Optimization Letters (2022) 16:1495–1511
https://doi.org/10.1007/s11590-021-01787-0

ORIGINAL PAPER

Convergence of the proximal bundle algorithm for nonsmooth nonconvex optimization problems

N. Hoseini Monjezi[1] · S. Nobakhtian[2]

Received: 23 September 2020 / Accepted: 17 July 2021 / Published online: 2 August 2021

Abstract
A proximal bundle algorithm is proposed for solving unconstrained nonsmooth nonconvex optimization problems. At each iteration, using already generated information, the algorithm defines a convex model of the augmented objective function. Then by solving a quadratic subproblem a new candidate iterate is obtained and the algorithm is repeated. The novelty in our approach is that the objective function can be any arbitrary locally Lipschitz function without any additional assumptions. The global convergence, starting from any point, is also studied. At the end, some encouraging numerical results with a MATLAB implementation are reported.

Keywords Proximal bundle method · Nonsmooth optimization · Nonconvex optimization · Global convergence

1 Introduction

Consider the following unconstrained optimization problem

$$\begin{aligned} \min \quad & f(x), \\ & x \in \mathbb{R}^n, \end{aligned} \tag{1}$$

where $\mathbb{R}^n$ denotes the n-dimensional Euclidean space and $f : \mathbb{R}^n \to \mathbb{R}$ is a locally Lipschitz function, but possibly nonsmooth and nonconvex.

✉ S. Nobakhtian
nobakht@math.ui.ac.ir

N. Hoseini Monjezi
najmeh.hoseini@sci.ui.ac.ir

1 School of Mathematics, Institute for Research in Fundamental Sciences (IPM), Tehran 19395-5746,, Iran

2 Department of Applied Mathematics and Computer Science, Faculty of Mathematics and Statistics, University of Isfahan, Isfahan, Iran

Several numerical methods have been proposed for solving nonsmooth optimization problems. These methods include subgradient algorithms [1,3,23,24], bundle methods [5,8,9,11,12,20], gradient sampling algorithms [4,19], the trust region algorithm [10] and the conjugate gradient method [16].

Bundle methods were first introduced by Lemaréchal [20] and have been developed over the years for various problems. Proximal bundle methods are currently considered among the most efficient methods for nonsmooth problems, see for convex problems [21] and the nonconvex case [18]. Recently the proximal bundle algorithm is developed in [8] for the problem (1) with lower $-C^2$ and in [9] with lower $-C^1$ objective functions. Moreover this method is generalized for the problem (1) with lower $-C^1$ and upper $-C^1$ objective functions in [6] and with DC objective functions in [17]. In addition, this method is extended for constrained optimization problems in [5] with lower $-C^1$ or upper $-C^1$ functions and in [11,12] with regular functions. More recently, the proximal bundle algorithm is studied for nonsmooth multiobjective problems in [13,14]

In this paper, we study the redistributed proximal bundle method of [8,9] to the class of all locally Lipschitz functions, which is less restrictive than lower $-C^2$, lower $-C^1$ and upper $-C^1$ assumptions. We strengthen the existing convergence results for this algorithm and introduce a slightly revised version for which convergence results are established for locally Lipschitz objective function without requiring any additional assumptions. To handle nonconvexity, the augmented objective function and its corresponding piecewise linear approximation are used. Iterates of the proximal bundle algorithm are generated by solving a quadratic programming subproblem. Each quadratic programming is defined by means of the piecewise linear model of the augmented objective function, stabilized by a quadratic term centered at the best point obtained so far (which is referred to the latest serious step). The quadratic term is added to guarantee the existence and uniqueness of the minimum point and also to keep the approximation local enough. At the end by minimizing the obtained model a trial iterate and new bundle elements are obtained. Different from [8,9,12], we do not require any lower-C^2, lower-C^1 or regularity assumption and the augmented function can be convex or nonconvex. Moreover unlike [8,9], we employ the upper envelope model of [5,11,12] and by utilizing it we connect the convex model with the original nonconvex problem to analyze the global convergence. This modification and also employing different techniques in the convergence theory enable us to drive convergence results with weaker assumptions than [8,9]. We prove that if the algorithm stops with a finite number of iterations, then the latest serious iterate is a stationary point. On the other hand, if the algorithm generates a sequence of iterates, then two cases may happen. (I) If we have a finite number of serious iterates with infinite number of null iterates, then the latest serious iterate is a stationary point. (II) If we obtain an infinite number of serious iterates, then every accumulation point of this sequence is a stationary point. At the end, the algorithm is implemented in the MATLAB environment and applied on some nonsmooth test problems. Numerical results illustrate the efficiency of the proximal bundle algorithm in the practical computation.

Throughout the paper, we use the following notation and definitions. We denote by $\langle u, v\rangle := \sum_{i=1}^{n} u_i v_i$ the inner product of two vectors $u, v \in \mathbb{R}^n$ and by $\| \cdot \|$ the standard Euclidean norm. For $x \in \mathbb{R}^n$ and $\varepsilon > 0$, $B_\varepsilon(x)$ ($\bar{B}_\varepsilon(x)$) is an open (closed) ball of the radius ε centered at x.

The function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $f(\lambda x + (1-\lambda)y) \le \lambda f(x) + (1-\lambda) f(y)$, for all $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$. The subdifferential of a convex function f at x is given by $\partial_c f(x) := \{\xi \in \mathbb{R}^n | \ f(y) \ge f(x) + \langle \xi, y - x\rangle, \ \forall y \in \mathbb{R}^n\}$. For any $\varepsilon \ge 0$, the $\varepsilon-$subdifferential [21] of a convex function f at x is defined as

$$\partial_\varepsilon f(x) := \{\xi \in \mathbb{R}^n | \ f(y) \ge f(x) + \langle \xi, y - x\rangle - \varepsilon, \ \forall y \in \mathbb{R}^n\}. \tag{2}$$

A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be locally Lipschitz of rank $L > 0$ at $x \in \mathbb{R}^n$ if for some $\varepsilon > 0$ we have $|f(y) - f(z)| \le L\|y - z\|$ for all $y, z \in B_\varepsilon(x)$. The Clarke subdifferential (generalized gradient) of f at x is defined as $\partial f(x) := \{\xi \in \mathbb{R}^n | \ \langle \xi, d\rangle \le f^\circ(x; d), \ \forall d \in \mathbb{R}^n\}$ and it coincides with the convex subdifferential for every convex function, where $f^\circ(x; d)$ is a Clarke directional derivative. Each element $\xi \in \partial f(x)$ is called a subgradient of f at x. It is well-known that $\partial f(x)$ is a nonempty convex compact set in $\mathbb{R}^n$. Also the Clarke subdifferential $\partial f(x)$ is upper semicontinuous at every $x \in \mathbb{R}^n$.

This paper is organized as follows. The new algorithm and its convergence analysis are described in Sect. 2. The corresponding encouraging computational results are reported in Sect. 3.

2 The proximal bundle algorithm

Suppose that x^l is the latest serious iterate, l and k are the serious iteration counter and the iteration counter, respectively. We use the notations $\mathcal{L}_k$ and $\{y_j\}_{j\in\mathcal{L}_k}$ to denote the index set and the set of bundle points, respectively. Assume that the latest serious iterate will be one of the bundle points: $x^l \in \{y_j\}_{j\in\mathcal{L}_k}$. As usual in the bundle methods, generated information is used to obtain a piecewise linear model and also a new iterate. Here f is locally Lipschitz, therefore it is possibly nonconvex. Motivated by the presented method in [8,9,11,12] for nonconvex cases, we use the augmented function as follows $f_{\eta_k}(\cdot, x^l) := f(\cdot) + \frac{\eta_k}{2}\| \cdot - x^l\|^2$, where $\eta_k \in \mathbb{R}$ is a positive parameter, that is adjusted dynamically.

The piecewise linear model for the augmented function f_{η_k} is formed at the kth iteration as follows:

$$M^k(y, x^l) := f(x^l) + \max_{j\in\mathcal{L}_k}\{-c^l_j + \langle \xi^l_j, y - x^l\rangle\}, \tag{3}$$

where $\xi_j \in \partial f(y_j)$, $e^l_j := f(x^l) - f(y_j) - \langle \xi_j, x^l - y_j\rangle$, $c^l_j := e^l_j + \eta_k b^l_j$, $\xi^l_j := \xi_j + \eta_k d^l_j$, $d^l_j := y_j - x^l$ and $b^l_j := \frac{\|y_j - x^l\|^2}{2}$, the index set at the kth iteration is $\mathcal{L}_k \subseteq \{1, 2, \ldots, k\}$ and the bundle of information is defined as $\mathcal{B}_k \subseteq \bigcup_{j\in\mathcal{L}_k}\left\{\left(\xi_j, e^l_j, b^l_j, d^l_j\right)\right\}$. Our aim is to keep c^l_j, $j \in \mathcal{L}_k$ nonnegative, for this purpose in our setting we take

$$\eta_k \geq \max\{\max_{j\in\mathcal{L}_k, y_j\neq x^l} \frac{-2e_j^l}{\|y_j - x^l\|^2}, \omega\} + \omega, \tag{4}$$

where $\omega > 0$ is a positive constant. The parameter η_k is chosen motivated by [8,9,12]. By using (4), for all $j \in \mathcal{L}_k$ such that $y_j \neq x^l$, we have $c_j^l = e_j^l + \frac{\eta_k}{2}\|y_j - x^l\|^2 \geq \frac{\omega}{2}\|y_j - x^l\|^2 \geq 0$. On the other hand, if $y_j = x^l$ we deduce $c_j^l = 0$. Therefore $c_j^l \geq 0$ for all $j \in \mathcal{L}_k$. This term implies that the augmented linearization errors, i.e., $c_j^l = e_j^l + \eta_k b_j^l$ remain positive for all $j \in \mathcal{L}_k$ and $y_j \neq x^l$, and in addition we have

$$c_j^l = e_j^l + \eta_k b_j^l \geq \frac{\omega}{2}\|y_j - x^l\|^2. \tag{5}$$

The relation (5) is the main key in the proof of parts (ii) and (iv) of Lemma 1. Taking the maximum of the term in (4) with ω yields positivity of η_k, and adding the positive parameter ω makes η_k satisfy (5).

Since the latest serious iterate is one of the bundle points, it follows that there exists $j(l) \in \mathcal{L}_k$ such that $x^l = y_{j(l)}$. So we get $M^k(x^l, x^l) = f(x^l) + \max_{j\in\mathcal{L}_k}\{-c_j^l\} = f(x^l)$. In addition by (3) and $M^k(x^l, x^l) = f(x^l)$ we obtain $M^k(y, x^l) \geq M^k(x^l, x^l) + \langle \xi_j^l, y - x^l\rangle - c_j^l$, for all $y \in \mathbb{R}^n$ and $j \in \mathcal{L}_k$. Using the definition of the $\varepsilon-$subdifferential in (2) we conclude

$$\xi_j^l \in \partial_{c_j^l} M^k(x^l, x^l), \quad \forall j \in \mathcal{L}_k. \tag{6}$$

To generate the next iterate y_{k+1}, our bundle method chooses a proximal parameter $\mu_k > 0$ and solves the following quadratic program

$$\min_{y\in\mathbb{R}^n} \quad M^k(y, x^l) + \frac{\mu_k}{2}\|y - x^l\|^2. \tag{7}$$

Clearly y_{k+1} is unique, since the objective function is strictly convex. Set $d_{k+1} := y_{k+1} - x^l$ and $v_{k+1} := M^k(y_{k+1}, x^l) - f(x^l)$. If $y_{k+1} = x^l$, then $v_{k+1} = 0$ and the algorithm stops. Therefore we assume $y_{k+1} \neq x^l$. By uniqueness of y_{k+1} as the solution of the problem (7), we get $M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|y_{k+1} - x^l\|^2 < M^k(x^l, x^l) + \frac{\mu_k}{2}\|0\|^2 = M^k(x^l, x^l) = f(x^l)$. On the other hand, using $\frac{\mu_k}{2}\|y_{k+1} - x^l\|^2 \geq 0$, we have $M^k(y_{k+1}, x^l) < f(x^l)$ and $v_{k+1} < 0$.

The problem (7) can be rewritten in the following smooth form

$$\begin{aligned} \min \quad & v + \frac{\mu_k}{2}\|y - x^l\|^2 \\ & \langle \xi_j^l, y - x^l\rangle - c_j^l \leq v, \quad \forall j \in \mathcal{L}_k, \\ & y \in \mathbb{R}^n, \ v \in \mathbb{R}. \end{aligned} \tag{8}$$

The quadratic dual problem of the problem (8) is formulated as follows:

$$\begin{aligned} \min \quad & \frac{1}{2\mu_k} \| \sum_{j \in \mathcal{L}_k} \lambda_j \xi_j^l \|^2 + \sum_{j \in \mathcal{L}_k} \lambda_j c_j^l \\ & \sum_{j \in \mathcal{L}_k} \lambda_j = 1, \ \lambda_j \geq 0, \quad \forall j \in \mathcal{L}_k. \end{aligned} \tag{9}$$

If λ_j for all $j \in \mathcal{L}_k$ solve the problem (9), then a unique solution of the problem (8) (using the relationship between the primal and dual solutions) is obtained in the form

$$d_{k+1} = -\frac{1}{\mu_k} \sum_{j \in \mathcal{L}_k} \lambda_j \xi_j^l, \ \text{ and } \ v_{k+1} = -\Big(\frac{1}{\mu_k} \| \sum_{j \in \mathcal{L}_k} \lambda_j \xi_j^l \|^2 + \sum_{j \in \mathcal{L}_k} \lambda_j c_j^l \Big), \tag{10}$$

where $y_{k+1} = d_{k+1} + x^l$.

Now the new trial iterate is computed (i.e., y_{k+1}), we first check whether it provides sufficient decrease of the objective function as compared to the latest serious iterate. If the descent is sufficient, then the corresponding point is declared as a new serious iterate (a so-called serious iteration). More precisely, when y_{k+1} satisfies the sufficient descent test

$$f(y_{k+1}) \leq f(x^l) + m_L v_{k+1}, \tag{11}$$

where $m_L \in (0, 1)$, then we have a new serious iterate and set $x^{l+1} = y_{k+1}$. Therefore the model $M^k(\cdot, x^l)$ should be updated and any element in $\mathcal{B}_k$ that is depended on l should be redefined. Motivated by the formulae (12) in [8], we update the elements in $\mathcal{B}_k$ according to the following relations:

$$e_j^{l+1} = e_j^l + f(x^{l+1}) - f(x^l) - \langle \xi_j, x^{l+1} - x^l \rangle, \tag{12a}$$

$$b_j^{l+1} = b_j^l + \frac{1}{2} \|x^{l+1} - x^l\|^2 - \langle d_j^l, x^{l+1} - x^l \rangle, \tag{12b}$$

$$d_j^{l+1} = d_j^l - (x^{l+1} - x^l). \tag{12c}$$

If (11) does not hold, then y_{k+1} is a null iterate and the latest serious iterate x^l remains unchanged (a so-called null iteration). In this case the model will be improved by adding new information to the bundle. For this purpose we update the index set and the bundle, that is $\mathcal{L}_{k+1} = \mathcal{L}_k \bigcup \{k+1\}$ and also $\mathcal{B}_{k+1} = \mathcal{B}_k \bigcup \{(\xi_{k+1}, e_{k+1}^l, b_{k+1}^l, d_{k+1}^l)\}$. After that a new iterate y_{k+2} can be calculated and the algorithm is repeated. We perform null steps until (11) is satisfied and the sufficient descent is reached.

Now we are in position to state the proximal bundle algorithm to solve the problem (1).

Algorithm 1 : The proximal bundle algorithm

```
1: Initialization: Choose the parameter $m_L \in (0,1)$ and the stopping tolerance tol $\geq 0$. Choose a starting point $y_1 \in \mathbb{R}^n$, set $x^1 := y_1$, and compute $f(y_1)$ and $\xi_1 \in \partial f(y_1)$. Set $k := 1$, $l := 1$, $\mathcal{L}_1 := \{1\}$, $\mathcal{B}_1 = \{(\xi_1, e_1^1, d_1^1, b_1^1)\}$.
2: New point generation and stopping test: Select $\eta_k > 0$ as in (4) and using (3) formulate $M^k(y, x^l)$. Select a proximal parameter $\mu_k > 0$ and by solving the subproblem (7) obtain $y_{k+1}$. Compute $v_{k+1}$, if $-v_{k+1} \leq$ tol, stop.
3: Descent condition and improving the bundle: Compute $f(y_{k+1})$. If (11) satisfies, then $y_{k+1}$ is a serious iterate. Otherwise, $y_{k+1}$ is a null iterate. Set $\mathcal{L}_{k+1} = \mathcal{L}_k \cup \{k+1\}$ and $\mathcal{B}_{k+1} = \mathcal{B}_k \cup \{(\xi_{k+1}, e_{k+1}^l, d_{k+1}^l, b_{k+1}^l)\}$.
4: Updating the bundle: If $y_{k+1}$ is a serious iterate, then set $x^{l+1} = y_{k+1}$, $f(x^{l+1}) = f(y_{k+1})$ and $l = l+1$. Update the bundle elements $\mathcal{B}_k$ according to (12a)–(12c).
5: Loop: Set $k = k+1$ and go to Step 2.
```

Remark 1 In the sequel, we suppose that $\{\eta_k\}_k$ is bounded. Using (4), we deduce that $\eta_k \geq 2\omega$ for all k, therefore we assume that there exists $\bar{\eta}$ such that $2\omega \leq \eta_k \leq \bar{\eta}$ for all k. Since the value of $\bar{\eta}$ is not needed in the performance and the analysis of the algorithm, this assumption is not restrictive on the implementation of the algorithm. The boundedness of $\{\eta_k\}_k$ has been considered in [9] with lower $-C^1$ functions and in [11,12] with regular functions. However we consider this assumption with locally Lipschitz functions.

For the analytical purpose, motivated by [5,11,12,22], we define the upper envelope model associated with the cutting plane. Assume that $\bar{B}_R(x)$ is a fixed closed ball large enough such that it contains all possible trial steps y^+. Set $\mathcal{B}(x) := \{y^+ |\ y^+ \in \bar{B}_R(x),\ y^+ \text{ is a trial point}\}$. The upper envelope model $M^\uparrow(\cdot, x) : \mathbb{R}^n \to \mathbb{R}$ is defined, for the objective function f corresponding with a given point $x \in \mathbb{R}^n$, as

$$M^\uparrow(y, x) := f(x) + \sup_{2\omega \leq \eta \leq \bar{\eta},\ y^+ \in \mathcal{B}(x),\ \xi \in \partial f(y^+)} \{m_{y^+,\xi,\eta}(y, x)\},$$

where $2\omega \leq \eta \leq \bar{\eta}$ with $\bar{\eta}$ defined in Remark 1. The plane $m_{y^+,\xi,\eta}(y, x)$ is the cutting plane at the serious iterate x and the trial step y^+ which is defined by $m_{y^+,\xi,\eta}(y, x) := -\frac{\omega}{2}\|y^+ - x\|^2 + \langle \xi + \eta(y^+ - x), y - x \rangle$. The boundedness of $\bar{B}_R(x)$ and the definition of η imply that $M^\uparrow(\cdot, x)$ is defined everywhere. Some beneficial properties of the upper envelope model $M^\uparrow(\cdot, x)$ are stated in Lemma 1. These results can be found in [5, Lemma 6.1], however the definition of the upper envelope model $M^\uparrow(\cdot, x)$ and its cutting planes are different from [5]. The proof of these results follows immediately from [12, Lemma 5] with slight modifications.

Lemma 1 *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is a locally Lipschitz function, then the following hold:*

(i) $M^\uparrow(\cdot, x)$ *is a convex function.*
(ii) $M^k(\cdot, x) \leq M^\uparrow(\cdot, x)$, $\forall k$.
(iii) $M^\uparrow(x, x) = f(x)$.
(iv) $\partial_c M^\uparrow(x, x) \subseteq \partial f(x)$.

Now we examine the convergence properties of Algorithm 1. We consider different cases that may happen during the implementation of it and in each case we state the results.

Theorem 1 *Assume that f is a locally Lipschitz function, $tol = 0$ and $\mu_k > 0$, for all k. If Algorithm 1 stops with a finite number of iterations, then the latest serious iterate x^l is a stationary point for the problem (1).*

Proof According to the assumption, Algorithm 1 stops with a finite number of iterations. This can happen at Step 2 with $-v_{k+1} \leq 0$ and since $-v_{k+1} \geq 0$ we obtain $-v_{k+1} = 0$. By definition $-v_{k+1} = \frac{1}{\mu_k}\|\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l\|^2 + \sum_{j\in\mathcal{L}_k}\lambda_j c_j^l$ and using $\mu_k > 0$, we deduce

$$\|\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l\|^2 = 0, \quad \text{and} \quad \sum_{j\in\mathcal{L}_k}\lambda_j c_j^l = 0. \tag{13}$$

By (6) we have $\xi_j^l \in \partial_{c_j^l} M^k(x^l, x^l)$, and consequently $M^k(y, x^l) \geq M^k(x^l, x^l) + \langle\xi_j^l, y - x^l\rangle - c_j^l$, for all $y \in \mathbb{R}^n$. Taking into account that $\lambda_j \geq 0$ and $\sum_{j\in\mathcal{L}_k}\lambda_j = 1$, we get $M^k(y, x^l) \geq M^k(x^l, x^l) + \langle\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l, y - x^l\rangle - \sum_{j\in\mathcal{L}_k}\lambda_j c_j^l$. By $M^k(x^l, x^l) = f(x^l)$ and Lemma 1 (ii)–(iii), We conclude that $M^\uparrow(y, x^l) \geq M^\uparrow(x^l, x^l) + \langle\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l, y - x^l\rangle - \sum_{j\in\mathcal{L}_k}\lambda_j c_j^l$. Using (13) and Lemma 1 (i), (iv), we get $0 \in \partial_c M^\uparrow(x^l, x^l)$ and also $0 \in \partial f(x^l)$. Consequently x^l, i.e., the latest serious iterate is a stationary point and the proof is completed. □

From now on, we assume that Algorithm 1 does not stop and generates an infinite sequence of iterates. We consider different cases that may happen during the performance of Algorithm 1 with infinite cycles.

Theorem 2 *Assume that f is a locally Lipschitz function, $\{\eta_k\}_k$ is bounded above and $\mu_k \leq \mu_{k+1} \leq \bar{\mu}$ for all k. If Algorithm 1 is performed infinitely with a finite number of serious iterations, then the latest serious iterate x^l is a stationary point for the problem (1).*

Proof Assume that Algorithm 1 produced a finite number of serious iterates that follows with infinite number of null iterates. Suppose that y_{k+1} is the optimal solution of the problem (8) and $d_{k+1} = y_{k+1} - x^l$. In this process the latest serious point x^l is constant and we demonstrate that it is a stationary point. First, we show that the sequence $\{M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2\}_k$ is bounded above and nondecreasing.

Since the problem (7) is strictly convex, it follows that its solution (i.e., y_{k+1}) is unique. Thus $M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2 < M^k(y, x^l) + \frac{\mu_k}{2}\|y - x^l\|^2$, for all $y \in \mathbb{R}^n$. Now set $y = x^l$, then $M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2 < M^k(x^l, x^l) = f(x^l)$. Therefore the sequence $\{M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2\}_k$ is bounded from above by $f(x^l)$. It is good to note that x^l is constant here. Next let us prove that this sequence is nondecreasing

$$
\begin{aligned}
&M^{k+1}(y_{k+2}, x^l) + \frac{\mu_{k+1}}{2}\|d_{k+2}\|^2 \\
&\geq M^{k+1}(y_{k+2}, x^l) + \frac{\mu_k}{2}\|d_{k+2}\|^2 \\
&\geq f(x^l) - c_j^l + \langle \xi_j^l, y_{k+2} - x^l\rangle + \frac{\mu_k}{2}\|d_{k+2}\|^2 \\
&= f(x^l) - c_j^l + \langle \xi_j^l, d_{k+2} - d_{k+1}\rangle + \langle \xi_j^l, d_{k+1}\rangle + \frac{\mu_k}{2}\|d_{k+2} - d_{k+1} + d_{k+1}\|^2 \\
&= f(x^l) - c_j^l + \langle \xi_j^l, d_{k+2} - d_{k+1}\rangle + \langle \xi_j^l, d_{k+1}\rangle \\
&\quad + \frac{\mu_k}{2}\|d_{k+2} - d_{k+1}\|^2 + \frac{\mu_k}{2}\|d_{k+1}\|^2 + \mu_k\langle d_{k+2} - d_{k+1}, d_{k+1}\rangle,
\end{aligned}
$$

where the first inequality follows from $\mu_k \leq \mu_{k+1}$, the second inequality holds by the definition of $M^{k+1}(\cdot, x^l)$ and the other relations are obvious. The above relation is satisfied for all $j \in \mathcal{L}_{k+1}$ and also $\mathcal{L}_{k+1} = \mathcal{L}_k \bigcup\{k+1\}$. For all $j \in \mathcal{L}_k$, multiplying each relation with corresponding λ_j, summing up and by using the fact that $\sum_{j\in\mathcal{L}_k}\lambda_j = 1$, we arrive at

$$
\begin{aligned}
M^{k+1}(y_{k+2}, x^l) + \frac{\mu_{k+1}}{2}\|d_{k+2}\|^2 &\geq f(x^l) - \sum_{j\in\mathcal{L}_k}\lambda_j c_j^l + \langle \sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l, d_{k+2} - d_{k+1}\rangle \\
+ \langle \sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l, d_{k+1}\rangle &+ \frac{\mu_k}{2}\|d_{k+2} - d_{k+1}\|^2 + \frac{\mu_k}{2}\|d_{k+1}\|^2 + \mu_k\langle d_{k+2} - d_{k+1}, d_{k+1}\rangle.
\end{aligned}
$$

Since $d_{k+1} = -\frac{1}{\mu_k}\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l$, we have

$$
\begin{aligned}
&M^{k+1}(y_{k+2}, x^l) + \frac{\mu_{k+1}}{2}\|d_{k+2}\|^2 \\
&\geq f(x^l) - \sum_{j\in\mathcal{L}_k}\lambda_j c_j^l + \frac{\mu_k}{2}\|d_{k+1}\|^2 + \langle \sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l, -\frac{1}{\mu_k}\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l\rangle \\
&\quad + \frac{\mu_k}{2}\|d_{k+2} - d_{k+1}\|^2 \\
&= f(x^l) - \Big(\sum_{j\in\mathcal{L}_{,k}}\lambda_j c_j^l + \frac{1}{\mu_k}\|\sum_{j\in\mathcal{L}_k}\lambda_j\xi_j^l\|^2\Big) + \frac{\mu_k}{2}\|d_{k+1}\|^2 + \frac{\mu_k}{2}\|d_{k+2} - d_{k+1}\|^2 \\
&= f(x^l) + v_{k+1} + \frac{\mu_k}{2}\|d_{k+1}\|^2 + \frac{\mu_k}{2}\|d_{k+2} - d_{k+1}\|^2 \\
&= M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2 + \frac{\mu_k}{2}\|d_{k+2} - d_{k+1}\|^2 \\
&\geq M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2.
\end{aligned}
$$

Therefore the sequence $\{M^k(y_{k+1}, x^l) + \frac{\mu_k}{2}\|d_{k+1}\|^2\}_k$ is nondecreasing and by its boundedness, we deduce this sequence is convergent. Passing to the limit in the above inequality when $k \to \infty$ we have $\lim_{k\to\infty}\frac{\mu_k}{2}\|d_{k+1} - d_k\|^2 \leq 0$. By assumption that $\mu_k \leq \mu_{k+1} \leq \bar{\mu}$, we obtain $\mu_k \geq \mu_{k^*}$ that is $\{\mu_k\}_k$ is bounded below by

μ_{k^*}. This implies that $d_{k+1} - d_k \to 0$, as $k \to \infty$ and thus $\{d_k\}_k$ is bounded. Using the definition of $M^{k+1}(y_{k+2}, x^l)$, for all $j \in \mathcal{L}_{k+1}$ we have $M^{k+1}(y_{k+2}, x^l) \geq f(x^l) - c_j^l + \langle \xi_j^l, y_{k+2} - x^l \rangle$. Set $j = k+1$ in this relation, we obtain

$$\begin{aligned}
M^{k+1}(y_{k+2}, x^l) &\geq f(x^l) - c_{k+1}^l + \langle \xi_{k+1}^l, d_{k+2} \rangle \\
&= f(y_{k+1}) + \langle \xi_{k+1}, x^l - y_{k+1} \rangle - \frac{\eta_{k+1}}{2} \| y_{k+1} - x^l \|^2 + \langle \xi_{k+1}^l, d_{k+2} \rangle \\
&= f(y_{k+1}) - \langle \xi_{k+1}, d_{k+1} \rangle - \frac{\eta_{k+1}}{2} \| d_{k+1} \|^2 + \langle \xi_{k+1}^l, d_{k+2} \rangle \\
&\geq f(y_{k+1}) - \langle \xi_{k+1}, d_{k+1} \rangle - \eta_{k+1} \| d_{k+1} \|^2 + \langle \xi_{k+1}^l, d_{k+2} \rangle \\
&= f(y_{k+1}) - \langle \xi_{k+1}, d_{k+1} \rangle - \eta_{k+1} \langle y_{k+1} - x^l, d_{k+1} \rangle + \langle \xi_{k+1}^l, d_{k+2} \rangle \\
&= f(y_{k+1}) - \langle \xi_{k+1} + \eta_{k+1}(y_{k+1} - x^l), d_{k+1} \rangle + \langle \xi_{k+1}^l, d_{k+2} \rangle \\
&= f(y_{k+1}) + \langle \xi_{k+1}^l, d_{k+2} - d_{k+1} \rangle \\
&> f(x^l) + m_L v_{k+1} + \langle \xi_{k+1}^l, d_{k+2} - d_{k+1} \rangle.
\end{aligned}$$

By the definition of v_{k+1} we have $v_{k+2} = M^{k+1}(y_{k+2}, x^l) - f(x^l)$ and get

$$0 \leq -v_{k+2} = f(x^l) - M^{k+1}(y_{k+2}, x^l) < -m_L v_{k+1} - \langle \xi_{k+1}^l, d_{k+2} - d_{k+1} \rangle.$$

Since $d_{k+2} - d_{k+1} \to 0$, as $k \to \infty$, $m_L \in (0, 1)$ and $\{\xi_{k+1}^l\}_k$ is bounded (since $\{d_{k+1}^l\}_k$ and $\{\eta_k\}_k$ are bounded and f is locally Lipschitz), it follows that $-v_{k+1} \to 0$. By the definition of v_{k+1} in (10) and $c_j^l \geq 0$ for all $j \in \mathcal{L}_k$ and also $\mu_k \leq \bar{\mu}$ we get

$$\sum_{j \in \mathcal{L}_k} \lambda_j \xi_j^l \to 0, \quad \text{and} \quad \sum_{j \in \mathcal{L}_k} \lambda_j c_j^l \to 0, \quad k \to \infty. \tag{14}$$

Using (6) and by the definition of the $\varepsilon-$subdifferential we deduce $M^k(y, x^l) \geq f(x^l) + \langle \xi_j^l, y - x^l \rangle - c_j^l$. By multiplying $\lambda_j \geq 0$ in this relation, summing up and using the fact that $\sum_{j \in \mathcal{L}_k} \lambda_j = 1$ we obtain $M^k(y, x^l) \geq f(x^l) + \langle \sum_{j \in \mathcal{L}_k} \lambda_j \xi_j^l, y - x^l \rangle - \sum_{j \in \mathcal{L}_k} \lambda_j c_j^l$. From Lemma 1 (ii)–(iii), we get $M^{\uparrow}(y, x^l) \geq M^{\uparrow}(x^l, x^l) + \langle \sum_{j \in \mathcal{L}_k} \lambda_j \xi_j^l, y - x^l \rangle - \sum_{j \in \mathcal{L}_k} \lambda_j c_j^l$. Taking the limit as $k \to \infty$ and by (14), we deduce $M^{\uparrow}(y, x^l) \geq M^{\uparrow}(x^l, x^l) + \langle 0, y - x^l \rangle$. Hence by Lemma 1 (i) and (iv) we obtain that $0 \in \partial_c M^{\uparrow}(x^l, x^l)$ and so $0 \in \partial f(x^l)$. □

Theorem 3 *Assume that f is locally Lipschitz and there exists $\bar{\mu} > 0$ such that $\mu_k \leq \bar{\mu}$ for all k. In addition suppose that the level set $A(x^1) := \{x \in \mathbb{R}^n,\ f(x) \leq f(x^1)\}$ is bounded. Then every accumulation point of the serious iterate sequence is stationary for the problem (1).*

Proof By assumption there exists a sequence $\{x^l\}_l$. The method is descent type thus we have $\{x^l\}_l \subseteq A(x^1)$. Since f is locally Lipschitz and $A(x^1)$ is bounded, the sequence $\{f(x^l)\}_l$ is bounded below. On the other hand, for each l we have $f(x^{l+1}) \leq$

$f(x^l) + m_L v_{k+1(l)}$. Since f is bounded below we deduce $v_{k+1(l)} \to 0$, as $l \to \infty$. Using $\mu_k \leq \bar{\mu}$ we deduce

$$\sum_{j\in\mathcal{L}_k} \lambda_j \xi_j^l \to 0, \quad \text{and} \quad \sum_{j\in\mathcal{L}_k} \lambda_j c_j^l \to 0, \quad l \to \infty. \tag{15}$$

On the other hand, by assumption we have $A(x^1)$ is bounded. Therefore the sequence $\{x^l\}_l$ is bounded and it has a convergent subsequence $\{x^l\}_{l\in\mathcal{A}} \subseteq \{x^l\}_l$ thus there exists $x^* \in \mathbb{R}^n$ such that $x^l \to_{l\in\mathcal{A}} x^*$. By (6) we have $\xi_j^l \in \partial_{c_j^l} M^k(x^l, x^l)$ and consequently $M^k(y, x^l) \geq M^k(x^l, x^l) - \langle \xi_j^l, x - x^l \rangle - c_j^l$ for all $y \in \mathbb{R}^n$. By Lemma 1 (ii)–(iii), $\lambda_j \geq 0$ and $\sum_{j\in\mathcal{L}_k} \lambda_j = 1$ we have $M^{\uparrow}(y, x^l) \geq M^{\uparrow}(x^l, x^l) + \langle \sum_{j\in\mathcal{L}_k} \lambda_{i,j} \xi_j^l, y - x^l \rangle - \sum_{j\in\mathcal{L}_k} \lambda_j c_j^l$. Passing to the limit in the above relation when $l \in \mathcal{A}$ and $l \to \infty$ and using (15), we obtain $M^{\uparrow}(y, x^*) \geq M^{\uparrow}(x^*, x^*) + \langle 0, y - x^* \rangle$. By Lemma 1 (i) and (iv), we deduce $0 \in f(x^*)$ and thus x^* is a stationary point. □

3 Numerical experiments

In this section, we report the results of numerical experiments for the proposed algorithm and compare it with two existing algorithms in the literature. They are SolvOpt (Solver for local nonlinear optimization problems) [24] and HANSO 2.2 (Hybrid Algorithm for Nonsmooth Optimization, Version 2.2) [4,15]. All codes are written in MATLAB R2016a and run on a PC Intel Core I5 with CPU 2.5 GHZ and 4GB of RAM. The number of function evaluations and subgradient evaluations are considered as a measure of efficiency. For SolvOpt and HANSO 2.2, the parameters are set to the default values suggested by the respective codes and the stopping parameter is considered tol $:= 10^{-8}$. We set the parameters of Algorithm 1 as tol $:= 10^{-8}$, $m_L := 10^{-2}$ and $\omega := 1.2$. The proximal parameter is considered one in all iterations, i.e., $\mu_k := 1$ for all k. The size of the bundle needs to be limited, we choose $L_{\max} := n + 50$ as the maximal size for the bundle, where n is the dimension of each problem. The index set $\mathcal{L}_k$ and the bundle of information $\mathcal{B}_k$ are updated as in Step 3 if $|\mathcal{L}_k| < L_{\max}$, and if $|\mathcal{L}_k| = L_{\max}$, then the set $\mathcal{L}_{k+1} = \mathcal{L}_k \cup \{k+1\} \setminus \{k - L_{\max}\}$ is used and accordingly the bundle is updated. The quadratic programming solver is quadprog.m, which is available in the MATLAB optimization toolbox. In our results, to select η_k we use the relation (4) with equality.

To measure the efficiency of the considered algorithms two classes of test problems of [2] are used. The first class includes the problems 1–10 (denoted by P1–P10) and 21–30 (called by P11–P20) with constant number of variables and the second class can be formulated with any number of variables.

We first applied the algorithms for solving problems P1-P20 by applying the given starting points from [2]. Results are presented in Table 1. In this table, we state the value of the objective function at the final point by f_{final}, the number of function evaluations and the number of subgradient evaluations by n_f and n_ξ, respectively. The numerical experiments demonstrate that Algorithm 1 has an acceptable behaviour for nonsmooth problems, since it uses the least number of function evaluations for 17 problems and

Table 1 Results of P1–P20 with given starting points

No.	Problem	n	Algorithm 1		HANSO 2.2		SolvOpt		
			n_f, n_ξ	f_{final}	n_f, n_ξ	f_{final}	n_f	n_ξ	f_{final}
$P1$	CB2	2	48	1.9522	128	1.9522	99	32	1.9522
$P1$	CB2	2	47	1.9522	108	1.9522	92	31	1.9522
$P2$	CB3	2	20	2	192	2.0000	81	30	2.0000
$P3$	DEM	2	36	− 3.0000	92	− 2.9998	250	92	− 3.0000
$P4$	QL	2	24	7.2000	131	7.2000	85	27	7.2000
$P5$	LQ	2	17	− 1.4142	156	− 1.4142	59	14	− 1.4142
$P6$	Miffilin 1	2	128	− 1.0000	32	4.0000	55	34	− 0.8286
$P7$	Wolfe	2	69	− 8.0000	96	− 8.0000	120	34	− 8
$P8$	Rosen-Suzuki	4	96	− 44.0000	140	− 44.0000	147	55	− 44.0000
$P9$	Davidon	4	69	115.7064	356	115.7064	211	75	115.7064
$P10$	Shor	5	92	22.6002	410	22.6002	118	46	22.6002
$P11$	Crescent	2	15	0.0002	175	0.0000	261	50	0.0000
$P12$	Miffilin 2	2	28	− 1.0000	247	− 1.0000	85	27	− 1.0000
$P13$	WF	2	73	0.0000	91	0.0000	172	33	0.0000
$P14$	Spiral	2	615	0.0769	96	0.0774	291	128	0.0743
$P15$	EVD 52	3	53	3.5997	131	3.5997	144	46	3.5997
$P16$	PBC 3	3	7	0	17	0	55	4	0
$P17$	Brad	3	87	0.0508	468	0.0508	118	37	0.0508
$P18$	Kowalik-Osborne	4	615	0.0081	860	0.0081	248	79	0.0081
$P19$	Polak 6	4	102	− 43.9897	50	− 41.9430	674	225	− 38.6919
$P20$	OTE5	4	61	0.0074	2622	0.0029	2445	662	0.0027
	Average		115.1		329.9		290.5	88.05	

HANSO 2.2 needs the least number of function evaluations for 3 problems. On the other hand, SolvOpt uses the least number of subgradient evaluations for 12 problems and Algorithm 1 needs the least number of subgradient evaluations for 6 problems. For more details and comparison average values of evaluations see Table 1.

At the next step, we use 20 random generated starting points (by using randn.m function in MATLAB) for each problem and the starting points are the same for all algorithms. To compare the performance of the algorithms, we apply the indicators: n_b− the number of successful runs considering the best known solution. We say that an algorithm finds a solution to a problem with a tolerance $\varepsilon > 0$, if $|f_{\text{final}} - f^*|/(1 + |f^*|) \leq \varepsilon$, where f^* is the optimal value. In our experiments $\varepsilon = 5 \times 10^{-4}$. Results of these numerical experiments are presented in Table 2. We present the number n_b for each problem as well as the average of the objective function values (denoted by f_{ave}), the average number of the objective function evaluations and the subgradient evaluations (denoted by n_f^{ave} and n_ξ^{ave}, respectively) over 20 runs for each problem and each solver. We observe that Algorithm 1 solves successfully 355 examples, where HANSO 2.2 and SolvOpt solve 318 and 344 examples, respectively.

Table 2 Average results of P1–P20 with 20 random starting points

No.	Algorithm 1			HANSO 2.2			SolvOpt			
	n_b	$n_f^{\text{ave}}, n_\xi^{\text{ave}}$	f_{ave}	n_b	$n_f^{\text{ave}}, n_\xi^{\text{ave}}$	f_{ave}	n_b	n_f^{ave}	n_ξ^{ave}	f_{ave}
$P1$	20	50.4	1.9522	20	109.5	1.9522	20	90.25	29.55	1.9522
$P2$	20	33.8	2	20	193.95	2.0000	20	94.8	31.55	2.0000
$P3$	20	35.55	– 2.9995	13	115.05	– 2.9216	20	187.9	54.2	– 3.0000
$P4$	20	40.05	7.2000	20	120.9	7.2000	20	105.3	32.65	7.2000
$P5$	20	29.6	– 1.4142	20	170.65	– 1.4142	20	90.85	29.55	– 1.4142
$P6$	18	107.45	– 0.9963	14	103.35	v0.8540	11	222.15	97.8	– 0.8595
$P7$	20	70.6	– 8.0000	20	182.1	– 8.0000	20	95.65	29.25	– 8.0000
$P8$	20	102.1	– 44.0000	20	131.95	– 44.0000	20	135.85	48.35	– 44.0000
$P9$	20	87.9	115.7064	20	488.45	115.7064	20	255.4	84.6	115.7064
$P10$	20	140.15	22.6002	20	404.5	22.6002	20	158.4	55.85	22.6002
$P11$	10	29.65	0.0062	20	170.5	0.0000	20	219.15	42.8	0.0000
$P12$	20	29.3	– 1.0000	20	278.95	– 1.0000	20	85.8	26.45	– 1.0000
$P13$	10	37.25	4.0328	10	3.0250	0.0000	6	92.55	25.45	4.2350
$P14$	18	295.95	0.0077	1	39.25	0.3380	14	402.75	117.55	0.0517
$P15$	20	56.55	3.5997	20	128.15	3.5997	20	120.2	39.4	3.5997
$P16$	20	6.25	0	20	4.55	0	20	29.65	11.3	0
$P17$	20	98.9	0.0508	18	857.65	0.0620	18	132.8	41.65	0.1263
$P18$	19	917.2	0.0085	9	1990.1	0.0175	15	578.85	159.4	0.0105
$P19$	0	98.31	– 43.3210	0	83	– 36.7891	0	246.5	132	– 42.9768
$P20$	20	455.05	0.0027	13	1508	0.0183	20	2307.7	626.25	0.0027
Sum	355			318			344			

At the third step, we consider the generalization of MAXQ, GMXHIB and Chained crescent I [2]. These problems can be formulated with any number of variables. We have used here $n = 5, 10, 50, 100, 200$ and apply 4 different starting points. The starting points are the same for all test problems and three algorithms. The results are summarized in Table 3 and show that values of f_{final} obtained by Algorithm 1 are quite correct for all problems whereas HANSO 2.2 and SolvOpt can not calculate any descent directions for MAXQ with $x^1 = [1, 1, \ldots, 1]$ and $x^1 = [-1.5, 2, \ldots]$ and for $n = 5, 10, 50, 100, 200$ and $n = 50, 100, 200$ respectively. To compare better, check the results in Table 3.

In addition, in order to obtain better comparison of the considered algorithms, we analyze the results using the performance profile. More details regarding the definition of the performance profile can be found in [7]. Here we use the number of function evaluations and the number of subgradient evaluations as efficiency measures to define the performance profile. In the performance profiles, the value of $\rho_s(\tau)$ at $\log(\tau) = 0$ indicates the ratio of the test problems for which the solver s is the best – that is, the solver s uses the least number of function evaluations or the least number of subgradient evaluations. The value of $\rho_s(\tau)$ at the rightmost abscissa gives the ratio

Table 3 Results of large scale problems with various starting points and dimensions

n	Starting point	Algorithm 1		HANSO 2.2		SolvOpt		
		n_f, n_ξ	f_{final}	n_f, n_ξ	f_{final}	n_f	n_ξ	f_{final}
MaxQ								
5	$[1, 1, \ldots, 1]$	66	0.0000	32	1	467	95	0.0000
10		118	0.0000	32	1	579	135	0.0000
50		567	0.0000	32	1	18	18	1
100		1180	0.0000	32	1	18	18	1
200		2145	0.0000	32	1	18	18	1
5	$[1, 2, \ldots, 5]$	43	0.0000	242	0.0000	315	74	0.0000
10		121	0.0000	787	0.0000	680	157	0.0000
50		536	0.0000	2137	0.0000	3556	761	0.0000
100		1362	0.0000	2117	0.0000	8126	1706	0.0000
200		2378	0.0000	2220	0.0000	18624	3802	0.0000
5	$[-1.5, 2, \ldots]$	70	0.0000	32	4	362	76	0.0000
10		114	0.0000	32	4	511	125	0.0000
50		886	0.0000	32	4	16	16	4
100		1065	0.0000	32	4	16	16	4
200		3728	0.0000	32	4	16	16	4
5	$[1, 2, 3, -4, -5]$	43	0.0000	242	0.0000	315	74	0.0000
10		121	0.0000	787	0.0000	680	157	0.0000
50		536	0.0000	2137	0.0000	3556	761	0.0000
100		1362	0.0000	2117	0.0000	8126	1706	0.0000
200		2378	0.0000	2220	0.0000	18624	3802	0.0000
GMXHIB								
5	$[1, 1, \ldots, 1]$	111	0.0000	1647	0.0000	527	128	0.0000
10		333	0.0000	622	0.0000	667	169	0.0000
50		566	0.0000	1411	0.0000	915	243	0.0000
100		1551	0.0000	1235	0.0000	995	279	0.0000
200		4555	0.0000	2057	0.0000	1056	293	0.0000
5	$[1, 2, \ldots, 5]$	653	0.0000	1697	0.0000	495	123	0.0000
10		1165	0.0000	1509	0.0000	847	214	0.0000
50		4871	0.0000	1806	0.0000	983	272	0.0000
100		7321	0.0001	2666	0.0000	1112	286	0.0000
200		8364	0.0000	2610	0.0000	1068	301	0.0000
5	$[-1.5, 2, \ldots]$	451	0.0000	873	0.0000	613	142	0.0000
10		2732	0.0000	977	0.0000	722	184	0.0000
50		858	0.0000	2867	0.0000	989	277	0.0000
100		1597	0.0000	2815	0.0000	876	253	0.0000
200		661	0.0000	2661	0.0000	1021	294	0.0000
5	$[1, 2, 3, -4, -5]$	1429	0.0000	1955	0.0000	510	124	0.0000

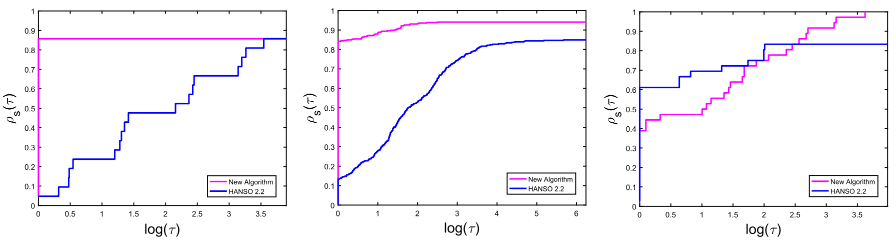
Table 3 continued

n	Starting point	Algorithm 1		HANSO 2.2		SolvOpt		
		n_f, n_ξ	f_{final}	n_f, n_ξ	f_{final}	n_f	n_ξ	f_{final}
10		1560	0.0000	2236	0.0000	910	223	0.0000
50		5873	0.0000	2930	0.0000	1011	271	0.0000
100		7321	0.0001	2862	0.0000	970	287	0.0002
200		8364	0.0000	2610	0.0000	1068	301	0.0000
Chained crescent I								
5	$[1, 1, \ldots, 1]$	125	0.0002	154	0.0000	450	106	0.0000
10		524	0.0002	168	0.0000	557	132	0.0000
50		826	0.0000	151	0.0000	2385	614	0.0000
100		1318	0.0001	151	0.0000	4642	1251	0.0000
200		1419	0.0002	159	0.0000	9389	2773	0.0000
5	$[1, 2, \ldots, 5]$	108	0.0000	166	0.0000	396	96	0.0000
10		152	0.0003	193	0.0000	456	107	0.0000
50		345	0.0001	164	0.0000	1653	394	0.0000
100		840	0.0000	164	-0.0000	3882	1023	0.0000
200		1067	0.0002	167	-0.0000	6238	1719	0.0000
5	$[-1.5, 2, \ldots]$	257	0.0001	168	0.0000	345	77	0.0000
10		334	0.0000	157	0.0000	541	134	0.0000
50		527	0.0002	168	0.0000	1467	335	0.0000
100		643	0.0002	153	0.0000	1380	325	0.0000
200		850	0.0000	144	0.0000	1096	270	0.0000
5	$[1, 2, 3, -4, -5]$	343	0.0000	145	0.0000	391	89	0.0000
10		435	0.0001	156	0.0000	552	132	0.0000
50		649	0.0005	177	0.0000	983	212	0.0000
100		1024	0.0001	157	0.0000	3462	868	0.0000
200		2144	0.0002	174	0.0000	6388	1734	0.0000

of the test problems that the solver s can solve – that is, the robustness of the solver s. In addition, the higher curve shows that its corresponding solver is better.

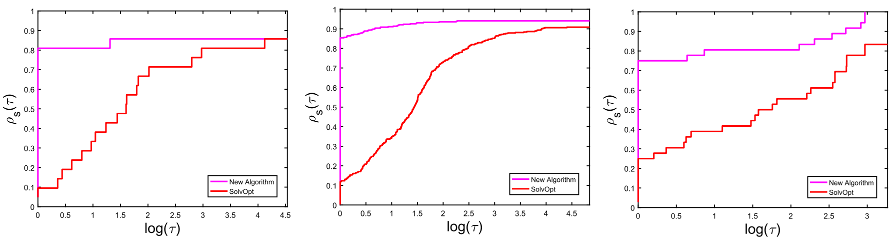
The results corresponding P1–P20 with the given starting points in [2] are reported in Figs. 1, 2 and 3 part (a). In part (b), the results of P1–P20 with 20 random starting points and the results of MAXQ, GMXHIB and Chained crescent I with dimensions $n = 5, 10$ are stated. In addition the results of MAXQ, GMXHIB and Chained crescent I with the dimensions $n = 50, 100, 200$ are reported in part (c) of figures.

From Figure 1, we deduce that Algorithm 1 is more efficient in comparison with HANSO 2.2 with the number of function and subgradient evaluations for the small problems with given and random starting points, since it is superior to HANSO 2.2 in Figure 1(a),(b). Algorithm 1 can solve more than 94% of the small problems with various starting point where HANSO 2.2 solves nearly 85%. For most of the small test problems, Algorithm 1 is better than HANSO 2.2, i.e., it uses the least number



(a) Given starting points **(b)** Several starting points **(c)** Big problems

Fig. 1 Comparison between Algorithm 1 and HANSO 2.2, Number of function and subgradient evaluations



(a) Given starting points **(b)** Several starting points **(c)** Big problems

Fig. 2 Comparison between Algorithm 1 and SolvOpt, Number of function evaluations

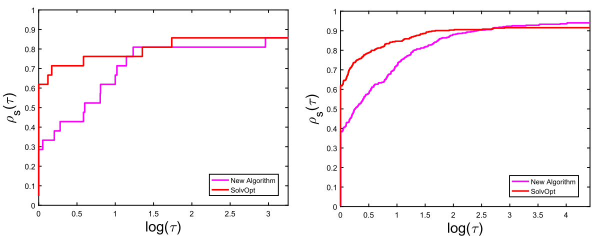
of function and subgradient evaluations. For big problems whose results are stated in Figure 1 (c), for 60% of problems HANSO 2.2 is better than Algorithm 1 and for 40% of problems Algorithm 1 is better than HANSO 2.2. Moreover Algorithm 1 can solve 100% of problems successfully, where HANSO 2.2 could solve 83% of problems.

From Figure 2, it is easy to understand that from the aspect of the number of the function evaluations Algorithm 1 is better than SolvOpt for all considered problems.

From Figure 3(a),(b), we deduce that for small test problems with the given and random starting points SolvOpt is more efficient than Algorithm 1 if the number of subgradient evaluations is considered as a measure of efficiency. For big problems, Figure 3(c), SolvOpt is better than Algorithm 1 for nearly 55% of problems, while Algorithm 1 is better than SolvOpt for 45% of problems. On the other hand, SolvOpt can solve nearly 83% of problems successfully, while Algorithm 1 solves 100% of problems. However, we can say that the performances of these two solvers are comparable.

Overall, our results show that, in general, the proposed method is efficient and more robust than other methods used in the numerical experiments.

At the end, we are interested in exploring the assumption that the parameter η_k remains bounded. In [9] the authors report that η_k was less than $2n + 2$ when solving 73 of the 75 exact unconstrained optimization problems; for one test problem η_k was between $2n + 2$ and $25n$ and for the remaining one η_k exceeded $25n$. In [12], the authors stated that $\eta_k = 2\omega$ for 85 problems, η_k is bounded by $2n$ for 55 problems and it is between $2n$ and $25n$ for 10 problems. In our results we note that $\eta_k = 2\omega$ for 315 examples, η_k is bounded by $2n$ for 127 cases and it is between $2n$ and $25n$ for 36



(a) Given starting points **(b)** Several starting points **(c)** Big problems

Fig. 3 Comparison between Algorithm 1 and SolvOpt, Number of subgradient evaluations

examples and for 3 cases η_k is bounded by $31n$. Altogether, the numerical experiments support that the boundedness assumption on the sequence $\{\eta_k\}$ is quite suitable.

Acknowledgements The first-named author was in part supported by a grant from IPM (No.1400900048). The authors would like to thank two anonymous referees for their comments that helped to improve the quality of the paper.

References

1. Bagirov, A.M., Jin, L., Karmitsa, N., Nuaimat, A., Sultanova, A.N.: Subgradient method for nonconvex nonsmooth optimization. J. Optim. Theory Appl. **157**, 416–435 (2013)
2. Bagirov, A.M., Karmitsa, N., Mäkelä, M.M.: Introduction to Nonsmooth Optimization: Theory, Practice and Software. Springer, New York (2014)
3. Bagirov, A.M., Taheri, S., Joki, K., Karmitsa, N., Mäkelä, M.M.: Aggregate subgradient method for nonsmooth DC optimization. Optim. Lett. **15**, 83–96 (2021)
4. Burke, J., Lewis, A., Overton, M.: A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. SIAM J. Optim. **15**, 571–779 (2005)
5. Dao, M.N.: Bundle method for nonconvex nonsmooth constrained optimization. J. Convex Anal. **22**(4), 1061–1090 (2015)
6. Dao, M.N., Gwinner, J., Noll, D., Ovcharova, N.: Nonconvex bundle method with application to a delamination problem. Comput. Optim. Appl. **65**, 173–203 (2016)
7. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
8. Hare, W., Sagastizábal, C.: A redistributed proximal bundle method for nonconvex optimization. SIAM J. Optim. **20**, 2442–2473 (2010)
9. Hare, W., Sagastizábal, C., Solodov, M.: A proximal bundle method for nonsmooth nonconvex functions with inexact information. Comput. Optim. Appl. **63**, 1–28 (2016)
10. Hoseini, N., Nobakhtian, S.: A new trust region method for nonsmooth nonconvex optimization. Optimization **67**, 1265–1286 (2018)
11. Hoseini Monjezi, N., Nobakhtian, S.: A filter proximal bundle method for nonsmooth nonconvex constrained optimization. J. Glob. Optim. **79**, 1–37 (2021)
12. Hoseini Monjezi, N., Nobakhtian, S.: A new infeasible proximal bundle algorithm for nonsmooth nonconvex constrained optimization. Comput. Optim. Appl. **74**(2), 443–480 (2019)
13. Hoseini Monjezi, N., Nobakhtian, S.: A proximal bundle-based algorithm for nonsmooth constrained multiobjective optimization problems with inexact data. Numer. Algor. (2021). https://doi.org/10.1007/s11075-021-01128-3
14. Hoseini Monjezi, N., Nobakhtian, S.: An inexact multiple proximal bundle algorithm for nonsmooth nonconvex multiobjective optimization problems. Ann. Oper. Res. (2020). https://doi.org/10.1007/s10479-020-03808-0
15. Lewis, A.S., Overton, M.L.: Nonsmooth Optimization via Quasi-Newton Methods. Math. Program. **141**(1–2), 135–163 (2013)

16. Li, Q.: Conjugate gradient type methods for the nondifferentiable convex minimization. Optim. Lett. **7**, 533–545 (2013)
17. Joki, K., Bagirov, A.M., Karmitsa, N., Mäkelä, M.M.: A proximal bundle method for nonsmooth DC optimization utilizing nonconvex cutting planes. J. Glob. Optim. **68**, 501–535 (2017)
18. Kiwiel, K.C.: A linearization algorithm for nonsmooth minimization. Math. Oper. Res. **10**, 185–194 (1985)
19. Kiwiel, K.C.: Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization. SIAM J. Optim. **18**, 379–388 (2007)
20. Lemaréchal, C.: Bundle methods in nonsmooth optimization. In: Lemaréchal, C., Mifflin, R. (eds.) Nonsmooth Optimization. IIASA Proc, Laxenburg (1977)
21. Mäkelä, M.M., Neittaanmäki, P.: Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control. World Scientific, Singapore (1992)
22. Noll, D.: Cutting plane oracles to minimize non-smooth non-convex functions. Set-Valued Var. Anal. **18**, 531–568 (2010)
23. Ruszczyński, A.: Convergence of a stochastic subgradient method with averaging for nonsmooth nonconvex constrained optimization. Optim. Lett. **14**, 1615–1625 (2020)
24. Shor, N.Z.: Minimization methods for non-differentiable functions. Springer, Berlin (1985)