A variable target value method for nondifferentiable optimization

Hanif D. Sherali *, Gyunghyun Choi, Cihan H. Tuncbilek

Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA

Received 1 March 1998; received in revised form 1 July 1999

Abstract

This paper presents a new Variable target value method (VTVM) that can be used in conjunction with pure or deflected subgradient strategies. The proposed procedure assumes no a priori knowledge regarding bounds on the optimal value. The target values are updated iteratively whenever necessary, depending on the information obtained in the process of the algorithm. Moreover, convergence of the sequence of incumbent solution values to a near-optimum is proved using popular, practically desirable step-length rules. In addition, the method also allows a wide flexibility in designing subgradient deflection strategies by imposing only mild conditions on the deflection parameter. Some preliminary computational results are reported on a set of standard test problems in order to demonstrate the viability of this approach. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Nondifferentiable optimization; Subgradient algorithm; Conjugate subgradient methods; Deflected subgradient directions; Variable target value method

1. Introduction

Consider the nondifferentiable optimization problem

\[ \text{NDO: Minimize } \{ f(x) : x \in X \} \]

where \( f \) is a convex function that is not necessarily differentiable, and \( X \) is a nonempty, closed, convex subset of \( \mathbb{R}^n \). We assume that (1) has an optimum \( x^* \) and that the set of subgradients of \( f \) over \( X \) is a bounded set.

One approach to solve such problems is to use subgradient optimization methods. Here, given an iterate \( x_k \in X, k \geq 1 \), a direction of motion \( d_k \) is generated based on the set of subgradients of \( f \) at or about \( x_k \), and a step-length \( \lambda_k \) is taken along this direction. The new iterate \( x_{k+1} \) is then computed according to

\[ x_{k+1} = P_X [x_k + \lambda_k d_k] \]

where \( P_X [\cdot] \) denotes the projection operation onto the set \( X \). The effectiveness of this scheme is strongly dependent on the choice of \( d_k \) and \( \lambda_k \). In this paper, we will focus on the more popular pure or deflected subgradient strategies that have been spurred by Lagrangian relaxation applications in which \( d_k \) is selected as either \(-g_k\) or \(-g_k + \psi d_{k-1}\), respectively, where \( g_k \) is a subgradient of \( f \) at \( x_k \), and where \( \psi \geq 0 \) is a suitable deflection parameter associated with the previous direction of motion, with \( d_0 \equiv 0 \). (See [3,6,10,15,17–19] for various subgradient deflection strategies.) Note that in our analysis,
we permit the use of any deflection strategy so long as the deflection parameters $\psi_k \geq 0$ are chosen such that
\[ \|d_k\| < M \text{ for all } k, \text{ for some sufficiently large number } M. \] (2)

This should be easy to ensure, given our assumption on the boundedness of the set of subgradients of $f$ over $X$. In particular, both the modified gradient technique (MGT) of Camerini et al. [3] and the average direction strategy (ADS) of Sherali and Ulular [18] satisfy condition (2).

As far as related step-length rules are concerned, these play an important role not only in ensuring convergence, but also in governing the practical rate of convergence to optimality (see [4,7,17] for example). The more effective step-length rules, popularized by Held and Karp [6] and Held et al. [7], are of the type
\[ \lambda_k = \beta_k \frac{f(x_k) - w}{\|d_k\|^2}, \] (3)
where $0 < \beta \leq \beta_k \leq \beta < 2$, and where $w$ is a target value.

Often, $w$ is taken as a fixed lower bound on the problem. On the other hand, Bazarra and Sherali [1] present some rules to choose the target value as a convex combination of a fixed lower bound and the current best objective value. A similar idea is used by Kim et al. [9] in their variable target value method for minimizing strongly convex functions. However, both these papers assume that some initial lower bound estimate is available, and moreover, Kim et al. [9] additionally assume that an upper bound on $\|x_1 - x^\ast\|$ is known, where $x^\ast$ is an optimal solution to NDO.

Our motivation in this paper is to present a new variable target value method (VTVM) for general nondifferentiable optimization problems that adopts an effective subgradient step-length rule of type (3) and that assumes no a priori knowledge whatsoever regarding the optimal objective value. Brämlund [2] describes an alternate scheme in his dissertation for such problems, Goffin and Kiwiel [5] present a convergence analysis for a variant of Brämlund’s method, and Kulikov and Fazylov [14] propose a fixed, short-step variant of our algorithm. For a more extensive discussion on related subgradient projection methods, we refer the reader to Kiwiel [12,13].

The remainder of this paper is organized as follows. In Section 2, we present the proposed algorithm VTVM, and we analyze its convergence properties in Section 3. Section 4, provides some preliminary computational results using a variety of standard test problems from the literature.

2. The algorithm VTVM

The proposed algorithm operates in an inner loop and an outer loop. The inner loop involves the main process of generating a sequence of iterates. Depending on the progress during such inner loop iterations, the target value and other related parameters are periodically updated in an outer loop adjustment step. The algorithm is designed to theoretically converge in objective value to within any a priori specified tolerance $\varepsilon > 0$ of the optimal value, while preserving a reasonable degree of computational effectiveness in practice. It accomplishes the latter by permitting the use of a variety of subgradient deflection direction strategies, and by employing the practically effective step-length rule (3).

Below, we highlight our notation and then present a statement of the algorithm. The principal algorithmic parameters are described in the Initialization Step, and recommended values of these parameters are provided later in Remark 1.

Notation:

- counters: $k \equiv$ total iteration counter, $\ell \equiv$ outer loop iteration counter, $\tau \equiv$ current inner loop’s iteration counter, and $\gamma \equiv$ counter of ongoing consecutive nonimprovements.
- For any iteration $k$: $x_k = \text{iterate}$, $f_k = f(x_k)$, $g_k = \text{subgradient of } f \text{ at } x_k$, $d_k = \text{direction}$, $\lambda_k = \text{step-length}$, $\beta_k = \text{step-length parameter}$, and $z_k = \text{incumbent solution value}$. Also, $\Delta_k = \text{accumulated improvements within the current set of inner loop iterations until the beginning of iteration } k$.
- For any outer loop $\ell$: $w_\ell = \text{target value}$, and $w_\ell = \text{acceptance tolerance}$ for declaring that the current incumbent value is close enough to the target value $w_\ell$.
- Optimum: $x^\ast \in \text{argmin}\{ f(x): x \in X \}$, and $f^\ast \equiv f(x^\ast)$. Also, $\bar{x} = \text{incumbent solution and } \bar{g} = \text{available subgradient of } f \text{ at } \bar{x}$.
Algorithm (VTVM)

Initialization. Select the step-length parameter tolerance 0 < \tilde{\varepsilon} \leq 1, and termination parameters \varepsilon_0 \geq 0 for the tolerance on subgradient norms, \varepsilon > 0 for the overall convergence tolerance, and \kappa_{\text{max}} \leq \infty for the limit on the maximum number of iterations. Select values for the algorithmic parameters \sigma \in (0, \frac{1}{2}], \eta \in (0, 1], \tilde{\tau}, \tilde{\gamma}, \gamma \in (0, 1], \tilde{\tau}, \tilde{\gamma}, \gamma \in (0, 1] (see Remark 1 below for recommended values).

Select a starting solution \( x_1 \in X \), compute \( f_1 = f(x_1) \) and let \( d_1 = -g_1 \). If \( \|g_1\| \leq \varepsilon_0 \), then stop with \( x_1 \) as the prescribed solution. Otherwise, set \( \tilde{x} = x_1 \) and \( \tilde{g} = g_1 \), and record \( z_1 = f_1 \) as the best known objective function value. Initialize the target value \( w_1 = \max \{LB, f_1 - \|g_1\|^2/2\} \) and the acceptance tolerance \( \varepsilon_1 = \sigma(f_1 - w_1) \), where \( LB \) is any known lower bound on \( f^* \), being taken as \( -\infty \) if no such lower bound is available. (Note that any reasonable value \( \varepsilon_1 \) would suffice for the second term in the maximand for \( w_1 \). The stated value corresponds to the minimum value of a second-order approximation of \( f \) at \( x_1 \) with assumed gradient \( g_1 \) and an identity Hessian.) Initialize \( k = 0 \), \( \tau = 1 \), \( \gamma = 0 \), and \( d_1 = 0 \).

Step 1 (Inner loop main iteration). If \( k > \kappa_{\text{max}} \), stop. Else, determine \( d_k = -g_k + \varphi_k d_{k-1} \), where \( \varphi_k \geq 0 \) is selected via any suitable strategy so long as (2) holds true, and where \( d_0 \equiv 0 \). If \( \|d_k\| \leq \varepsilon_0 \), then set \( d_k = -g_k \). Also, compute the step-length

\[
\lambda_k = \beta_k \frac{f_k - w_\tau}{\|d_k\|^2}.
\]

Find the new iterate \( x_{k+1} = P_X[x_k + \lambda_k d_k] \), and determine \( f_{k+1} \) and \( g_{k+1} \). If \( \|g_{k+1}\| \leq \varepsilon_0 \), terminate the algorithm with \( x_{k+1} \) as the prescribed solution. Update \( A_{k+1} = A_k + \max\{0, z_k - f_{k+1}\} \). If \( f_{k+1} < z_k \), go to Step 2(a), and otherwise, go to Step 2(b).

Step 2(a) (Improvement in the inner loop). Put \( \gamma = 0 \), \( z_{k+1} = f_{k+1} \), and update \( \tilde{x} = x_{k+1} \) and \( \tilde{g} = g_{k+1} \). If \( z_{k+1} \leq w_\tau + \varepsilon_\tau \), then go to Step 3(a). Otherwise, if \( \tau \geq \tilde{\tau} \), go to Step 3(b); else, set \( \beta_{k+1} = \beta_k \), increment \( k \) and \( \tau \) by one, and return to Step 1.

Step 2(b) (Nonimprovement in the inner loop). Put \( z_{k+1} = z_k \), and increment \( \gamma \) by one. If \( \gamma \geq \tilde{\gamma} \) or \( \tau \geq \tilde{\tau} \), go to Step 3(b). Otherwise, set \( \beta_{k+1} = \beta_k \), increment \( k \) and \( \tau \) by one, and return to Step 1.

Step 3(a) (Outer loop success iteration: \( z_{k+1} \leq w_\tau + \varepsilon_\tau \)). Compute

\[
w_{\tau+1} = z_{k+1} - \varepsilon_\tau - \eta A_{k+1}
\]

and

\[
\varepsilon_{\tau+1} = \max\{(z_{k+1} - w_{\tau+1}) \gamma, \varepsilon_\tau\}.
\]

(See Remark 1 below.) Put \( \tau = 1, A_{k+1} = 0, \beta_{k+1} = \beta_k \), increment \( \gamma \) and \( k \) by one, and return to Step 1.

Step 3(b) (Outer loop failure iteration: \( z_{k+1} > w_\tau + \varepsilon_\tau \)). Compute

\[
w_{\tau+1} = \frac{(z_{k+1} - \varepsilon_\tau) + w_\tau}{2}
\]

and

\[
\varepsilon_{\tau+1} = \max\{(z_{k+1} - w_{\tau+1}) \gamma, \varepsilon_\tau\}.
\]

(Optionally, if \( \gamma \geq \tilde{\gamma} \), adjust \( \tilde{\gamma} \) as recommended in Remark 1 and restart with the incumbent solution.)

Remark 1. From a practical efficiency point of view, to ensure adequate step-lengths during improving phases of the algorithm, we can replace the target value update in (5) by \( w_{\tau+1} = z_{k+1} - \max\{\varepsilon_\tau + \eta A_{k+1}, r|z_{k+1}|\} \), where \( 0 < r < 1 \), and where \( r \) is divided by some \( \tilde{r} > 1 \) whenever \( w_{\tau+1} \) is determined by the second term in this maximand. (We used \( r = 0.08 \) and \( \tilde{r} = 1.08 \) in our computations and this strategy gave an improved computational performance.)

The convergence analysis of Section 3 continues to hold with this modification. Other recommended values of the parameters are \( \tilde{\varepsilon} = 10^{-6}, \varepsilon_0 = 10^{-6}, \varepsilon \in [10^{-6}, 10^{-1}] \) (we used \( \varepsilon = 0.1 \) in our computations), \( \kappa_{\text{max}} = 2000, \beta_1 = 0.95, \sigma = 0.15, \eta \in [0.75, 0.95] \) (we used \( \eta = 0.75 \), \( \tilde{\varepsilon} = 75 \), and \( \tilde{\gamma} \) being initialized at 20 and then incremented by 10 each time this limit is reached at Step 3(b) up to a maximum value of 50.
Remark 2 (Some practical considerations). Note that a restarting technique is often an important computational ingredient of subgradient procedures (see [1,7,18]). In the same spirit, for VTVM, whenever the target needs to be increased at Step 3(b) due to \( r \) consecutive failures, we restart the algorithm by setting \( x_k = \hat{x}, \ g_k = \hat{g}, \) and \( f_k = z_k \) at the end of Step 3(b), and then at the next visit to Step 1, we adopt \( d_k = -g_k. \) (For convergence purposes, no restarts are performed after some finite number of iterations.) Also, if \( w_{k+1} - w_k \leq 0 \cdot 1 \max \{1, |z_{k+1}| \} \), we replace \( \beta_k \) by \( \beta_{k+1} = \max \{ \beta_k / 2, \varepsilon \}. \)

3. Convergence analysis

In this section, we establish the convergence of Algorithm VTVM under \( k_{\max} = \infty \) and \( \varepsilon_0 = 0. \)

Lemma 1. Consider Algorithm VTVM, and for convenience, let us denote the target value at any (inner) iteration \( k \) by \( \hat{w}_k. \) Suppose that there exist values \( w \) and \( \tilde{w} \) such that

\[
f^* \leq w < \hat{w} < \tilde{w} \leq f_k \quad \text{for all} \ k.
\]

Then we must have both \( \{ \hat{w}_k \} \to \tilde{w} \) and \( \{ f_k \} \to \tilde{w}. \)

Proof. Let us first show that

\[
d_k(x^* - x_k) \geq 0 \quad \text{for all} \ k.
\]

Note that this is trivially true for \( k = 1, \) since \( d_0 = 0. \) By induction, consider any \( k \geq 2, \) and assume that \( d_{k-1}(x^* - x_{k-1}) \geq 0. \) Using the definition of \( d_{k-1}, \) the induction hypothesis, the convexity of \( f, \) Eq. (7), the fact that \( \beta_k \in [\tilde{\beta}, 1], \) and the Cauchy–Schwarz inequality, we obtain the following string of relations:

\[
d_k(x^* - x_k) = d_k(x^* - x_k) + d_k(x_k - x_{k-1})
\]

\[= -d_{k-1}(x^* - x_{k-1}) + \psi_k d_{k-1}(x^* - x_{k-1}) + \beta_k(f_{k-1} - f_k) - \|d_{k-1}\| ||x_k - x_{k-1}||
\]

\[= \beta_k(f_{k-1} - f_k) - \|d_{k-1}\||x_k - x_{k-1}||
\]

\[\leq \beta_k(f_{k-1} - f_k) - \|d_k\||x_k - x_{k-1}||
\]

\[\leq \beta_k(f_{k-1} - f_k) - \|d_k\||x_k - x_{k-1}||
\]

Now, since \( x_{k-1} \in X, \) we have, using (4), (7) and the nonexpansiveness of \( P_X \) that

\[
d_k(x^* - x_k) \geq \beta_k(f_{k-1} - w_k) - \|d_{k-1}\|
\]

\[\times \|x_{k-1} + \lambda_{k-1}d_{k-1} - x_k\|
\]

\[= \beta_k(f_{k-1} - w_k) - \|d_k\| ||x_k - x_{k-1}||
\]

\[= \beta_k(f_{k-1} - w_k) - \|d_k\| ||x_k - x_{k-1}||
\]

\[= \beta_k(f_{k-1} - w_k) - \|d_k\| ||x_k - x_{k-1}||
\]

Hence, assertion (8) holds true. Using the definition of \( d_k, \) we get

\[
\left\| x^* - x_{k+1} \right\|^2 \leq \left\| x^* - x_k - \lambda_k d_k \right\|^2
\]

\[\leq \left\| x^* - x_k - \lambda_k d_k \right\|^2
\]

\[= \left\| x^* - x_k \right\|^2 + \lambda_k^2 ||d_k||^2
\]

\[-2\lambda_k d_k(x^* - x_k)
\]

\[= \left\| x^* - x_k \right\|^2 + \lambda_k^2 ||d_k||^2
\]

\[+2\lambda_k g_k d_k(x^* - x_k)
\]

\[-2\lambda_k g_k d_k(x^* - x_k)
\]

\[\leq \left\| x^* - x_k \right\|^2 + \lambda_k^2 ||d_k||^2
\]

\[+2\lambda_k(f^* - f_k).
\]

The last inequality holds from (8) and the definition of a subgradient. Now, from (4) and (7), we get \( \beta_k(f^* - f_k) \leq \lambda_k(\hat{w}_k - f_k) = -\beta_k(\hat{w}_k - f_k)^2/||d_k||^2. \) Hence, using this in (9) along with (4) and the assumption on \( \beta_k \) we have

\[
\left\| x^* - x_{k+1} \right\|^2 \leq \left\| x^* - x_k \right\|^2 + \beta_k^2(\hat{w}_k - f_k)^2/||d_k||^2
\]

\[-2\beta_k(\hat{w}_k - f_k)^2/||d_k||^2
\]

\[= \left\| x^* - x_k \right\|^2 + \beta_k(\beta_k - 2) \left( \frac{f_k - \hat{w}_k)^2}{||d_k||^2} \right)
\]

\[< \left\| x^* - x_k \right\|^2.
\]

Hence, \{ \left\| x^* - x_k \right\|^2 \} is a bounded monotone decreasing sequence and is therefore convergent.

Consequently, we have, \( \lim_{k \to \infty} (f_k - \hat{w}_k)^2/||d_k||^2 = 0. \) From (2) and (7), this can happen only if both \( \{ f_k \} \to \tilde{w} \) and \( \{ \hat{w}_k \} \to \tilde{w}, \) and this completes the proof. □

It is interesting to note that this analysis requires \( \beta_k \leq 1, \) while the ordinary pure subgradient approach
with the step-length (4) permits $1 < \beta_k \leq \tilde{c} < 2$ as well. Now, observe that \{\{z_k\}\} is a monotone nonincreasing sequence that is bounded below by $f^*$, and is hence a convergent sequence. Let $\tilde{z}$ be the limit of this sequence, and consider the following results.

**Lemma 2.** For Algorithm VTVM, there exists an outer iteration $L$ such that $\varepsilon_L \equiv \varepsilon$ for all $\ell \geq L$.

**Proof.** Consider Algorithm VTVM for iterations $k \geq K$ such that $z_k < \tilde{z} + \delta$ for all $k \geq K$, where $0 < \delta < \varepsilon$. Let us examine any outer loop $\ell$ for which $k \geq K$. Given $w_\ell$ and $\varepsilon_\ell$, note that $w_{\ell+1}$ and $\varepsilon_{\ell+1}$ for the subsequent outer loop iteration are given by either (5) or (6). In the case of (5), having computed $w_{\ell+1} = z_{k+1} - \varepsilon_\ell - \eta \Delta_{k+1}$, and noting that $\eta \Delta_{k+1} \leq \Delta_{k+1} < \delta$ since all objective values are confined in $[\tilde{z}, \tilde{z} + \delta]$, we get

$$
(z_{k+1} - w_{\ell+1})\sigma = \sigma(\varepsilon_\ell + \eta \Delta_{k+1}) < \sigma(\varepsilon_\ell + \delta) < \sigma(\varepsilon_\ell + \varepsilon) \leq \frac{1}{2} \sigma.
$$

Similarly, in the case of (6), having computed $w_{\ell+1}$, we have that

$$
(z_{k+1} - w_{\ell+1})\sigma = \left[ z_{k+1} - \left( \frac{z_{k+1} - \varepsilon_\ell}{2} + \frac{w_{\ell+1}}{2} \right) \right] \sigma = \frac{\varepsilon_\ell \sigma}{2} + \frac{(z_{k+1} - w_{\ell+1})\sigma}{2}.
$$

**Case (i):** Suppose that $\varepsilon_\ell = \varepsilon$. Then from (5) or (6) at the previous outer loop for some iteration $k' < k$, we must have had $(z_{k' + 1} - w_{\ell})\sigma \leq \varepsilon$, which implies that

$$
(z_{k+1} - w_{\ell})\sigma \leq (z_{k' + 1} - w_{\ell})\sigma \leq \varepsilon.
$$

Now, if $\varepsilon_{\ell+1}$ is given by (5), we have from (10) that $\varepsilon_{\ell+1} = \varepsilon$. On the other hand, if $\varepsilon_{\ell+1}$ is given by (6), we have from (11) and (12) that $(z_{k' + 1} - w_{\ell})\sigma \leq \varepsilon/6 + \varepsilon/2 = 2\varepsilon/3$, and so $\varepsilon_{\ell+1} = \varepsilon$ as well. Hence, we have shown that $\varepsilon_\ell = \varepsilon$ implies that $\varepsilon_{\ell+1} = \varepsilon$ also, once $k \geq K$ for such outer loops.

**Case (ii):** Suppose that $\varepsilon_\ell > \varepsilon$. Hence, we have from (5) or (6) at the previous outer loop that for some $k' < k$,

$$
\varepsilon_\ell = (z_{k' + 1} - w_{\ell})\sigma \geq (z_{k+1} - w_{\ell})\sigma.
$$

Now, if $\varepsilon_{\ell+1}$ is given by (5), then we either have $\varepsilon_{\ell+1} = \varepsilon$, or else from (10), we get $\varepsilon_{\ell+1} \leq 2\varepsilon/3$. Similarly, if $\varepsilon_{\ell+1}$ is given by (6), then we either have $\varepsilon_{\ell+1} = \varepsilon$, or else from (11) and (13), we get $\varepsilon_{\ell+1} = \sigma(z_{k+1} - w_{\ell+1}) \leq \sigma \varepsilon/2 + 2\varepsilon/3 \leq 2\varepsilon/3$. In either case, \{\{\varepsilon_\ell\}\} decreases at least at a geometric rate, ultimately becoming equal to $\varepsilon$ at some finite outer loop iteration $L$, after which it remains at $\varepsilon$ from Case (i) above. This completes the proof. \qed

**Lemma 3.** Suppose that for Algorithm VTVM, we have $\tilde{z} - \varepsilon - f^* \geq \delta > 0$ for some $\delta$ satisfying $0 < \delta < \varepsilon$. Then there exists an outer iteration $\ell$ such that

$$
\tilde{z} - \varepsilon - \delta < w_\ell < \tilde{z} - \varepsilon + \delta \quad \text{for all } \ell \geq \ell.
$$

**Proof.** Following the proof of Lemma 2, let us consider the algorithmic process once we have $k \geq K$ that is sufficiently large so that $z_k < \tilde{z} + \delta$ for all $k \geq K$, and that the outer loop index $\ell \geq L$, where $L$ is large enough so that $\varepsilon_L \equiv \varepsilon$, for all $\ell \geq L$.

Suppose that $w_\ell \leq \tilde{z} - \varepsilon - \delta$ for some outer loop $\ell$. Then, $z_{k+1} > z_{k+1} - \delta \geq \tilde{z} - \delta \geq w_\ell + \varepsilon$ for all corresponding inner loop iteration $k$. Hence, the algorithm continues to increase $\tau$, and ultimately, increases the target value at Step 3(b). By successively increasing the target value according to (6) in this fashion, we will reach an outer iteration $\ell$ such that $w_\ell > \tilde{z} - \varepsilon - \delta$. Moreover, since any such increase via (6) while $w_\ell \leq \tilde{z} - \varepsilon - \delta$ and $z_{k+1} < \tilde{z} + \delta$ yields

$$
w_{\ell+1} < \frac{(\tilde{z} + \delta) - \varepsilon}{2} + (\tilde{z} - \varepsilon - \delta)
$$

we also have $w_\ell < \tilde{z} - \varepsilon + \delta$.

On the other hand, suppose that $w_\ell \geq \tilde{z} - \varepsilon + \delta$ for some outer loop $\ell \geq L$. Note that when $(k + 1) \geq K$, we have $z_{k+1} < \tilde{z} + \delta \leq w_\ell + \varepsilon \equiv w_\ell + \varepsilon$. But the condition $z_{k+1} < w_\ell + \varepsilon$ occurs in the algorithm only when an improvement in objective value at Step 2(a) causes $z_{k+1}$ to fall below $w_\ell + \varepsilon$. Therefore in this case, we must have had $\varepsilon_\ell = \varepsilon$ and $w_\ell \geq \tilde{z} - \varepsilon + \delta$ when for the first time after $k + 1 \geq K$, an improvement caused $z_{k+1}$ to fall below $\tilde{z} + \delta$, and hence below $w_\ell + \varepsilon$, thereby triggering a transfer to Step 3(a). Using (5) with $\varepsilon_\ell = \varepsilon$, the consequent decrease in the target value yields $w_{\ell+1} = z_{k+1} - \varepsilon - \eta \Delta_{k+1} < (\tilde{z} + \delta) - \varepsilon - \eta \Delta_{k+1} < \tilde{z} - \varepsilon + \delta$. Incrementing $\ell$ by 1, either this revised $w_\ell$ then satisfies the lower bound in (14) as well, or else, we have $w_\ell \leq \tilde{z} - \varepsilon - \delta$. In the latter
case, as above, we will again obtain (14) holding for some \( \ell \) after successive increases in the target value. Therefore, we have shown thus far that for some inner iteration \( \hat{k} > K \) during an outer iteration \( \hat{\ell} \), we have
\[
Z - e - \delta < w_{\ell} < Z - e + \delta.
\]
(15)
Now, it remains to show that (15) continues to hold for all \( \ell' \geq \ell \). Let \( j(1) \) be the first iteration after \( k \) at which the target value is changed at the next outer loop, so that either the target value is decreased via (5) or it is increased via (6) to yield, respectively,
\[
w_{\ell+1} = (z_{j(1)} - e) - \eta \Delta_{j(1)}
\]
or
\[
w_{\ell+1} = (z_{j(1)} - e) + w_{\ell}.
\]
For the first case in (16), we have \( \eta \Delta_{j(1)} < \Delta_{j(1)} < \delta \) because for \( k > K \) the objective value improves by less than \( \delta \). Hence, \( Z - e + \delta > w_{\ell} > w_{\ell+1} = (z_{j(1)} - e) - \eta \Delta_{j(1)} > Z - e - \delta \). The second case in (16) yields, using (15) that
\[
Z - e - \delta < w_{\ell} < w_{\ell+1} = \frac{(z_{j(1)} - e) + w_{\ell}}{2}
\]
\[
< \frac{(Z + \delta - e) + (Z - e + \delta)}{2} = Z - e + \delta.
\]
Hence, in either case, \( w_{\ell+1} \) continues to satisfy (15). By induction, this completes the proof. \( \square \)

**Theorem 1.** Algorithm VTVM generates a sequence \( \{z_k\} \rightarrow \bar{z} \), where \( Z - e \leq f^* \).

**Proof.** Consider the algorithm after the final restart has been performed. Assume on the contrary that \( Z - e > f^* \). We can therefore choose a \( \delta \) satisfying \( e > \delta > 0 \) such that \( Z - e - f^* > \delta > 0 \). By Lemma 3, we can find an outer iteration \( \hat{\ell} \) such that
\[
\bar{w} \equiv Z - e - \delta < w_{\ell} < Z - e + \delta = \bar{w}\text{ for all } \ell \geq \hat{\ell}.
\]
Since \( f^* \leq \bar{w} < \bar{w} < \bar{z} \leq f_k \) for all \( k \) and \( \ell \) sufficiently large, we get by Lemma 1 that \( \{f_k\} \rightarrow \bar{w} \). But this is a contradiction because \( \bar{w} < Z \) since \( e > \delta \), and so the proof is complete. \( \square \)

**Remark 3.** As evident from the proofs of Lemma 3 and Theorem 1, Algorithm VTVM can be operated with \( e \) held fixed at \( e \forall \ell \), and we would still have \( \{z_k\} \rightarrow \bar{z} \) where \( Z - e \leq f^* \). However, from the viewpoint of computational efficiency (as verified by our experiments), it is important to permit variable acceptance tolerances \( w_{\ell} \), as prescribed by the stated procedure. This is so because a small, fixed acceptance tolerance of \( \varepsilon_{\ell} = e \) can possibly lead to a sequence of increases in the target value until the gap \( (f_k - w_{\ell}) \) becomes relatively small, thereby resulting in small step-lengths via (4), and inducing a slow progress at iterates that are as yet remote from optimality. \( \square \)

4. Computational experience

We now present some preliminary computational results using 15 convex test problems from the literature. Table 1 gives the sizes and the sources of these problems, along with the standard, prescribed, starting solutions.

Table 2 gives the results obtained. All the algorithms tested were coded in C and run on an IBM RS/6000 computer. Run 1 corresponds to Algorithm VTVM with the ADS de/CRection strategy (see Section 1) and with a maximum limit of \( k_{\text{max}} = 2000 \) iterations. Note that we have used a fixed set of parameter values as prescribed in Remark 1, along with the strategies of Remarks 1 and 2, and with an initial lower bound \( LB = -\infty \) for all the runs. Run 2 corresponds to Kim et al.’s [9] algorithm run for 2000 iterations using their recommended parameters, including a strong convexity constant of 1 as used in their runs, and with an initial lower bound of \(-10^6 \) for all the problems. Run 3 is the same as Run 1, but with an additional improvement-based stopping criterion. Note that as stated, the algorithm is terminated whenever the iteration count exceeds \( k_{\text{max}} \) or when the norm of the current subgradient becomes sufficiently small. Additionally, we can terminate the algorithm based on its progress in improving the incumbent value. Hence, in Run 3, we terminate the algorithm when each of the following conditions holds. (i) \( k > 500 \), (ii) the algorithm has executed Step 3(a) (outer loop success iteration) at least once, and (iii) Step 3(b) is visited four consecutive times via Step 2(b), with the average of the relative improvements \( \Delta_{k+1}/(z_{k+1} - w_{\ell}) \) over these four visits being less than or equal to 0.05.

The results indicate that Algorithm VTVM is fairly robust and viable for a variety of problems, yielding near-optimal solutions with reasonable effort. This
Table 1
Test problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>n</th>
<th>Starting solution $x_1$</th>
<th>Source(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>(0, 0, 0, 0, 1)</td>
<td>Shor’s Problem [11]: Test 1</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>(1, …, 1)</td>
<td>Lemarechal and Mifflin [16, p. 151] (MAXQUAD)</td>
</tr>
<tr>
<td>3</td>
<td>50</td>
<td>$(i - (n + 1)/2, i = 1, …, n)$</td>
<td>Goffin’s polyhedral problem [11]: Test 3</td>
</tr>
<tr>
<td>4</td>
<td>48</td>
<td>(0, …, 0)</td>
<td>Lemarechal and Mifflin [16, p. 161] (TR48)</td>
</tr>
<tr>
<td>5</td>
<td>48</td>
<td>(0, …, 0)</td>
<td>Lemarechal and Mifflin [16, p. 165] (A48)</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>(0, …, 0)</td>
<td>Kiwiel [11]: Test 5</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>(0, …, 0)</td>
<td>Kiwiel [11]: Test 6</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>(0, 0, 0)</td>
<td>Streit’s Problem no. 1, Kiwiel [11]: Test 8</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>(0, 0, 0)</td>
<td>Streit’s Problem no. 2, Kiwiel [11]: Test 9</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>(0, 0, 0, 0, 0)</td>
<td>Streit’s Problem no. 3, Kiwiel [11]: Test 10</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>(0, 0, 0, 1)</td>
<td>Hock and Schittkowski [8, p. 105]</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>(0, 0, 0)</td>
<td>Hock and Schittkowski [8, p. 66]</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>(15, 22, 26, 11)</td>
<td>Chatelon et al.’s Minimax Location Problem [11]: Test 13</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>(0, 0, 0, 0, 0)</td>
<td>Chatelon et al.’s Minimax Location Problem [11]: Test 14</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>(0, …, 0)</td>
<td>Kiwiel [11]: Test 7</td>
</tr>
</tbody>
</table>

Table 2
Computational results for runs 1, 2, and 3

<table>
<thead>
<tr>
<th>Problem</th>
<th>$f(x^*)$</th>
<th>Run 1: VTVM+ADS</th>
<th>Run 2: Kim et al. (1991)</th>
<th>Run 3: Run 1+Improvement-based stopping criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f(x_{best})$</td>
<td>(cpu s)</td>
<td>$f(x_{best})$</td>
<td>(cpu s)</td>
</tr>
<tr>
<td>1</td>
<td>-22.60016</td>
<td>22.600162</td>
<td>0.05</td>
<td>-22.685788</td>
</tr>
<tr>
<td>2</td>
<td>-0.841408</td>
<td>-0.801792</td>
<td>0.10</td>
<td>-0.786540</td>
</tr>
<tr>
<td>3</td>
<td>0.0</td>
<td>0.000002</td>
<td>0.06</td>
<td>481.154344</td>
</tr>
<tr>
<td>4</td>
<td>-63565</td>
<td>-626175.71</td>
<td>1.15</td>
<td>-560439.57</td>
</tr>
<tr>
<td>5</td>
<td>-9870</td>
<td>-9870.00</td>
<td>1.12</td>
<td>-9546.11</td>
</tr>
<tr>
<td>6</td>
<td>0.0</td>
<td>0.025849</td>
<td>0.67</td>
<td>1.055688</td>
</tr>
<tr>
<td>7</td>
<td>0.0</td>
<td>0.398131</td>
<td>0.18</td>
<td>0.058670</td>
</tr>
<tr>
<td>8</td>
<td>0.077107</td>
<td>0.707107</td>
<td>0.05</td>
<td>0.794324</td>
</tr>
<tr>
<td>9</td>
<td>1.014214</td>
<td>1.014214</td>
<td>0.04</td>
<td>1.014360</td>
</tr>
<tr>
<td>10</td>
<td>0.014706</td>
<td>0.128475</td>
<td>0.67</td>
<td>0.119763</td>
</tr>
<tr>
<td>11</td>
<td>-32.348679</td>
<td>-32.3254</td>
<td>0.04</td>
<td>-28.144825</td>
</tr>
<tr>
<td>12</td>
<td>-44.0</td>
<td>-43.950579</td>
<td>0.04</td>
<td>-43.971590</td>
</tr>
<tr>
<td>13</td>
<td>23.886767</td>
<td>24.818542</td>
<td>0.42</td>
<td>26.708094</td>
</tr>
<tr>
<td>14</td>
<td>68.82856</td>
<td>68.831859</td>
<td>0.60</td>
<td>68.836753</td>
</tr>
<tr>
<td>15</td>
<td>-0.368166</td>
<td>-0.339827</td>
<td>0.13</td>
<td>-0.299131</td>
</tr>
</tbody>
</table>

Legend: $f(x^*)$: optimal objective value.
$f(x_{best})$: best objective value obtained by the corresponding algorithm.
cpu s: total execution time (in s) on an IBM RS/6000 computer.
Iters: total number of iterations until termination occurred for Run 3.

might be acceptable in Lagrangian relaxation applications, for example. Moreover, it is simple to implement. In contrast, the bundle method implemented in Kiwiel [11] typically yields more accurate solutions in significantly fewer iterations, although each iteration is more complex in that it requires the solution of a quadratic program.

The results for Run 2 indicate that Kim et al.’s algorithm is quite sensitive to the strong convexity assumption, which does not necessarily hold for these
test problems. The results for Run 3 indicate that the improvement-based stopping criterion prescribed above offers a reasonable alternative to the criterion based simply on the maximum number of iterations. Note that for most test problems, progress that is acceptable to this criterion continues until close to the limit of 2000 iterations.

Finally, we comment that for Run 1, we also attempted the MGT and the pure subgradient strategies (see Section 1). The ADS strategy performed the same or better than the MGT (respectively, the pure subgradient) strategy on 12 (respectively, 13) out of the 15 test problems. Also, Algorithm VTVM performed the same or better on 13 out of the 15 test problems than its variant in which $\varepsilon$ is held fixed at the value $\varepsilon$ (see Remark 3). In addition, we attempted to solve some larger sized, randomly generated, dual assignment test problems. For example, using test cases of sizes $200 \times 200$, $250 \times 250$, and $300 \times 300$ having optimal values $-2157$, $-2697$, and $-3246$, respectively, Algorithm VTVM terminated within 500 iterations in each case, finding solutions of objective values $-2156.46$, $-2696.78$, and $-3245.88$, consuming a total of 26.8, 42.4 and 61.3 cpu s, respectively.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. DMI-9521398 and DMI-9812047 and the Air Force Office of Scientific Research under Grant No. F-49620-96-1-0274.

References