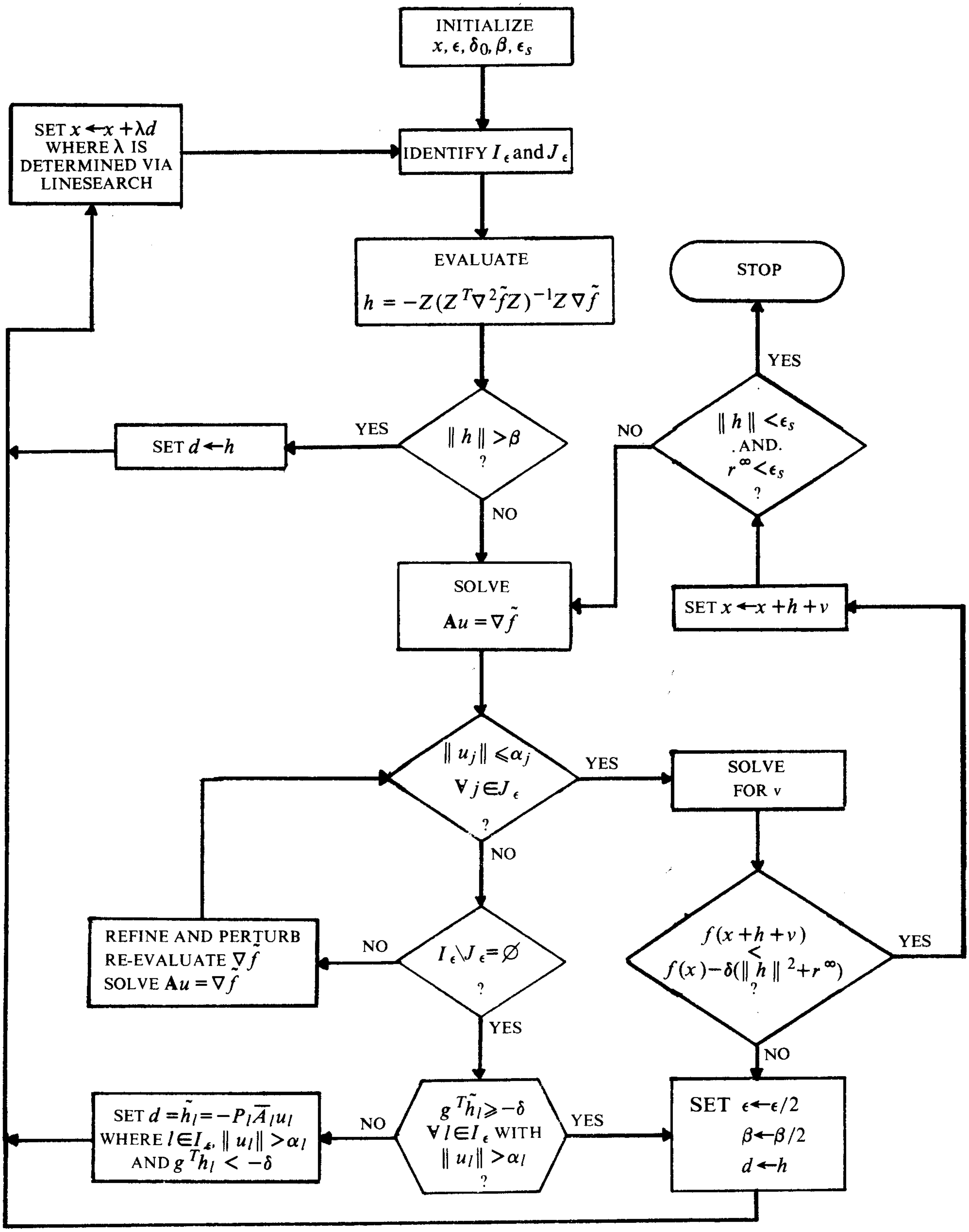
$$\epsilon \leftarrow \epsilon/2$$

and

$$\beta \leftarrow \beta/2.$$

The parameters β and ϵ are adjusted whenever the step $h + v$ is unsuccessful or the step $\tilde{h}_l$ fails. This, in effect, refines our tests for dead-point neighbourhoods and nondifferentiability.

What follows is a flowchart of our algorithm for solving the continuous multifacility location problem involving Euclidean distances. The performance of this method is affected by the initial choice of the parameters ϵ and β (it is assumed that the chosen values for δ_0 and ϵ_s are "reasonable" and that they therefore have little or no effect on the algorithm's efficiency). The decision as


INITIALIZE
$x, \epsilon, \delta_0, \beta, \epsilon_s$
SET $x \leftarrow x + \lambda d$ WHERE λ IS DETERMINED VIA LINESEARCH
IDENTIFY I_ϵ and J_ϵ
EVALUATE
$h = -Z(Z^T \nabla^2 \tilde{f} Z)^{-1} Z \nabla \tilde{f}$
STOP
YES
YES
SET $d \leftarrow h$
$\| h \| > \beta$?
NO
$\| h \| < \epsilon_s$ .AND. $r^\infty < \epsilon_s$?
NO
SOLVE
$\mathbf{A}u = \nabla \tilde{f}$
SET $x \leftarrow x + h + v$
$\| u_j \| \leqslant \alpha_j$ $\forall j \in J_\epsilon$?
YES
SOLVE FOR v
NO
REFINE AND PERTURB
RE-EVALUATE $\nabla \tilde{f}$
SOLVE $\mathbf{A}u = \nabla \tilde{f}$
NO
$I_\epsilon \backslash J_\epsilon = \emptyset$?
$f(x+h+v) < f(x) - \delta(\| h \|^2 + r^\infty)$?
YES
YES
NO
SET $d = \tilde{h}_l = -P_l \bar{A}_l u_l$ WHERE $l \in I_\epsilon$, $\| u_l \| > \alpha_l$ AND $g^T h_l < -\delta$
NO
$g^T \tilde{h}_l \geqslant -\delta$ $\forall l \in I_\epsilon$ WITH $\| u_l \| > \alpha_l$?
YES
SET $\epsilon \leftarrow \epsilon/2$
$\beta \leftarrow \beta/2$
$d \leftarrow h$

MINIMIZATION FLOWCHART

to what are optimal (or even appropriate) values for the parameters ϵ and β is, by no means, trivial and goes beyond the scope of this paper. It should be noted however that the global convergence properties will be unaffected by this choice. The selection of the starting point is also left up to the user.

The linesearch used in our algorithm is fully described in the next section.

3.8 Line Search Algorithm

In this section we present our method for chosing the steplength λ whenever the line search is invoked in the minimization process.

As in most descent methods our criteria for accepting a steplength is based on convergence requirements (see, for example, [9] and [16]).

In order to ensure that the objective function "decreases sufficiently" with respect to the chosen steplength λ and direction d, we insist that the following condition be met:

$$f(x) - f(x + \lambda d) \geqslant -\mu * \lambda * d^T g(x) \tag{3.8.1}$$

where μ is a preassigned scalar in the range $0 < \mu < 1$. We also ensure that the chosen steplength is large enough by restricting our choice of candidates to those that satisfy the condition:

$$| d^T g(x + \lambda d)| < -\xi * d^T g(x) \tag{3.8.2}$$

where ξ is a preassigned scalar in the range $0 < \xi < 1$. This test also determines the accuracy to which the stepsize approximates the minimum along the line. (The optimal choice of the parameters μ and ξ is not obvious. In our current implementation we have had acceptable results with the values 0.1 and 0.9 respectively.)

Now that we have defined our acceptance criteria lets look at our method of generating trial steplengths.

If we let

$$\Phi'_{+/-}(\cdot) = \lim_{\Delta \to 0^{+/-}} \frac{\Phi(\cdot + \Delta) - \Phi(\cdot)}{\Delta}$$

then λ^* is a minimum of $f(x + \lambda d)$ only if $f'(x + \lambda^* d) = 0$ or λ^* is a derivative discontinuity of $f(x + \lambda d)$ with $f'_-(x + \lambda^* d) \leqslant 0$ and $f'_+(x + \lambda^* d) \geqslant 0$.

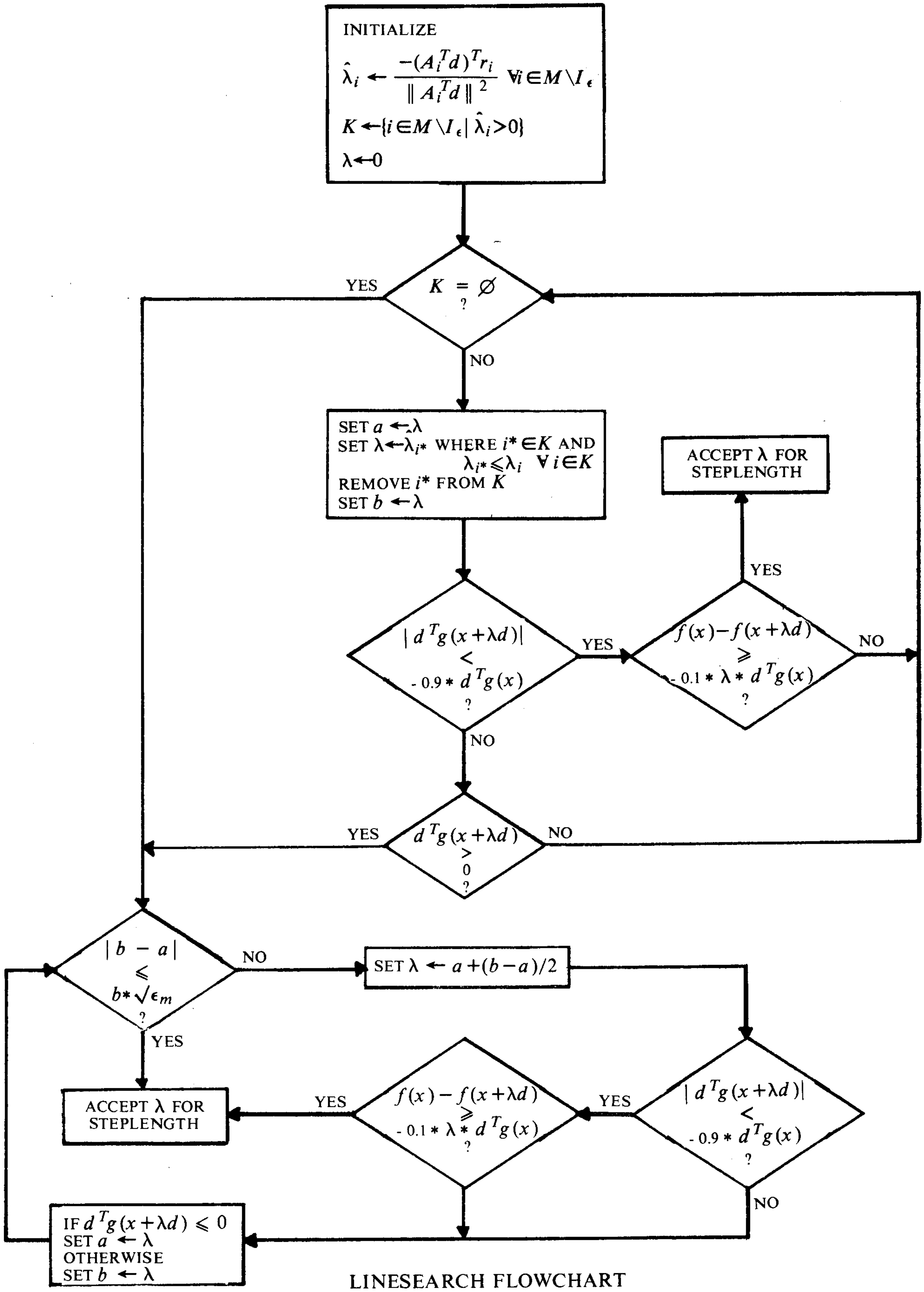
It can be shown that if derivative discontinuities exist along the direction d, then they occur at the values $\lambda = \lambda^*_i$ that exactly satisfy the equations

$$r_i(x + \lambda_i d) = r_i(x) + \lambda_i A_i^T d = 0 \quad i \in M \backslash I_\epsilon \tag{3.8.3}$$

(we exclude the set I_ϵ since $r_i(x + \lambda d) = r_i(x) \;\; \forall i \in I_\epsilon$). In addition, if $r_i(x + \bar{\lambda} d) = 0$ for some $i \in M \backslash I_\epsilon$ then $r_{i_-}'(x + \bar{\lambda} d) \leqslant 0$ and $r_{i_+}'(x + \bar{\lambda} d) \geqslant 0$.

Consider the set $K = \{ i \in M \backslash I_\epsilon | \hat{\lambda}_i > 0 \}$ where the $\hat{\lambda}_i$'s are the least-squares solutions to (3.8.3) (i.e. $\hat{\lambda}_i = -(A_i^T d)^T r_i(x) / \| A_i^T d \|^2 \;\; \forall i \in K$). Any number of these λ_i's may define derivative discontinuities of f along d; moreover, any number may satisfy our acceptance criteria. It therefore seems appropriate to consider, as trial steplengths, the values $\hat{\lambda}_i \; i \in K$. (It should be clear that $\lambda^* \in (0, \hat{\lambda}_{i_{\max}}]$ where $i_{\max} \in K$ and $\hat{\lambda}_{i_{\max}} \geqslant \hat{\lambda}_i \;\; \forall i \in K$.)

A flowchart of our linesearch algorithm follows. In this procedure we progress sequentially through a sorted list of the trial steplengths. If one of the trial steplengths satisfies the acceptance criteria, we use it. Otherwise, we perform an iterative bisection starting with the trial steplengths that most closely bound the minimum along the line. This bisection terminates whenever the interval of uncertainty becomes "small" or a bisection point satisfies the acceptance criteria. (The constant ϵ_M which appears in the line search flowchart is defined as the smallest number satisfying


INITIALIZE
$\hat{\lambda}_i \leftarrow \frac{-(A_i^T d)^T r_i}{\| A_i^T d \|^2} \ \forall i \in M \setminus I_\epsilon$
$K \leftarrow \{ i \in M \setminus I_\epsilon \mid \hat{\lambda}_i > 0 \}$
$\lambda \leftarrow 0$
$K = \varnothing$?
YES
NO
SET $a \leftarrow \lambda$
SET $\lambda \leftarrow \lambda_{i^*}$ WHERE $i^* \in K$ AND $\lambda_{i^*} \leqslant \lambda_i \quad \forall i \in K$
REMOVE i^* FROM K
SET $b \leftarrow \lambda$
$|d^T g(x+\lambda d)| < -0.9 * d^T g(x)$?
YES
NO
$f(x) - f(x+\lambda d) \geqslant -0.1 * \lambda * d^T g(x)$?
YES
NO
ACCEPT λ FOR STEPLENGTH
$d^T g(x+\lambda d) > 0$?
YES
NO
$|b - a| \leqslant b * \sqrt{\epsilon_m}$?
NO
YES
SET $\lambda \leftarrow a + (b-a)/2$
$|d^T g(x+\lambda d)| < -0.9 * d^T g(x)$?
YES
NO
$f(x) - f(x+\lambda d) \geqslant -0.1 * \lambda * d^T g(x)$?
YES
ACCEPT λ FOR STEPLENGTH
IF $d^T g(x+\lambda d) \leqslant 0$
SET $a \leftarrow \lambda$
OTHERWISE
SET $b \leftarrow \lambda$

LINESEARCH FLOWCHART

$1 \oplus \epsilon_M > 1$ where $\oplus$ represents floating point addition.)

There are several variants of this linesearch which are currently under investigation by the authors. The most noteworthy involves using, as trial steplengths, only those λ_i's which "closely" approximate discontinuities along the line (this can be easily determined by computing the values $\| r_i(x + \lambda_i d) \|$ to see if they are "small"). If one of these trial steplengths is acceptable, we use it. Otherwise, we bracket the minimum and use an appropriate method to estimate the minimum.

3.9 Extensions

In the foregoing sections of this paper we presented a second-order method for solving the continuous unconstrained multifacility location problem involving Euclidean distances. Here we present some of the details for extending this technique to a wider class of location problems.

First, we consider the linearly constrained location problem:

$$\text{minimize } f(x) = \sum_{i \in M} \| A_i^T x - b_i \| \tag{3.9.1}$$

such that

$$r_i(x) = a_i^T x - b_i \geqslant 0 \quad i \in LI$$
$$r_i(x) = a_i^T x - b_i = 0 \quad i \in LE.$$

If we transform problem (3.9.1) into the unconstrained problem,

$$\text{minimize } F(x, \mu) = \mu f(x) - \sum_{i \in LI} \min\,[0,\, r_i(x)] + \sum_{i \in LI} | r_i(x) | \tag{3.9.2}$$

where μ is a positive parameter, then the second-order method we have described can be easily modified to solve problem (3.9.2). Details of the necessary modifications are presented in [2].

As with these constrained problems there are many practical justifications for considering l_p distance location problems (see, for example, [13]). Fortunately, many of the results we have presented are valid for the problem:

$$\text{minimize } f(x) = \sum_{i \in M} \| A_i^T x - b_i \|_p \quad 1 \leqslant p \leqslant \infty \tag{3.9.3}$$

where

$\| \cdot \|_p$ is the general l_p norm, and

$$\| x \|_p = \begin{cases} \left(\sum_{c=1}^{q} | x_c |^p \right)^{1/p} & 1 \leqslant p < \infty \\ \max \{ | x_1 |, \ldots, | x_q | \} & p = \infty . \end{cases}$$

Only a few modifications become necessary when we consider solving this problem using our second-order technique when $1 < p < \infty$ or our first-order technique [2] when $p = 1$ or $p = \infty$. These modifications revolve around the following optimality conditions:

The point x^* is a minimum for problem (3.9.3) if and only if

i) $P \nabla \tilde{f}(x^*) = 0$

ii) $I_1(x^*) = \emptyset$

iii) $\| u_i^* \|_{p'} \leqslant \alpha_i \quad \forall\, i \in I_\epsilon(x^*)$

where

$$p' = \begin{cases} \infty & p = 1, \\ \dfrac{p}{1-p} & 1 < p < \infty, \\ 1 & p = \infty. \end{cases}$$

Thus the minimization strategy for solving problem (3.9.3) involves taking a step in the direction h whenever $\| h \|_2 > \beta$ (CASE 1 § 3.7); attempting the step $h + v$ whenever $\| h \|_2 \leqslant \beta$ and $\| u_i \|_{p'} \leqslant \alpha_i \;\; \forall i \in I_\epsilon$ (CASE 2 § 3.7); and attempting a step in the direction $\tilde{h}_l$ whenever $\| h \|_2 \leqslant \beta$ and $\| u_l \|_{p'} > \alpha_l \;\; l \in I_\epsilon$ (CASE 3 § 3.7). In this latter instance, $\tilde{h}_l$ is defined as follows:

$$\tilde{h}_l = -\gamma P_l \bar{A}_l \hat{u}_l$$

where, for $1 < p \leqslant \infty$,

$$[\hat{u}_l]_c = \operatorname{sgn}[u_l]_c * | [u_l]_c |^{p'/p} \qquad c = 1, \ldots, q$$

and, for $p = 1$,

$$[\hat{u}_l]_c = \begin{cases} \operatorname{sgn}[u_l]_c & c = c^* \\ 0 & \text{otherwise} \end{cases}$$

where c^* is the index to the component of the vector u_l that is largest in magnitude (i.e. $| [u_l]_{c^*} | \geqslant | [u_l]_c | \quad c = 1, \ldots, q$).

As a consequence of this last extension we are also able to solve the continuous multifacility location problem where the distances are combinations of the l_p norms (see [18]). This problem has the form

$$\text{minimize } f(x) = \sum_{k \in K} \sum_{i \in M(k)} \| A_i^T x - b_i \|_{p(k)} \tag{3.9.4}$$

where

$$K = \{k_1, \ldots, k_L\} \qquad L < \infty$$

$$\bigcup_{j=1}^{L} M(k_j) = M$$

$$M(k_j) \cap M(k_{j'}) = \varnothing \qquad j' \neq j \qquad k_j, k_{j'} \in K$$

and

$$1 \leqslant p(k) \leqslant \infty \qquad \forall k \in K$$

Although this particular problem has received very little attention in the literature it is quite obvious that different norms may appear simultaneously in many practical facility location problems.

4. IMPLEMENTATION

4.1 Concepts and Definitions

To explain the implementation of our method we present some basic ideas about graph theory and some additional definitions .

Consider a graph consisting of n vertices having a one-to-one correspondence to the n new facilities (i.e. vertex j corresponds to $NF_j, j = 1, \ldots, n$). Edge γ_{jk}, $1 \leqslant j < k \leqslant n$, is found between vertex j and vertex k if there is an interaction between NF_j and NF_k (i.e. $v_{jk} \neq 0$). A set of edges $\{\gamma_{jk}\}$ form a tree if

a) the edges generate a connected subgraph

b) the edges contain no cycles.

A subgraph is connected if there is a chain joining every pair of distinct vertices in the subgraph. If we consider any sequence of vertices, say $i_1, \ldots, i_l$, then a possible chain would consist of the sequence of edges $\gamma_{i_1 i_2}, \gamma_{i_2 i_3}, \ldots, \gamma_{i_{l-1} i_l}$. The initial vertex of this chain would be i_1 and the terminal vertex would be i_l. A cycle is a chain whose initial vertex and terminal vertex are identical. For example, the sequence of edges $\gamma_{13}, \gamma_{34}, \gamma_{41}$ form a cycle.

Finally let E_j be the $qn \times q$ matrix defined by

$$E_j^T = [\, 0_{j-1} \quad I_q \quad 0_{n-j} \,]$$

where

0_k is a zero matrix of dimension $q \times qk$

and

I_q is a $q \times q$ identity matrix,

and let the sets $J(*)$, $-\lfloor n/2 \rfloor \leqslant * \leqslant \lfloor n/2 \rfloor$, and K^* be defined as

$$J(*) = \{j \mid \text{TREE}(j) = *\} = \{\beta_1, \dots, \beta_{J_*}\}$$

and

$$K^* = \{k \mid J(k) \neq \emptyset\}$$

where

$$\beta_i = \beta_i(*) \qquad i = 1, \dots, J_*$$

$$\beta_1 < \beta_2 < \cdots < \beta_{J_*}$$

$$J_* = |\, J(*) \,|$$

and where the vector TREE is defined in § 4.2.

4.2 Active Trees and Masking

Each term $\| r_i \|$ $i \in M$ in our objective function represents either an interaction between two new facilities ($i \leqslant \eta$) or an interaction between a new facility and some existing facility ($\eta < i \leqslant \tau$). In the former instance the interaction between the two *NF*'s is represented by the edge in our graph joining the two *NF* vertices. Those two *NF*'s are included as vertices in an *active* tree under the following conditions:

a) $\| r_i \| \leqslant \epsilon$

b) the inclusion of the edge does not form a cycle

c) both *NF*'s are not *masked* (this term is defined below).

When the objective function term involves the interaction between a *NF* and some *EF* we *mask* the *NF* vertex down 1 under the following conditions:

a) $\| r_i \| \leqslant \epsilon$

b) the vertex is not already masked.

In both instances the first condition implies that the corresponding objective function term is nondifferentiable (or near-nondifferentiable) whereas the remaining conditions detect degeneracies.

Now suppose that at each stage in our minimization process we have the n-vector TREE whose i-th element is set according to the following rules:

a) TREE(i) ← 0 if and only if NF_i is not a vertex in any active tree and is not masked by any *EF*.

b) TREE(i) ← k if and only if NF_i is a vertex in the k-th active tree and no *NF* in that tree is masked by any *EF*.

c) TREE(i) $\leftarrow -k$ if and only if NF_i is a vertex in the k-th active tree and some NF in that tree is masked by some EF.

d) TREE(i) $\leftarrow -\lfloor n/2 \rfloor$-1 if and only if NF_i is not a vertex in any active tree but is masked by some EF.

(Since there can be no more than $\lfloor n/2 \rfloor$ distinct trees in a graph with n vertices we satisfy $0 \leqslant k \leqslant \lfloor n/2 \rfloor$ in the foregoing definitions.)

As we shall see in subsequent sections, once we have the n-vector TREE we can very easily:

1) Identify I_ϵ,
2) Form a basis **A** for A_i, $i \in I_\epsilon$,
3) Construct Z.

The method used for constructing the n-vector TREE is a simple adaptation of the classical Spanning Tree Algorithm (see [14]). What follows is a flowchart for this construction. In this flowchart (and in subsequent subsections) i^*,j^* and k^* are defined as they were in § 2.

4.3 Identifying I_ϵ

For each term $\| r_i \|$, $i = 1, \ldots, \tau$ in our objective function we have defined the values j^* and k^* (for $i \leqslant \eta$) and j^* and i^* (for $\eta < i \leqslant \tau$). We add the index i to the set I_ϵ under either of the following two conditions:

$$\text{If} \quad i \leqslant \eta \quad \text{and} \quad \text{TREE}(j^*) = \text{TREE}(k^*) \neq 0 \quad \text{or} \quad \text{TREE}(j^*) < 0 \text{ and } \text{TREE}(k^*) < 0\,. \tag{4.3.1}$$

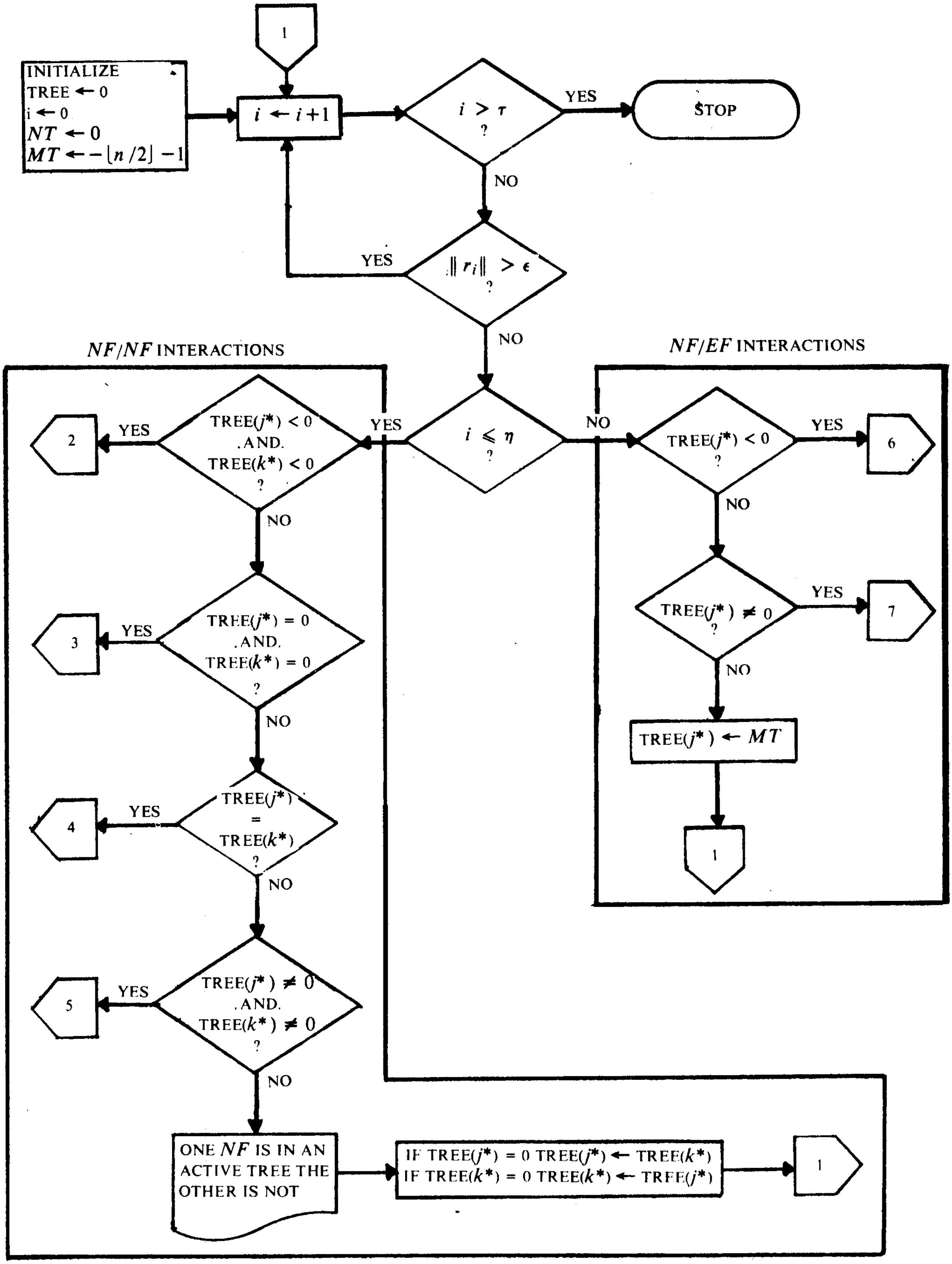
$$\text{If} \quad \eta < i \leqslant \tau \quad \text{and} \quad \text{TREE}(j^*) < 0\,. \tag{4.3.2}$$

A brief examination of the TREE VECTOR FLOWCHART should convince the reader that all terms $\| r_i \|$, $i = 1, \ldots, \tau$ satisfying the inequality $\| r_i \| \leqslant \epsilon$ will also satisfy (4.3.1) or (4.3.2). However, there can be indices $i \in I_\epsilon$ that do not satisfy this inequality. Consider, for instance, the following situation:

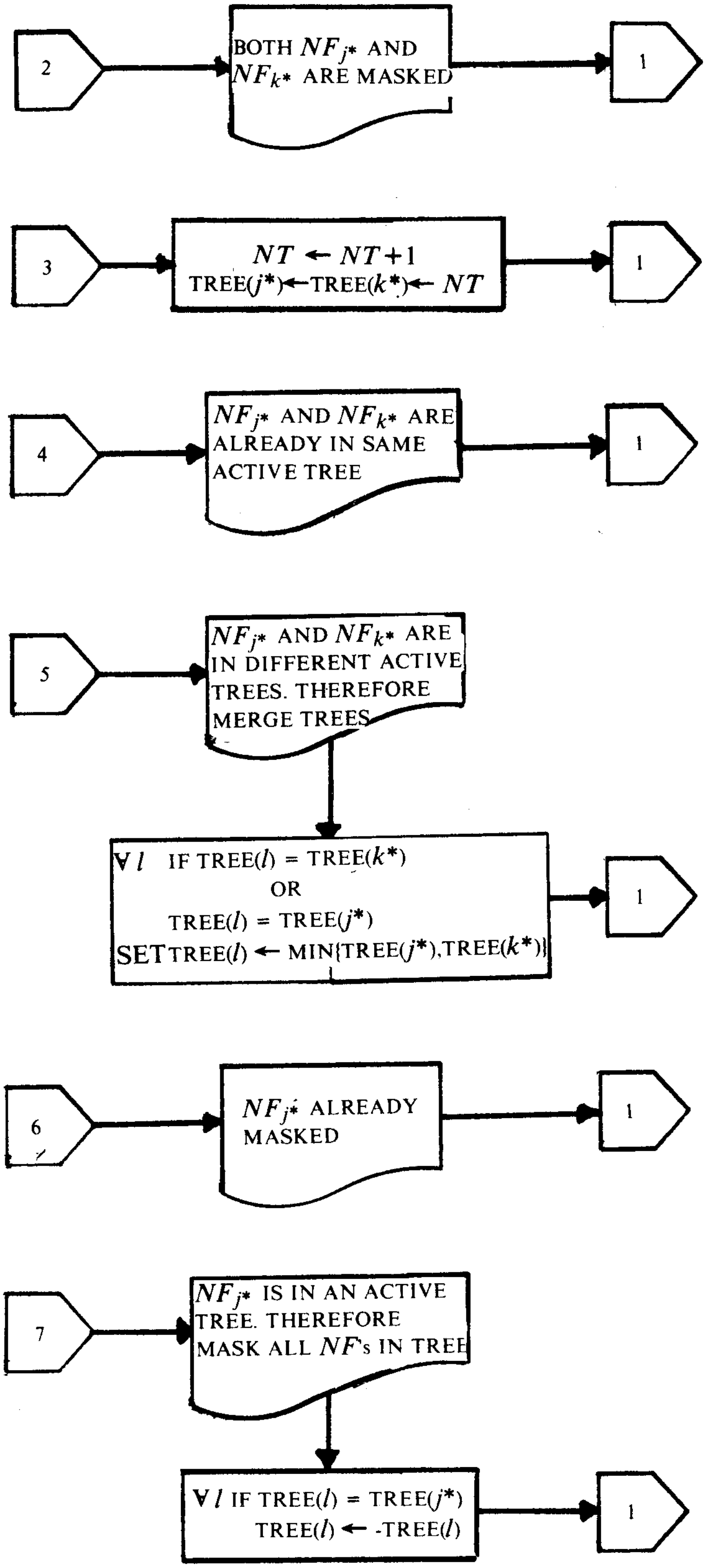
Example 1: $n = 2$, $m = 1$, $q = 2$, $p = 2$. $v_{12} = w_{11} = w_{21} = 1$. $x_1^T = \{\epsilon, 0\}$, $x_2^T = \{0, \epsilon\}$, $p_1^T = \{0, 0\}$. We have $\| r_1 \| = \sqrt{2} * \epsilon$ and $\| r_2 \| = \| r_3 \| = \epsilon$. We also have TREE(1) = TREE(2) = -2. Now, for $i = 1$, $j^* = 1$, $k^* = 2$ and TREE(j^*) = TREE(k^*) $\neq 0$. Therefore, according to (4.3.1), the index $i = 1$ is added to I_ϵ even though $\| r_1 \| > \epsilon$.

The reasoning for this apparent inconsistency is simple. Consider Example 1 again. Since $\| r_2 \| \leqslant \epsilon$ and $\| r_3 \| \leqslant \epsilon$ index 2 and 3 belong in I_ϵ. The fact that $\overline{A}_i^T d = 0 \; \forall i \in I_\epsilon$ guarantees that $\overline{A}_1^T d = 0$ since $\overline{A}_1 = \overline{A}_2 - \overline{A}_3$. Thus index 1 can be added to I_ϵ without increasing the rank or complicating the construction of **A** (see § 4.4). In addition, whenever a facility is in the "neighbourhood" of another facility we have a term that is potentially nondifferentiable and whose index may therefore belong in I_ϵ. Since x_1 and x_2 are both in ϵ-neighbourhoods of P_1 (i.e. $\| r_2 \| = \| r_3 \| = \epsilon$) indices 2 and 3 are added to I_ϵ. In making this decision we, in effect, "pretend" that x_1 and x_2 coincide with P_1. Under this pretension, x_1 must coincide with x_2 (making $\| r_1 \| = 0$). We may therefore include index 1 in I_ϵ.

It can be shown that $\forall i \in I_\epsilon$ either:


1
INITIALIZE
TREE ← 0
i ← 0
$NT \leftarrow 0$
$MT \leftarrow -\lfloor n/2 \rfloor - 1$
$i \leftarrow i+1$
$i > \tau$?
YES
STOP
NO
$\| r_i \| > \epsilon$?
YES
NO
NF/NF INTERACTIONS
NF/EF INTERACTIONS
$i \leqslant \eta$?
YES
NO
TREE(j^*) < 0 .AND. TREE(k^*) < 0 ?
YES
2
NO
TREE(j^*) = 0 .AND. TREE(k^*) = 0 ?
YES
3
NO
TREE(j^*) = TREE(k^*) ?
YES
4
NO
TREE(j^*) ≠ 0 .AND. TREE(k^*) ≠ 0 ?
YES
5
NO
ONE NF IS IN AN ACTIVE TREE THE OTHER IS NOT
IF TREE(j^*) = 0 TREE(j^*) ← TREE(k^*)
IF TREE(k^*) = 0 TREE(k^*) ← TREE(j^*)
1
TREE(j^*) < 0 ?
YES
6
NO
TREE(j^*) ≠ 0 ?
YES
7
NO
TREE(j^*) ← MT
1

TREE VECTOR FLOWCHART


2
BOTH NF_{j^*} AND NF_{k^*} ARE MASKED
1
3
$NT \leftarrow NT+1$
TREE(j^*)←TREE(k^*)← NT
1
4
NF_{j^*} AND NF_{k^*} ARE ALREADY IN SAME ACTIVE TREE
1
5
NF_{j^*} AND NF_{k^*} ARE IN DIFFERENT ACTIVE TREES. THEREFORE MERGE TREES
∀ l IF TREE(l) = TREE(k^*)
OR
TREE(l) = TREE(j^*)
SET TREE(l) ← MIN{TREE(j^*),TREE(k^*)}
1
6
NF_{j^*} ALREADY MASKED
1
7
NF_{j^*} IS IN AN ACTIVE TREE. THEREFORE MASK ALL NF's IN TREE
∀ l IF TREE(l) = TREE(j^*)
TREE(l) ← -TREE(l)
1

TREE VECTOR FLOWCHART (continued)

a) $\| r_i \| \leqslant \epsilon$

or b) there exists vectors y_i such that $\overline{A}_i = \sum_{j \in I_\epsilon^*} \overline{A}_j y_j$ where $I_\epsilon^* = \{ i \in I_\epsilon | \; \| r_i \| \leqslant \epsilon \}$.

4.4 Constructing A

What follows is our method for constructing the matrix **A** which is a basis for the column space of the matrices $\overline{A}_i \;\; \forall i \in I_\epsilon$. This method takes full advantage of the structure of this problem and its graphical interpretation.

For $j = 1, ..., n$, the j-th step of the construction process is as follows:

a) If TREE(j) = 0 no contribution is made to **A**.

b) If TREE(j) < 0 augment **A** with the matrix E_j.

c) If TREE(j) = l , $1 \leqslant l \leqslant \lfloor n/2 \rfloor$, then let $k > j$ be the next index satisfying TREE(k) = TREE(j). If no such index exists then proceed to the next step in the construction process, otherwise augment **A** with the matrix $[E_j - E_k]$.

We can now prove that **A** is a basis for the space spanned by the columns of $\overline{A}_i \;\; \forall i \in I_\epsilon$ (i.e. $\forall i \in I_\epsilon \;\; \overline{A}_i \in C(\mathbf{A})$ and **A** is full rank.)

Proof that $\overline{A}_i \in C(\mathbf{A}) \;\; \forall i \in I_\epsilon$

For $i \in I_\epsilon^* \;\; \| r_i \| \leqslant \epsilon$ and either

1) : $i \leqslant \eta$ and TREE(j^*) = TREE(k^*) > 0 .

or 2) : $i \leqslant \eta$ and TREE(j^*) < 0 and TREE(k^*) < 0 .

or 3) : $i > \eta$ and TREE(j^*) < 0 .

For case 1), assume that TREE(j^*) = TREE(k*) = l and let $\{j_1, j_2, \cdots, j_L\} = \{ j \mid \text{TREE}(j) = l, j^* \leqslant j \leqslant k^* \}$ where $j_1 < j_2 < \cdots < j_L$.

Thus

$$\begin{aligned} \overline{A}_i &= E_{j_1} - E_{j_L} \\ &= [E_{j_1} - E_{j_2}] + [E_{j_2} - E_{j_3}] + \cdots + [E_{j_{L-1}} - E_{j_L}]. \end{aligned}$$

But $[E_{j_1} - E_{j_2}], [E_{j_2} - E_{j_3}], \ldots, [E_{j_{L-1}} - E_{j_L}]$ are the matrices that were augmented to **A** in its construction process when $j = j_1, \ldots, j_{L-1}$ respectively. Therefore $\overline{A}_i \in C(\mathbf{A})$.

In case 2), $\overline{A}_i = E_{j^*} - E_{k^*}$. But E_{j^*} and E_{k^*} are the matrices augmented to **A** in its construction process when $j = j^*$ and $j = k^*$ respectively. Therefore $\overline{A}_i \in C(\mathbf{A})$.

For case 3), $\overline{A}_i = E_{j^*}$. But E_{j^*} is the matrix augmented to **A** in its construction process when $j = j^*$. Thus $\overline{A}_i \in C(\mathbf{A})$.

We have shown that $\forall i \in I_\epsilon^* \; \overline{A}_i \in C(\mathbf{A})$. This, in turn, guarantees that $\overline{A}_i \in C(\mathbf{A}) \;\; \forall i \in I_\epsilon$ since, for $i \in I_\epsilon \backslash I_\epsilon^* \;\; \overline{A}_i = \sum_{j \in I_\epsilon^*} \overline{A}_j y_j$ for some vectors y_j.

Proof that A is full rank

At the j-th step in the construction process there are, at most, $q(j-1)$ columns in **A**. Any augmentation to **A** at this j-th step involves the matrix E_j or the matrix $[E_j - E_k]$ both of which have zeros in rows 1 through $q(j-1)$ and a $q \times q$ identity matrix in the next q rows. Thus **A** is full rank (and lower-trapezoidal) and rank(**A**) = $q * (n_1 + n_2)$ where

$n_1 \underline{\Delta}$ number of NF's that are masked but not in active trees, and

$n_2 \underline{\Delta}$ number of NF's in active trees minus the number of unmasked active trees.

4.5 Constructing Z

The matrix Z, defined by (3.3.2) and (3.3.3), is an orthonormal basis for the null space of the matrix $\mathbf{A}$ (i.e. $\mathbf{A}^T Z = 0$ and $Z^T Z = I_{q(n-t)}$ where rank($\mathbf{A}$) $= qt = q(n_1 + n_2)$). In [2] it is suggested that Z be computed using the QR factorization of $\mathbf{A}$. However, the structure of the problem allows a much more efficient method for constructing Z.

We construct the matrix Z, using the information in the vector TREE, as follows:

Step 1. If there are no zero entries in the vector TREE proceed to Step 2; otherwise, $\forall k \in \{j \mid \text{TREE}(j) = 0\}$ augment Z with the matrix E_k.

Step 2. If there are no strictly positive entries in the vector TREE, the construction process is complete; otherwise, for $l = 1, \ldots, \lfloor n/2 \rfloor$ if $J(l) \neq \emptyset$ augment Z with the matrix $\frac{1}{\sqrt{J_l}} \sum_{j \in J(l)} E_j$.

With this construction process in mind we can prove that the matrix Z is, in fact, a basis for the null space of $\mathbf{A}$. (The proof that Z is orthonormal is trivial.)

Proof that $A^T Z = 0$ and rank$(Z) = q(n-t)$

The matrix Z has only zero entries in rows $q(j-1)+1$ through qj when TREE(j) < 0. Therefore $E_j^T Z = 0$ for all indices j satisfying TREE(j) < 0. Similarily, there are only q columns in Z with nonzero entries in rows $q(j-1)+1$ through qj or $q(k-1)+1$ through qk when TREE(j) = TREE(k) = $l > 0$. These nonzero entries are all in the matrix $\frac{1}{\sqrt{J_l}} \sum_{j \in J(l)} E_j$ that was augmented to Z in step 2 of its construction process. Therefore $(E_j - E_k)^T Z = 0$ for all indices j and k satisfying TREE(j) = TREE(k) = $l > 0$. Since $\mathbf{A}$ is composed only of submatrices E_j, where TREE(j) < 0 , and $(E_j - E_k)$, where TREE(j) = TREE(k) = $l > 0$, we have $\mathbf{A}^T Z = 0$.

Each matrix augmenting Z in the construction process has rank q and no two such matrices have nonzero entries in any of the same row positions, therefore

$$
\begin{aligned}
\text{rank}(Z) &= q * \text{number of matrices augmented to } Z \\
&= q * (n_3 + n_4)
\end{aligned}
$$

where

$n_3 \underline{\Delta}$ number of NFs that are unmasked and do not belong to any active trees, and

$n_4 \underline{\Delta}$ number of unmasked active trees.

But

$$
\begin{aligned}
(n_3 + n_4) &= n - (n_1 + n_2) \\
&= n - t.
\end{aligned}
$$

Therefore rank$(Z) = q(n-t)$.

4.6 The Direction $\tilde{h}_l$

Our minimization algorithm sometimes requires an iteration using the direction

$$\tilde{h}_l = -\gamma P_l \bar{A}_l u_l. \tag{4.6.1}$$

As we shall see below, the vector TREE is easily modified to indicate index l no longer belongs in the set I_ϵ (i.e. the columns of $\bar{A}_l$ are to be dropped from the matrix **A**). Once this is accomplished the matrix Z can be reconstructed. Since Z then forms an orthogonal basis for the space $S_l^{\perp}$, the projection matrix P_l can be computed as

$$P_l = Z Z^t \tag{4.6.2}$$

and the direction $\tilde{h}_l$ can be formed.

In section 4.4. we learned that the matrix $\bar{A}_l$ was constructed from either the matrix E_j where $1 \leqslant j \leqslant n$ or the matrix $(E_j - E_k)$, where $1 \leqslant j < k \leqslant n$. With this thought in mind the vector TREE is modified as follows:

1) If $\bar{A}_l$ is constructed from E_j then set TREE(j) ← 0.
2) If $\bar{A}_l$ is constructed from $[E_j - E_k]$ then let $l^* \in \{l \mid 1 \leqslant l \leqslant \lfloor n/2 \rfloor, l \notin K^*, -l \notin K^*\}$ and set TREE(i) ← l^* for all $i \geqslant k$ with TREE(i) = TREE(j).

4.7 The Second-Order Direction

The direction h which is obtained by finding the solution $w = w^*$ to

$$Z^T \nabla^2 \tilde{f} Z w = -Z^T \nabla \tilde{f} \tag{4.7.1}$$

and then setting

$$h = Z w^* \tag{4.7.2}$$

is called the projected Newton direction (ie h is the Newton step to the minimum of problem (3.3.1)). The matrix $Z^T \nabla^2 \tilde{f} Z$ and the vector $Z^T \nabla \tilde{f}$ are respectively called the projected Hessian and the projected gradient.

Since $\tilde{f}$ is convex, $Z^T \nabla^2 \tilde{f} Z$ is positive semi-definite. When $Z^T \nabla^2 \tilde{f} Z$ is not positive definite or when it is positive definite but very ill-conditioned, we cannot (stably) solve (4.7.1). We can, however, apply a numerically stable modified Cholesky factorization to $Z^T \nabla^2 \tilde{f} Z$ to obtain the system

$$L D L^T w = -Z^T \nabla \tilde{f} \tag{4.7.3}$$

where $LDL^T = Z^T \nabla^2 \tilde{f} Z + E$, L is a lower-triangular matrix, D is a diagonal matrix and E is a diagonal matrix that is zero when $Z^T \nabla^2 \tilde{f} Z$ is "sufficiently" positive definite. The solution $w = w^*$ to (4.7.3) can then be computed, using a forward and backward substitution, and used to obtain the modified projected Newton direction $h = Z w^*$. (The reader is referred to [10] for a complete description of the modified Cholesky factorization.)

It should be noted that at each stage in the minimization process we use the true projected Hessian and not some approximation to it (it is possible to use some Quasi-Newton approach). This choice is justifiable since the construction of Z and the computation of $\nabla^2 \tilde{f}$ are relatively inexpensive.

4.8 The Refinement Step

In section 3.5 we showed that the refinement step v could be obtained by solving the linear system

$$\left[\begin{array}{c} \mathbf{A}^T \\ \hline Z^T \end{array}\right] v = \left[\begin{array}{c} -\bar{r}(x+h) \\ \hline 0 \end{array}\right]. \tag{4.8.1}$$

As we shall soon see, solving this system is a trivial process as a result of its structure.

If we let $v^T = \{v_1^T \cdots v_n^T\}$ and $\bar{r}(x+h)^T = \{\bar{r}_1^T \cdots \bar{r}_t^T\}$ where v_j^T and $\bar{r}_i^T$ are $1 \times q$ vectors, then we can solve for v as follows:

Step 1. If $J(l) = \emptyset$ for $l = -1, \ldots, -\lfloor n/2 \rfloor$ then go to Step 2; otherwise, $\forall j \in J(l)$, $l = -1, \ldots, -\lfloor n/2 \rfloor$, the matrix E_j augmented **A** in its construction (see § 4.4). If E_j was the ξ_j^{th} matrix to augment **A** then, from (4.8.1), we have $E_j^T v = v_j = -\bar{r}_{\xi_j}$.

Step 2. If $J(0) = \emptyset$ go to step 3; otherwise, for all $j \in J(0)$ the matrix E_j augmented Z in its construction (see § 4.5). Then, from (4.8.1), we have

$$E_j^T v = v_j = 0.$$

Step 3. For $l = 1, \ldots, \lfloor n/2 \rfloor$, if $J(l) \neq \emptyset$ then the matrix $\frac{1}{\sqrt{J_l}} \sum_{j \in J(l)} E_j$ augmented Z in its construction (see § 4.5) and the matrices $[E_{\beta_1} - E_{\beta_2}], \ldots, [E_{\beta_{J_l-1}} - E_{\beta_{J_l}}]$ augmented **A** in its construction (see § 4.4). If, for $i = 1, \ldots, J_l - 1$, the matrix $[E_{\beta_i} - E_{\beta_{i+1}}]$ was the $\xi_{\beta_i}^{\text{th}}$ matrix to augment **A** then, from 4.8.1, we have

$$\begin{bmatrix} E_{\beta_1}{}^T - E_{\beta_2}{}^T \\ \vdots \\ E_{\beta_{J_l-1}}{}^T - E_{\beta_{J_l}}{}^T \\ E_{\beta_1}{}^T + \cdots + E_{\beta_{J_l}}{}^T \end{bmatrix} v = \begin{bmatrix} -\bar{r}_{\xi_{\beta_1}} \\ \vdots \\ -\bar{r}_{\xi_{\beta_{J_l-1}}} \\ 0 \end{bmatrix}$$

or, equivalently,

$$v_{\beta_{J_l}} = \left[\bar{r}_{\xi_{\beta_1}} + 2\bar{r}_{\xi_{\beta_2}} + \cdots + (J_l - 1)\bar{r}_{\xi_{\beta_{J_l-1}}}\right] / J_l$$

and

$$v_{\beta_i} = v_{\beta_{i+1}} - \bar{r}_{\xi_{\beta_i}} \qquad i = J_l - 1, \ldots, 1.$$

4.9 The Dual Estimates

In our minimization algorithm we sometimes compute the least-squares solution to the system

$$\mathbf{A}u = \nabla \tilde{f}. \tag{4.9.1}$$

If we factor **A** into the product

$$Q^T \mathbf{A} = P \begin{bmatrix} R \\ 0 \end{bmatrix}, \tag{4.9.2}$$

where Q^T is an orthonormal matrix, P is a permutation matrix and R is an upper-triangular matrix, then this least-squares solution is efficiently obtained by solving the system

$$P\begin{bmatrix}R\\0\end{bmatrix}u = Q^T\nabla\tilde{f} . \tag{4.9.3}$$

To describe the factorization given by (4.9.2), let the Givens reflection matrices $\overline{G}_i$, $i = 1, 2, \cdots$ be defined as

$$\overline{G}_i = \begin{bmatrix} c_iI_q & s_iI_q \\ s_iI_q & -c_iI_q \end{bmatrix}$$

where

$$c_0 = 0, \qquad s_0 = 1$$

$$c_i = \frac{1}{\surd(1+s_{i-1}^2)}, \quad s_i = \frac{-s_{i-1}}{\surd(1+s_{i-1}^2)} \quad i = 1, 2, \ldots,$$

and let the matrix $G_i(j,k)$ be the $qn \times qn$ matrix obtained by imbedding $\overline{G}_i$ in the qn-dimensional identity matrix as follows:

$$G_i(j, k) = \begin{array}{c} \\ 1 \\ \\ \\ j \\ \\ \\ \\ k \\ \\ \\ n \end{array}\!\!\begin{array}{cccccccc} 1 & & j & & & k & & n \\ \left[\begin{matrix} I_q & & & & & & & \\ & \ddots & & & & & & \\ & I_q & & & & & & \\ & & c_iI_q & & & s_iI_q & & \\ & & & I_q & & & & \\ & & & & \ddots & & & \\ & & & & & I_q & & \\ & & s_iI_q & & & -c_iI_q & & \\ & & & & & & I_q & \\ & & & & & & \ddots & \\ & & & & & & & I_q \end{matrix}\right] \end{array}$$

We then take

$$Q^T = \prod_{\substack{k \in K^* \\ k>0}} \prod_{i=1}^{J_k-1} G_{J_k-1}(\beta_i, \beta_{J_k})$$

where K^*, J_k and β_i are as defined in § 4.1.

The effect of premultiplying **A** by Q^T is equivalent to having augmented **A** with the matrices

$$\left[\left(c_{J_k-1} - s_{J_k-1}s_{J_k-2}\right)E_{\beta_1} - c_{J_k-2}E_{\beta_2}\right], \ldots, \left[\left(c_1 - s_1s_0\right)E_{\beta_{J_k-1}} - c_0E_{\beta_{J_k}}\right] \tag{4.9.4}$$

instead of the matrices

$$\left[E_{\beta_1} - E_{\beta_2}\right], \ldots, \left[E_{\beta_{J_k-1}} - E_{\beta_{J_k}}\right] \tag{4.9.5}$$

when $j = \beta_1, \beta_2, \ldots, \beta_{J_k - 1}$ in **A**'s construction (see § 4.4). Thus we form $Q^T\mathbf{A}$ or more appropriately, $P\begin{bmatrix} R \\ 0 \end{bmatrix}$ by transforming **A** using the relationships given by (4.9.4) and (4.9.5) for all $k \in K^*$, $k > 0$.

If we compare (4.9.4) and (4.9.5) it becomes clear that there is no fill-in whatsoever when **A** is transformed in the described manner. However, since $c_0 E_{\beta_{J_k}} = 0 \; k \in K^* \; k > 0$, each augmentation of the matrices given by (4.9.4) results in q zero rows replacing q nonzero rows of **A** (These q nonzero rows of **A** contained, for their nonzero entries, the matrix $-I_q$ which resulted from augmenting **A** with the matrix $\left[E_{\beta_{J_k - 1}} - E_{\beta_{J_k}}\right]$.) It is the introduction of these zero rows that allows us to form the factorization given by (4.9.2).

As a result of the manner in which **A** is formed (§ 4.4) and transformed (by premultiplication by Q^T) the upper-triangular matrix **R** is obtained by simply disregarding the zero rows of $Q^T\mathbf{A}$ (ie the permutation matrix P does not re-order the rows of **R**). Therefore, once $Q^T\nabla f$ is computed we can solve for the dual estimate u in (4.9.3) by simple forward-substitution.

If we let $\nabla \tilde{f} = [f_1^T \cdots f_n^T]^T$ and $Q^T \nabla \tilde{f} = [\bar{f}_1^T \cdots \bar{f}_n^T]^T$, where f_i and $\bar{f}_i$ are $q \times 1$ vectors, then $Q^T\nabla f$ is obtained by performing the following algorithm:

```
SET $\bar{f}_i \leftarrow f_i \quad i = 1, \ldots, n.$
DO  for all $k \in K^*, \quad k > 0$
      DO for  $i = 1, \ldots, J_K - 1$
            TEMP $\leftarrow c_i \bar{f}_{\beta_{J_k - i}} + s_i \bar{f}_{\beta_{J_k}}$
            $\bar{f}_{\beta_{J_k}} \leftarrow s_i \bar{f}_{\beta_{J_k - i}} - c_i \bar{f}_{\beta_{J_k}}$
            $\bar{f}_{\beta_{J_k - i}} \leftarrow$ TEMP
```

5. PRELIMINARY NUMERICAL RESULTS

In this section we provide a cursory comparison between the performance of the projected Newton method (PNM) described in this paper, the hyperboloid approximation procedure (HAP [6]) and a projected steepest descent method (PSDM [2]). These three algorithms were implemented in FORTRAN on a Honeywell 66/60 using single precision arithmetic.

Six small problems were run as a basis for this comparison. The first three problems are given in [8] (as exercises #5.23, #5.6 and #5.7 respectively), the fourth is reported in [6] and the last two appear in [1].

The results of these test runs are summarized in Table 1. Except for the last row, the figures in the table refer to the number of iterations required to reach the solution.

In the last row an estimate of the total number of addition operations (in units of one thousand) required in solving the six problems is given (approximately the same number of multiplications would be required). The reader should note that the number of addition operations quoted in [2], for the projected steepest descent method, is greater than the number quoted here. This is because the structure of the problem was not taken into account when this method was originally implemented and tested.

In all problems, except #5, the projected Newton method outperformed the other methods in terms of both the number of iterations and the number of addition operations. The performance on problem #5 could be improved by an alternate choice of the free parameters (for all six problems the free parameters for PNM were set as follows: $\epsilon = 10^{-1}, \beta = 10^{-2}, \delta_0 = 10^{-5}, \epsilon_s = 10^{-8}$ and $\epsilon_M = 7.45 \times 10^{-9}$).

#	HAP	PSDM	PNM
1	1661	64	17
2	647	17	6
3	87	8	4
4	45	17	12
5	142	26	29
6	242	18	6
TOTAL	2824	150	74
+OPs/1000	387	49	59

Table 1 Comparative Test Results

A much more thorough investigation into the performance of this second-order method is currently under way and is intended for future publication.

6. CONCLUDING REMARKS

Our objective has been to provide a unified and numerically stable approach for solving facility location problems. To achieve this goal we have presented a second-order method, involving projected Newton steps, that can be applied to a wide class of location problems. For efficiency, the method has been designed to exploit the sparsity and structure that are inherent in all these problems regardless of their scale. In addition, the degeneracies that occur quite frequently in multifacility location problems are easily resolved using the proposed method.

ACKNOWLEDGEMENTS

The authors would like to thank Mary Wang, Brian Finch and Marney Heatley for their help in typesetting this paper using Troff and the Photon Econosetter.

REFERENCES

[1] Calamai, P.H., and Charalambous, C., "Solving Multifacility Location Problems Involving Euclidean Distances", Naval Res. Log. Quart., 27, 609-620, (1980).

[2] Calamai, P.H., and Conn, A.R., "A Stable Algorithm for Solving the Multifacility Location Problem Involving Euclidean Distances", SIAM J. Sci. Stat. Comput., 1, 512-525, (1980).

[3] Chatelon, J.A., Hearn, D.W., and Lowe, T.J., "A Subgradient Algorithm for Certain Minimax and Minisum Problems", Math Prog., 15, 130-145, (1978).

[4] Coleman, T.F., and Conn, A.R., "Nonlinear Programming Via an Exact Penalty Function: Global Analysis", Math. Prog., (to appear).

[5] Coleman, T.F., and Conn, A.R., "Nonlinear Programming Via an Exact Penalty Function: Asymptotic Analysis", Math. Prog., (to appear).

[6] Eyster, J.W., White, J.A., and Wierwille, W.W., "On Solving Multifacility Location Problems Using a Hyperboloid Approximation Procedure", AIIE Trans., 5, 1-6, (1973).

[7] Francis, R.L., and Goldstein, J.M., "Location Theory: A Selective Bibliography", Oper. Res., 22, 400-410, (1974).

[8] Francis, R.L., and White, J.A., "Facility Layout and Location: An Analytic Approach", Prentice-Hall, New Jersey, (1974).

[9] Gill, P.E., and Murray, W., "Safeguarded Steplength Algorithms for Optimization Using Descent Methods", Report NAC 37, National Physical Laboratory, England, (1974).

[10] Gill, P.E., and Murray, W., "Newton-type Methods for Unconstrained and Linearly Constrained Optimization", Math. Prog., 7, 311-350, (1974).

[11] Lea, A.C., "Location-Allocation Systems: An Annotated Bibliography", Discussion Paper No. 13, Univ. of Toronto, Department of Geography, Canada, (1973).

[12] Love, R.F., "Locating Facilities in Three-dimensional Space by Convex Programming", Naval Res. Log. Quart., 16., 503-516, (1969).

[13] Love, R.F., and Morris, J.G., "Modelling Inter-city Road Distances by Mathematical Functions", Opnl. Res. Quart., 23, 61-71, (1972).

[14] Minieka, E., "Optimization Algorithms for Networks and Graphs", Industrial Engineering Series: Volume 1, Marcel Dekker Inc., (1978).

[15] Morris, J.G., "A Linear Programming Solution to the Generalized Rectangular Distance Weber Problem", Naval Res. Log. Quart., 22, 155-164, (1975).

[16] Murray, W., and Overton, M.L., "Steplength Algorithms for Minimizing a Class of Non-Differentiable Functions", Computing, 23, 309-331, (1979).

[17] Overton, M.L., "A Quadratically Convergent Method for Minimizing a Sum of Euclidean Norms", Tech. Report #030, Dept. of Comp. Sci., Courant Inst. of Math. Sci., (1981).

[18] Planchart, A., and Hurter, A.P., "An Efficient Algorithm for the Solution of the Weber Problem With Mixed Norms", SIAM J. Control, 13, 650-665, (1975).

[19] Vergin, R.C., and Rogers, J.D., "An Algorithm and Computational Procedure for Locating Economic Facilities", Management Sci., 13, B240-B254, (1967).

[20] Wesolowsky, G.O., and Love, R.F., "A Nonlinear Appoximation Method for Solving a Generalized Rectangular Distance Weber Problem", Management Sci., 18, 656-663, (1972).

[21] Wesolowsky, G.O., and Love, R.F., "The Optimal Location of New Facilities Using Rectangular Distances", Oper. Res., 19, 124-130, (1971).