J Glob Optim (2018) 70:517–549
https://doi.org/10.1007/s10898-017-0565-2

A proximal bundle method for constrained nonsmooth nonconvex optimization with inexact information

Jian Lv[1] · Li-Ping Pang[2] · Fan-Yun Meng[3]

Received: 19 November 2016 / Accepted: 29 August 2017 / Published online: 14 September 2017

Abstract We propose an inexact proximal bundle method for constrained nonsmooth nonconvex optimization problems whose objective and constraint functions are known through oracles which provide inexact information. The errors in function and subgradient evaluations might be unknown, but are merely bounded. To handle the nonconvexity, we first use the redistributed idea, and consider even more difficulties by introducing inexactness in the available information. We further examine the modified improvement function for a series of difficulties caused by the constrained functions. The numerical results show the good performance of our inexact method for a large class of nonconvex optimization problems. The approach is also assessed on semi-infinite programming problems, and some encouraging numerical experiences are provided.

Keywords Constrained optimization · Nonconvex optimization · Nonsmooth optimization · Inexact oracle · Proximal bundle method

Mathematics Subject Classification 90C26 · 49J52 · 93B40

Partially supported by Huzhou science and technology plan on No. 2016GY03 and Natural Science Foundation of China Grant 11626051.

✉ Li-Ping Pang
lppang@dlut.edu.cn

Jian Lv
lvjian328@163.com

1 School of Finance, Zhejiang University of Finance and Economics, Hangzhou 310018, China

2 School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, China

3 School of Computer, Qingdao Technological University, Qingdao 266033, China

1 Introduction

We consider the constrained nonconvex nonsmooth optimization problem

$$\begin{aligned} \min_{x\in\mathbf{R}^n} \quad & f(x) \\ s.t. \quad & c(x) \le 0, \quad x \in \mathcal{X}, \end{aligned} \tag{1.1}$$

where f, c: $\mathbf{R}^n \to \mathbf{R}$ are locally Lipschitz functions, and $\mathcal{X}$ is a convex compact subset of $\mathbf{R}^n$. Further, we assume that for fixed accuracy tolerances, the approximate function values and subgradients of f and c can be obtained by some inexact oracle for each given point. The errors in the function values and subgradients might be unknown, but are bounded by universal constants.

The above framework (1.1) arises in many optimization problems from real-life applications. For some problems, as in H_∞-control problems [2,42], SIP problems [43] and stochastic programming problems, computing exact information is too expensive, or even out-of-reach, whereas evaluating some inexact information is still possible. Most methods based on exact information can only be used for solving some simplifications of these problems.

Nonsmooth optimization problems are, in general, difficult to solve, even when they are unconstrained. Bundle methods are currently recognized as the most robust and reliable for nonsmooth optimization; see e.g., [1,21,24,31] for more detailed comments. In particular, bundle methods can be thought of as replacing the objective function f by a cutting-plane model to formulate a subproblem whose solution gives the next iteration and obtains the new bundle elements.

Bundle methods for nonconvex problems using exact information were developed in [14, 15,18,26,29,36,37,40,54,55] for more detailed comments. Except for [18,55], all of these methods that desired positive linearization errors did so by redefining them. That is, most of previous methods handle nonconvexity by downshitfing the so-called linearization errors if they are negative. The cutting-planes models in [18,55] are special in the sense that they no longer model the objective function f but rather certain local convexification. Our proximal bundle method here is along the lines of [18].

Bundle methods capable of handling inexact oracles have received much attention in the last few years. They have been discussed in [25] since 1985. Inexact evaluations of subgradient in subgradient methods date back to [5,30,41] in the convex setting, and [47] for nonconvex optimization. Different from earlier works [41,47], allow nonvanishing errors. Nonvanishing errors of both functions and subgradient values in convex bundle methods were first considered in [48], and further studied in [31]. Besides, several variants have been developed, i.e., [20,31] for general approaches [9], for energy planning [6,10], for a stochastic optimization framework [33], for a combinatorial context [43], for SIP problems, and [7] for a detailed summary. As far as we know, the only two other works dealing with inexact information in bundle methods for nonconvex optimization are [19,42].

For constrained problems, such as problem (1.1) considered here, more sophisticated methods need to come into play. One possibility is to solve an equivalent unconstrained problem with an exact penalty objective function; see [27,28] for convex constrained problems, and [55] for nonconvex case. These approaches, however, possess some shortcomings, which are typical whenever a penalty function is employed. Specifically, computing a suitable value of the penalty parameter is sometimes too expensive. Besides, if a large value of the penalty parameter is required to guarantee the exactness of a given penalty function, then numerical difficulties arise. Hence, bundle methods in [35,46] introduce a so-called improvement function

$$h_\tau(x) = \max\{f(x) - \tau, c(x)\}, \tag{1.2}$$

where τ is generated by the former iterative points. It is one of the most effective tools to handle constrained optimization. In this paper, we change τ into the $f(x^k)$ plus a weighted measure of its feasibility. Thus there is a balance between the search for feasibility and the reduction of the objective function. The details will be discussed in Sect. 2.1

The cutting-plane model in our method no longer approximates penalty function of the objective function as in [55], or the improvement function as in [24,46]. It is given by a maximum value function of the piecewise linear models of the local convexification of the objective function f and constraint function c. Hence, our approach can not be viewed as an unconstrained proximal bundle method applied to the function $h_\tau(x)$. Another feature of our algorithm is infeasibility in this paper. With respect to [24,39], the advantage is that it is not necessary to compute a feasible point to start the algorithm. Also, since serious steps can be infeasible, monotonicity in f is not enforced. Infeasible bundle methods are very rare and valuable.

In this paper, we propose an infeasible proximal bundle method with inexact information for solving (1.1). Above all, the sequence of iterative points of our inexact bundle method converges to an approximate optimal point at most ϵ asymptotically under a mild assumption. To be specific, the limit value f^* and c^* of the sequences $\{f^k\}$ and $\{c^k\}$, respectively, satisfy

$$\text{if } c^* \le 0\text{: } f^* \le f(y) + \epsilon$$

for all y in a neighbourhood. If the feasible set of (1.1) is empty, then $c^* \le c(y) + \epsilon$ for all y.

This paper is organized as follows. In Sect. 2, we state basic conceptual comments, inexact oracle and basic properties of the improvement function. The cutting-planes model and the algorithm itself are stated in Sect. 3, where some preliminary properties also are established. Asymptotic analysis is considered in Sect. 4, and convergence analysis is provided in Sect. 5. Numerical experiments are presented in Sect. 6.

2 Background, assumptions and notation

The description of this work starts with conceptual comments and results of variational analysis, then passes to technical details gradually. We assume that the objective function f and constraint function c are proper [44, p. 5], regular [44, Def 7.25], and locally Lipschitz with full domain.

2.1 Background and assumptions

In this subsection, we first recall concepts and results of variational analysis that will be of use in this paper. The closed ball in $\mathbf{R}^n$ with the center in $x \in \mathbf{R}^n$ and radius $\rho > 0$ is denoted by $B_\rho(x)$. We shall use $\partial f(\bar{x})$ to denote the subdifferential of f at the point $\bar{x}$. By noting the regularity of f, the subdifferential mapping is well-defined and is given by

$$\partial f(\bar{x}) := \left\{v \in \mathbf{R}^n\colon f(x) \ge f(\bar{x}) + \langle v, x - \bar{x}\rangle + \circ(|x - \bar{x}|),\ \text{for all } x \in \mathbf{R}^n\right\}, \tag{2.1}$$

and each element of this subdifferential is called subgradient. Besides, the Clarke directional derivative [3] in x in direction $d \in \mathbf{R}^n$ is given as

$$f^0(x, d) := \lim_{h\to 0,}\ \sup_{t\downarrow 0} \frac{f(x + h + td) - f(x + h)}{t},$$

and alternative definition is

$$f^0(x,d) := \max_{g\in\partial f(x)} g^{\mathrm{T}} d.$$

Given an open set $\mathcal{O}$ containing $\mathcal{X}$, a locally Lipschitz function $f\colon \mathcal{O} \to \mathbf{R}$, is called to be lower-$C^1$, if on some neighborhood V of every $\tilde{y} \in \mathcal{O}$, there exists an expression as [49]:

$$f(y) := \max_{s\in S} g(y,s), \tag{2.2}$$

where the derivative $D_y g$ is jointly continuous and the index set S is a compact set. Our nonconvex proximal bundle method is given for lower-C^1 functions, and we define the following equivalence given in [8, The 2, Cor 3] and [49, Prop 2.4] as

- The locally Lipschitz function f is lower-C^1 on $\mathcal{O}$.
- Its Clarke subdifferential ∂f is submonotone at every $\tilde{y} \in \mathcal{O}$. That means for every $\tilde{y} \in \mathcal{O}$ and for all $\theta > 0$ there exists $\rho > 0$ such that
$$\left\langle q^1 - q^2, z^1 - z^2 \right\rangle \geq -\theta \|z^1 - z^2\|,$$
$$\text{for all } z^1, z^2 \in B_\rho(\tilde{y}) \text{ and } q^1 \in \partial f(z^1), q^2 \in \partial f(z^2).$$
- For all $\tilde{y} \in \mathcal{O}$ and for all $\theta > 0$ there exists $\rho > 0$ such that $\forall\, y \in B_\rho(\tilde{y})$ and $q \in \partial f(y)$
$$f(y+u) - f(x) \geq \langle q, u\rangle - \theta\|u\|, \tag{2.3}$$
whenever $\|u\| < \rho$ and $y + u \in B_\rho(\tilde{y})$.
- For every $\tilde{y} \in \mathcal{O}$ and for all $\theta > 0$ there exists $\rho > 0$ such that
$$f(ty^1 + (1-t)y^2) \leq tf(y^1) + (1-t)f(y^2) + \theta t(1-t)\|y^1 - y^2\|,$$
for all $t \in (0,1)$ and $y^1, y^2 \in B_\rho(\tilde{y})$.

If the optimal value $f_{\min}$ of the problem (1.1) is known in advance, the solution of (1.1) can be obtained by minimizing $h_{\min}(y) = \max\left\{f(y) - f_{\min}, c(x)\right\}$. Notice that if x^* is a solution of the problem (1.1), then $h_{\min}(x^*) = 0$. Hence, Lemaréchal [35] first changed $f_{\min}$ into a lower bound $\underline{f}^k$. The sequence $\{\underline{f}^k\}$, however, is not available, thus it is not possible for proximal bundle methods. Some alternatives were proposed in [1,23,32,35,43,46]. For instance [43], replaced $\underline{f}^k$ by the objective function value at the current stability center x^k as

$$h_{x^k}(y) = \max\left\{f(y) - f(x^k),\ c(y)\right\}. \tag{2.4}$$

However, all of them may not be suitable for nonconvex optimization with inexact oracle in this paper. Based on [32], we replace τ by $f(x^k)$ plus a weighted measure of its feasibility

$$h_{x^k}(y) := \max\left\{f(y) - \theta_1^{x^k}, c(y) - \theta_2^{x^k}\right\}, \tag{2.5}$$

with

$$\theta_1^{x^k} := f(x^k) + s_k \max\{0, c(x^k)\},\ \ \theta_2^{x^k} := t_k \max\{0, c(x^k)\}, \tag{2.6}$$

where $s_k > 0$ and $t_k > 0$ are penalty parameters. The penalty parameters s_k and t_k are bounded, and updated depending on a constant along the iterative process. Moreover, as shown below $h_{x^k}(\cdot)$ is nonnegative at each stability center, and used for the optimality measure.

We introduce the following extended MFCQ (see [22,51]) for this nonsmooth SIP.

Assumption 1 (*The constraint qualification*) Suppose that x^* is a local solution of the problem (1.1). There exists some direction d of $\mathcal{F} := \{x \in \mathcal{X}: c(x) \leq 0\}$ at x^*, satisfies

$$c^0(x^*, d) < 0. \tag{2.7}$$

We now give the following necessary condition of optimality measure.

Lemma 2.1 *Suppose x^* is a local solution of the problem (1.1), and there exists a constant ρ^* such that $f(y) \geq f(x^*)$ for all $y \in B_{\rho^*}(x^*) \bigcap \mathcal{X}$. Then the following statements hold:*

(i) *For all $y \in \mathcal{X}$, it holds that*

$$\min\{h_{x^*}(y): y \in \mathcal{X}\} = h_{x^*}(x^*) = 0, \ \textit{for all } y \in B_{\rho^*}(x^*) \cap \mathcal{X}. \tag{2.8}$$

(ii) $0 \in \partial h_{x^*}(x^*)$, *and there exist μ_0 and μ_1 satisfying*

$$\begin{aligned} &0 \in \mu_0 \partial f(x^*) + \mu \partial c(x^*) + \partial i_{\mathcal{X}}(x^*), \\ &\mu \geq 0, \ \mu + \mu_0 = 1, \ \mu c(x^*) = 0, \ c(x^*) \leq 0, \end{aligned} \tag{2.9}$$

where $i_{\mathcal{X}}(x^*)$ denotes the indicator function of the set $\mathcal{X}$.

(iii) *If, in addition, Assumption 1 holds, then x^* is a KKT point of the problem (1.1).*

Proof Since x^* is a local solution of the problem (1.1), we conclude that

$$c(x^*) \leq 0 \ \text{ and } \ f(x^*) \leq f(y), \ \textit{for all } y \in B_{\rho^*}(x^*) \cap \mathcal{X}.$$

Together with the relation (2.6), we can obtain

$$\theta_1^{x^*} = f(x^*) \ \text{ and } \ \theta_2^{x^*} = 0.$$

That means, for all $y \in B_{\rho^*}(x^*) \cap \mathcal{X}$,

$$h_{x^*}(y) = \max\Big\{f(y) - f(x^*), c(y)\Big\} \geq 0 = h_{x^*}(x^*),$$

which implies that the relation (2.8) holds. From here, the first item (i) follows.

It follows from [24, Lemma 2.15] that $0 \in \partial(h_{x^*}(x^*) + i_{\mathcal{X}}(x^*))$. Hence the point x^* is a FJ point of (1.1) and the conditions (2.9) hold. To see item (iii),

$$\begin{aligned} 0 &\in \partial(h_{x^*}(x^*) + i_{\mathcal{X}}(x^*)) \\ &= conv\Big\{\partial f(x^*) \cup \partial c(x^*)\Big\} + \partial i_{\mathcal{X}}(x^*). \end{aligned}$$

If μ_0 is zero in condition (2.9) we obtain $\mu = 1$ and $-g^* \in \partial i_{\mathcal{X}}(x^*)$ for some $g^* \in \partial c(x^*)$. By noting d is a feasible direction of $\mathcal{F}$, we get that $\langle g^*, d\rangle \geq 0$. Thus we obtain

$$0 \leq \max\big\{\langle g, d\rangle | g \in \partial c(x^*)\big\} = c^0(x^*, d),$$

which is an contradiction with (2.7). Hence the constant μ_0 is strictly positive. Together with the conditions (2.9), we obtain that

$$\begin{aligned} &0 \in \partial f(x^*) + \nu \partial c(x^*) + \partial i_{\mathcal{X}}(x^*), \\ &\nu \geq 0, \ c(x^*) \leq 0, \ \nu c(x^*) = 0, \end{aligned}$$

since one may take $\nu = \frac{\mu}{\mu_0}$. Hence, x^* is a KKT point of the problem (1.1). □

2.2 Available information

The algorithm herein will hinge around previously generated information. Suppose that y^j is an iterative point at jth step, and x^k denotes the kth stability center. Let j_k denote the jth iteration giving the stability center x^k, i.e., $x^k := y^{j_k}$. The stability center is considered as the "best" known point up to the kth iteration.

We assume that at any y^j, the oracle provides

$$\begin{cases} f^j \text{ and } c^j & \text{estimate for the function values, and} \\ g_f^j \text{ and } g_c^j & \text{estimate for the respective subgradients.} \end{cases} \tag{2.10}$$

For each $y^j \in \mathcal{X}$, the oracle provides us with

$$\begin{cases} f\text{-oracle information} \begin{bmatrix} f^j = f(y^j) - \sigma_j \text{ and} \\ g_f^j \in \partial f(y^j) + B_{\varepsilon_j}(0); \end{bmatrix} \\ c\text{-oracle information} \begin{bmatrix} c^j = c(y^j) - \sigma_j \text{ and} \\ g_c^j \in \partial c(y^j) + B_{\varepsilon_j}(0). \end{bmatrix} \end{cases} \tag{2.11}$$

At each stability center x^k, for simplicity, we denote

$$\begin{bmatrix} \hat{f}^k = f(x^k) - \hat{\sigma}_k \text{ and} \\ \hat{g}_f^k \in \partial f(x^k) + B_{\hat{\varepsilon}_k}(0); \end{bmatrix} \quad \begin{bmatrix} \hat{c}^k = c(x^k) - \hat{\sigma}_k \\ \hat{g}_c^k \in \partial c(x^k) + B_{\hat{\varepsilon}_k}(0). \end{bmatrix} \tag{2.12}$$

Since the sign of error σ_j is not specified, the exact function values may be either underestimated or overestimated by f^j and c^j. Throughout this section we will make the assumption that the error on each of these estimates is bounded, i.e., there exist corresponding bounds $\bar{\sigma}$ and $\bar{\varepsilon}$, such that

$$-\bar{\sigma} \le \sigma_j \le \bar{\sigma} \text{ and } 0 \le \varepsilon_j \le \bar{\varepsilon} \text{ for all } j,$$

where both bounds $\bar{\sigma}$ and $\bar{\varepsilon}$ are generally unknown.

3 Defining the inexact algorithm

3.1 The model

Based on the modified improvement function (2.5) and the inexact oracle (2.11), we define the kth inexact modified improvement function at the stability center x^k as follows:

$$h_k(y) := \max\left\{f^y - \theta_1^k, c^y - \theta_2^k\right\}, \tag{3.1}$$

where $f^y := f(y) - \sigma_y$, $c^y := c(y) - \sigma_y$, and

$$\theta_1^k = \hat{f}^k + s_k \max\{0, \hat{c}^k\}, \quad \theta_2^k = t_k \max\{0, \hat{c}^k\}. \tag{3.2}$$

Both penalty parameters s_k and t_k are updated depending on a constant $\varpi > 0$, and satisfy

$$s_k \ge 0, \quad t_k \in [0, 1] \text{ satisfying } s_k - t_k \ge \varpi. \tag{3.3}$$

Since penalty parameters s_k and t_k are bounded, they must be well-defined along the iterative process. Considering (3.1) and the oracle assumption, we obtain

$$h_k(x^k) = \max\Big\{-s_k\max\{0,\hat{c}^k\}, \hat{c}^k - t_k\max\{0,\hat{c}^k\}\Big\}$$
$$= \begin{cases} 0, & \text{if } \hat{c}^k \le 0, \\ (1-t_k)\hat{c}^k, & \text{if } \hat{c}^k > 0. \end{cases} \tag{3.4}$$

The strategy (3.3) for updating s_k and t_k can ensure the nonnegativity $h_k(x^k) \ge 0$.

It is worth noting that the past information gives the cutting plane model. Just as a classic method in [46], if f and c are convex functions with exact oracle, then the cutting plane model was given as

$$\varphi_l(y) = \max_{i\in\mathcal{B}_l}\Big\{h_{x^k}(y^i) + \langle g_h^i, y - y^i\rangle\Big\}, \quad \text{where } g_h^i \in \partial h_{x^k}(y^i). \tag{3.5}$$

However, considering the negativity of the linearization error and the non-monotonicity for the nonconvex function $h_{x^k}(\cdot)$, as a result many more difficulties are introduced. Besides, the computations for the value $h_k(y^i)$ and its subgradient g_h^i, at each iteration, are relatively expensive. Thus, the formula (3.5) does not apply to our nonconvex optimization in our setting. Then we have to deal with them adequately along the iterative process.

The oracle output is collected along the iterative process to form the "Bundle" of information

$$\begin{aligned} \mathcal{B}_k^f &:= \Big\{(x^j, f^j, g_f^j)\colon j \in L_k^f\Big\} \text{ for } L_k^f \subset \{1,\ldots,k\}; \\ \mathcal{B}_k^c &:= \Big\{(x^j, c^j, g_c^j)\colon j \in L_k^c\Big\} \text{ for } L_k^c \subset \{1,\ldots,k\}, \end{aligned} \tag{3.6}$$

where L_k^f and L_k^c are respectively index sets for the objective function and the constrained function at the kth iteration. We now define respectively the linearization errors for f and c at x^k as

$$\begin{aligned} e_{f_j}^k &:= \hat{f}^k - f^j - \Big\langle g_f^j, x^k - y^j\Big\rangle, \quad j \in L_k^f, \\ e_{c_j}^k &:= \hat{c}^k - c^j - \Big\langle g_c^j, x^k - y^j\Big\rangle, \quad j \in L_k^c. \end{aligned} \tag{3.7}$$

Possible negativity of linearization errors is more strictly linked to nonconvexity of f and c compared to inexact oracle information. Some appropriate adjustments should be introduced, such as by adding the second-order relation

$$p_j^k := \frac{\mu_k}{2}\|y^j - x^k\|^2$$

where μ_k is the so-called convexification parameter.

Having this information, the kth approximate piecewise-linear models for f, c are generated by

$$\begin{aligned} \check{f}_k(y) &= \max_{j\in L_k^f}\Big\{f_j^k + \Big\langle \hat{g}_{f_j}^k, y - y^j\Big\rangle\Big\} \\ &= \hat{f}^k + \max_{j\in L_k^f}\Big\{-\hat{e}_{f_j}^k + \Big\langle \hat{g}_{f_j}^k, y - x^k\Big\rangle\Big\}, \end{aligned} \tag{3.8a}$$

$$\begin{aligned} \check{c}_k(y) &= \max_{j\in L_k^c}\Big\{c_j^k + \Big\langle \hat{g}_{c_j}^k, y - y^j\Big\rangle\Big\} \\ &= \hat{c}^k + \max_{j\in L_k^c}\Big\{-\hat{e}_{c_j}^k + \Big\langle \hat{g}_{c_j}^k, y - x^k\Big\rangle\Big\}, \end{aligned} \tag{3.8b}$$

where

$$f_j^k := f^j + p_j^k \text{ and } \hat{e}_{f_j}^k := e_{f_j}^k + p_j^k, \text{ for all } j \in L_k^f, \tag{3.9a}$$

$$c_j^k := c^j + p_j^k \text{ and } \hat{e}_{c_j}^k := e_{c_j}^k + p_j^k, \text{ for all } j \in L_k^c, \tag{3.9b}$$

$$\hat{g}_{f_j}^k := g_f^j + \mu_k(y^j - x^k), \quad j \in L_k^f \text{ and } \hat{g}_{c_j}^k := g_c^j + \mu_k(y^j - x^k), \quad j \in L_k^c. \tag{3.9c}$$

The challenge is therefore to select μ_k sufficiently large that $\hat{e}_{f_j}^k \geq 0$ and $\hat{e}_{c_j}^k \geq 0$ for all j but sufficiently small to remain manageable. Inspired by [17], we define the parameter as follows:

$$\mu_k := \max\left\{ \max_{j \in L_k^f/\{j_k\}} \frac{-2e_{f_j}^k}{\|y^j - x^k\|^2}, \max_{j \in L_k^c/\{j_k\}} \frac{-2e_{c_j}^k}{\|y^j - x^k\|^2}, 0\right\} + \iota, \text{ where } \iota > 0. \tag{3.10}$$

Different from (3.5), having piecewise-linear models (3.8), the kth inexact improvement function is modelled by the following model function:

$$\Psi_k(y) = \left\{ \check{f}_k(y) - \theta_1^k, \check{c}_k(y) - \theta_2^k \right\}, \tag{3.11}$$

By this mean, the model function will maintain powerful relationships with the original functions f and c.

3.2 Inexact proximal bundle method

Let $\|\cdot\|$ be the Euclidean norm, $\|\cdot\|_k$ primal metrics and $|\cdot|_k$ dual metrics. The iterative point y^{k+1} is nothing but the computation of the proximal point of the model function (3.11), with prox-center (stability center) x^k and a variable prox-metric depending on matrices M_k. More specifically, the positive definite matrix M_k of order n has the form

$$M_k := A_k + \eta_k I,$$

where the parameter $\eta_k > 0$, A_k is a symmetric $n \times n$ matrix, and I the identity matrix of order n. As a result, we introduce the corresponding primal and dual metrics for each $z \in \mathbf{R}^n$ as follows:

$$\texttt{primal metrics:}\, \|z\|_k^2 := z^{\mathrm{T}} M_k z \quad \text{and} \quad \texttt{dual metrics:}\, |z|_k^2 := z^{\mathrm{T}} (M_k)^{-1} z. \tag{3.12}$$

Let $\lambda_{\max}$ and $\lambda_{\min}$ denote, respectively, the maximum eigenvalue and the minimum eigenvalue, then

$$\lambda_{\max}(M_k) = \lambda_{\max}(A_k) + \eta_k \text{ and } \lambda_{\min}((M_k)^{-1}) = \frac{1}{\lambda_{\max}(A_k) + \eta_k}. \tag{3.13}$$

As a result, by the Euclidean norm, it holds that

$$\|z\|_k^2 \geq (\lambda_{\min}(A_k) + \eta_k)\|z\|^2 \quad \text{and} \quad |z|_k^2 \geq \frac{1}{(\lambda_{\max}(A_k) + \eta_k)} \|z\|^2. \tag{3.14}$$

Given a positive definite matrix M_k, the next iterative point y^{k+1} is generated by solving the following quadratic programming (QP) problem

$$\min_{y \in \mathcal{X}} \Psi_k(y) + \frac{1}{2}\|y - x^k\|_k^2. \tag{3.15}$$

If y^{k+1} satisfies the descent condition, then $x^{k+1} = y^{k+1}$ declares a serious point (a new stability center). Otherwise, it declares a null step, i.e., $x^{k+1} = x^k$ and only updates the next piecewise linear model. With (3.15) as a QP($\mathcal{B}_k$) with an extra scalar variable r as follows

$$\begin{aligned} \min_{(y,r)\in\mathcal{X}\times\mathbf{R}} \quad & r + \tfrac{1}{2}\|y - x^k\|_k^2, \\ s.t. \quad & \hat{f}^k - \theta_1^k - \hat{e}_{f_j}^k + \left\langle \hat{g}_{f_j}^k, y - x^k \right\rangle \le r, \quad j \in L_k^f, \\ & \hat{c}^k - \theta_2^k - \hat{e}_{c_j}^k + \left\langle \hat{g}_{c_j}^k, y - x^k \right\rangle \le r, \quad j \in L_k^c. \end{aligned} \tag{3.16}$$

More precisely, let $(\lambda^k, \gamma^k) \in \mathbf{R}_+^{|L_k^f|} \times \mathbf{R}_+^{|L_k^c|}$ denote the optimal solution of the dual problem of (3.16), considering the primal and dual optimal variables, then the following relations are easily recognized to hold:

$$y^{k+1} = x^k - M_k^{-1}(G^k + \alpha^k) \tag{3.17}$$

where

$$G^k := \sum_{j\in L_k^f} \lambda_j^k \hat{g}_{f_j}^k + \sum_{j\in L_k^c} \gamma_j^k \hat{g}_{c_j}^k \in \partial\Psi_k(y^{k+1}), \quad \alpha^k \in \partial i_{\mathcal{X}}(y^{k+1}), \tag{3.18a}$$

$$\sum_{j\in L_k^f} \lambda_j^k + \sum_{i\in L_k^c} \gamma_j^k = 1, \text{ with } \lambda_j^k \ge 0, \ j \in L_k^f \text{ and } \gamma_j^k \ge 0, \ j \in L_k^c. \tag{3.18b}$$

After solving the problem (3.16), the aggregate linearization

$$\psi_k(y) := \Psi_k(y^{k+1}) + \left\langle G^k, y - y^{k+1} \right\rangle \tag{3.19}$$

which is an affine function, and implies that $\psi_k(y^{k+1}) = \Psi_k(y^{k+1}), \ G^k = \nabla\psi_k(y)$. Clearly, because $G^k \in \partial\Psi_k(y^{k+1})$,

$$\psi_k(y) \le \Psi_k(y), \quad \text{for all } y \in \mathbf{R}^n. \tag{3.20}$$

The another ingredient in the bundle method is given by the aggregate error, defined by

$$\begin{aligned} V_k &:= h_k(x^k) - \psi_k(x^k) \\ &= h_k(x^k) - \Psi_k(y^{k+1}) - \left\langle G^k, x^k - y^{k+1} \right\rangle. \end{aligned} \tag{3.21}$$

For our setting, the noise introduced by the nonconvexity and inordinate noise is judged "too large" when the function value at the stability center is below the minimum model value. To judge the inordinate noise, we check the following noise measurement quantity as

$$h_k(x^k) - \left(\Psi_k(y^{k+1}) + \frac{1}{2}\|y^{k+1} - x^k\|_k^2 \right) < 0. \tag{3.22}$$

To measure progress towards the solution of (1.1), we define the predicted decrease

$$\delta_k := h_k(x^k) - \Psi_k(y^{k+1}) + \left\langle \alpha^k, x^k - y^{k+1} \right\rangle. \tag{3.23}$$

By using the relations (3.12), (3.17) and (3.21), we can obtain

$$\begin{aligned} \delta_k &= V_k + \left\langle G^k + \alpha^k, x^k - y^{k+1} \right\rangle \\ &= V_k + \|x^k - y^{k+1}\|_k^2 \\ &= V_k + |G^k + \alpha^k|_k^2. \end{aligned} \tag{3.24}$$

Only when the measurement (3.22) does not hold, i.e., the noise is acceptable, then we examine the new iterative point y^{k+1} by checking the descent condition

$$\begin{cases} f^{k+1} \le \hat{f}^k - m\delta_k \ \text{ and } \ c^{k+1} \le 0, & \text{if } \ \hat{c}^k \le 0, \\ c^{k+1} \le \hat{c}^k - m\delta_k, & \text{if } \ \hat{c}^k > 0 \end{cases} \tag{3.25}$$

to ensure whether y^{k+1} is good enough to develop into a new stability center. Otherwise, it declares a null step, and the stability center is fixed. The descent condition (3.25) measures the progress towards the solution of (1.1) from two ways. If $\hat{c}^k \le 0$, it reduces the objective value without losing feasibility by checking the first condition in (3.25). Otherwise, when $\hat{c}^k > 0$, the emphasis is put on reducing infeasibility by checking the second condition in (3.25).

Lemma 3.1 *Suppose* (r_k, y^{k+1}) *is the optimal solution of* $QP(\mathcal{B}_k)$. *The aggregate linearization error* V_k *satisfies the following relation:*

$$V_k \ge \sum_{j\in L_k^f} \lambda_j^k \hat{e}_{f_j}^k + \sum_{j\in L_k^c} \gamma_j^k \hat{e}_{c_j}^k > 0 \tag{3.26}$$

Proof The Lagrange function of (3.16) can be rewritten as

$$\begin{aligned} L(y^{k+1}, \lambda^k, \gamma^k) = &\sum_{j\in L_k^f} \lambda_j^k(\hat{f}^k - \theta_1^k) + \sum_{j\in L_k^c} \gamma_j^k(\hat{c}^k - \theta_2^k) + \tfrac{1}{2}\|y^{k+1} - x^k\|_k^2 \\ &+ \sum_{j\in L_k^f} \lambda_j^k \left(-\hat{e}_{f_j}^k + \left\langle \hat{g}_{f_j}^k, y^{k+1} - x^k \right\rangle\right) + \sum_{j\in L_l^c} \gamma_j^l \left(-\hat{e}_{c_j}^k + \left\langle \hat{g}_{c_j}^k, y^{k+1} - x^k \right\rangle\right), \end{aligned}$$

which implies that

$$\begin{aligned} L(y^{k+1}, \lambda^k, \gamma^k) = &\sum_{j\in L_k^f} \lambda_j^k(\hat{f}^k - \theta_1^k) + \sum_{j\in L_k^c} \gamma_j^k(\hat{c}^k - \theta_2^k) + \frac{1}{2}\|y^{k+1} - x^k\|_k^2 \\ &- \left(\sum_{j\in L_k^f} \lambda_j^k \hat{e}_{f_j}^k + \sum_{j\in L_k^c} \gamma_j^k \hat{e}_{c_j}^k \right) + \left\langle G^k, y^{k+1} - x^k \right\rangle. \end{aligned}$$

On the other hand,

$$L(y^{k+1}, \lambda^k, \gamma^k) = \Psi_k(y^{k+1}) + \frac{1}{2}\|y^{k+1} - x^k\|_k^2,$$

implies that

$$\begin{aligned} \sum_{j\in L_k^f} \lambda_j^k \hat{e}_{f_j}^k + \sum_{j\in L_k^c} \gamma_j^k \hat{e}_{c_j}^k = &\sum_{j\in L_k^f} \lambda_j^k(\hat{f}^k - \theta_1^k) \\ &+ \sum_{j\in L_k^c} \gamma_j^k(\hat{c}^k - \theta_2^k) - \Psi_k(y^{k+1}) + \left\langle G^k, y^{k+1} - x^k \right\rangle. \end{aligned} \tag{3.27}$$

By noting (3.1), we can obtain that

$$h_k(x^k) = \max\left\{\hat{f}^k - \theta_1^k, \hat{c}^k - \theta_2^k\right\} \ge \sum_{j\in L_k^f} \lambda_j^k \left(\hat{f}^k - \theta_1^k\right) + \sum_{j\in L_k^c} \gamma_j^k \left(\hat{c}^k - \theta_2^k\right).$$

Together with the functional relations (3.19) and (3.21), we obtain that

$$V_k \geq \sum_{j\in L_k^f} \lambda_j^k \hat{e}_{f_j}^k + \sum_{j\in L_k^c} \gamma_j^k \hat{e}_{c_j}^k .$$

It follows from (3.10) for selecting μ_k that the relation (3.26) holds. From here, the required result follows. □

Our method is based on the ideas of [31], and extend them to nonconvex constrained optimization problems. We now give our bundle method for solving problem (1.1)

Algorithm 3.1

Step 0 Initialization.

Choose an initial point $x^0 \in \mathbf{R}^n$, set $y^0 = x^0$, and compute $g_f^0 \in f(y^0),\ g_c^0 \in c(y^0)$. Initialize $k = 0$, and set the bundle index sets $L_0^f := \{0\}$, and $L_0^c := \{0\}$.
Choose a stopping tolerance $tol_{stop} \geq 0$, an Armijo-like parameter $m \in (0, 1)$, a positive constant $\iota > 0$.
Select two initial parameters η_0 and μ_0, penalty parameters s_0 and t_0 satisfying (3.3).
Choose a symmetric matrix A_0 of order n and a prox-parameter η_0 ensuring that the matrix $M_0 = A_0 + \eta_0 I$ is positive definite.

Step 1 Trial step generation.

Having the current model $\Psi_k(\cdot)$ defined by (3.11), find an unique solution y^{k+1} of the quadratic programming (3.15), i.e.,

$$y^{k+1} = \arg\min_{y\in\mathcal{X}} \left\{\Psi_k(y) + \frac{1}{2}\|y - x^k\|_k^2\right\}.$$

Compute the aggregate error V_k and the predicted decrease δ_k,

$$V_k := h_k(x^k) - \psi_k(x^k) \text{ and } \delta_k := V_k + \|y^{k+1} - x^k\|_k^2.$$

Call the inexact oracle at y^{k+1} to compute $f^{k+1},\ c^{k+1},\ g_f^{k+1},\ g_c^{k+1}$ satisfying (2.11).

Step 2 Stopping test.

If $\delta_k \leq tol_{\text{stop}}$, stop.

Step 3 Acceptance test.

If the descent condition (3.25) holds, declare a new stability center $x^{k+1} = y^{k+1}$, $\hat{f}^{k+1} = f^{k+1}$, and $\hat{c}^{k+1} = c^{k+1}$. Compute a new symmetric matrix A_{k+1} and a prox-parameter $\eta_{k+1} \geq 0$ such that $M_{k+1} = A_{k+1} + \eta_{k+1} I$ is positive definite. Choose two new penalty parameters $s_{k+1},\ t_{k+1}$ satisfying (3.3).
Otherwise, declare a null step $x^{k+1} = x^k$, $\hat{f}^{k+1} = \hat{f}^k$, and $\hat{c}^{k+1} = \hat{c}^k$. Let $A_{k+1} := A_k$, and select $\eta_{k+1} \geq \eta_k,\ M_{k+1} = A_{k+1} + \eta_{k+1} I$. Let the penalty parameters $s_{k+1} = s_k$ and $t_{k+1} = t_k$.

Step 4 Bundle management.

Select the new parameter μ_{k+1} as (3.10). Choose the next index set satisfying

$$\begin{cases} L_{k+1}^f \supseteq \{k+1,\ j_k\} \text{ and } L_{k+1}^f \supseteq \{j \in L_k^f : \lambda_j^k > 0\}, \\ L_{k+1}^c \supseteq \{k+1,\ j_k\} \text{ and } L_{k+1}^c \supseteq \{j \in L_k^c : \gamma_j^k > 0\}. \end{cases} \tag{3.28}$$

Increase k by 1 and go to Step 1.

We now suppose that $\{\mu_k\}$ is bounded. Boundedness of the sequence of $\{\mu_k\}$ has been proved in [17,55] for the lower-C^2 functions with an exact oracle. In theory, inexactness may result in an unbounded μ_k in our setting. The numerical experiments in Sect. 6 show that $\{\mu_k\}$ is bounded under various kinds of perturbations, and our inexact method is satisfactory.

4 Asymptotic analysis

We now analyze the different cases that can arise when Algorithm 3.1 loops forever. As usual in the convergence analysis of bundle methods, we consider the following two possible cases:

- either there are infinitely many serious steps, or
- there is an finite number of serious steps, followed by infinitely many null steps.

We start with the case of infinitely many serious steps.

4.1 Infinite serious steps

Theorem 4.1 *Suppose that Algorithm 3.1 generates an infinite sequence $\{x^k\}$ of serious steps. Let $\mathcal{L}_s$ denote the set gathering indices of serious steps. Then $\delta_k \to 0$ and $V_k \to 0$ as $\mathcal{L}_s \ni k \to \infty$.*

- *In addition, if the series $\sum_{k\in\mathcal{L}_s} \frac{1}{\lambda_{\max}(A_k)+\eta_k}$ is divergent, then $\liminf_{k\to\infty} \|G^k+\alpha^k\| = 0$. Besides, there exists some subsequence $\mathcal{K}_s \subseteq \mathcal{L}_s$ and an accumulation $\hat{x}$ such that $x^k \to \hat{x}$ and $G^k \to \hat{G}$ as $\mathcal{K}_s \ni k \to \infty$.*
- *If, instead of the above weaker condition, the sequence $\{\lambda_{\max}(A_k) + \eta_k\}$ is bounded above, then $\lim_{k\to\infty} \|G^k + \alpha^k\| = 0$, and same assertions hold for all accumulation points of the sequence $\{x^k\}$.*

Proof We examine the following two different possibilities respectively. In the first case, if there exists some $\tilde{k}$ such that $\hat{c}^{\tilde{k}} \le 0$, together with satisfaction of the first condition in (3.25), this means that

$$\hat{c}^{k+1} \le 0, \quad \text{for all } \mathcal{L}_s \ni k \ge \tilde{k}.$$

and

$$0 < m\delta_k \le \hat{f}^k - \hat{f}^{k+1}, \quad \text{for all } \mathcal{L}_s \ni k \ge \tilde{k}. \tag{4.1}$$

Together with the basic assumptions on f, we can obtain that the sequence $\{\hat{f}^k\}_{\mathcal{L}_s}$ is decreasing and bounded. Hence there exists some $\hat{f}$ such that the sequence $\{\hat{f}^k\}_{\mathcal{L}_s}$ converges to $\hat{f}$ satisfying $\hat{f} \le \hat{f}^k$ for all $k \ge \tilde{k}$ in $\mathcal{L}_s$. Summing up the relations (4.1), we obtain that

$$\sum_{\mathcal{L}_s \ni k \ge \tilde{k}} \delta_k \le \frac{1}{m} \sum_{\mathcal{L}_s \ni k \ge \tilde{k}} (\hat{f}^k - \hat{f}^{k+1}) = \frac{1}{m}(\hat{f}^{\tilde{k}} - \hat{f}),$$

which implies $\delta_k \to 0$ as $\mathcal{L}_s \ni k \to \infty$.

On the other hand, if $\hat{c}^k > 0$ for all $k \in \mathcal{L}_s$, then (3.25) implies that

$$0 < \delta_k \le \frac{1}{m}(\hat{c}^k - \hat{c}^{k+1}).$$

By the same argument, the sequence $\{\hat{c}^k\}$ is decreasing and bounded below. Then it converges to a value $\hat{c}$ satisfying $\hat{c} \le \hat{c}^k$ for all $k \ge \tilde{k}$ in $\mathcal{L}_s$. As a result, summing up the above relation over all $\mathcal{L}_s$, we see that

$$\sum_{k\in\mathcal{L}_s} \delta_k \le \frac{1}{m} \sum_{k\in\mathcal{L}_s} (\hat{c}^k - \hat{c}^{k+1}) = \frac{1}{m}(\hat{c}^0 - \hat{c}),$$

which means $\lim_{k\in\mathcal{L}_s} \delta_k = 0$. Together with (3.14) and (3.24) we see that

$$0 \le V_k \le \delta_k \quad \text{and} \quad \frac{1}{\lambda_{\max}(A_k) + \eta_k} \|G^k + \alpha^k\|^2 \le | G^k + \alpha^k |_k^2 \le \delta_k,$$

and, hence $V_k \to 0$ as $\mathcal{L}_s \ni k \to \infty$.

By noting the relation (3.24), we obtain that

$$\sum_{k\in\mathcal{L}_s} \delta_k = \sum_{k\in\mathcal{L}_s} \left(V_k + | G^k + \alpha^k |_k^2 \right) < \infty.$$

Since all the quantities are nonnegative, and together with (3.14), it holds that

$$0 < \sum_{k\in\mathcal{L}_s} \frac{1}{\lambda_{\max}(A_k) + \eta_k} \|G^k + \alpha^k\|^2 \le \sum_{k\in\mathcal{L}_s} | G^k + \alpha^k |_k^2 < \infty.$$

If a weaker condition holds, i.e., $\sum_{k\in\mathcal{L}_s} \frac{1}{\lambda_{\max}(A_k)+\eta_k}$ is divergent, then we can obtain $\liminf_{k\to\infty} \|G^k + \alpha^k\| = 0$. Passing onto the subsequence $\mathcal{K}_s$, if necessary, we can suppose $\{x^k\} \to \hat{x}$ and $G^k \to \hat{G}$ as $\mathcal{K}_s \ni k \to \infty$. From here first item is now proven. Furthermore, if the sequence $\{\lambda_{\max}(A_k) + \eta_k\}$ is bounded above, we obtain that

$$\|G^k + \alpha^k\| \to 0, \quad \mathcal{L}_s \ni k \to \infty, \tag{4.2}$$

which implies that the same assertions can be proven for all accumulations of $\{x^k\}$. From here all results have been proven. □

4.2 Infinitely many consecutive null steps

The another case refers to finitely many serious steps, which implies Algorithm 3.1 makes infinitely many consecutive null steps. Let $\hat{k}$ denote the last serious index, and $\hat{x} := x^{\hat{k}}$ be the last stability center. By adjusting the strategy of the bundle management, the model is ensured to satisfy the following conditions whenever k and $k+1$ are two consecutive null indexes:

$$\begin{aligned} \check{f}_{k+1}(y) &\ge \hat{f}^{k+1} - \hat{e}^{k+1}_{f_{k+1}} + \left\langle \hat{g}^{k+1}_{f_{k+1}}, y - \hat{x} \right\rangle, \\ \check{c}_{k+1}(y) &\ge \hat{c}^{k+1} - \hat{e}^{k+1}_{c_{k+1}} + \left\langle \hat{g}^{k+1}_{c_{k+1}}, y - \hat{x} \right\rangle, \end{aligned} \tag{4.3}$$

and

$$\Psi_{k+1}(y) \ge \psi_k(y). \tag{4.4}$$

Moreover, let r_k denote the objective function optimal value of the (3.15), i.e.,

$$r_k := \Psi_k(y^{k+1}) + \frac{1}{2}\|y^{k+1} - x^k\|_k^2. \tag{4.5}$$

We first prove the following preliminary result regarding relations between consecutive terms of the sequence $\{r_k\}$.

Lemma 4.1 *Suppose that Algorithm 3.1 takes a finite number of serious steps. The optimal values r_{k+1} and r_k satisfy*

$$r_{k+1} \geq r_k + \frac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2. \tag{4.6}$$

Then, the following relations hold:

$$\begin{aligned} &\lim_{k\to\infty} \|y^{k+2} - y^{k+1}\|_k^2 = 0 \\ &\lim_{k\to\infty} \left(r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2\right) = 0. \end{aligned} \tag{4.7}$$

Proof By noting the definition of ψ_k in (3.19), it is easy to see that

$$\begin{aligned} \psi_k(y) &= \Psi_k(y^{k+1}) + \left\langle G^k, y - y^{k+1}\right\rangle \\ &= \Psi_k(y^{k+1}) + (\hat{x} - y^{k+1})^{\mathrm{T}} M_k(y - y^{k+1}) - \left\langle \alpha^k, y - y^{k+1}\right\rangle \\ &\geq \Psi_k(y^{k+1}) + \frac{1}{2}\|y^{k+1} - \hat{x}\|_k^2 + \frac{1}{2}\|y - y^{k+1}\|_k^2 - \frac{1}{2}\|y - \hat{x}\|_k^2 \\ &= r_k + \frac{1}{2}\|y - y^{k+1}\|_k^2 - \frac{1}{2}\|y - \hat{x}\|_k^2, \end{aligned}$$

where the inequality has used $\alpha^k \in \partial i_{\mathcal{X}}(y^{k+1})$, and the last equality follows from (4.5). Hence we can obtain that

$$\psi_k(y) + \frac{1}{2}\|y - \hat{x}\|_k^2 \geq r_k + \frac{1}{2}\|y - y^{k+1}\|_k^2. \tag{4.8}$$

By evaluating at $y = \hat{x}$, it follows that

$$r_k + \frac{1}{2}\|\hat{x} - y^{k+1}\|_k^2 \leq \psi_k(\hat{x}) \leq \Psi_k(\hat{x}),$$

where the last inequality has used (3.20), i.e., $\psi_k(y) \leq \Psi_k(y)$. Hence the sequence $\{r_k\}$ is bounded from above. From (4.4), (4.8), and by noting the fact $M_{k+1} \succeq M_k$, we can obtain that

$$r_k + \frac{1}{2}\|y - y^{k+1}\|_k^2 \leq \Psi_{k+1}(y) + \frac{1}{2}\|y - \hat{x}\|_k^2 \leq \Psi_{k+1}(y) + \frac{1}{2}\|y - \hat{x}\|_{k+1}^2.$$

By evaluating at $y = y^{k+2}$, we obtain that

$$r_k + \frac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2 \leq r_{k+1} = \Psi_{k+1}(y^{k+2}) + \frac{1}{2}\|y^{k+2} - \hat{x}\|_{k+1}^2,$$

which implies that the relation (4.6) holds. Since the sequence $\{r_k\}$ is monotone increasing and bounded from above, we also obtain the result (4.7) holds. □

Theorem 4.2 *Suppose that Algorithm 3.1 generates a finite sequence of serious steps, and the sequence $\{\mu_k\}$ be bounded. Let $\mathcal{L}_n$ denote the set gathering indices of iterations larger than $\hat{k}$. Then $\delta_k \to 0$ and $V_k \to 0$ as $\mathcal{L}_n \ni k \to \infty$.*

- *If, in addition, $\liminf_{k\to\infty} \frac{1}{\eta_k} > 0$, then there exists a subsequence $\mathcal{K}_n \subseteq \mathcal{L}_n$, such that $\|G^k + \alpha^k\| \to 0$, $y^k \to \hat{x}$ and $G^k \to \bar{G}$ as $\mathcal{K}_n \ni k \to \infty$.*
- *If, instead of the above weaker condition, there exists $\eta_{\max} > 0$ such that $\eta_k \leq \eta_{\max}$, then $\|G^k + \alpha^k\| \to 0$ and $y^k \to \hat{x}$ as $\to \infty$.*

Proof For convenience, let $\hat{x} = x^k$, $\hat{f} = \hat{f}^k$, $\hat{c} = \hat{c}^k$, $\hat{A} = A_{\hat{k}}$ $\hat{\theta}_1 = \theta_1^k$, $\hat{\theta}_2 = \theta_2^k$, and $\hat{h} = h_k(x^k)$ for all $k \geq \hat{k}$. By noting the definition of the inexact improvement function, we can obtain that

$$h_k(y^{k+1}) = \max\left\{f^{k+1} - \theta_1^k, c^{k+1} - \theta_2^k\right\}$$
$$= \begin{cases} \max\left\{f^{k+1} - \hat{f} - s_k\hat{c}, c^{k+1} - t_k\hat{c}\right\}, & \text{if } \hat{c} > 0, \\ \max\left\{f^{k+1} - \hat{f}, c^{k+1}\right\}, & \text{if } \hat{c} \leq 0 \end{cases}$$

If $\hat{c} > 0$, yields,

$$\begin{aligned} h_{x^k}(y^{k+1}) &\geq c^{k+1} - t_k\hat{c} \\ &\geq \hat{c} - t_k\hat{c} - m\delta_k \\ &= \hat{h} - m\delta_k, \end{aligned} \tag{4.9}$$

where the second inequality follows from (3.25), and the last equality has used the relation (3.4).

On the other hand, if $\hat{c} \leq 0$, it follows from (3.4) that $\hat{h} = 0$. By combining with (3.25), we get

$$\begin{aligned} h_k(y^{k+1}) &\geq \begin{cases} f^{k+1} - \hat{f}, & \text{and} \\ c^{k+1} \end{cases} \\ &> \begin{cases} -m\delta_k, & \text{if } f^{k+1} > \hat{f} - m\delta_k, \\ 0, & \text{if } c^{k+1} > 0 \end{cases} \\ &\geq \hat{h} - m\delta_k. \end{aligned} \tag{4.10}$$

By adding δ_k to both terms, we can obtain that

$$0 \leq (1-m)\delta_k < h_k(y^{k+1}) - \hat{h} + \delta_k \leq h_k(y^{k+1}) - \Psi_k(y^{k+1}), \tag{4.11}$$

where the last inequality has used the relation (3.23), that is

$$-\hat{h} + \delta_k = -\Psi_k(y^{k+1}) + \langle \alpha^k, \hat{x} - y^{k+1}\rangle \leq -\Psi_k(y^{k+1}).$$

Following (3.9) and (4.3), we can observe that

$$\begin{aligned} \check{f}_{k+1}(y^{k+2}) &\geq \hat{f} - \hat{e}_{f_{k+1}}^{k+1} + \left\langle \hat{g}_{f_{k+1}}^{k+1}, y^{k+2} - \hat{x}\right\rangle \\ &= f^{k+1} + \left\langle g_{f_{k+1}}, \hat{x} - y^{k+1}\right\rangle - \frac{\mu_{k+1}}{2}\|y^{k+1} - \hat{x}\|^2 + \left\langle g_{f_{k+1}} + \mu_{k+1}(y^{k+1} - \hat{x}), y^{k+2} - \hat{x}\right\rangle \\ &\geq f^{k+1} + \left\langle g_{f_{k+1}}, y^{k+2} - y^{k+1}\right\rangle + \mu_{k+1}\left\langle y^{k+1} - \hat{x}, y^{k+2} - y^{k+1}\right\rangle \\ &= f^{k+1} + \left\langle \hat{g}_{f_{k+1}}^{k+1}, y^{k+2} - y^{k+1}\right\rangle. \end{aligned} \tag{4.12}$$

By the same argument, we obtain that

$$\check{c}_{k+1}(y^{k+2}) \geq c^{k+1} + \left\langle \hat{g}_{c_{k+1}}^{k+1}, y^{k+2} - y^{k+1}\right\rangle. \tag{4.13}$$

Besides, from (3.11), (4.12) and (4.13), we get

$$\begin{aligned} \Psi_{k+1}(y^{k+2}) &= \left\{\check{f}_{k+1}(y^{k+2}) - \hat{\theta}_1, \check{c}_{k+1}(y^{k+2}) - \hat{\theta}_2\right\} \\ &\geq \left\{f^{k+1} + \left\langle \hat{g}_{f_{k+1}}^{k+1}, y^{k+2} - y^{k+1}\right\rangle - \hat{\theta}_1, c^{k+1} + \left\langle \hat{g}_{c_{k+1}}^{k+1}, y^{k+2} - y^{k+1}\right\rangle - \hat{\theta}_2\right\}. \end{aligned}$$

By noting the sequence $\{y^k\} \subset \mathcal{X}$, there exists a positive constant $N > 0$ such that $\|y^k - \hat{x}\| \leq N$ for all k. Therefore,

$$\begin{aligned}\Psi_{k+1}(y^{k+2}) &\geq \left\{f^{k+1} - \theta_1^k, c^{k+1} - \theta_2^k\right\} - (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\| \\ &= h_k(y^{k+1}) - (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\|,\end{aligned}$$

where the inequality has used the fact that g_c^{k+1} and g_f^{k+1} are bounded (recall that f and c are locally Lipschitzian). By combining with (4.11), we obtain that

$$\begin{aligned}&0 \leq (1-m)\delta_k \\ &< \Psi_{k+1}(y^{k+2}) - \Psi_k(y^{k+1}) + (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\| \\ &= r_{k+1} - \frac{1}{2}\|y^{k+2} - \hat{x}\|_{k+1}^2 - r_k + \frac{1}{2}\|y^{k+1} - \hat{x}\|_k^2 + (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\|,\end{aligned}$$

where the last equality comes from the relation (4.5). From the relation (3.14) and the condition $M_{k+1} \succ M_k$, we get

$$\begin{aligned}&0 \leq (1-m)\delta_k \\ &\quad < r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - \hat{x}\|_k^2 + \tfrac{1}{2}\|y^{k+1} - \hat{x}\|_k^2 + (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\| \\ &\quad = \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2 - \tfrac{1}{2}\|y^{k+2} - \hat{x}\|_k^2 + \tfrac{1}{2}\|y^{k+1} - \hat{x}\|_k^2 \\ &\qquad + (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\| + r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2 \\ &\quad = \langle y^{k+2} - y^{k+1}, M_k^{-1}(G^k + \alpha^k)\rangle + (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\| + r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2 \\ &\quad \leq \langle y^{k+2} - y^{k+1}, \tfrac{1}{\lambda_{\min}(\hat{A}) + \hat{\eta}}(G^k + \alpha^k)\rangle + (L + \mu_{k+1}N)\|y^{k+2} - y^{k+1}\| + r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2 \\ &\quad \leq \left(L + \mu_{k+1}N + \tfrac{1}{\lambda_{\min}(\hat{A}) + \hat{\eta}}\|G^k\|\right)\|y^{k+2} - y^{k+1}\| + r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2 \\ &\quad \leq \left(L + \mu_{k+1}N + \tfrac{L}{\lambda_{\min}(\hat{A}) + \hat{\eta}}\right)\|y^{k+2} - y^{k+1}\| + r_{k+1} - r_k - \tfrac{1}{2}\|y^{k+2} - y^{k+1}\|_k^2,\end{aligned}$$

where the last inequality follows from $G^k \in conv\{g_f^j,\ g_c^j,\ j \in L_k\}$ and $\hat{\eta} := \eta_{\hat{k}} \leq \eta_k$ for all $k \geq \hat{k}$. Passing to the limit as $k \to \infty$, and using (4.7) in Lemma 4.1, we obtain that $\lim_{k\to\infty} \delta_k = 0$. Together with (3.14) and (3.24) we see that

$$0 \leq V_k \leq \delta_k \quad \text{and} \quad \frac{1}{\left(\lambda_{\max}(\hat{A}) + \eta_k\right)}\|G^k + \alpha^k\|^2 \leq | G^k + \alpha^k |_k^2 \leq \delta_k,$$

Since $\liminf_{k\to\infty} \frac{1}{\eta_k} > 0$, we obtain that $\lim_{\mathcal{K}\ni k\to\infty} V_k = 0$ and $\lim_{\mathcal{K}\ni k\to\infty} \|G^k + \alpha^k\| = 0$ for some subsequence $\mathcal{K}$. Furthermore, it is follows from (3.17) and $M_{k+1} \succeq M_k$ that $\lim_{\mathcal{K}\ni k\to\infty} y^k = \hat{x}$. If $\eta_k \leq \eta_{\max}$, it is obviously that $\|G^k + \alpha^k\| \to 0$ and $y^k \to \hat{x}$ as $\to \infty$. From here the results have been proved. □

5 Convergence results

One of purposes of conditions (3.3) is to ensure satisfaction of the relations stated in the following lemma. Let x^* denote an accumulation point of $\{x^k\}$, (f^*, c^*) and (s_*, t_*) be the limit points of $\{(f^k, c^k)\}$ and $\{(x_k, t_k)\}$ respectively. We can define the constant ϖ in (3.3) satisfying $\varpi > \frac{1}{c^*}(\max_{x\in\mathcal{X}}\{f(y) - c(y)\} - f^*)$, which implies that the penalty parameters s_k, t_k satisfy

$$s_k - t_k > \left(\max_{x\in\mathcal{X}}\{f(y) - c(y)\} - f^*\right)/c^*. \tag{5.1}$$

Theorem 5.1 *Suppose that Algorithm 3.1 solves (1.1) satisfying (2.11), (2.12) with penalty parameters s_k and t_k satisfying (3.3).*

If the algorithm loops forever, then for each $\theta^* > 0$, there exists a positive constant ρ^* such that

- $$(1 - t^*) \max\{c^*, 0\} \le \max\left\{ f(y) - f^* - s_* \max\{c^*, 0\}, c(y) - t_* \max\{c^*, 0\} \right\} + \epsilon, \tag{5.2}$$

  for all $y \in B_{\rho^*}(x^*) \bigcap \mathcal{X}$, where $\epsilon := (\theta^* + \bar{\varepsilon})\rho^* + \bar{\sigma}$.
- *If $c^* > 0$ then*
  $$c^* \le c(y) + \epsilon \ \text{ for all } y.$$
- *If $c^* < 0$ and $X_\epsilon := \{y \in X : c(y) < -\epsilon\}$ is not empty set, it holds that*
  $$f^* \le f(y) + \epsilon \ \text{ for all } \ y \in X_\epsilon.$$

Proof If Algorithm 3.1 loops forever, since the set $\mathcal{X}$ is compact and $x^k \in \mathcal{X}$, there exists an index set $\mathcal{L}'$ such that $\{x^k\}_{k\in\mathcal{L}'} \to x^*$, and recalling that when $\mathcal{L}' = \mathcal{L}_s$ eventually $x^* = \lim_{k\to\infty} y^k$. Let $q_f^j \in \partial f(x^j)$ and by noting (3.7) and (3.9), we obtain that

$$\begin{aligned} &\hat{f}^k + \left\langle g_f^j, y - x^k \right\rangle + p_j^k - \hat{e}_{f_j}^k \\ &\quad = f^j + \left\langle g_f^j, y - y^j \right\rangle \\ &\quad = f(y^j) + \left\langle q_f^j, y - y^j \right\rangle - \sigma_j + \left\langle g_f^j - q_f^j, y - y^j \right\rangle, \end{aligned} \tag{5.3}$$

where the last equality follows from (2.11). By noting the definition of lower-$\mathcal{C}^1$, then for all y^j and $\theta^j > 0$, there exists $\rho^j > 0$ such that

$$f(y) - f(y^j) \ge \langle q_f^j, y - y^j \rangle - \theta^j \|y - y^j\|, \ \text{for all } y \in B_{\rho^j}(y^j),$$

which combined with the relation (5.3), implies that

$$\begin{aligned} &\hat{f}^k + \left\langle g_f^j, y - x^k \right\rangle + p_j^k - \hat{e}_{f_j}^k \\ &\quad \le f(y) + (\theta^j + \varepsilon_j)\|y - y^j\| - \sigma_j \\ &\quad \le f(y) + (\theta^j + \varepsilon_j)\rho^j - \sigma_j. \end{aligned} \tag{5.4}$$

Together with satisfaction of (3.9c), i.e., $g_f^j = \hat{g}_{f_j}^k - \mu_k(y^j - x^k)$, it holds that

$$\begin{aligned} &f(y) + (\theta^j + \varepsilon_j)\rho^j - \sigma_j \\ &\quad \ge \hat{f}^k + \left\langle g_f^j, y - x^k \right\rangle + p_j^k - \hat{e}_{f_j}^k \\ &\quad = \hat{f}^k - \hat{e}_{f_j}^k + \left\langle \hat{g}_{f_j}^k, y - x^k \right\rangle - \mu_k \left\langle y^j - x^k, y - x^k \right\rangle + p_j^k \\ &\quad \ge \hat{f}^k - \hat{e}_{f_j}^k + \left\langle \hat{g}_{f_j}^k, y - x^k \right\rangle - \mu_k \left\langle y^j - x^k, y - x^k \right\rangle, \end{aligned}$$

where the last inequality follows from $p_j^k \ge 0$. The above relation will eventually mean that

$$f(y) \ge \hat{f}^k - \hat{e}_{f_j}^k + \left\langle \hat{g}_{f_j}^k, y - x^k \right\rangle - \mu_k \left\langle y^j - x^k, y - x^k \right\rangle - (\theta^j + \varepsilon_j)\rho^j + \sigma_j. \tag{5.5}$$

By same argument, it holds that

$$c(y) \geq \hat{c}^k - \hat{e}^k_{c_j} + \langle \hat{g}^k_{c_j}, y - x^k \rangle - \mu_k \langle y^j - x^k, y - x^k \rangle - (\theta^j + \varepsilon_j)\rho^j + \sigma_j. \quad (5.6)$$

It follows from (5.5) and (5.6) that

$$\max\left\{f(y) - \theta^k_1, c(y) - \theta^k_2\right\} + (\theta^j + \bar{\varepsilon})\rho^j + \bar{\sigma}$$
$$\geq -\left(\sum_{j \in L^f_k} \lambda^k_j \hat{e}^k_{f_j} + \sum_{j \in L^c_k} \gamma^k_j \hat{e}^k_{c_j}\right) + \langle G^k, y - x^k \rangle + \sum_{j \in L^f_k} \lambda^k_j (\hat{f}^k - \theta^k_1) + \sum_{j \in L^c_k} \gamma^k_j (\hat{c}^k - \theta^k_2)$$
$$- \mu_k \langle \sum_{j \in L^f_k} \lambda^k_j (y^j - x^k) + \sum_{j \in L^c_k} \gamma^k_j (y^j - x^k), y - x^k \rangle.$$

Besides, by noting (3.27) and (3.23) it holds that

$$\begin{aligned}
&\sum_{j \in L^f_k} \lambda^k_j (\hat{f}^k - \theta^k_1) + \sum_{j \in L^c_k} \gamma^k_j (\hat{c}^k - \theta^k_2) \\
&\quad = \sum_{j \in L^f_k} \lambda^k_j \hat{e}^k_{f_j} + \sum_{j \in L^c_k} \gamma^k_j \hat{e}^k_{c_j} + \Psi_k(y^{k+1}) - \langle G^k, y^{k+1} - x^k \rangle \\
&\quad = \sum_{j \in L^f_k} \lambda^k_j \hat{e}^k_{f_j} + \sum_{j \in L^c_k} \gamma^k_j \hat{e}^k_{c_j} + h_k(x^k) - \delta_k + \mid G^k + \alpha^k \mid^2_k \\
&\quad = \sum_{j \in L^f_k} \lambda^k_j \hat{e}^k_{f_j} + \sum_{j \in L^c_k} \gamma^k_j \hat{e}^k_{c_j} + h_k(x^k) - V_k, \qquad (5.7)
\end{aligned}$$

which implies that

$$\max\left\{f(y) - \theta^k_1, c(y) - \theta^k_2\right\} + (\theta^j + \bar{\varepsilon})\rho^j + \bar{\sigma}$$
$$\geq h_k x^k) - V_k + \left\langle G^k, y - x^k \right\rangle - \mu_k \left\langle \sum_{j \in L^f_k} \lambda^k_j (y^j - x^k) + \sum_{j \in L^c_k} \gamma^k_j (y^j - x^k), y - x^k \right\rangle.$$

By noting Theorem 4.1, Theorem 4.2 and Lemma 5.1, and passing to the limit in above relation as $k \to \infty$, we obtain that

$$\max\left\{f(y) - f^* - s_* \max\{c^*, 0\}, c(y) - t_* \max\{c^*, 0\}\right\} + (\theta^* + \bar{\varepsilon})\rho^* + \bar{\sigma}$$
$$\geq (1 - t_*) \max\{c^*, 0\} + \langle \bar{G}, y - \bar{x} \rangle.$$

As already seen, $G^k + \alpha^k \to G^* + \alpha^* = 0$ and $\alpha^* \in \partial i_{\mathcal{X}}(x^*)$, implies that $-G^* \in \partial i_{\mathcal{X}}(x^*)$, thus $\langle -G^*, y - x^* \rangle \leq 0$. From here the result in first item has been proven.

To show second item, consider the condition of $c^* > 0$. Recalling the relation (5.1), yields

$$(s_* - t_*)c^* > f(y) - c(y) - f^*,$$

which implies that

$$f(y) - f^* - s_* \max\{c^*, 0\} < c(y) - t_* \max\{c^*, 0\}.$$

Together with the relation (5.2), we obtain that

$$c^* \leq c(y) + \epsilon.$$

Then the result in second item has been proven.

Finally, to see item (iii), consider condition $c^* < 0$. According to (5.2), it holds that

$$0 \leq \max\left\{f(y) - f^*, c(y)\right\} + \epsilon,$$

which means $f^* \leq f(y) + \epsilon$ for $x \in X_\epsilon$. From here all results have been proven. $\square$

From the third statement of lower-C^1 in (2.3), the constants θ^* and ρ^* for the point x^* can be set small enough. Thus Theorem 5.1 states that, as long as θ^* and ρ^* are chosen properly, the constant ϵ should meet the required precision accordingly. Besides, Theorem 5.1 also indicate that, if the parameter s_k is updated quickly (see e.g., $s_{k+1} = 2s_k$ if $\hat{c}^k > 0$, and $s_{k+1} = s_k$ otherwise), Algorithm 3.1 will eventually find a point infeasible up to the accuracy problem (1.1). Meanwhile, by noting the relation $c(y) > c^* - \epsilon$, then the feasible set of (1.1) may be empty set or very small. Otherwise, the set X_ϵ should be nonempty as long as the accuracy ϵ is small enough and an approximate local solution of (1.1) will be detected.

Our inexact bundle method can also be given for lower-C^2 functions. The function f is lower-C^2 on a open set $\mathcal{M}$ as long as it is finite value on an open set $\mathcal{M}$ and for any point $\bar{x} \in \mathcal{M}$ there exists a constant $R_{\text{th}} > 0$ such that $f + \frac{r}{2}|-\bar{x}|^2$ is convex on an open neighborhood $\mathcal{M}'$ of $\bar{x}$ for all $r > R_{\text{th}}$. It has been shown the lower-C^2 functions are locally Lipschitz continuous.

The prox-regularity is essential when working with proximal points in a nonconvex setting. Specifically, if f is a prox-regular locally Lipschitz function, then it is lower-C^2 function. The finite convex function is lower-C^2 [44, Theorem 10.31]. There are also some other lower-C^2 functions, i.e., C^2 functions, strongly amenable functions, and if f is l.s.c., proper, and prox-bounded, then the opposite of Moreau envelopes $e_\lambda f$ is lower-C^2.

By applying [38, Prop. 10.54] and [55, Lemma 4.2], we obtain that there exist two thresholds μ_f, μ_c, such that for all $\mu \geq \rho^{id} := \max\{\mu_f, \mu_c\}$, and any given point $y \in \mathcal{L}_0$ (a nonempty compact set), the functions $f(\cdot) + \frac{\mu}{2}|\cdot - y|^2$, $c(\cdot) + \frac{\mu}{2}|\cdot - y|^2$ are convex on $\mathcal{L}_0$. Besides, the positive threshold ρ^{id} can be updated along iterations, by using data in the bundle generated by Algorithm 3.1. Besides, from [55, Lemma 4.2] the parameter μ_k remain unchanged eventually, that is there exist $\hat{k} > 0$ and $\bar{\mu} > 0$, such that $\mu_k = \bar{\mu}$, for all $k > \hat{k}$. The similar results as Theorem 5.1 can be proved naturally.

6 Numerical experiments

To assess practical performance of our inexact proximal bundle method, we coded Algorithm 3.1 in Matlab and ran it on a PC with 1.80 GHz CPU. Quadratic programming solver is `QuadProg.m`, which is available in the Optimization Toolbox.

6.1 Parameters for the proximal bundle method

We first set, respectively, the minimum and maximum positive thresholds to be $\eta_{\min} = 10^{-5}$ and $\eta_{\max} = 10^9$. Once the number of active elements in the bundle $\mathcal{G}_l$ is more than $num(\mathcal{G}_{\max}) = 50$, the bundle should be compressed. The initial value of the prox-parameter is started $\eta_0 = 1$.

We first compute $\tilde{\eta}_k$ by using the reverse quasi-Newton scalar

$$\tilde{\eta}_k := \left\{ \frac{|\chi^k - \chi^{k-1}|^2}{(\chi^k - \chi^{k-1})^{\mathrm{T}}(y^k - y^{k-1})} : \chi^k = G^k + \alpha^k,\ \chi^{k-1} = G^{k-1} + \alpha^{k-1} \right\}.$$

Just as [37], the prox-parameter η_{k+1} will be updated at each serious steps as below.

$$\eta_{k+1} = \min\{\eta, \eta_{\max}\},$$

where

$$\eta := \begin{cases} \max\{\eta_{\min}, \eta_k, -1.01\lambda_{\min}(A_k)\}, & \text{if } \lambda_{\min}(A_k) < 0 \\ \max\{\eta_{\min}, \tilde{\eta}_k, \}, & \text{if } \lambda_{\min}(A_k) = 0 \\ \max\{0, \lambda_{\min}(A_k) - \tilde{\eta}_k\}, & \text{if } \lambda_{\min}(A_k) > 0. \end{cases} \tag{6.1}$$

If the kth iteration declares a null step, then the prox-parameter η_k remains constant. This strategy satisfies the condition of the parameters in Theorems 4.1 and 4.2. If necessary, η_{k+1} is projected so that $\eta_{k+1} = [\eta_{\min}, \eta_{\max}]$.

Let $\iota = 2$, and update the convexification parameter μ_k by using the relation (3.10), i.e.,

$$\mu_k := \max\left\{ \max_{j \in L_k^f, y^j \neq x^k} \frac{-e_{f_j}^k}{\frac{1}{2}\|y^j - x^k\|^2}, \max_{j \in L_k^c, y^j \neq x^k} \frac{-e_{c_j}^k}{\frac{1}{2}\|y^j - x^k\|^2}, 0 \right\} + \iota,$$

which is similar to the strategy in [18]. And in [18, Lem 3] and [55, Lem 4.2], for the exact oracle (i.e., $\bar{\sigma} = 0$ and $\bar{\varepsilon} = 0$) it has proven that the sequence $\{\mu_k\}$ is boundedness. Yet the boundedness of $\{\mu_k\}$ would be difficult to prove without additional coercive assumptions on the behavior of the errors in our setting. It is worth noting that the penalty parameters s_k and t_k is updated, which depends on a constant ϖ defining in (5.1). Considering the basic assumptions on f, c and $\mathcal{X}$, we can obtain that the values of both functions on the set $\mathcal{X}$ are bounded. Thus the constant ϖ is bounded, and as a result s_k and t_k are well-defined in this work.

6.2 Examples for nonconvex optimization problems

In this subsection, we first introduce the nonconvex test problems. We prefer a series of polynomial functions developed in [11]; see also [12,18]. For each $i = 1, 2, \ldots, n$, the function $h_i : R^n \to R$ is defined by

$$h_i(x) = \sum_{j=1}^{n} x_j + (i x_i^2 - 2x_i), \tag{6.2}$$

where M is a fixed constant. There are five classes of test functions defined by h_i in [12] as objective functions

$$f_1(x) := \sum_{i=1}^{n} |h_i(x)|, \tag{6.3a}$$

$$f_2(x) := \max_{i=1,\ldots,n} |h_i(x)|, \tag{6.3b}$$

$$f_3(x) := \sum_{i=1}^{n} |h_i(x)| + \frac{1}{2}|x|^2, \tag{6.3c}$$

$$f_4(x) := \sum_{i=1}^{n} |h_i(x)| + \frac{1}{2}|x|, \tag{6.3d}$$

It has been proved in [12,18] that they are nonconvex, globally lower-C^1, bounded on compact $\mathcal{X}$, and level coercive. We can obtain that $0 = \min_x f_i$ and $\{0\} \subseteq \arg\min_x f_i$ for $i = 1, 2, 3, 4$. Thus we can define the compact $\mathcal{X} := B_{15}(0)$.

For constraint functions, we consider the pointwise maximum of a finite collection of quadratic functions developed in [17], i.e.,

$$c(x) := \max_{i=1,2,\ldots,n} \{\langle x, A_i x\rangle + \langle B_i, x\rangle + C_i\}, \tag{6.4}$$

where A_i are $n \times n$ matrices, $B_i \in \mathbf{R}^n$, and $C_i \in \mathbf{R}$ for $i = 1, 2, \ldots, n$. Here all the coefficients A_i, B_i, and C_i are uniformly distributed in $[-5, 5]$, which are chosen randomly by matlab code. The functions as (6.4) have many important practical advantages. Firstly, they are always lower-$\mathcal{C}^2$ (semiconvex), prox-bounded, and prox-regular, but may be nonconvex since A_i are not necessary to be positive definite. Secondly, the sequence of the convexification parameter $\{\mu_k\}$ for $c(\cdot)$ is fixed as

$$\mu = \max\left\{|A_i + A_i^{\mathrm{T}}| : i = 1, 2, \ldots n\right\}.$$

As a result, the large enough convexification parameters μ for $c(\cdot)$ can be estimated in advance. Thirdly, many different examples are easily obtained just by randomly generating A_i, b_i and c_i and choosing values for n and N. Lastly, the oracles (2.11) and (2.12) are easy to obtain the inexact information of the constrained functions.

We considered the following test problems

$$\begin{array}{ll} \min\limits_{x \in \mathbf{R}^n} & f_i(x) \\ s.t. & c(x) \le 0, \quad x \in \mathcal{X}, \end{array}$$

for $i = 1, \ldots, 4$ and $n = 5, \ldots, 15, 20$. For each test, it is well known that the optimum point is $x^* = (0, \ldots, 0) \in \mathbf{R}^n$. Hence, each f_i is clearly bounded below by 0. To check the precision, we will report the objective function value in last serious point. The remaining parameters of all of these tests are the same and listed below:

- stopping tolerance $tol_{stop} = 1.0\mathrm{e}-06$, initial starting point $x_0 = (1, \ldots, 1)$;
- prox-parameter $\eta_0 = 5$, convexification parameter $\mu_0 = 5$;
- Armijo-like parameter $m = 0.55$, positive constant $\iota = 2$;
- penalty parameters $s_0 = 15$ and $t_0 = 0$, stopping tolerance $\mathrm{tol}_{stop} = 10^{-6}$;
- initial variable prox-metric $A_0 = I$ ($n \times n$ identity matrix).

6.3 Comparison with penalty proximal bundle method

In this subsection, we examine the performances of Algorithm 3.1 with inexact oracles. At each evaluation, we use inexact function values and subgradients as $f(y^j) - \sigma_j$, $g_f^j = \tilde{g}_f^j + \varepsilon_j$ and $c(y^j) - \sigma_j$, $g_c^j = \tilde{g}_c^j + \varepsilon_j$, where $\tilde{g}_f^j \in \partial f(y^j)$ and $\tilde{g}_c^j \in \partial c(y^j)$. To provide a brief comparison of Algorithm 3.1 to other research, we compared our results with those obtained by the penalty proximal bundle method (PPBM) in [55].

Our results for deterministic tests are summarized in following tables, in which n denotes dimension. The following notations are used as

`Time` −−the CPU time(sec.)	x^* −−the optimal point,
f_{final} −−the final objective value,	c_{final} −−the final constrained value,
`Ni` −−the number of iterations,	`Ki` −−the number of serious iterations.

Table 1 Comparison between Algorithm 3.1 and PPBM for f_1 with $n = 5, \dots 10$

n	Alg.	x^*	f_{final}	c_{final}	Ki	Ni	Time
5	Algorithm 3.1	1.0e−04 (−0.3254, 0.3775, −0.1408, −0.0421, −0.5506)	2.88e−004	-4.9981	16	23	1.1040
	PPBM	1.0e−004 (0.09, −0.17, 0.09, 0.40, −0.20)	1.83e−004	-5.0012	–	22	1.3703
6	Algorithm 3.1	(−0.0001, 0.0000, −0.0001, 0.0000 0.0001, −0.0000)	0.0001	-11.0000	18	29	1.0271
	PPBM	(−0.0165, −0.0099, −0.0091, −0.0081, 0.0176, 0.0088)	0.1060	-10.0756	–	20	1.4273
7	Algorithm 3.1	1.0e−06 (−0.6163, −0.0445, −0.2178, 0.4750, −0.0514, −0.2268, 0.4710)	4.03e−006	-1.0000	21	33	2.4432
	PPBM	(0.0020, 0.0025, −0.0188, 0.0101, 0.0024, 0.0024, 0.0024)	0.0713	-1.3740	–	38	3.2513
8	Algorithm 3.1	1.0e−04 (0.1636, −0.1932, −0.1637, 0.0126, −0.3128, 0.0874, 0.2132, 0.0102)	2.68e−04	-53.9991	24	32	1.9312
	PPBM	(0.0219, 0.0221, 0.0407, 0.0227, 0.0229, 0.0068, 0.0026, −0.0014)	0.2712	-55.0488	–	36	2.4381
9	Algorithm 3.1	1.0e−03 (−0.1036, −0.0289, 0.0083, −0.0563, −0.0079, 0.1961, −0.1785, 0.0436, 0.0894)	0.0014	-6.9989	32	41	1.3706
	PPBM	(−0.1165, 0.0286, 0.0290, 0.0295, −0.0208, 0.0305, 0.0202, 0.0318, 0.0223)	0.4327	-10.3862	–	39	2.3319
10	Algorithm 3.1	1.0e−03 (−0.0598, −0.7086, −0.1750, 0.25, 0.1534, 0.0865, 0.2471, 0.0470, −0.1707, 0.0044)	0.0049	−16.9802	27	36	1.7860
	PPBM	(−0.0003, 0.0116, −0.0015, −0.0017, −0.0017, −0.0031, −0.0017, −0.0017, −0.0017, −0.0017)	0.0321	-16.7810	–	38	2.4939

Our numerical results for general nonconvex examples are reported in Tables 1, 2, 3, 4, 5, 6, 7 and 8, which show a reasonable performance of Algorithm 3.1.

We analyze the results in more detail as follows:

- For objective function f_1, we tested 12 nonconcex examples with $n = 5, \dots, 15, 20$. We use constant noise $\sigma_j = 0.01$ and $\varepsilon_j = 0.01 * \texttt{ones}(n, 1)$ for all j. Tables 1 and 2 show that values of f_{final} obtained by Algorithm 3.1 are two orders of magnitude smaller than PPBM in most cases. Thus, Algorithm 3.1 succeeds in obtaining a reasonably high accuracy, while spend less time in all cases. Besides, the percentage of serious steps (Ki) in total iterations (Ni) is above 90% in many cases, which means that most iterations are reliable and Algorithm 3.1 is effective.
- To introduce errors in the available information, we use random noises $\sigma_j = 0.1 * \texttt{random}(\text{'}Normal\text{'}, 0, 0.1)$ and $\varepsilon_j = 0.1 * \texttt{random}(\text{'}Normal\text{'}, 0, 0.1, n, 1)$, which generates random numbers from the normal distribution with mean 0 and standard deviation 0.1, and scalars n and 1 are the row and column dimensions. Tables 3 and 4 show the results of Algorithm 3.1 as compared to PPBM for the objective function f_2 with

Table 2 Comparison between Algorithm 3.1 and PPBM for objective function f_1 with $n = 11, \dots, 15, 20$

n	Alg.	f_{final}	c_{final}	Ki	Ni	Time
11	Algorithm 3.1	0.0051	−16.9514	39	63	0.9206
	PPBM	0.0204	−17.9861	–	54	1.7934
12	Algorithm 3.1	0.0018	−17.0068	49	61	1.1334
	PPBM	0.0079	−11.1753	–	74	4.0303
13	Algorithm 3.1	2.4845e−004	−17.0000	49	70	1.1524
	PPBM	0.0037	−14.0561	–	96	5.3781
14	Algorithm 3.1	2.0912e−004	−16.9997	63	82	2.0177
	PPBM	0.0028	−17.0030	–	104	4.0706
15	Algorithm 3.1	1.9762e−004	−16.9999	51	72	1.5128
	PPBM	0.0711	−14.0310	–	132	4.7832
20	Algorithm 3.1	1.8412e−004	−16.9992	55	80	1.8021
	PPBM	0.0003	−17.1310	–	179	7.8291

Table 3 Comparison between Algorithm 3.1 and PPBM for f_2 with $n = 5, \dots, 10$

n	Alg.	x^*	f_{final}	c_{final}	Ki	Ni	Time
5	Algorithm 3.1	1.0e−06 (−0.0680, −0.0680, −0.0728, 0.1409, −0.0712)	4.20e−007	−35.0000	37	37	1.1875
	PPBM	1.0e−06 (−0.0684, 0.0198, 0.0214, 0.0225, −0.3134)	3.63e−07	−35.0000	55	55	4.5731
6	Algorithm 3.1	1.0e−06 (−0.1243, 0.3156, −0.0936, −0.0796, −0.1176, −0.0961)	8.26e−07	−11.0000	21	25	1.4540
	PPBM	(0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,)	1.20e−05	−11.0006	–	24	1.5329
7	Algorithm 3.1	1.0e−04 (0.0357, 0.0348, 0.2194, −0.1765, −0.1295, 0.0302, −0.0472)	4.71e−05	−1.0600	23	31	1.4573
	PPBM	1.0e−05 (3.30, 3.30, 3.30, 3.30, 3.30, 3.30, 3.30)	1.67e−04	−1.0076	–	30	4.1749
8	Algorithm 3.1	1.0e−06 (0.0988, 0.1196, 0.1308, 0.0588, 0.1333, −0.0459, −0.3481, 0.1167)	9.60e−07	−54.0000	31	39	2.01268
	PPBM	1.0e−4 (1.16, 1.16, 1.16, 1.16, 1.16, 1.16, 1.16, 1.16)	6.99e−04	−54.0273	–	31	2.3108
9	Algorithm 3.1	1.0e−06 (−0.0423, −0.0258, −0.0369, 0.4380, −0.1142, −0.1310, −0.0820, −0.0416, −0.0416)	9.53e−007	−7.0000	29	39	3.1590
	PPBM	1.0e−04 (4.27, 4.28, 4.28, 4.28, 4.28, 4.28, 4.28, 4.28, 4.28)	0.0030	−7.1867	–	38	4.9357
10	Algorithm 3.1	1.0e−05 (0.1017, −0.6883, 0.0995, 0.0238, 0.1000, 0.0739, 0.2273, 0.0975, 0.0893, 0.0523)	1.5536e−005	−17.0001	41	52	5.8139
	PPBM	1.0e−04 (3.30, 3.30, 3.30, 3.30, 3.30, 3.30, 3.30, 3.30, 3.30, 3.30,)	0.0026	−17.2078	–	36	6.0814

Table 4 Comparison between Algorithm 3.1 and PPBM for f_2 with $n = 11, \ldots, 15, 20$

n	Alg.	f_{final}	c_{final}	Ki	Ni	Time
11	Algorithm 3.1	4.29e−005	−17.0004	107	119	8.6199
	PPBM	0.0141	−17.1286	–	134	13.4316
12	Algorithm 3.1	3.84e−006	−10.3833	107	118	7.9933
	PPBM	0.0073	−16.9998	–	162	25.6711
13	Algorithm 3.1	0.0022	−17.0316	107	115	9.0743
	PPBM	0.0139	−17.0075	–	142	14.6779
14	Algorithm 3.1	8.03e−004	−8.1707	107	113	8.9709
	PPBM	0.0375	−18.2312	–	133	14.4552
15	Algorithm 3.1	0.0071	−17.0847	134	142	16.0570
	PPBM	0.0177	−17.4762	–	150	19.6917
20	Algorithm 3.1	0.0060	−7.2063	113	125	9.1701
	PPBM	0.0843	−17.5780	–	169	28.3983

$n = 5, \ldots, 15, 20$. We observe that Algorithm 3.1 only needs one-half of the cpu time of PPBM. In particular, if $n = 20$, the value of `time` of our method is just one-third of PPBM. We also see that Algorithm 3.1 always produces better (lower) objective values than PPMB for all examples.

- For the objective function f_3, we also tested 12 nonconvex examples with $n = 5, \ldots, 15, 20$ from randomly generated initial points. For these nonconvex examples, we introduce random noises $\sigma_j = 0.1 * \texttt{unifrnd}(0, 1)$ and $\varepsilon_j = 0.1 * \texttt{unifrnd}(0, 1, n, 1)$. The matlab code $\texttt{unifrnd}(0, 1, n, 1)$ returns an $n-$dimensional column vector, and all elements are generated from the same distribution in the interval $[0, 1]$. From Tables 5 and 6, it observes that Algorithm 3.1 always produces better (lower) objective values than PPMB, while spend less time in most cases. Thus, Algorithm 3.1 performs better than PPBM for the objective function f_3.
- For the objective function f_4, we also tested 12 nonconvex examples with $n = 5, \ldots, 15, 20$ from a fixed initial point $x^0 = ones(n, 1)$. For these nonconvex examples, we introduce random noises $\sigma_j = 0.1 * \texttt{normrnd}(0, 0.2)$ and $\varepsilon_j = 0.1 * \texttt{normrnd}(0, 0.2, n, 1)$. The matlab code $\texttt{normrnd}(0, 0.2, n, 1)$ generates random numbers from the normal distribution with mean 0 and standard deviation 0.2, where scalars n and 1 are the row and column dimensions. From Tables 7 and 8, it seems that our method succeeds in obtaining better objective values, at the price of less time than PPBM. In most of cases, the number of serious steps (`Ki`) is much more than the number of null steps, which implies that our inexact method is high efficiency.

6.4 Impact of noise on solution accuracy

To analyse the errors in the available information, we test five different types of inexact oracle:

- `NN` (No noise): $\sigma_j = \bar{\sigma} = 0$ and $\varepsilon_j = \bar{\varepsilon} = 0$, for all j,
- `CN` (Constant noise): $\bar{\sigma} = \sigma_j = 0.01$ and $\varepsilon_j = \bar{\varepsilon} = 0.01$, for all j,
- `VN` (Vanishing noise): $\bar{\sigma} = 0.01$, $\sigma_j = \min\{0.01, \frac{\|y^j\|}{100}\}$, $\varepsilon_j = \min\{0.01, \frac{\|y^j\|}{100}\}$, for all j,

Table 5 Comparison between Algorithm 3.1 and PPBM for objective function f_3 with $n = 5, \ldots, 10$

n	Alg.	x^*	f_{final}	c_{final}	Ki	Ni	Time
5	Algorithm 3.1	1.0e−07 (0.5703, −0.1791, −0.2646, 0.0749, 0.3721)	3.1958e−007	−35.0000	37	37	0.6183
	PPBM	1.0e−07 (0.5703, −0.1791, −0.2646, 0.0749, 0.3721)	2.4010e−07	−35.0000	39	39	0.7031
6	Algorithm 3.1	1.0e−07 (−0.1302, −0.0660, −0.0649, 0.1863, 0.1840, −0.1119)	1.48e−007	−11.0000	27	32	0.6342
	PPBM	1.0e−05 (0.19, −0.58, 0.19, 0.19, 0.19, 0.19)	1.56e−05	−11.0000	–	35	1.2176
7	Algorithm 3.1	1.0e−06 (−0.0558, −0.6342, 0.0814, −0.9637, 0.6600, −0.3018, 0.4242)	7.18e−006	−1.0000	31	49	0.5457
	PPBM	(−0.0033, −0.0033, 0.0132, −0.0033, −0.0033, −0.0033, −0.0033)	0.0328	−0.6933	–	33	1.3189
8	Algorithm 3.1	1.0e−04 (0.1681, −0.2523, −0.2771, −0.4971, 0.5511, −0.6311, 0.5777, 0.0761)	6.06e−004	−177	34	44	0.8901
	PPBM	(−0.0024, 0.0004, −0.0110, 0.0104, 0.0005, 0.0004, 0.0021, 0.0004)	0.0520	−53.9515	–	48	5.8792
9	Algorithm 3.1	1.0e−05 (0.0766, 0.0606, −0.1501, 0.0269, −0.2047, −0.0171, −0.0621, 0.1573, −0.0162)	1.81e−005	−7.0000	31	39	1.4310
	PPBM	(0.0219, −0.0110, −0.0029, −0.0029, −0.0029, −0.0029, −0.0029, −0.0005, −0.0017)	0.0732	−6.3529	–	48	2.3984
10	Algorithm 3.1	1.0e−05 (−0.1146, −0.0321, −0.1081, −0.0787, 0.2241, 0.3012, 0.1565, 0.0618, −0.2185, −0.0004)	3.10e−005	−17.0003	39	50	0.8654
	PPBM	1.0e−04 (1.03, −0.067, −0.402, 0.045, 0.045, −0.513, −0.402, −0.401, 0.045, 0.045)	6.87e−04	−16.9944	–	58	4.3127

- CGN (Constant Gradient noise): $\bar{\sigma} = \sigma_j = 0$ and $\varepsilon_j = \bar{\varepsilon} = 0.05$, for all j,
- VGN (Vanishing Gradient noise): $\bar{\sigma} = \sigma_j = 0$ and $\varepsilon_j = \min\{0.01, \frac{\|y^j\|}{100}\}$, for all j.

Considering the optimum value of all the test problems is zero, we can use the formula Precision= $|\log_{10}(f_{final})|$ to check the performance of the various methods. Figures 1 and 2 report the average performance of Algorithm 3.1 when noise is introduced (the results are averaged across all 10 runs). To better reveal the influence of noise, we also report the results of NN (no noise). Figure 1 reports the results for constant noise (CN and CGN) with $\texttt{tol} = 10^{-6}$. We can observe that Algorithm 3.1 can achieve an acceptable accuracy for the type CN. The results of CGN are better than those of CN, although still notably deviations than when exact calculations are available. Figure 2 reports the precision results of Algorithm 3.1 for vanishing noise (VN and VGN). The vanishing noise results in reducing the accuracy for $\texttt{tol} = 10^{-6}$, generally better than in the constant noise cases.

Table 6 Comparison between Algorithm 3.1 and PPBM for f_3 with $n = 11, \dots, 15, 20$

n	Alg.	f_{final}	c_{final}	Ki	Ni	Time
11	Algorithm 3.1	1.50e−004	−16.9995	49	68	1.1577
	PPBM	2.16e−004	−16.9995	–	72	1.3178
12	Algorithm 3.1	5.13e−006	−17.0000	55	71	1.5879
	PPBM	5.70e−007	−17.0000	–	83	1.6647
13	Algorithm 3.1	2.52e−004	−16.9988	53	76	1.3958
	PPBM	6.15e−004	−17.0039	–	91	2.7469
14	Algorithm 3.1	2.10e−004	−17.0002	59	80	1.5253
	PPBM	2.01e−004	−16.9997	–	74	2.3840
15	Algorithm 3.1	0.0021	−17.0003	46	66	0.9866
	PPBM	0.0430	−21.8734	–	64	1.1668
20	Algorithm 3.1	0.0047	−17.0346	79	107	5.0972
	PPBM	0.0537	−19.8410	–	123	11.8231

Table 7 Comparison between Algorithm 3.1 and PPBM for objective function f_4 with $n = 5, \dots, 10$

n	Alg.	x^*	f_{final}	c_{final}	Ki	Ni	Time
5	Algorithm 3.1	1.0e−03 (0.0972, −0.3890, −0.1623, 0.3057, −0.2618)	0.0028	−34.9938	26	27	0.2066
	PPBM	(0.0010, 0.0002, −0.0012, 0.0001, −0.0001)	0.0067	−34.97004	–	30	0.6941
6	Algorithm 3.1	(0.0011, −0.0003, −0.0000, −0.0004, −0.0006, −0.0004)	0.0056	−10.9829	25	34	0.3667
	PPBM	(0.0027, −0.0011, 0.0004, −0.0022, −0.0006, −0.0027)	0.0244	−10.8834	–	29	0.3413
7	Algorithm 3.1	1.0e−03 (−0.2013, −0.0510, 0.0403, 0.0701, 0.3010, 0.0610, −0.0804)	0.0020	−1.0024	39	46	1.7304
	PPBM	1.0e−03 (−0.21, −0.05, 0.04, 0.07, 0.30, 0.06, −0.08)	0.0020	−1.0020	–	77	5.9351
8	Algorithm 3.1	(0.0003, −0.0000, 0.0000, −0.0005, 0.0012, 0.0000, −0.0002, −0.0009)	0.0080	−53.9897	31	39	0.4567
	PPBM	(−0.0005, 0.0005, −0.0013, −0.0005, 0.0010, −0.0007, 0.0004, 0.0001)	0.0129	−53.9849	–	40	1.2455
9	Algorithm 3.1	1.0e−03 (−0.2814, −0.4280, 0.4075, −0.3852, −0.3361, 0.4444, 0.0724, −0.6250, 0.2739)	0.0084	−6.9478	32	56	1.4601
	PPBM	(0.0032, −0.0004, 0.0003, −0.0003, −0.0010, −0.0003, −0.0018, 0.0006, −0.0007)	0.0207	−6.9562	–	49	1.9936
10	Algorithm 3.1	(−0.0003, 0.0001, −0.0001, −0.0005, −0.0016, 0.0006, 0.0023, −0.0008, 0.0006, −0.0005)	0.0181	−16.9381	18	37	1.1475
	PPBM	(0.0004, −0.0001, −0.0005, −0.0006, −0.0004, −0.0004, 0.0005, 0.0010, −0.0015, 0.0008)	0.0155	−16.9304	–	88	7.2973

Table 8 Comparison between Algorithm 3.1 and PPBM for f_4 with $n = 11, \dots, 15, 20$

n	Alg.	f_{final}	c_{final}	Ki	Ni	Time
11	Algorithm 3.1	0.0091	−16.9981	40	57	1.6717
	PPBM	0.0117	−16.9787	–	81	11.2496
12	Algorithm 3.1	0.0122	−17.0241	29	571	1.3205
	PPBM	0.0124	−16.9781	–	79	1.6228
13	Algorithm 3.1	0.0104	−17.0109	28	52	0.3696
	PPBM	0.0503	−17.0037	–	99	2.4906
14	Algorithm 3.1	0.0131	−17.0043	33	54	1.4138
	PPBM	0.0593	−16.9997	–	74	2.3840
15	Algorithm 3.1	0.0014	−17.0409	49	76	1.0356
	PPBM	0.0463	−17.0313	–	84	3.7320
20	Algorithm 3.1	0.0169	−17.0802	37	66	1.7420
	PPBM	0.0778	−17.0141	–	119	5.8107

6.5 Comparison with FSQP-GS and SLQP-GS

In this subsection, we aim to compare the practical effectiveness of Algorithm 3.1 with a sequential quadratic programming algorithm (SLQP-GS) [4] and a feasible SQP-GS algorithm (FSQP-GS) [52]. We test the following two examples, which are taken from [4,45] respectively. In all numerical experiments, we choose the same initial points as FSQP-GS in [52] for all examples in this subsection.

To analyse the errors in the available information, we consider four different types of inexact oracle:

- Constant noise (CN): $\sigma_j = 0.01$ and $\varepsilon_j = 0.01 * \mathtt{ones}(n, 1)$ for all j;
- $\sigma_j = 0.1 * \mathtt{random}(\text{‘}Normal\text{’}, 0, 0.1)$ and $\varepsilon_j = 0.1 * \mathtt{random}(\text{‘}Normal\text{’}, 0, 0.1, n, 1)$ for all j;
- $\sigma_j = 0.1 * \mathtt{unifrnd}(0, 1)$ and $\varepsilon_j = 0.1 * \mathtt{unifrnd}(0, 1, n, 1)$ for all j;
- $\sigma_j = 0.1 * \mathtt{normrnd}(0, 0.2)$ and $\varepsilon_j = 0.1 * \mathtt{normrnd}(0, 0.2, n, 1)$ for all j.

Example 1 Constrained nonsmooth Rosenbrock problem [4]:

$$\begin{aligned} \min_x \quad & 8|x_1^2 - x_2| + (1 - x_1)^2 \\ s.t. \quad & \max\{\sqrt{2}x_1, 2x_2\} \le 1 \end{aligned}$$

The solution of this problem is $x^* = (\frac{\sqrt{2}}{2}, \frac{1}{2})$ at which both the objective function and the constraint function are nondifferentiable. In this numerical experiment, we set the initial points $(0.066661, -0.350366)^{\mathrm{T}}$ and $(0.433746, -1.447530)^{\mathrm{T}}$ respectively.

Example 2 Rosen–Suzuki problem [45]:

$$\begin{aligned} \min_x \quad & f(x) = \max\{f_i(x): i = 1, \dots, 4\} \\ s.t. \quad & \max\{c_i(x): i = 1, 2, 3\} \le 0, \end{aligned}$$

where $f_1(x) = x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4$, $f_2(x) = f_1(x) + 10c_1(x)$, $f_3(x) = f_1(x) + 10c_2(x)$, $f_4(x) = f_1(x) + 10c_3(x)$, $c_1(x) = x_1^2 + x_2^2 + x_3^2 +$

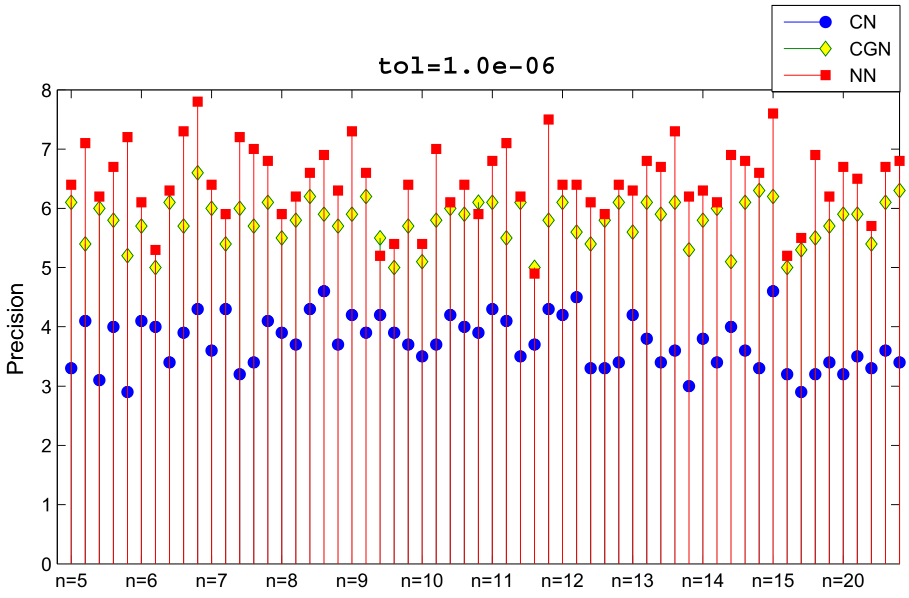

Fig. 1 Precision for Algorithm 3.1 for noise forms `NN`, `CN` and `CGN` with `tol` $= 10^{-6}$

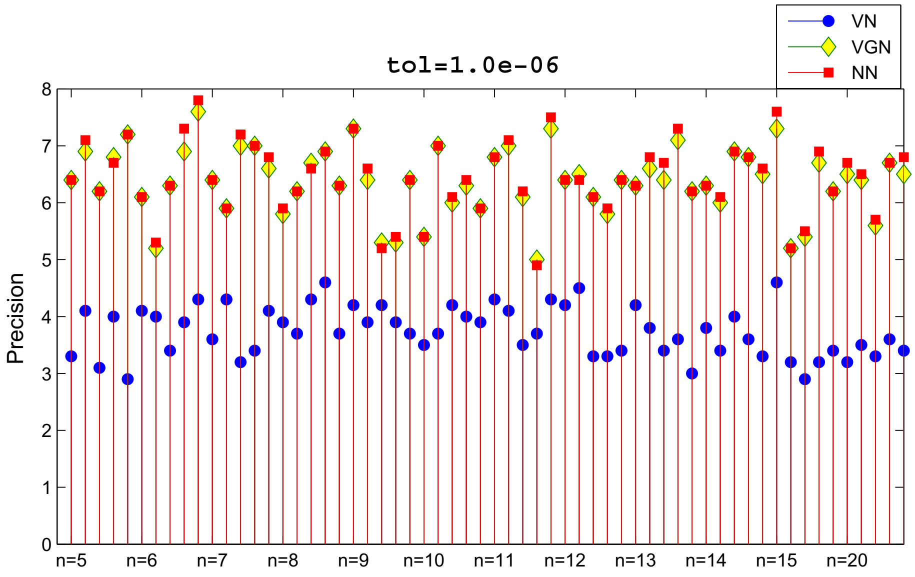

Fig. 2 Precision for Algorithm 3.1 for noise forms `NN`, `VN` and `VGN` with `tol` $= 10^{-6}$

$x_4^2 + x_1 - x_2 + x_3 - x_4 - 8,\ c_2(x) = x_1^2 + 2x_2^2 + x_3^2 + 2x_4^2 - x_1 - x_4 - 10,\ c_3(x) = x_1^2 + x_2^2 + x_3^2 + 2x_1 - x_2 - x_4 - 5$. In our numerical experiment, we set the initial points $(1, 1, 1, 1)^T$ and $(0, 0, 0, 0)^T$ respectively.

From Tables 9 and 10, it observes that Algorithm 3.1 (`CN`) needs a fewer number of iterations than FSQP-GS and SLQP-GS for all examples. Tables 9 and 10 also show that

Table 9 Comparison with FSQP-GS and SLQP-GS for Examples 1

x^0	`Alg.`	f_{final}	`Ki`	`Ni`	`Time`
$(0.0666, -0.3504)^T$	Algorithm 3.1(`CN`)	0.0857850	56	64	1.8
	Algorithm 3.1(`rand`)	0.0857863	63	71	2.0
	Algorithm 3.1(`unifrnd`)	0.0857982	59	73	2.4
	Algorithm 3.1(`normrnd`)	0.0861583	53	68	3.7
	SLQP-GS	0.0857869	–	69	3.0
	FSQP-GS	0.0857870	–	65	2.4
$(0.4337, -1.4475)^T$	Algorithm 3.1(`CN`)	0.0857851	46	57	1.6
	Algorithm 3.1(`rand`)	0.0857993	52	63	1.9
	Algorithm 3.1(`unifrnd`)	0.0857824	50	59	2.1
	Algorithm 3.1(`normrnd`)	0.0857863	38	61	2.8
	SLQP-GS	0.0857864	–	59	2.1
	FSQP-GS	0.0857910	–	68	2.2

Table 10 Comparison with FSQP-GS and SLQP-GS for Examples 2

x^0	`Alg.`	f_{final}	`Ki`	`Ni`	`Time`
$(1, 1, 1, 1)^T$	Algorithm 3.1(`CN`)	43.99800	39	47	3.405
	Algorithm 3.1(`rand`)	43.99840	43	52	3.817
	Algorithm 3.1(`unifrnd`)	44.00027	37	49	3.701
	Algorithm 3.1(`normrnd`)	43.99910	46	55	4.504
	SLQP-GS	44.00000	–	43	4.619
	FSQP-GS	43.99800	–	51	4.011
$(0, 0, 0, 0)^T$	Algorithm 3.1(`CN`)	43.99820	49	54	3.346
	Algorithm 3.1(`rand`)	43.99900	52	67	3.716
	Algorithm 3.1(`unifrnd`)	44.00024	59	71	3.970
	Algorithm 3.1(`normrnd`)	43.99902	50	64	3.281
	SLQP-GS	44.00000	–	51	4.093
	FSQP-GS	43.99905	–	57	3.887

Algorithm 3.1 (`CN`) always obtains similar objective values with SLQP-GS and FSQP-GS, while spends less time for most cases. In general, Algorithm 3.1(`CN`) performs slightly better than FSQP-GS and SLQP-GS for all examples. Besides, we also consider random noises, which are generated randomly by using matlab codes `random`, `unifrnd` and `normrnd`. From Tables 9 and 10, it observes that Algorithm 3.1(`rand`) and Algorithm 3.1(`unifrnd`) are comparable for FSQP-GS and SLQP-GS for most cases. Algorithm 3.1(`normrnd`) spends much time for some cases, than FSQP-GS and SLQP-GS.

6.6 Results for semi-infinite programming problems

For benchmarking purposes we will investigate the computational behaviour and convergence results for SIP problems. We will compare the performance of Algorithm 3.1 by using

Constant noise with the central cutting plane algorithm (CCPA) [34] and the SIP solver *fseminf* in MATLAB toolbox. The SIP problem has the following abstract structure

$$\begin{aligned} &\min_{x} \; f(x), \\ &s.t. \;\; g(x, v) \le 0, \;\text{ for all } v \in V, \end{aligned} \tag{6.5}$$

where $f(x)\colon \mathbf{R}^n \to \mathbf{R}$ is locally Lipschitz and not necessary differentiable; $g\colon \mathbf{R}^n \times V \to \mathbf{R}$ is twice continuously differentiable; and V is a nonempty compact subset of $\mathbf{R}^n$.

We first consider the so-called lower level problem

$$\min_{v \in V} -g(\bar{x}, v), \tag{6.6}$$

to obtain the global solution. The difficulty lies in the fact that $-g(\bar{x}, v_{glob})$ is the globally optimal value of (6.6) which might be hard to obtain numerically. In fact, most of standard nonlinear problem solvers can only be expected to output a local minimizer v_{loc}. Many works aim at structuring a sequence of convexifications of the lower level problem by using the technologies in [13,50] to solve the auxiliary problems with convex lower levels. In fact, the function c needs only an inexact value with a given fixed accuracy ω. That is we only want to obtain an approximate solution $v(\bar{x})$ such that $c(\bar{x}, v(\bar{x})) \ge c(\bar{x}) - \omega$. The iterative rules in [14,19] can generate more and more accurate solutions to (6.6), until a ω-optimal solution.

The numerical results are listed in Table 11, where the following notations are used as

x^*: the optimal point,	f^*: the final objective value,
Time(s): the CPU time of Algorithm 3.1,	Time$_{\max}(s)$: the CPU time for (6.6),
Iter: the number of iterations of Algorithm 3.1,	

Example 3 Finding the Chebyshev approximation of the function, and has been tested by Floudas and Stein [13, Example 5.1]:

$$\begin{aligned} \min \quad & f(x) = x_4, \\ s.t. \quad & g_1(x, y) = sin(\pi y) - x_3 y^2 - x_2 y - x_1 - x_4 \le 0, \\ & g_2(x, y) = -sin(\pi y) + x_3 y^2 + x_2 y + x_1 - x_4 \le 0, \\ & \text{for all } Y = [0, 1]. \end{aligned}$$

Example 4 The following SIP problem is tested by Kortanek and No [34], and stemmed from Tichatschke and Nebeling [53]:

$$\begin{aligned} \min \quad & f(x) = (x_1 - 2)^2 + (x_2 - 0.2)^2, \\ s.t. \quad & g(x, y) = (5sin(\pi\sqrt{y})/(1 + y^2))x_1^2 - x_2 \le 0, \\ & \text{for all } Y = [0, 1]. \end{aligned}$$

Example 5 The problem is discussed by Goberna and Löpez [16], and Zhang and Wu [56]:

$$\begin{aligned} \min \quad & f(x) = x_1^2 + x_2^2, \\ s.t. \quad & g(x, y) = \cos(y)x_1 + \sin(y)x_2 - (1 + \cos(y) + \sin(y)) \le 0, \\ & \text{for all } Y = [\pi, \tfrac{3}{2}\pi], \end{aligned}$$

Although the feasible set is not bounded, the objective function is level bounded on the feasible set.

Example 6 Consider the following problem, which was tested by Kortanek and No [34]:

$$\begin{aligned} \min \quad & f(x) = x_1^2 + x_2^2 + x_2^3, \\ s.t. \quad & g(x, y) = x_1 + x_2 \exp(x_3 y) + \exp(2y) - 2\sin(4y) \le 0, \\ & \text{for all } Y = [0, 1]. \end{aligned}$$

Table 11 Comparison of results for Examples 3–7

Example	Algorithm	x^*	f^*	Time	Iter	Time$_{max}$	Iter$_{max}$
3	Algorithm 3.1 (CN)	(0.0275, 4.0000, −4.0000, 0.0274)	0.0274	11.14	23	2.06	15
	CCPA	(0.0275, 4.0000, −4.0000, 0.0274)	0.0274	25.12	17	–	–
	fseminf	(0.0283, 4.0000, −3.7010, 0.0289)	0.0289	25.51	23	–	–
4	Algorithm 3.1 (CN)	(0.2055, 0.2000)	3.2202	1.20	21	0.92	7
	CCPA	(0.2052, 0.2000)	3.2211	1.75	47	–	–
	fseminf	(0.2052, 0.2000)	3.2211	2.09	16	–	–
5	Algorithm 3.1 (CN)	(0.2052, 0.2000)	0.0821	2.96	18	0.43	8
	CCPA	(0.2052, 0.2000)	0.0821	7.32	32	–	–
	fseminf	(0.3012, 0.3101)	0.1868	1.98	53	–	–
6	Algorithm 3.1 (CN)	(−0.2131, −1.3611, −1.8530)	5.3317	1.41	21	0.54	7
	CCPA	(−0.2133, −1.3615, −1.8535)	5.3346	2.71	27	–	–
	fseminf	(−0.2133, −1.3615, −1.8535)	5.3346	3.17	46	–	–
7	Algorithm 3.1 (CN)	(0.5858, 0.5858)	0.6863	0.24	15	0.21	5
	CCPA	(0.5858, 0.5858)	0.6863	0.30	9	–	–
	fseminf	(0.5858, 0.5858)	0.6863	17.01	107	–	–

Example 7 The following convex SIP problem has been tested by Kortanek and No [34]:

$$\begin{array}{ll} \min & f(x) = x_1^2 + x_2^2, \\ s.t. & c(x, y) = (x_1^2 + x_2^2 - 4)y_1 + ((x_1 - 2)^2 + (x_2^2 - 2)^2 - 4)y_2 \leq 0, \\ & x_1 \in [0, 2],\ x_2 \in [0, 2],\ Y = [0, 1] \times [0, 1]. \end{array}$$

Observe that the problem satisfies the Slater CQ and the feasible set is bounded.

The numerical results of Examples 3–7 are reported in Table 11. One can observe in Table 11 that Algorithm 3.1 performs well and provides an optimal solution to each example. Besides, the inexact solution of subproblem (6.6) can be obtained effectively, which means the inexact oracle is reasonable. To obtain a similar optimal solution, the SIP solver fseminf and the CCPA took much time than our method. In our opinion, the performance of Algorithm 3.1 for solving Examples 3–7 is better than that of the CCPA and the SIP solver fseminf.

Acknowledgements The authors wish to thank Editor-in-Chief Sergiy Butenko, the preceding managing editors and the anonymous reviewers for their helpful comments on the earlier version of this paper, which considerably improved both the presentation and the numerical experiments. We also gratefully acknowledge the support of the Huzhou science and technology plan on No. 2016GY03 and Natural Science Foundation of China Grant 11626051.

References

1. Ackooij, W., Sagastizábal, C.: Constrained bundle methods for upper inexact oracles with application to joint chance constrained energy problems. SIAM J. Optim. **24**, 733–765 (2014)
2. Apkarian, P., Noll, D., Prot, O.: A proximity control algorithm to minimize nonsmooth and nonconvex semi-infinite maximum eigenvalue functions. J. Convex Anal. **16**, 641–666 (2009)
3. Borwein, J.M., Lewis, A.S.: Convex Analysis and Nonlinear Optimization: Theory and Examples, 2nd edn. Springer, Berlin (2006)

4. Curtis, F.E., Overton, M.L.: A sequential quadratic programming algorithm for nonconvex, nonsmooth constrained optimization. SIAM J. Optim. **22**, 474–500 (2012)
5. d'Antonio, G., Frangioni, A.: Convergence analysis of deflected conditional approximate subgradient methods. SIAM. J. Optim. **20**, 357–386 (2009)
6. de Oliveira, W., Sagastizábal, C., Scheimberg, S.: Inexact bundle methods for two-stage stochastic programming. SIAM J. Optim. **21**, 517–544 (2011)
7. de Oliveira, W., Sagastizábal, C., Lemaréchal, C.: Convex proximal bundle methods in depth: a unified analysis for inexact oracles. Math. Program. **148**, 241–277 (2014)
8. Daniilidis, A., Georgiev, P.: Approximate convexity and submonotonicity. J. Math. Anal. Appl. **291**, 292–301 (2004)
9. Emiel, G., Sagastizábal, C.: Incremental-like bundle methods with application to energy planning. Comput. Optim. Appl. **46**, 305–332 (2010)
10. Fábián, C., Szöke, Z.: Solving two-stage stochastic programming problems with level decomposition. Comput. Manag. Sci. **4**, 313–353 (2007)
11. Ferrier, C.: Bornes Duales de Problémes d'Optimisation Polynomiaux, Ph.D. thesis. Laboratoire Approximation et Optimisation, Université Paul Sabatier, Toulouse (1997)
12. Ferrier, C.: Computation of the distance to semi-algebraic sets. ESAIM Control Optim. Calc. Var. **5**, 139–156 (2000)
13. Floudas, C.A., Stein, O.: The adaptive convexification algorithm: a feasible point method for semi-infinite programming. SIAM J. Optim. **18**, 1187–1208 (2007)
14. Fuduli, A., Gaudioso, M., Giallombardo, G.: A DC piecewise affine model and a bundling technique in nonconvex nonsmooth optimization. Optim. Methods Softw. **19**, 89–102 (2004)
15. Fuduli, A., Gaudioso, M., Giallombardo, G.: Minimizing nonconvex nonsmooth functions via cutting planes and proximity control. SIAM J. Optim. **14**, 743–756 (2004)
16. Goberna, M.A., López, M.A.: Linear Semi-Infinite Optimization. Wiley, New-York (1998)
17. Hare, W., Sagastizábal, C.: Computing proximal points of nonconvex functions. Math. Program. **116**, 221–258 (2009)
18. Hare, W., Sagastizábal, C.: A redistributed proximal bundle method for nonconvex optimization. SIAM J. Optim. **20**, 2442–2473 (2010)
19. Hare, W., Sagastizábal, C., Solodov, M.: A proximal bundle method for nonconvex functions with inexact oracles. Comput. Optim. Appl. **63**, 1–28 (2016)
20. Hintermüller, M.: A proximal bundle method based on approximate subgradients. Comput. Optim. Appl. **20**, 245–266 (2001)
21. Hiriart-Urruty, J.-B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms. II, Volume 306 of Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer, Berlin (1993). Advanced theory and bundle methods
22. Jongen, H.T., Rückmann, J.-J., Stein, O.: Generalized semi-infinite optimization: a first order optimality condition and examples. Math. Program. **83**, 145–158 (1998)
23. Karas, E., Ribeiro, A., Sagastizàbal, C., Solodov, M.: A bundle-filter method for nonsmooth convex constrained optimization. Math. Program. **116**, 297–320 (2009)
24. Kiwiel, K.C.: Methods of Descent for Nondifferentiable Optimization, Lecture Notes in Mathematics. Springer, Berlin (1985)
25. Kiwiel, K.C.: An algorithm for nonsmooth convex minimization with errors. Math. Comput. **45**, 171–180 (1985)
26. Kiwiel, K.C.: A linearization algorithm for nonsmooth minimization. Math. Oper. Res. **10**, 185–194 (1985)
27. Kiwiel, K.C.: An exact penalty function algorithm for nonsmooth convex constrained minimization problems. IMA J. Numer. Anal. **5**, 111–119 (1985)
28. Kiwiel, K.C.: Exact penalty functions in proximal bundle methods for constrained convex nondifferentiable minimization. Math. Program. **52**, 285–302 (1991)
29. Kiwiel, K.C.: Restricted step and Levenberg–Marquardt techniques in proximal bundle methods for nonconvex nondifferentiable optimization. SIAM J. Optim. **6**, 227–249 (1996)
30. Kiwiel, K.C.: Convergence of approximate and incremental subgradient methods for convex optimization. SIAM. J. Optim. **14**, 807–840 (2004)
31. Kiwiel, K.C.: A proximal bundle method with approximate subgradient linearizations. SIAM. J. Optim. **16**, 1007–1023 (2006)
32. Kiwiel, K.C.: A method of centers with approximate subgradient linearizations for nonsmooth convex optimization. SIAM J. Optim. **18**, 1467–1489 (2008)
33. Kiwiel, K.C., Lemaréchal, C.: An inexact bundle variant suited to column generation. Math. Program. **118**, 177–206 (2009)

34. Kortanek, K.O., No, H.: A central cutting plane algorithm for convex semi-infinite programming problems. SIAM J. Optim. **3**, 901–918 (1993)
35. Lemaréchal, C., Nemirovskii, A., Nesterov, Y.: New variants of bundle methods. Math. Program. **69**, 111–147 (1995)
36. Lukšan, L., Vlček, J.: A bundle-Newton method for nonsmooth unconstrained minimization. Math. Program. **83**, 373–391 (1998)
37. Lv, J., Pang, L.P., Wang, J.H.: Special backtracking proximal bundle method for nonconvex maximum eigenvalue optimization. Appl. Math. Comput. **265**, 635–651 (2015)
38. Mifflin, R.: An algorithm for constrained optimization with semismooth functions. Math. Oper. Res. **2**, 191–207 (1977)
39. Mifflin, R.: A modification and extension of Lemarechal's algorithm for nonsmooth minimization. Math. Program. Stud. **17**, 77–90 (1982)
40. Mifflin, R.: A quasi-second-order proximal bundle algorithm. Math. Program. **73**, 51–72 (1996)
41. Nedić, A., Bertsekas, D.P.: The effect of deterministic noise in subgradient methods. Math. Program. **125**, 75–99 (2010)
42. Noll, D.: Bundle method for non-convex minimization with inexact subgradients and function values. Comput. Anal. Math. **50**, 555–592 (2013)
43. Pang, L.P., Lv, J.: Constrained incremental bundle method with partial inexact oracle for nonsmooth convex semi-infinite programming problems. Comput. Optim. Appl. **64**, 433–465 (2016)
44. Rockafellar, R.T., Wets, J.J.-B.: Variational Analysis. Springer, Berlin (1998)
45. Rustem, B., Nguyen, Q.: An algorithm for the inequality-constrained discrete minimax problem. SIAM J. Optim. **8**, 265–283 (1998)
46. Sagastizábal, C., Solodov, M.: An infeasible bundle method for nonsmooth convex constrained optimization without a penalty function or a filter. SIAM J. Optim. **16**, 146–169 (2005)
47. Solodov, M.V., Zavriev, S.K.: Error stabilty properties of generalized gradient-type algorithms. J. Optim. Theory Appl. **98**, 663–680 (1998)
48. Solodov, M.V.: On approximations with finite precision in bundle methods for nonsmooth optimization. J. Optim. Theory Appl. **119**, 151–165 (2003)
49. Spingarn, J.E.: Submonotone subdifferentials of Lipschitz functions. Trans. Am. Math. Soc. **264**, 77–89 (1981)
50. Stein, O.: Bi-Level Strategies in Semi-Infinite Programming. Kluwer, Boston (2003)
51. Stein, O.: On constraint qualifications in nonsmooth optimization. J. Optim. Theory. Appl. **121**, 647–671 (2004)
52. Tang, C.M., Liu, S., Jian, J.B., Li, J.L.: A feasible SQP-GS algorithm for nonconvex, nonsmooth constrained optimization. Numer. Algor. **65**, 1–22 (2014)
53. Tichatschke, R., Nebeling, V.: A cutting plane method for quadratic semi-infinite programming problems. Optimization. **19**, 803–817 (1988)
54. Wolfe, P.: A method of conjugate subgradients for minimizing nondifferentiable functions. In: Balinski, M.L., Wolfe, P. (eds.) Nondifferentiable Optimization. Math. Program. Stud., 3, pp. 145–173. North-Holland, Amsterdam (1975)
55. Yang, Y., Pang, L.P., Ma, X.F., Shen, J.: Constrained nonconvex nonsmooth optimization via proximal bundle method. J. Optim. Theory Appl. **163**, 900–925 (2014)
56. Zhang, L.P., Wu, S.-Y., López, M.A.: A new exchange method for convex semi-infinite programming. SIAM J. Optim. **20**, 2959–2977 (2010)