Chapter IV

Solving Euclidean Distance Multifacility Location Problems Using Conjugate Subgradient and Line-Search Methods

As demonstrated in the previous chapters, the objective function of EMFLP has undefined first partial derivatives whenever any new facility coincides with either an existing or another new facility with which it interacts, and because of this feature, standard differentiable optimization techniques cannot be directly applied.

In this chapter, we explore the use of conjugate or deflected subgradient techniques along with suitable subgradient generation and line-search strategies in order to derive an effective scheme for solving EMFLP. The basic algorithmic framework that we employ is the Variable Target Value Method (VTVM) of Sherali *et al*.(1995). This method is a general globally convergent approach for solving convex, nondifferentiable optimization problems. It assumes no prior knowledge regarding bounds on the optimal value, and it manipulates a target that estimates the optimal value in order to induce convergence to an optimum. We employ this algorithmic framework because of its simplicity and its wide flexibility in permitting the special design of both a strategy for computing step-lengths, as well as a strategy for employing a subgradient deflection scheme, while preserving convergence properties. However, we modify the restart feature and the termination criteria prescribed by this procedure in order to improve its performance for the class of problems EMFLP. Within this modified framework, we employ the Average Direction Strategy (ADS) of Sherali and Ulular (1989) to generate deflected subgradient based search directions. The subgradients themselves are obtained using two methods. In the first method, zero values are assigned to contributions from the nondifferentiable terms in the objective function. The second method attempts to obtain improved search directions by deriving a low-norm subgradient based on using valid contributions other than zeros for the nondifferentiable terms. We also test the use of an alternative step-size strategy

known as the block-halving (BH) scheme (Sherali and Ulular, 1989) within the framework of VTVM, and denote this by VTVM + BH. In addition, motivated by the work of Calamai and Conn (1987), and Overton (1983), a Newton- based inexact line-search method is developed and tested in conjunction with both step-size strategies. Our results show that the modified VTVM algorithm when operated using the ADS deflection approach in concert with a particular combination of the two subgradient generation strategies and the proposed line-search technique, yields a computationally effective procedure. Furthermore, the results show that VTVM + BH also yields a competitive performance, but more strongly requires the use of the proposed line-search technique to be computationally effective.

The remainder of this chapter is organized as follows. In Section 4.1 we present a our proposed two subgradient generation strategies. Section 4.2 describes our modification of the VTVM approach including the use of the BH step-size scheme, and Section 4.3 develops our proposed inexact line-search method. Finally, Section 4.4 presents some computational results and comparisons using a collection of test problems.

4.1. Subgradient Generation Strategies

Charalambous (1985), Plastria (1992) and our proposed Lemma 1 of the previous chapter present various equivalent characterizations for the set of subgradients of the objective function f of EMFLP at any point $(\bar{x}, \bar{y})$. Using this characterizartion, various subgradient generation strategies can be designed based on the selection of (ζ_1, ζ_2) associated with the nondifferentiable terms in $f(\bar{x}, \bar{y})$. We propose two different strategies to be implemented in conjunction with a conjugate subgradient scheme within VTVM. The first strategy simply designates zero subgradients for the nondifferentiable terms, i.e., it sets $\zeta_1 = \zeta_2 = 0$ whenever $\bar{a}_{ij} = 0$ for $(i, j) \in A_{NE}$, or whenever $\bar{b}_{kl} = 0$ for $(k, l) \in A_{NN}$. The second strategy derives a low-norm subgradient of the overall objective function by adopting the solution of a simple minimization problem associated with each nondifferentiable term considered in a particular sequential fashion. Note that a least norm

subgradient is guaranteed to be a descent direction at a nonoptimal solution (see Bazaraa *et al.*, 1993, for example). The following are the essential steps of this second strategy.

Low-Norm Subgradient Strategy

Step 1- Arrange the weights w and then v in nonincreasing order.(Naturally, this is performed only once at the beginning of VTVM.)

Step 2- Initialize the subgradient vectors $z_x = z_y = 0 \in R^n$, and consider the terms $w_{ij}\{(x_i - a_j)^2 + (y_i - b_j)^2\}^{1/2}$ and $v_{kl}\{(x_k - x_l)^2 + (y_k - y_l)^2\}^{1/2}$ for every pair (i, j) and (k, l) in A_{NE} and A_{NN}, respectively. For each differentiable term, add the corresponding gradient component appropriately to z_x and z_y, and then proceed to Step 3.

Step 3- Using the order prescribed at Step 1, consider each nondifferentiable term $w_{ij}\{(x_i - a_j)^2 + (y_i - b_j)^2\}^{1/2}$, where $(i, j) \in A_{NE}$ and where currently, we have $(x_i, y_i) \equiv (a_j, b_j)$.

Let p and q be the current components of z_x and z_y corresponding to x_i and y_i, respectively. If $p = q = 0$, go to Step 4. Else, choose ξ_1 and ξ_2 as the solution to

$\min\{(p + wx_1)^2 + (q + wx_2)^2 : x_1^2 + x_2^2 \le 1\}$, where w " w_{ij}.

This yields

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -p/\theta \\ -q/\theta \end{pmatrix},$$

where $\theta = \max.\{w, \sqrt{p^2 + q^2}\}$. Replace the components p and q of z_x and z_y by $p + wx_1$, $q + wx_2$, respectively.

Step 4- Using the order prescribed at Step 1, consider each nondifferentiable term $v_{kl}\{(x_k - x_l)^2 + (y_k - y_l)^2\}^{1/2}$, $(k, l) \in A_{NE}$. Let $\begin{pmatrix} p \\ q \end{pmatrix}$ and $\begin{pmatrix} r \\ s \end{pmatrix}$ be the current components of z_x and z_y, respectively, corresponding to $\{x_k$ and $x_l\}$, and to $\{y_k$ and $y_l\}$, respectively. Choose x_1 and x_2 as the solution to

$\min\{(p + vx_1)^2 + (q - vx_1)^2 + (r + vx_2)^2 + (s - vx_2)^2 : x_1^2 + x_2^2 \le 1\}$, where v" $v_{kl.}$

As in Step 3, we obtain $\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} (q-p)/\theta \\ (s-r)/\theta \end{pmatrix}$,

where $\theta = \max.\{\, 2v,\ \sqrt{(q-p)^2 + (s-r)^2}\,\}$.

Replace the components p, q, r, and s of the vectors z_x and z_y by $p + v\,\xi_1$, $q - v\,x_1$, $r + v\,x_2$, and $s - v\,x_2$, respectively. Stop at the end of this pass through all the nondifferentiable terms.

4.2 Conjugate Subgradient Algorithm

This section briefly describes the conjugate subgradient algorithm VTVM of Sherali *et al*. (1995) and its proposed variant VTVM + BH. It also addresses our modification of VTVM regarding the strategy of periodically restarting the algorithm with the current best (incumbent) solution, and regarding the termination criteria, both of which contribute to the acceleration of the convergence and the early stopping of the algorithm within an acceptable percentage of optimality. The developed VTVM method is operated and tested in conjunction with each of the two subgradient generation strategies proposed in Section 3, as well as with a combination thereof, in order to solve problem EMFLP.

4.2.1 VTVM Algorithm

The VTVM procedure generates a sequence of incumbent solutions whose corresponding objective values converge to within any specified optimality tolerance $\varepsilon > 0$. This is achieved by employing two essential loops, the inner loop and the outer loop. In the inner loop of the algorithm, given an iterate $z^k \equiv (x^k, y^k)$, a direction of motion d^k is selected as either $-g^k$ or $-g^k + \gamma_k\, d^{k-1}$, where g^k is a subgradient of f at z^k, d^{k-1} is the search direction at the previous iteration, and where γ_k is a deflection parameter selected via the Average Direction Strategy (ADS) of Sherali and Ulular (1989). A prescribed step-size

$$\bar{\lambda} = \beta_k \frac{f(z^k) - T_k}{\left\| d^k \right\|^2} \tag{4.1}$$

is then computed, where $0 < \beta_k < 2$ is the step-size parameter and where T_k is the current target value. The new iterate z^{k+1} is then computed according to $z^{k+1} = z^k + \bar{\lambda} d^k$. Periodically, the algorithm is triggered to visit the outer loop where it adjusts the target value and other related algorithmic parameters based on the information obtained from the inner loop. We modify the criterion used by VTVM to increase the target value due to non-improvements to include the case when the objective value of the current iterate is greater than twice the incumbent objective value. When this case is encountered, the number of consecutive non-improvements permitted within the inner loop, which is denoted by $\bar{\gamma}$ and which is initialized as $\bar{\gamma} = 10$, is replaced by $\min\{\bar{\gamma} + 5, 50\}$ within the outer loop, and the procedure is reset to the incumbent solution. A second modification used is in regard to the stopping criteria employed by the algorithm. Besides using the standard termination rule based on reaching a specified maximum number of iterations k_{max}, the algorithm is allowed to terminate whenever $\left\| g^k \right\| < 10^{-3} f(z^*)$, where z^* is the current incumbent solution. Another stopping criterion that is employed by the original VTVM algorithm is based on the relative accumulated improvement in the incumbent value over the current set of inner loop iterations. Whenever after 500 iterations, denoted by *kup*, the algorithm has performed at least one decrease in the target value during a visit to the outer loop, then if the aforementioned relative improvement is less than 0.05 over four consecutive target increases in the outer loop step that are triggered by the inner loop non-improvement step, the procedure terminates. We proposed a slight change to this criterion by terminating the algorithm whenever after *kup* = 50 iterations the former criterion is satisfied, without requiring that the algorithm must have performed at least one decrease in the target value.

4.2.2 **VTVM + BH Algorithm**

Sherali and Ulular (1989) describe a block-halving (BH) strategy to control the parameter β_k when using the prescribed step-size formula (4.1). In the spirit of this BH scheme, we divide the maximum number of inner loop iterations into blocks, each having an equal number of iterations. For the first block (k = 1), the procedure initializes with β_k = 0.95. Then, for each inner loop, the value of β_k is re-initialized at the beginning of each subsequent block at 0.25 less than the value of this parameter at the beginning of the previous block. Furthermore, within each block, if the number of consecutive non-improvements in the objective value of the incumbent reaches the limit set by VTVM, the current β - parameter is halved and the algorithm is reset to the incumbent solution. In order to avoid having near-zero step-sizes, the value of β_k is not permitted to fall below 10^{-6} in this scheme.

4.3 Line-Search Strategy

Several researchers have considered using second-order line-search methods in their algorithmic approaches for solving EMFLP. Motivated by the algorithms developed by Overton (1983) and Calamai and Conn (1987), we designed a new line-search technique that exploits the structure of the problem in order to derive a procedure that combines Newton's method and the quadratic fit approach (see Bazaraa *et al*. (1993) for a description of the latter procedure). Details of this method are presented below. We mention here that we also studied four other line-search schemes including Overton's (1983) method, a procedure based on the hyperboloid approximation, an exact search using the golden section method, and a direct application of an inexact quadratic fit search. In comparison, only the last of these methods gave competitive results with respect to the procedure described below, and we provide some related computational results in Section 4.5.

To present the proposed line-search, suppose that we are given some solution z and a direction d that is detected to be a descent direction upon having implemented the prescribed step-length $\bar{\lambda}$ of Equation (4.1). The purpose of the line-search scheme is to solve the problem (perhaps inexactly)

$$\underset{\lambda > 0}{\text{Minimize}} \quad F(\lambda) \equiv f(z + \lambda d). \tag{4.2}$$

Toward this end, let $M = \{ A_{NE} \cup A_{NN} \}$, and let us refer to the various index pairs defining M as simply $t \in M$. Accordingly, we will let u_t denote the positive interaction weight w_{ij} if $t \equiv (i, j)$ in A_{NE}, and $u_t = v_{kl}$ if $t \equiv (k, l)$ in A_{NN}. Then, letting $A_j \equiv (a_j, b_j) \ \forall j = 1,.., m$, and denoting $z_i \equiv (x_i, y_i)$ with the corresponding components of the direction vector d being $d_i \equiv (d_{ix}, d_{iy})$ for each $i = 1,..., n$, we can rewrite the function $F(\lambda)$ in (4.2) as

$$F(\lambda) = \sum_{t \in M} \Phi_t(\lambda),$$

where

$$\Phi_t(\lambda) = \begin{cases} w_{ij} \left\| z_i + \lambda d_i - A_j \right\| & \text{if } t \equiv (i, j) \in A_{NE} \\ v_{kl} \left\| z_k + \lambda d_k - z_l - \lambda d_l \right\| & \text{if } t \equiv (k, l) \in A_{NN}. \end{cases} \tag{4.3}$$

Note that whenever $\Phi_t(\lambda) \neq 0$, we have

$$\Phi_t'(\lambda) = \begin{cases} w_{ij}^2 \dfrac{d_i \cdot (z_i + \lambda d_i - A_j)}{\Phi_t(\lambda)} & \text{if } t \equiv (i, j) \in A_{NE} \\ v_{kl}^2 \dfrac{(d_k - d_l) \cdot (z_k + \lambda d_k - z_l - \lambda d_l)}{\Phi_t(\lambda)} & \text{if } t \equiv (k, l) \in A_{NN}, \end{cases} \tag{4.4a}$$

$$\Phi_t''(\lambda) = \begin{cases} w_{ij}^2 \dfrac{\left[\Phi_t(\lambda) \left\| d_i \right\|^2\right] - \left[(\Phi_t'(\lambda))^2 \Phi_t(\lambda)\right]}{[\Phi_t(\lambda)]^2} & \text{if } t \equiv (i, j) \in A_{NE} \\ v_{kl}^2 \dfrac{\left[\Phi_t(\lambda) \left\| (d_k - d_l) \right\|^2\right] - \left[(\Phi_t'(\lambda))^2 \Phi_t(\lambda)\right]}{[\Phi_t(\lambda)]^2} & \text{if } t \equiv (k, l) \in A_{NN}. \end{cases} \tag{4.4b}$$

Observe that for any $t \in M$, if the corresponding coefficient of λ in (4.3) is zero, then $\Phi_t(\lambda)$ is invariant with λ. Accordingly, let us define

$$M' = \{ t \in M : d_i \neq 0 \text{ if } t \equiv (i, j) \in A_{NE}, \text{ and } d_k \neq d_l \text{ if } t \equiv (k, l) \in A_{NN} \}. \tag{4.5}$$

Then, we have that

$$F(\lambda) = (\text{constant}) + \sum_{t \in M'} \Phi_t(\lambda) \qquad (4.6)$$

and so, the one-dimensional line-search problem is concerned with minimizing only the variable part of (4.6) in order to find an (approximate) optimum λ^*. However, one difficulty in minimizing (4.6) using standard derivative-based line-search approaches is that $F(\lambda)$ is not everywhere differentiable. To circumvent this nondifferentiability problem, we first determine a perturbed step-length $\hat{\lambda}$ (≥ 0) at which F is twice differentiable. The basic idea of our line-search is to then perform a standard Newton iteration at $\hat{\lambda}$ to determine an unsafeguarded step, say s. Next, if some specified conditions regarding the absolute value of the derivative of F at $\hat{\lambda}$ and the positivity of the Newton step s are not satisfied, the algorithm uses the available points such as 0, $\hat{\lambda}$, s and the prescribed step-size $\bar{\lambda}$ to establish a Three- Point-Pattern (TPP) (see Bazaraa *et al.*, 1993) in order to perform one quadratic fit step (or more, if needed) in order to obtain the recommended step-size λ^*.

In this process, to find the point $\hat{\lambda}$ where F is twice differentiable and hence to determine a Newton step s, we adopt Overton's (1983) strategy, but we use M' in lieu of M, to define

$$J = \{ t \in M' : \Phi_t(0) \leq (\text{machine epsilon})^{3/4} \} \qquad (4.7)$$

and its complement $\bar{J} = M' - J$. Note that if $\bar{J}$ is empty, then the direction d is not a (sufficiently) descent direction, and we could then skip this line-search step. Furthermore, if J is empty, then F is twice differentiable at 0 itself, and hence we can set $\hat{\lambda} = 0$. Otherwise, noting the convexity of $\Phi_t(\lambda)$ for $t \in \bar{J}$, we use a linear approximation at $\lambda = 0$ to obtain a lower bound on the step at which Φ_t might become zero for each $t \in \bar{J}$, and

we set $\hat{\lambda}$ to be half of the minimum of these estimated zeros of $\Phi_t(\lambda)$ for $t \in \bar{J}$. Consequently, at $\hat{\lambda}$, $\Phi_t(\lambda) \neq 0 \ \forall \ t \in M'$, and so F defined by (4.6) is then twice differentiable at $\hat{\lambda}$. The following is a formal statement of this procedure.

Statement of the Line-Search Procedure

Set the parameter $\varepsilon_0 = 10^{-3}$, and let TOL be a termination tolerance on the length of the interval of uncertainty (we used TOL = 10^{-4}).

Step 1 Derive each of the sets M', J and $\bar{J}$ as defined in Equations (4.5) and (4.7).

If $\bar{J} = \varphi$, then abort this line-search procedure.

Step 2 If $J = \varphi$, then set $\hat{\lambda} = 0$, and otherwise, let $\hat{\lambda} = \frac{1}{2} \min \left\{ \frac{-f_t(0)}{f'_t(0)}, t \in \bar{J} \right\}$.

Step 3 Compute $F'(\hat{\lambda}) = \sum_{t \in M'} \Phi'_t(\hat{\lambda})$ via Equations (4.4a) and (4.6). If $| F'(\hat{\lambda})| \leq \varepsilon_0$,

then set $\lambda^* = \hat{\lambda}$ and stop. Otherwise, compute $F''(\hat{\lambda}) = \sum_{t \in M'} \Phi''_t(\hat{\lambda})$ via

Equations (4.4b) and (4.6), and determine the Newton step $s = \hat{\lambda} - \frac{F'(\hat{\lambda})}{F''(\hat{\lambda})}$.

Step 4 If $F'(\hat{\lambda}) > 0$ then if $s > 0$, set $\lambda^* = s$;

else, establish a TPP $(\alpha_1, \alpha_2, \alpha_3)$ by appropriately selecting points from the values 0, $\hat{\lambda}$ and $\bar{\lambda}$, and by suitably halving or doubling the values of $\hat{\lambda}$ or $\bar{\lambda}$, as necessary. If $(\alpha_3 - \alpha_1) \leq$ TOL, pick $\lambda^* = \alpha_2$, and terminate. Otherwise, perform one quadratic fit to find λ^*.

If $F'(\hat{\lambda}) < 0$ (so that $s > \hat{\lambda}$), establish a TPP (α_1, α_2 , α_3 by using $\hat{\lambda}$, $\bar{\lambda}$ and s, as appropriate. If $(\alpha_3 - \alpha_1) \leq$ TOL, pick $\lambda^* = \alpha_2$, and terminate. Otherwise, perform one quadratic fit to find λ^*.

Step 5 If a quadratic fit has been performed at Step 4 and if the current best step-size λ^* is equal to $\bar{\lambda}$ with $(\alpha_3 - \alpha_1) >$ TOL, perform an additional quadratic fit step and stop.

4.4 Computational Experience

In this section, we discuss the computational experiments that we have conducted to study the effect of the two subgradient generation strategies on the performance of each of the developed VTVM and VTVM + BH algorithms, as well as on using the proposed line-search procedure along with these two algorithms. In these experiments we used 8 test problems (TP). The first 4 problems are the same standard test problem that we used in the previous chapter whereas the other test problems were generated randomly. Table 1 gives the sizes of these test problems.

Table 1: Size of the Randomly Generated Test Problems.

TP	n	m
5	10	20
6	20	30
7	20	10
8	30	50

All the algorithms tested were coded in FORTRAN and run on an IBM RS/6000 computer, with a fixed set of parameter values as prescribed or recommended by the original VTVM algorithm (see Sherali *et al.*, 1995), except that the limit on the maximum number of iterations was set to k_{max} = 400. The block size in the VTVM + BH algorithm was chosen to be equal to 100 iterations. Also, the ADS deflection strategy was used in

all the runs. In the tables given below, the total execution time (in seconds) of the algorithm is denoted by cpu, and the percentage of optimality is denoted by OPT(%) and is computed as $f(x_{exact})*100/ f(x_{best})$, where x_{exact} and x_{best} are the known exact solution and the best solution found by the algorithm upon termination, respectively. In all of the test problems, the starting solution has been chosen to be the optimum to the corresponding squared Euclidean distance problem (see Francis *et al*., 1991)

Run 1 and **Run 3** correspond to algorithm VTVM and VTVM + BH, respectively, when the first subgradient strategy is used. Notationally, we will denote these algorithmic runs by VTVM1 and VTVM1 + BH, respectively. **Run 2** and **Run 4** correspond to algorithm VTVM and VTVM + BH, respectively, when the second (low-norm) subgradient strategy is used, and we will similarly denote these two algorithms by VTVM2 and VTVM2 + BH. In each of these four runs, we let the algorithm terminate only when the maximum number of iterations is reached. The additional modified stopping criteria prescribed in Section 3 are tested subsequently. Moreover, in order to exhibit the improvement behavior of the incumbent solution, we record the iteration number at which the final incumbent solution was found, and denote this by k^*. Table 2 gives the results obtained.

Table 2: Comparative Test Results for VTVM and VTVM + BH Using the Two Subgradient Strategies.

TP	Run 1: VTVM1		Run 2: VTVM2		Run 3: VTVM1 + BH		Run 4: VTVM2 + BH	
	OPT(%)	k^*	**OPT(%)**	k^*	**OPT(%)**	k^*	**OPT(%)**	k^*
1	**99.99**	375	**99.86**	32	23.6	**399**	23.15	**399**
2	**99.97**	356	**92.86**	1	97.92	**400**	92.86	**1**
3	**99.96**	364	**97.4**	1	99.35	**206**	97.4	**1**
4	**99.99**	386	**99.99**	308	99.99	**199**	99.99	**98**
5	**99.90**	364	**99.57**	376	99.83	**216**	99.52	**353**
6	**99.96**	386	**98.58**	249	99.69	**226**	98.72	**307**
7	**99.98**	392	**99.38**	381	99.57	**231**	98.84	**399**
8	**99.97**	371	**70.0**	397	91.03	**400**	66.16	**399**

When we compare the results of Run1 and Run 3, we observe that if the first subgradient strategy is adopted (and no line-search is performed), then our modification of VTVM attains a percentage of optimality within 99.9% for all of the test problems , while VTVM in conjunction with the BH scheme attains unsatisfactory accuracy in some cases as for the test problems 1, 2 and 8. Furthermore, if we compare the results of Runs 2 and 4 corresponding to the second subgradient strategy, we observe that both VTVM and VTVM in conjunction with the BH scheme exhibit an inadequate performance in certain instances as seen from the results for problems 2, 3 and 8, in particular. Note that problems 2 and 3 did not detect any descent direction based on the prescribed step-size (4.1). On the other hand, sometimes, better quality solutions appear to be detected earlier using this second subgradient strategy. On an overall comparison, when no line-search is performed, VTVM yields the best performance when operated in conjunction with the first subgradient strategy.

Next, we attempted to use the early improvement- based stopping criteria discussed in Section 3 in conjunction with VTVM. Table 3 presents the results obtained. **Run 5** is the same as Run 1, but under the activation of the stopping criteria, and **Run 6** is similar to Run 5, but with the modification that after 150 iterations, the algorithm switches over to using the second low- norm subgradient strategy. In Table 3, **Iters** represents the total

number of iterations executed, and the values of S.C, given within parentheses, represent the type of stopping criteria that caused the termination. These values are designated as follows.

$$\text{S.C} = \begin{cases} \text{(1) if the algorithm stops because } \left\| g^k \right\| < 10^{-3} f(z^*), \\ \text{(2) if the algorithm stops based on the cumulative relative improvement criterion,} \\ \text{(3) if the algorithm stops becuase } k_{\max} (= 400) \text{ iterations have been performed.} \end{cases}$$

Table 3: Test Results Using the Modified Stopping Criteria and a Combination of the Two Subgradient Strategies in VTVM.

TP	**Run 5: (VTVM1 + stopping criteria)**				**Run 6: (Run 5 + using the low-norm subgradient strategy after 150 iterations**)			
	cpu	**Iters**	**S. C**	**Opt(%)**	**cpu**	**Iters**	**S. C**	**Opt(%)**
1	.253	146	(2)	**99.96**	.253	146	(2)	**99.96**
2	.013	93	(2)	**99.96**	.013	93	(2)	**99.96**
3	.079	198	(2)	**99.87**	.08	199	(2)	**99.62**
4	.033	30	(1)	**99.98**	.033	30	(1)	**99.98**
5	.298	130	(2)	**99.78**	.298	130	(2)	**99.78**
6	1.612	204	(2)	**99.47**	1.706	219	(2)	**99.77**
7	1.272	400	(3)	**99.97**	.765	261	(2)	**99.58**
8	4.189	207	(2)	**99.53**	4.127	207	(2)	**99.53**

By comparing the results of Runs 5 and 6 in Table 3, we see that when the low-norm subgradient strategy is activated after 150 iterations, in most of the test problems, the convergence of the algorithm either stays the same or accelerates as in the case of test problem 7. A comparison of these two runs with Run 1 demonstrates the efficiency of the proposed stopping criteria in terminating the algorithm significantly earlier while having yet attained a solution within 99.5% of the optimality. For example, stopping criterion (1) caused the early termination of test problem 4 with a solution that attains 99.98% of optimality after only 30 iterations.

For the sake of demonstrating the effect of our proposed line-search procedure on the convergence performance of both VTVM and VTVM + BH, we considered **Runs 7, 8 and 9** as described in Table 4. Here, **line-search 1** refers to the procedure discussed in Section 4. In these runs, a line-search is invoked whenever a descent direction of motion d is encountered and when $\left[\frac{f(z^{k+1})-f(z^k)}{\|d\|\bar{l}}\right] \le -0.8\|g^k\|$, where $z^{k+1} = z^k + \bar{l}\ d$. This latter condition is similar to Powell's (1977) criterion for restarting conjugate gradient methods in smooth optimization, and its intent here is to avoid a line-search if the rate of descent is not sufficient. We also used a block-size of 50 for the BH strategy, and TOL=10^{-4} for the line-search procedure.

Table 4: Comparative Test Results on Using the Proposed Line-Search Procedure.

TP	**Run 7 : VTVM1+ BH + SC + line-search 1**				**Run 8 : VTVM1 + SC + line-search 1**			
	cpu	**Iters**	**S.C**	**Opt(%)**	**cpu**	**Iters**	**S.C**	**Opt(%)**
1	.007	19	(1)	**99.99**	.037	165	(2)	**99.92**
2	.159	400	(2)	**99.58**	.015	89	(2)	**99.54**
3	.088	185	(2)	**99.8**	.052	114	(2)	**97.49**
4	.030	16	(1)	**99.99**	.044	22	(1)	**99.99**
5	.595	144	(1)	**99.93**	.467	153	(2)	**99.8**
6	2.25	184	(2)	**99.91**	1.705	186	(2)	**99.33**
7	3.488	400	(3)	**98.99**	.721	195	(2)	**99.34**
8	5.43	164	(2)	**99.56**	3.113	122	(2)	**98.91**

TP	**Run 9 : Run 6 + line-search 1**			
	cpu	**Iters**	**S.C**	**Opt(%)**
1	.037	165	(2)	**99.92**
2	.015	89	(2)	**99.45**
3	.052	114	(2)	**97.49**
4	.044	22	(1)	**99.99**
5	.472	153	(2)	**99.81**
6	1.645	174	(2)	**99.2**
7	.695	198	(2)	**98.02**
8	3.11	122	(2)	**98.9**

From Run 7 of Table 4 and Run 3 of Table 2, we can see that under the proposed line-search, the performance of VTVM1 + BH is significantly enhanced, particularly for test cases 1 and 8, and also to some extent for test problem 2. Moreover, Run 7 terminates in test problems 1, 4 and 5 due to stopping criterion (1) with very accurate solutions, and in particular, having consumed fewer iterations and less cpu time than VTVM1 of Run 8 for test problems 1 and 4. However, Run 8 yields a better performance than Run 7 for the larger problem instances 6, 7, and 8. Furthermore, when we compare the results of Runs 5 and 6 with those of Runs 8 and 9, we observe that the latter runs again yield an improved performance for the larger test problems 6, 7, and 8. This comparison indicates that the incorporation of the proposed line-search procedure improves the convergence performance of VTVM1, particularly for the larger sized test problems.

Motivated by the above observed improvement in VTVM1 + BH when employing line-searches, we investigated the convergence behavior of this algorithm using another inexact line-search procedure based on performing two quadratic fits, neglecting nondifferentiability issues. Here, the initial interval of uncertainty is determined by sequentially doubling $\bar{l}$ until a TPP is established. Next, the line-search algorithm performs two quadratic fit steps in order to obtain the prescribed step-size l^* and then terminates. However, in case l^* turns out to be equal to $\bar{l}$ and the length of the interval of uncertainty is greater than a specified tolerance TOL (we used TOL = 10^{-3}), the algorithm performs an additional quadratic fit step before termination. Let this procedure be denoted by **line-search 2**.

Table 5 gives the results for **Run 10** that substitutes line-search 2 for line-search 1 in Run 7. Comparing Runs 7 and 10, we see that the results are roughly competitive, but that Runs 8 and 9 continue to dominate the overall performance.

Table 5: Test Results for the Two-Quadratic Fit Line-Search Procedure.

	Run 10 : VTVM1 + BH + S.C + line-search 2			
TP	**cpu**	**Iters**	**S.C**	**Opt(%)**
1	.012	46	(1)	**99.99**
2	.088	263	(2)	**99.88**
3	.053	124	(2)	**99.6**
4	.039	18	(1)	**99.99**
5	.708	175	(2)	**99.94**
6	1.975	128	(2)	**99.89**
7	1.356	197	(2)	**99.57**
8	6.160	157	(2)	**99.74**

Therefore, we conclude that the modified variant of VTVM that either employs the first or the combined subgradient strategy, and uses the proposed line-search procedure (Runs 8 and 9), yields the best overall performance in our experiments. The competing BH step-size strategy more strongly requires the use of line-searches to yield good quality results, and affords an alternative second choice for solving the Euclidean multifacility location problem.

In order to present some additional comparative results for the approaches developed in the previous and the present chapter for solving Problem EMFLP, we employed the algorithm of Run 8, which performed as one of the best conjugate subgradient methods developed in this chapter, and compared its results with those of using MINOS to solve the differentiable reformulations REMFLP and DEMFLP of chapter 3. Table 6 provides the results obtained using 8 test problems. Among these are the 4 test problems obtained from the literature that we have used in the runs presented in the present and the previous chapters. The other 4 test problems are randomly generated, and the data for these test problems are provided in the Appendix.

Observe that due to the developed stopping criterion (2), Run 8 performs quite efficiently in attaining a near optimal solution. However, with respect to the number of iterations, REMFLP appears to be faster as the size of the problem increases, while with respect to accuracy, DEMFLP yields the most accurate results in relatively acceptable cpu time.

We conclude from these experiments that the conjugate subgradient method consumes the least cpu time while providing fairly accurate solutions, and the differentiable reformulations REMFLP and DEMFLP provide somewhat more accurate solutions with slightly additional efforts, although still within acceptable cpu time.

Table 6. Comparative Results for the Algorithm of Chapters 3 and 4 on Additional Test Problems.

TP	**(*n*, *m*)**	**Run 8**				**REMFLP**		
		f_{best}	**S.C**	**Iters**	**cpu sec**	f_{best}	**Iters**	**cpu sec**
1	(2,5)	67.292	(2)	165	.037	67.681	24	.190
2	(2,3)	173.0416	(2)	89	.015	172.256	20	.180
3	(5,3)	40.004	(2)	114	.052	39.000	5	.220
4	(9,5)	201.891	(1)	22	.044	201.872	7	.440
5	(3,40)	14086.02	(1)	32	.0642	14081.492	25	.580
6	(5,20)	11512.885	(2)	152	.2744	11512.331	26	.550
7	(7,15)	2886.232	(2)	115	.239	2878.714	139	.550
8	(10,15)	16896.596	(2)	142	.4623	1687.978	33	.820

TP	**DEMFLP**		
	f_{best}	**Iters**	**cpu sec**
1	67.239	110	.160
2	172.256	9	.160
3	39.000	6	.191
4	201.872	312	.320
5	14080.899	60	.380
6	11511.841	63	.380
7	2876.142	1085	.420
.8	16878.404	310	.540