



A proximal subgradient algorithm with extrapolation for structured nonconvex nonsmooth problems

Tan Nhat Pham^{1,3} · Minh N. Dao² · Rakibuzzaman Shah³ · Nargiz Sultanova¹ · Guoyin Li⁴ · Syed Islam³

Received: 10 November 2022 / Accepted: 1 April 2023 / Published online: 20 June 2023
© The Author(s) 2023

Abstract

In this paper, we consider a class of structured nonconvex nonsmooth optimization problems, in which the objective function is formed by the sum of a possibly nonsmooth nonconvex function and a differentiable function with Lipschitz continuous gradient, subtracted by a weakly convex function. This general framework allows us to tackle problems involving nonconvex loss functions and problems with specific nonconvex constraints, and it has many applications such as signal recovery, compressed sensing, and optimal power flow distribution. We develop a proximal subgradient algorithm with extrapolation for solving these problems with guaranteed subsequential convergence to a stationary point. The convergence of the whole sequence generated by our algorithm is also established under the widely used Kurdyka–Łojasiewicz property. To illustrate the promising numerical performance of the proposed algorithm, we conduct numerical experiments on two important nonconvex models. These include a compressed sensing problem with a nonconvex regularization and an optimal power flow problem with distributed energy resources.

Keywords Composite optimization problem · Difference of convex · Distributed energy resources · Extrapolation · Optimal power flow · Proximal subgradient algorithm

Mathematics Subject Classification (2010) 90C26 · 49M27 · 65K05

✉ Minh N. Dao
minh.dao@rmit.edu.au

Extended author information available on the last page of the article

1 Introduction

In this work, we consider the structured optimization problem

$$\min_{x \in C} F(x) := f(x) + h(Ax) - g(x), \quad (\text{P})$$

where C is a nonempty closed subset of a finite-dimensional real Hilbert space \mathcal{H} , A is a linear mapping from \mathcal{H} to another finite-dimensional real Hilbert space \mathcal{K} , $f: \mathcal{H} \rightarrow (-\infty, +\infty]$ is a proper lower semicontinuous (possibly nonsmooth and nonconvex) function, $h: \mathcal{K} \rightarrow \mathbb{R}$ is a differentiable (possibly nonconvex) function whose gradient is Lipschitz continuous with modulus ℓ , and $g: \mathcal{H} \rightarrow (-\infty, +\infty]$ is a continuous weakly convex function with modulus β on an open convex set containing C . This broad optimization problem has many important applications in diverse areas, including power control problems [1], compressed sensing [2], portfolio optimization, supply chain problem, image segmentation, and others [3].

In particular, the model problem (P) covers two of the most general models in the literature. Firstly, in statistical learning, the following optimization model is often used

$$\min_{x \in \mathbb{R}^d} (\varphi(x) + \gamma r(x)), \quad (1)$$

where φ is called a loss function which measures the data misfitting, r is a regularization which promotes specific structure in the solution such as sparsity, and $\gamma > 0$ is a weighting parameter. Typical choices of the loss function are the least square loss function $\varphi(x) = \frac{1}{2} \|Ax - b\|^2$ where $A \in \mathbb{R}^{m \times d}$ and $b \in \mathbb{R}^m$ and the logistic loss function, which are both convex. In the literature, nonconvex loss functions have also received increased attentions. Some popular nonconvex loss functions include the ramp loss function [4, 5] and the Lorentzian norm [6]. In addition, [7] recently showed that many regularization r used in the literature can be written as difference of two convex functions, and so, the model (1) can be formulated into problem (P). These include popular regularizations such as the smoothly clipped absolute deviation (SCAD) [8], the indicator function of cardinality constraint [9], $L_1 - L_2$ regularization [2], or minimax concave penalty (MCP) [10]. Therefore, problem (P) can be interpreted as a problem with the form (1) whose objective function is the sum of a nonconvex and nonsmooth loss function and a regularization which can be expressed as a specific form of difference-of-(possibly) nonconvex functions.¹ Secondly, in the case when $C = \mathbb{R}^d$ and A is the identity mapping, problem (P) reduces to

$$\min_{x \in \mathbb{R}^d} (f(x) + h(x) - g(x)), \quad (2)$$

¹ Indeed, note that any smooth function with Lipschitz gradient function is weakly convex. By adding and subtracting $\alpha \|x\|^2$ for large $\alpha > 0$, our model problem (P) can also be mathematically reduced to the form (1) whose objective function is the sum of a nonconvex and nonsmooth loss function and a difference-of-convex regularization.

referred as the general difference-of-convex (DC) program, which is a broad class of optimization problems studied in the literature. To solve problem (2) under the convexity of g , a *generalized proximal point algorithm* was developed in [11]. For the case when both f and g are convex, [12] provided an *accelerated difference-of-convex algorithm* incorporating Nesterov's acceleration technique into the standard *difference-of-convex algorithm* (DCA) to improve its performance, while [13] proposed an *inexact successive quadratic approximation method*. When f , h , and g are all required to be convex, a *proximal difference-of-convex algorithm with extrapolation* (pDCAe) was proposed in [14], and there are also other existing studies (e.g., [15, 16]) that developed algorithms to solve such a problem.

In the cases where the loss function f is smooth and the regularization r is prox-friendly in the sense that its proximal operator can be computed efficiently, the proximal gradient method is a widely used algorithm for solving (1) (for example, see [17]). Moreover, incorporating information from previous iterations to accelerate the proximal algorithm while trying not to significantly increase the computational cost has also been a research area which receives a lot of attention. One such approach is to make use of the extrapolation technique. In this approach, *momentum* terms that involve the information from previous iterations are used to update the current iteration. Such techniques have been successfully implemented and achieved significant results, including Polyak's heavy ball method [18], Nesterov's techniques [19, 20], and the fast iterative shrinking-threshold algorithm (FISTA) [21]. In particular, extrapolation techniques have shown competitive results for optimization problems that involve sum of convex functions [22], difference of convex functions [14, 16], and ratio of nonconvex and nonsmooth functions [23].

In view of these successes, this paper proposes an extrapolated proximal subgradient algorithm for solving problem (P). In our work, comparing to the literature, the convexity and smoothness of the loss functions f are relaxed. We also allow a closed feasible set C instead of optimizing over the whole space. This general framework allows us to tackle problems involving nonconvex loss functions such as Lorentzian norm and problems with specific nonconvex constraints such as spherical constraint. We then prove that the sequence generated by the algorithm is bounded and any of its cluster points is a stationary point of the problem. We also prove the convergence of the full sequence under the assumption of Kurdyka–Łojasiewicz property. We then evaluate the performance of the proposed algorithm on a compressed sensing problem for both convex and nonconvex loss functions together with the recently proposed nonconvex $L_1 - L_2$ regularization. Finally, we formulate an optimal power flow problem considering photovoltaic systems placement, and address it using our algorithm.

The rest of this paper is organized as follows. In Section 2, we provide preliminary materials used in this work. In Section 3, we introduce our algorithm, and establish subsequential convergence and full sequential convergence of the proposed algorithm. In Section 4, we present numerical experiments for several case studies. Finally, we conclude the paper in Section 5.

2 Preliminaries

Throughout this paper, \mathcal{H} is a finite-dimensional real Hilbert space with inner product $\langle \cdot, \cdot \rangle$ and the induced norm $\| \cdot \|$. We use the notation \mathbb{N} for the set of nonnegative integers, \mathbb{R} for the set of real numbers, \mathbb{R}_+ for the set of nonnegative real numbers, and \mathbb{R}_{++} for the set of the positive real numbers.

Let $f: \mathcal{H} \rightarrow [-\infty, +\infty]$. The *domain* of f is $\text{dom } f := \{x \in \mathcal{H} : f(x) < +\infty\}$ and the *epigraph* of f is $\text{epi } f := \{(x, \rho) \in \mathcal{H} \times \mathbb{R} : f(x) \leq \rho\}$. The function f is *proper* if $\text{dom } f \neq \emptyset$ and it never takes the value $-\infty$, *lower semicontinuous* if its epigraph is a closed set, and *convex* if its epigraph is a convex set. We say that f is *weakly convex* if $f + \frac{\alpha}{2} \| \cdot \|^2$ is convex for some $\alpha \in \mathbb{R}_+$. The *modulus* of the weak convexity is the smallest constant α such that $f + \frac{\alpha}{2} \| \cdot \|^2$ is convex. Given a subset C of \mathcal{H} , the *indicator function* ι_C of C is defined by $\iota_C(x) := 0$ if $x \in C$, and $\iota_C(x) := +\infty$ if $x \notin C$. If $f + \iota_C$ is weakly convex with modulus α , then f is said to be weakly convex on C with modulus α . Some examples of weakly convex functions are quadratic functions, convex functions, and differentiable functions with Lipschitz continuous gradient.

Let $x \in \mathcal{H}$ with $|f(x)| < +\infty$. The *Fréchet subdifferential* of f at x is defined by

$$\widehat{\partial} f(x) := \left\{ x^* \in \mathcal{H} : \liminf_{y \rightarrow x} \frac{f(y) - f(x) - \langle x^*, y - x \rangle}{\|y - x\|} \geq 0 \right\}$$

and the *limiting subdifferential* of f at x is defined by

$$\partial_L f(x) := \left\{ x^* \in \mathcal{H} : \exists x_n \xrightarrow{f} x, x_n^* \rightarrow x^* \text{ with } x_n^* \in \widehat{\partial} f(x_n) \right\},$$

where the notation $y \xrightarrow{f} x$ means $y \rightarrow x$ with $f(y) \rightarrow f(x)$. In the case where $|f(x)| = +\infty$, both Fréchet subdifferential and limiting subdifferential of f at x are defined to be the empty set. The *domain* of $\partial_L f$ is given by $\text{dom } \partial_L f := \{x \in \mathcal{H} : \partial_L f(x) \neq \emptyset\}$. It can be directly verified from the definition that the limiting subdifferential has the *robustness property*

$$\partial_L f(x) = \left\{ x^* \in \mathcal{H} : \exists x_n \xrightarrow{f} x, x_n^* \rightarrow x^* \text{ with } x_n^* \in \partial_L f(x_n) \right\}.$$

Next, we revisit some important properties of the limiting subdifferential.

Lemma 2.1 (Sum rule) *Let $x \in \mathcal{H}$ and let $f, g: \mathcal{H} \rightarrow (-\infty, +\infty]$ be proper lower semicontinuous functions. Suppose that f is finite at x and g is locally Lipschitz around x . Then $\partial_L(f + g)(x) \subseteq \partial_L f(x) + \partial_L g(x)$. Moreover, if g is strictly differentiable at x , then $\partial_L(f + g)(x) = \partial_L f(x) + \nabla g(x)$.*

Proof This follows from [24, Proposition 1.107(ii) and Theorem 3.36]. \square

The following result, whose proof is included for completeness, is similar to [25, Lemma 2.9].

Lemma 2.2 (Upper semicontinuity of subdifferential) *Let $f : \mathcal{H} \rightarrow [-\infty, +\infty]$ be Lipschitz continuous around $x \in \mathcal{H}$, let $(x_n)_{n \in \mathbb{N}}$ be a sequence in \mathcal{H} converging to x , and let, for each $n \in \mathbb{N}$, $x_n^* \in \partial_L f(x_n)$. Then $(x_n^*)_{n \in \mathbb{N}}$ is bounded with all cluster points contained in $\partial_L f(x)$.*

Proof By the Lipschitz continuity of f around x , there are a neighborhood V of x and a constant $\ell_V \in \mathbb{R}_+$ such that f is Lipschitz continuous on V with modulus ℓ_V . Then, by [24, Corollary 1.81], for all $v \in V$ and $v^* \in \partial_L f(v)$, one has $\|v^*\| \leq \ell_V$. Since $x_n \rightarrow x$ as $n \rightarrow +\infty$, there is $n_0 \in \mathbb{N}$ such that, for all $n \geq n_0$, $x_n \in V$, which implies that $\|x_n^*\| \leq \ell_V$. This means $(x_n^*)_{n \in \mathbb{N}}$ is bounded.

Now, let x^* be a cluster point of $(x_n^*)_{n \in \mathbb{N}}$, i.e., there exists a subsequence $(x_{k_n}^*)_{n \in \mathbb{N}}$ such that $x_{k_n}^* \rightarrow x^*$ as $n \rightarrow +\infty$. On the other hand, we have from the convergence of $(x_n)_{n \in \mathbb{N}}$ and the Lipschitz continuity of f around x that $x_{k_n} \xrightarrow{f} x$. Therefore, $x^* \in \partial_L f(x)$ due to the robustness property of the limiting subdifferential. \square

We end this section with the definitions of stationary points for the problem (P). A point $\bar{x} \in C$ is said to be a

- *stationary point* of (P) if $0 \in \partial_L(f + \iota_C + h \circ A - g)(\bar{x})$,
- *lifted stationary point* of (P) if $0 \in \partial_L(f + \iota_C)(\bar{x}) + A^* \nabla h(A\bar{x}) - \partial_L g(\bar{x})$.

Here A^* is the adjoint mapping of the linear mapping A .

3 Proximal subgradient algorithm with extrapolation

We now propose our extrapolated proximal subgradient algorithm for solving problem (P) with guaranteed convergence to stationary points.

Algorithm 1 Proximal subgradient algorithm with extrapolation

▷ **Step 1.** Let $x_{-1} = x_0 \in C$ and set $n = 0$. Let $\bar{\lambda} \in \mathbb{R}_+$, $\bar{\mu} \in \mathbb{R}_+$, and $\delta \in \mathbb{R}_{++}$.

▷ **Step 2.** Let $g_n \in \partial_L g(x_n)$, $u_n = x_n + \lambda_n(x_n - x_{n-1})$ with $\lambda_n \in [0, \bar{\lambda}]$, and $v_n = x_n + \mu_n(x_n - x_{n-1})$ with $\mu_n \in [0, \bar{\mu}\tau_n]$. Choose $\tau_n \in \left(0, 1/(\beta + 2\delta + \ell\|A\|^2(2\bar{\lambda} + 1) + 2\bar{\mu})\right]$ and compute

$$x_{n+1} \in \operatorname{argmin}_{x \in C} \left(f(x) + \frac{1}{2\tau_n} \|x - v_n + \tau_n A^* \nabla h(Au_n) - \tau_n g_n\|^2 \right).$$

▷ **Step 3.** If a termination criterion is not met, set $n = n + 1$ and go to Step 2.

Remark 3.1 (Discussion of the algorithm structure and extrapolation parameters) Some comments on Algorithm 1 are in order.

- (i) Recalling that the proximal operator of a proper function $\phi : \mathcal{H} \rightarrow (-\infty, +\infty]$ is defined by

$$\operatorname{prox}_\phi(x) = \operatorname{argmin}_{y \in \mathcal{H}} \left(\phi(y) + \frac{1}{2} \|y - x\|^2 \right),$$

we see that the update of x_{n+1} in Step 2 can be written as

$$x_{n+1} \in \text{prox}_{\tau_n(f+\iota_C)}(v_n - \tau_n A^* \nabla h(Au_n) + \tau_n g_n).$$

Therefore, Step 2 essentially boils down to the computation of the proximal operator of $\tau_n(f + \iota_C)$. This can be done efficiently for various specific structures of f and C .

- First, in the case where $f \equiv 0$, computing the proximal operator of $\tau_n(f + \iota_C)$ is equivalent to computing the projection onto the set C . This can be efficiently computed in many applications or even admits a closed form solution, for example, when C is a ball, a sphere, or takes the form $\{x \in \mathcal{H} : \|x\|_0 \leq r\}$ with $r > 0$. Here, $\|x\|_0$ denotes the cardinality of the vector x .
 - Second, in the case where $C = \mathcal{H} = \mathbb{R}^d$, this task reduces to computing the proximal operator of $\tau_n f$, which can also have closed form solution for several nonconvex/nonsmooth functions f . These include various popular regularization functions in the literature, such as the $L_{1/2}$ regularization $f(x) = \sum_{i=1}^d |x_i|^{\frac{1}{2}}$ [26, Theorem 1], the $L_1 - L_2$ regularization $f(x) = \|x\|_1 - \alpha \|x\|$ ($\alpha \in \mathbb{R}_+$) [2, Lemma 1], and the sum entropy and sparsity regularization terms [27, Section 3].
 - In addition, when f is a convex quadratic function and C is a polyhedral set, computing the proximal operator of $\tau_n(f + \iota_C)$ is equivalent to solving a convex quadratic programming problem. When f is a nonconvex quadratic function and C is the unit ball, this reduces to a trust region problem which can be solved as a generalized eigenvalue problem or a semi-definite programming problem. For further tractable cases, see, e.g., [23, Remark 4.1].
- (ii) Let us consider the case when A is the identity mapping and $C = \mathcal{H}$. We fix an arbitrary $\tau \in (0, 1/\ell)$ and choose $\bar{\lambda} = \bar{\mu} = 0$ (which yields $\lambda_n = \mu_n = 0$), $\delta \in (0, 1/(2\tau) - \ell/2)$, and $\tau_n = \tau$. Then the update of x_{n+1} in Step 2 becomes

$$x_{n+1} \in \text{prox}_{\tau f}(x_n - \tau \nabla h(x_n) + \tau g_n),$$

which is the so-called *generalized proximal point algorithm* (GPPA) in [11], where g is assumed to be convex (In this case, $\beta = 0$ and $1/\tau_n = 1/\tau > 2\delta + \ell = \beta + 2\delta + \ell \|A\|^2 (2\bar{\lambda} + 1) + 2\bar{\mu}$).

- (iii) In the case where $h \equiv 0$, the objective function F reduces to $f - g$ and the update of x_{n+1} in Step 2 reduces to

$$x_{n+1} \in \text{prox}_{\tau_n(f+\iota_C)}(v_n + \tau_n g_n).$$

In turn, if $C = \mathcal{H}$ and $\mu_n = 0$, Algorithm 1 reduces to the *proximal linearized algorithm* proposed in [28] which requires that f and g are convex.

- (iv) When $g \equiv 0$, A is the identity mapping, and $C = \mathcal{H}$, the objective function reduces to $f + h$. By choosing $\lambda_n = \mu_n$, we have

$$x_{n+1} \in \text{prox}_{\tau_n f}(u_n - \tau_n \nabla h(u_n))$$

and Algorithm 1 reduces to the *inertial forward-backward algorithm* studied in [22], in which an additional requirement of the convexity of h is imposed.

- (v) Motivated by the popular parameter used in FISTA and also its variants (such as the restarted FISTA scheme) [17, Chapter 10], a plausible option for extrapolation parameters λ_n and μ_n (which will be used in our computation later) is that

$$\lambda_n = \frac{\bar{\lambda} \kappa_{n-1} - 1}{\kappa_n} \quad \text{and} \quad \mu_n = \frac{\bar{\mu} \tau_n \kappa_{n-1} - 1}{\kappa_n},$$

where $\kappa_{-1} = \kappa_0 = 1$ and $\kappa_{n+1} = \frac{1 + \sqrt{1 + 4\kappa_n^2}}{2}$. It can be seen that, for all $n \in \mathbb{N}$, $1 \leq \kappa_{n-1} < \kappa_n + 1$, and so $\lambda_n \in [0, \bar{\lambda}]$ and $\mu_n \in [0, \bar{\mu} \tau_n]$. We can also reset $\kappa_{n-1} = \kappa_n = 1$ whenever n is a multiple of some fix integer n_0 .

From now on, let $(x_n)_{n \in \mathbb{N}}$ be a sequence generated by Algorithm 1. Under suitable assumptions, we show in the next theorem that $(x_n)_{n \in \mathbb{N}}$ is bounded and any of its cluster points is a stationary point of problem (P).

Theorem 3.2 (Subsequential convergence) *For problem (P), suppose that the function F is bounded from below on C and that the set $C_0 := \{x \in C : F(x) \leq F(x_0)\}$ is bounded. Set $c := \frac{1}{2}(\ell \|A\|^2 \bar{\lambda} + \bar{\mu})$. Then the following statements hold:*

- (i) For all $n \in \mathbb{N}$,

$$(F(x_{n+1}) + c \|x_{n+1} - x_n\|^2) + \delta \|x_{n+1} - x_n\|^2 \leq F(x_n) + c \|x_n - x_{n-1}\|^2 \quad (3)$$

and the sequence $(F(x_n))_{n \in \mathbb{N}}$ is convergent.

- (ii) The sequence $(x_n)_{n \in \mathbb{N}}$ is bounded and $x_{n+1} - x_n \rightarrow 0$ as $n \rightarrow +\infty$.
 (iii) Suppose that $\liminf_{n \rightarrow +\infty} \tau_n = \bar{\tau} > 0$ and let \bar{x} be a cluster point of $(x_n)_{n \in \mathbb{N}}$. Then $\bar{x} \in C \cap \text{dom } f$, $F(x_n) \rightarrow F(\bar{x})$, and \bar{x} is a lifted stationary point of (P). Moreover, \bar{x} is a stationary point of (P) provided that g is strictly differentiable on an open set containing $C \cap \text{dom } f$.

Proof (i) & (ii): We see from Step 2 of Algorithm 1 that, for all $n \in \mathbb{N}$, $x_n \in C$ and

$$\begin{aligned} x_{n+1} &\in \operatorname{argmin}_{x \in C} \left(f(x) + \frac{1}{2\tau_n} \|x - v_n + \tau_n A^* \nabla h(Au_n) - \tau_n g_n\|^2 \right) \\ &= \operatorname{argmin}_{x \in C} \left(f(x) + \frac{1}{2\tau_n} \|x - v_n\|^2 + \langle A^* \nabla h(Au_n) - g_n, x - v_n \rangle \right) \\ &= \operatorname{argmin}_{x \in C} \left(f(x) + \langle A^* \nabla h(Au_n), x - u_n \rangle - \langle g_n, x - v_n \rangle \right. \\ &\quad \left. + \langle A^* \nabla h(Au_n), u_n - v_n \rangle + \frac{1}{2\tau_n} \|x - v_n\|^2 \right) \\ &= \operatorname{argmin}_{x \in C} \left(f(x) + \langle \nabla h(Au_n), Ax - Au_n \rangle - \langle g_n, x - v_n \rangle + \frac{1}{2\tau_n} \|x - v_n\|^2 \right). \end{aligned}$$

Therefore, for all $n \in \mathbb{N}$ and all $x \in C$,

$$\begin{aligned} &f(x) + \langle \nabla h(Au_n), Ax - Au_n \rangle - \langle g_n, x - v_n \rangle + \frac{1}{2\tau_n} \|x - v_n\|^2 \\ &\geq f(x_{n+1}) + \langle \nabla h(Au_n), Ax_{n+1} - Au_n \rangle - \langle g_n, x_{n+1} - v_n \rangle + \frac{1}{2\tau_n} \|x_{n+1} - v_n\|^2, \end{aligned}$$

or equivalently,

$$\begin{aligned} f(x) &\geq f(x_{n+1}) + \langle \nabla h(Au_n), Ax_{n+1} - Ax \rangle - \langle g_n, x_{n+1} - x \rangle \\ &\quad + \frac{1}{2\tau_n} (\|x_{n+1} - v_n\|^2 - \|x - v_n\|^2). \end{aligned} \quad (4)$$

By the Lipschitz continuity of ∇h , we derive from [20, Lemma 1.2.3] that

$$\begin{aligned} &\langle \nabla h(Au_n), Ax_{n+1} - Ax_n \rangle \\ &= \langle \nabla h(Ax_n), Ax_{n+1} - Ax_n \rangle + \langle \nabla h(Au_n) - \nabla h(Ax_n), Ax_{n+1} - Ax_n \rangle \\ &\geq h(Ax_{n+1}) - h(Ax_n) - \frac{\ell}{2} \|Ax_{n+1} - Ax_n\|^2 \\ &\quad - \|\nabla h(Au_n) - \nabla h(Ax_n)\| \|Ax_{n+1} - Ax_n\| \\ &\geq h(Ax_{n+1}) - h(Ax_n) - \frac{\ell \|A\|^2}{2} \|x_{n+1} - x_n\|^2 - \ell \|A\|^2 \|u_n - x_n\| \|x_{n+1} - x_n\|. \end{aligned}$$

As $g_n \in \partial_L g(x_n)$ and $x_n, x_{n+1} \in C$, it follows from the weak convexity of g and [23, Lemma 4.1] that

$$\langle g_n, x_{n+1} - x_n \rangle \leq g(x_{n+1}) - g(x_n) + \frac{\beta}{2} \|x_{n+1} - x_n\|^2.$$

Letting $x = x_n \in C$ in (4) and combining with the last two inequalities, we obtain that

$$\begin{aligned} & f(x_n) + h(Ax_n) - g(x_n) \\ & \geq f(x_{n+1}) + h(Ax_{n+1}) - g(x_{n+1}) - \left(\frac{\ell \|A\|^2}{2} + \frac{\beta}{2} \right) \|x_{n+1} - x_n\|^2 \\ & \quad - \ell \|A\|^2 \|u_n - x_n\| \|x_{n+1} - x_n\| + \frac{1}{2\tau_n} (\|x_{n+1} - v_n\|^2 - \|x_n - v_n\|^2). \end{aligned}$$

By the definition of u_n and v_n , we have $x_n - u_n = -\lambda_n(x_n - x_{n-1})$, $x_{n+1} - v_n = (x_{n+1} - x_n) - \mu_n(x_n - x_{n-1})$, $x_n - v_n = -\mu_n(x_n - x_{n-1})$, and so

$$\begin{aligned} F(x_n) & \geq F(x_{n+1}) - \left(\frac{\ell \|A\|^2}{2} + \frac{\beta}{2} \right) \|x_{n+1} - x_n\|^2 - \ell \|A\|^2 \lambda_n \|x_n - x_{n-1}\| \|x_{n+1} - x_n\| \\ & \quad + \frac{1}{2\tau_n} (\|x_{n+1} - x_n\|^2 - 2\mu_n \langle x_{n+1} - x_n, x_n - x_{n-1} \rangle) \\ & \geq F(x_{n+1}) + \left(\frac{1}{2\tau_n} - \frac{\ell \|A\|^2}{2} - \frac{\beta}{2} \right) \|x_{n+1} - x_n\|^2 \\ & \quad - \left(\ell \|A\|^2 \lambda_n + \frac{\mu_n}{\tau_n} \right) \|x_{n+1} - x_n\| \|x_n - x_{n-1}\| \\ & \geq F(x_{n+1}) - \left(\frac{\ell \|A\|^2 \lambda_n}{2} + \frac{\mu_n}{2\tau_n} \right) \|x_n - x_{n-1}\|^2 \\ & \quad + \left(\frac{1}{2\tau_n} - \frac{\ell \|A\|^2}{2} - \frac{\beta}{2} - \frac{\ell \|A\|^2 \lambda_n}{2} - \frac{\mu_n}{2\tau_n} \right) \|x_{n+1} - x_n\|^2, \end{aligned}$$

where we have used $\langle x_{n+1} - x_n, x_n - x_{n-1} \rangle \leq \|x_{n+1} - x_n\| \|x_n - x_{n-1}\| \leq \frac{1}{2} (\|x_{n+1} - x_n\|^2 + \|x_n - x_{n-1}\|^2)$. Rearranging terms yields

$$\begin{aligned} & F(x_n) + \left(\frac{\ell \|A\|^2 \lambda_n}{2} + \frac{\mu_n}{2\tau_n} \right) \|x_n - x_{n-1}\|^2 \\ & \geq F(x_{n+1}) + \left(\frac{1}{2\tau_n} - \frac{\ell \|A\|^2}{2} - \frac{\beta}{2} - \frac{\ell \|A\|^2 \lambda_n}{2} - \frac{\mu_n}{2\tau_n} \right) \|x_{n+1} - x_n\|^2. \end{aligned}$$

Since $\lambda_n \in [0, \bar{\lambda}]$, $\mu_n \in [0, \bar{\mu}\tau_n]$, and $1/\tau_n \geq \beta + 2\delta + \ell \|A\|^2 (2\bar{\lambda} + 1) + 2\bar{\mu}$, it follows that

$$\begin{aligned} & F(x_n) + \frac{1}{2} (\ell \|A\|^2 \bar{\lambda} + \bar{\mu}) \|x_n - x_{n-1}\|^2 \geq F(x_{n+1}) \\ & \quad + \frac{1}{2} (2\delta + \ell \|A\|^2 \bar{\lambda} + \bar{\mu}) \|x_{n+1} - x_n\|^2, \end{aligned}$$

which proves (3).

Recalling $c = \frac{1}{2}(\ell\|A\|^2\bar{\lambda} + \bar{\mu})$ and setting $\mathcal{F}_n := F(x_n) + c\|x_n - x_{n-1}\|^2$, we have

$$\mathcal{F}_n \geq \mathcal{F}_{n+1} + \delta\|x_{n+1} - x_n\|^2. \quad (5)$$

Since $\delta > 0$, the sequence $(\mathcal{F}_n)_{n \in \mathbb{N}}$ is nonincreasing. Since F is bounded below on C , the sequence $(\mathcal{F}_n)_{n \in \mathbb{N}}$ is bounded below, and it is therefore convergent. After rearranging (5) and performing telescoping, we obtain that, for all $m \in \mathbb{N}$,

$$\delta \sum_{n=0}^m \|x_{n+1} - x_n\|^2 \leq \sum_{n=0}^m (\mathcal{F}_n - \mathcal{F}_{n+1}) = \mathcal{F}_0 - \mathcal{F}_{m+1}.$$

Denoting $\bar{\mathcal{F}} := \lim_{n \rightarrow +\infty} \mathcal{F}_n$ and letting $m \rightarrow +\infty$, we obtain that

$$\sum_{n=0}^{+\infty} \|x_{n+1} - x_n\|^2 \leq \frac{1}{\delta}(\mathcal{F}_0 - \bar{\mathcal{F}}) < +\infty.$$

Therefore, as $n \rightarrow +\infty$, $x_{n+1} - x_n \rightarrow 0$, and so $F(x_n) = \mathcal{F}_n - c\|x_n - x_{n-1}\|^2 \rightarrow \bar{\mathcal{F}}$, which means that the sequence $(F(x_n))_{n \in \mathbb{N}}$ is convergent.

Now, we observe that

$$F(x_n) = \mathcal{F}_n - c\|x_n - x_{n-1}\|^2 \leq \mathcal{F}_n \leq \mathcal{F}_0 = F(x_0),$$

which implies $x_n \in C_0 = \{x \in C : F(x) \leq F(x_0)\}$. Hence, $(x_n)_{n \in \mathbb{N}}$ is bounded due to the boundedness of C_0 .

(iii): As \bar{x} is a cluster point of the sequence $(x_n)_{n \in \mathbb{N}}$, there exists a subsequence $(x_{k_n})_{n \in \mathbb{N}}$ of $(x_n)_{n \in \mathbb{N}}$ such that $x_{k_n} \rightarrow \bar{x}$ as $n \rightarrow +\infty$. Then $\bar{x} \in C$ and, since $x_{n+1} - x_n \rightarrow 0$, one has $x_{k_n-1} \rightarrow \bar{x}$, so as u_{k_n-1} and v_{k_n-1} . Since $g + \frac{\beta}{2}\|\cdot\|^2$ is a continuous convex function on an open set O containing C , we obtain from [29, Example 9.14] that g is locally Lipschitz continuous on O . In view of Lemma 2.2, since $x_{k_n} \rightarrow \bar{x}$ as $n \rightarrow +\infty$, passing to a subsequence if necessary, we can assume that $g_{k_n} \rightarrow \bar{g} \in \partial_L g(\bar{x})$ as $n \rightarrow +\infty$.

Replacing n in (4) with $k_n - 1$, we have for all $n \in \mathbb{N}$ and all $x \in C$ that

$$\begin{aligned} f(x) &\geq f(x_{k_n}) + \langle \nabla h(Au_{k_n-1}), Ax_{k_n} - Ax \rangle - \langle g_{k_n-1}, x_{k_n} - x \rangle \\ &\quad + \frac{1}{2\tau_{k_n-1}} (\|x_{k_n} - v_{k_n-1}\|^2 - \|x - v_{k_n-1}\|^2). \end{aligned} \quad (6)$$

As $\liminf_{n \rightarrow +\infty} \tau_n = \bar{\tau} > 0$, letting $x = \bar{x}$ and $n \rightarrow \infty$, we obtain that $f(\bar{x}) \geq \limsup_{n \rightarrow +\infty} f(x_{k_n})$. Since f is lower semicontinuous, it follows that $\lim_{n \rightarrow +\infty} f(x_{k_n}) = f(\bar{x})$. On the other hand, $\lim_{n \rightarrow +\infty} g(x_{k_n}) = g(\bar{x})$ and $\lim_{n \rightarrow +\infty} h(Ax_{k_n}) = h(A\bar{x})$ due to the continuity of g and h . Therefore,

$$\begin{aligned} \lim_{n \rightarrow +\infty} F(x_n) &= \lim_{n \rightarrow +\infty} F(x_{k_n}) = \lim_{n \rightarrow +\infty} (f(x_{k_n}) + h(Ax_{k_n}) - g(x_{k_n})) \\ &= f(\bar{x}) + h(A\bar{x}) - g(\bar{x}) = F(\bar{x}). \end{aligned}$$

Next, by letting $n \rightarrow +\infty$ in (6), for all $x \in C$,

$$f(x) \geq f(\bar{x}) + \langle \nabla h(A\bar{x}), A\bar{x} - Ax \rangle - \langle \bar{g}, \bar{x} - x \rangle - \frac{1}{2\bar{\tau}} \|x - \bar{x}\|^2,$$

which can be rewritten as

$$\begin{aligned} f(x) + \langle \nabla h(A\bar{x}), Ax - A\bar{x} \rangle - \langle \bar{g}, x - \bar{x} \rangle + \frac{1}{2\bar{\tau}} \|x - \bar{x}\|^2 \\ \geq f(\bar{x}) + \langle \nabla h(A\bar{x}), A\bar{x} - A\bar{x} \rangle - \langle \bar{g}, \bar{x} - \bar{x} \rangle + \frac{1}{2\bar{\tau}} \|\bar{x} - \bar{x}\|^2. \end{aligned}$$

This means \bar{x} is a minimizer of the function $(f + \langle \nabla h(A\bar{x}), A \cdot - A\bar{x} \rangle - \langle \bar{g}, \cdot - \bar{x} \rangle + \frac{1}{2\bar{\tau}} \|\cdot - \bar{x}\|^2)(x)$ over C . Hence, $0 \in \partial_L(f + \langle \nabla h(A\bar{x}), A \cdot - A\bar{x} \rangle - \langle \bar{g}, \cdot - \bar{x} \rangle + \frac{1}{2\bar{\tau}} \|\cdot - \bar{x}\|^2 + \iota_C)(\bar{x}) = \partial_L(f + \iota_C)(\bar{x}) + A^* \nabla h(A\bar{x}) - \bar{g}$, and we must have $\bar{x} \in C \cap \text{dom } f$. Since $\bar{g} \in \partial_L g(\bar{x})$, we deduce that $0 \in \partial_L(f + \iota_C)(\bar{x}) + A^* \nabla h(A\bar{x}) - \partial_L g(\bar{x})$, i.e., \bar{x} is a lifted stationary point of (P). In addition, if we further require that g is strictly differentiable, then Lemma 2.1 implies that \bar{x} is a stationary point of (P). \square

Next, we establish the convergence of the full sequence generated by Algorithm 1. In order to do this, we recall that a proper lower semicontinuous function $G: \mathcal{H} \rightarrow (-\infty, +\infty]$ satisfies the Kurdyka–Łojasiewicz (KL) property [30, 31] at $\bar{x} \in \text{dom } \partial_L G$ if there exist $\eta \in (0, +\infty]$, a neighborhood V of \bar{x} , and a continuous concave function $\phi: [0, \eta) \rightarrow \mathbb{R}_+$ such that ϕ is continuously differentiable with $\phi' > 0$ on $(0, \eta)$, $\phi(0) = 0$, and, for all $x \in V$ with $G(\bar{x}) < G(x) < G(\bar{x}) + \eta$,

$$\phi'(G(x) - G(\bar{x})) \text{dist}(0, \partial_L G(x)) \geq 1.$$

We say that G is a KL function if it satisfies the KL property at any point in $\text{dom } \partial_L G$. If G satisfies the KL property at $\bar{x} \in \text{dom } \partial_L G$, in which the corresponding function ϕ can be chosen as $\phi(t) = ct^{1-\theta}$ for some $c \in \mathbb{R}_{++}$ and $\theta \in [0, 1)$, then G is said to satisfy the KL property at \bar{x} with exponent θ . The function G is called a KL function with exponent θ if it is a KL function and has the same exponent θ at any $x \in \text{dom } \partial_L G$.

Theorem 3.3 (Full sequential convergence) *For problem (P), suppose that F is bounded from below on C , that the set $C_0 := \{x \in C : F(x) \leq F(x_0)\}$ is bounded, that g is differentiable on an open set containing $C \cap \text{dom } f$ whose gradient ∇g is Lipschitz continuous with modulus ℓ_g on $C \cap \text{dom } f$, and that $\liminf_{n \rightarrow +\infty} \tau_n = \bar{\tau} > 0$. Define*

$$G(x, y) := F(x) + \iota_C(x) + c\|x - y\|^2,$$

where $c = \frac{1}{2}(\ell\|A\|^2\bar{\lambda} + \bar{\mu})$, and suppose that G satisfies the KL property at (\bar{x}, \bar{x}) for every $\bar{x} \in C \cap \text{dom } f$. Then

- (i) The sequence $(x_n)_{n \in \mathbb{N}}$ converges to a stationary point x^* of (P) and $\sum_{n=0}^{+\infty} \|x_{n+1} - x_n\| < +\infty$.
- (ii) Suppose further that G satisfies the KL property with exponent $\theta \in [0, 1)$ at (\bar{x}, \bar{x}) for every $\bar{x} \in C \cap \text{dom } f$. The following statements hold:

- (a) If $\theta = 0$, then $(x_n)_{n \in \mathbb{N}}$ converges to x^* in a finite number of steps.
 (b) If $\theta \in (0, \frac{1}{2}]$, then there exist $\gamma \in \mathbb{R}_{++}$ and $\rho \in (0, 1)$ such that, for all $n \in \mathbb{N}$,
 $\|x_n - x^*\| \leq \gamma \rho^{\frac{n}{2}}$ and $|F(x_n) - F(x^*)| \leq \gamma \rho^n$.
 (c) If $\theta \in (\frac{1}{2}, 1)$, then there exists $\gamma \in \mathbb{R}_{++}$ such that, for all $n \in \mathbb{N}$, $\|x_n - x^*\| \leq \gamma n^{-\frac{1-\theta}{2\theta-1}}$ and $|F(x_n) - F(x^*)| \leq \gamma n^{-\frac{2-2\theta}{2\theta-1}}$.

Proof For each $n \in \mathbb{N}$, let $z_n = (x_{n+1}, x_n)$. According to Theorem 3.2, we have that, for all $n \in \mathbb{N}$,

$$G(z_{n+1}) + \delta \|x_{n+2} - x_{n+1}\|^2 \leq G(z_n),$$

that the sequence $(z_n)_{n \in \mathbb{N}}$ is bounded, that $z_{n+1} - z_n \rightarrow 0$ as $n \rightarrow +\infty$, and that for every cluster point \bar{z} of $(z_n)_{n \in \mathbb{N}}$, $\bar{z} = (\bar{x}, \bar{x})$, where $\bar{x} \in C \cap \text{dom } f$ is a stationary point of (P) and $G(z_n) = F(x_{n+1}) + c\|x_{n+1} - x_n\|^2 \rightarrow F(\bar{x}) = G(\bar{z})$ as $n \rightarrow +\infty$.

Let $n \in \mathbb{N}$. It follows from the update of x_{n+1} in Step 2 of Algorithm 1 that

$$0 \in \partial_L(f + \iota_C)(x_{n+1}) + \frac{1}{\tau_n}(x_{n+1} - v_n + \tau_n A^* \nabla h(Au_n) - \tau_n \nabla g(x_n)),$$

which implies that

$$\nabla g(x_n) - A^* \nabla h(Au_n) - \frac{1}{\tau_n}(x_{n+1} - v_n) \in \partial_L(f + \iota_C)(x_{n+1}).$$

Noting that $G(z_n) = (f + \iota_C)(x_{n+1}) + h(Ax_{n+1}) - g(x_{n+1}) + c\|x_{n+1} - x_n\|^2$ and that

$$\begin{aligned} \partial_L G(z_n) &= \{\partial_L(f + \iota_C)(x_{n+1}) + A^* \nabla h(Ax_{n+1}) - \nabla g(x_{n+1}) + 2c(x_{n+1} - x_n)\} \\ &\quad \times \{2c(x_n - x_{n+1})\}, \end{aligned}$$

we obtain

$$\begin{aligned} \text{dist}(0, \partial_L G(z_n)) &\leq \|\nabla g(x_n) - A^* \nabla h(Au_n) - \frac{1}{\tau_n}(x_{n+1} - v_n) + A^* \nabla h(Ax_{n+1}) \\ &\quad - \nabla g(x_{n+1}) + 2c(x_{n+1} - x_n)\| + 2c\|x_n - x_{n+1}\| \\ &\leq \ell_g \|x_{n+1} - x_n\| + \ell \|A\| \|x_{n+1} - u_n\| + \frac{1}{\tau_n} \|x_{n+1} - v_n\| \\ &\quad + 4c\|x_{n+1} - x_n\|. \end{aligned}$$

Since $\|x_{n+1} - u_n\| \leq \|x_{n+1} - x_n\| + \lambda_n \|x_n - x_{n-1}\|$ and $\|x_{n+1} - v_n\| \leq \|x_{n+1} - x_n\| + \mu_n \|x_n - x_{n-1}\|$, we derive that

$$\begin{aligned} \text{dist}(0, \partial_L G(z_n)) &\leq \left(\ell_g + \ell \|A\| + \frac{1}{\tau_n} + 4c \right) \|x_{n+1} - x_n\| \\ &\quad + \left(\ell \|A\| \lambda_n + \frac{\mu_n}{\tau_n} \right) \|x_n - x_{n-1}\|. \end{aligned}$$

Since $\liminf_{n \rightarrow +\infty} \tau_n = \bar{\tau} > 0$, there exists $n_0 \in \mathbb{N}$ such that, for all $n \geq n_0$, $\tau_n \geq \bar{\tau}/2$. Recalling that $\lambda_n \leq \bar{\lambda}$ and $\frac{\mu_n}{\tau_n} \leq \bar{\mu}$, we have for all $n \geq n_0$ that

$$\text{dist}(0, \partial_L G(z_n)) \leq \eta_1 \|x_{n+1} - x_n\| + \eta_2 \|x_n - x_{n-1}\|,$$

where $\eta_1 = \ell_g + \ell \|A\| + \frac{2}{\bar{\tau}} + 4c$ and $\eta_2 = \ell \|A\| \bar{\lambda} + \bar{\mu}$. Now, the first conclusion follows by applying [23, Theorem 5.1] with $I = \{1, 2\}$, $\lambda_1 = \frac{\eta_1}{\eta_1 + \eta_2}$, $\lambda_2 = \frac{\eta_2}{\eta_1 + \eta_2}$, $\Delta_n = \|x_{n+2} - x_{n+1}\|$, $\alpha_n \equiv \delta$, $\beta_n \equiv \frac{1}{\eta_1 + \eta_2}$, and $\varepsilon_n \equiv 0$. The remaining conclusions follow a rather standard line of argument as used in [23, 32, 33], see also [34, Theorem 3.11]. \square

Remark 3.4 (KL property and KL exponents) In the preceding theorem, the convergence of the full sequence generated by Algorithm 1 requires the KL property of the function G with the form that $G(x, y) := F(x) + \iota_C(x) + c\|x - y\|^2$, where F is the objective function of the model problem (P), C is the feasible region of problem (P) and $c > 0$. We note that this assumption holds for a broad class of model problem (P) where F is a semi-algebraic function and C is a semi-algebraic set. More generally, it continues to hold when F is a definable function and C is a definable set (see [30, 35]).

As simple illustrations, in our case study in the next section, we will consider the following two classes of functions:

- (i) $F(x) = \varphi(Ax) + \gamma(\|x\|_1 - \alpha\|x\|)$, where $\varphi(z) = \frac{1}{2}\|z - b\|^2$ (least square loss) or $\varphi(z) = \|z - b\|_{L_{2,1}} = \sum_{i=1}^m \log(1 + |z_i - b_i|^2)$ (Lorentzian norm loss [6]), $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$, $\alpha \in \mathbb{R}_+$, and $\gamma \in \mathbb{R}_{++}$.
- (ii) $F(x) = \frac{1}{2}x^T Mx + u^T x + r$, where M is an $(d \times d)$ symmetric matrix, $u \in \mathbb{R}^d$, and $r \in \mathbb{R}$.

Let $G(x, y) := F(x) + \iota_C(x) + c\|x - y\|^2$, where $c > 0$ and C is a semi-algebraic set in \mathbb{R}^d . Then, in both cases, G is definable, and so, it satisfies the KL property at (\bar{x}, \bar{x}) for all $\bar{x} \in C \cap \text{dom } F$. Moreover, for case (ii), if C is further assumed to be a polyhedral set, then as shown in [33] the KL exponent for G is $\frac{1}{2}$, and by Theorem 3.3, the proposed algorithm exhibits a linear convergence rate.

4 Case studies

In this section, we provide the numerical results of our proposed algorithm for two case studies: compressed sensing with $L_1 - L_2$ regularization, and optimal power flow problem which considers photovoltaic systems placement for a low voltage network. All of the experiments are performed in MATLAB R2021b on a 64-bit laptop with Intel(R) Core(TM) i7-1165G7 CPU (2.80GHz) and 16GB of RAM.

4.1 Compressed sensing with $L_1 - L_2$ regularization

We consider the compressed sensing problem

$$\min_{x \in \mathbb{R}^d} (\varphi(Ax) + \gamma(\|x\|_1 - \alpha\|x\|)), \quad (7)$$

where $A \in \mathbb{R}^{m \times d}$ is an underdetermined sensing matrix of full row rank, $\gamma \in \mathbb{R}_{++}$, and $\alpha \in \mathbb{R}_{++}$. Here, φ can be the least square loss function and the Lorentzian norm loss function mentioned in Remark 3.4.

In our numerical experiments, we let $\alpha = 1$ to be consistent with the setting in [2]. We first start with the least square loss function. By letting $\varphi(z) = \frac{1}{2}\|z - b\|^2$, where $b \in \mathbb{R}^m \setminus \{0\}$, the problem (7) now becomes

$$\min_{x \in \mathbb{R}^d} \left(\frac{1}{2}\|Ax - b\|^2 + \gamma(\|x\|_1 - \|x\|) \right). \quad (8)$$

This is known as the regularized least square problem, which has many applications in signal and image processing [2, 36, 37]. To solve problem (8), we use Algorithm 1 with $f = \gamma\|\cdot\|_1$, $h = \varphi$, and $g = \gamma\|\cdot\|$. Then the update of x_{n+1} in Step 2 of Algorithm 1 reads as

$$x_{n+1} = \operatorname{argmin}_{x \in \mathbb{R}^d} (\gamma\|x\|_1 + \frac{1}{2\tau_n}\|x - w_n\|^2) = \operatorname{prox}_{\gamma\tau_n\|\cdot\|_1}(w_n),$$

where $w_n = v_n - \tau_n A^*(Au_n - b) + \gamma\tau_n g_n$, and where $g_n \in \partial_L \|\cdot\|(x_n)$ is given by

$$g_n = \begin{cases} 0 & \text{if } x_n = 0, \\ \frac{x_n}{\|x_n\|} & \text{if } x_n \neq 0. \end{cases}$$

In this case, the proximal operator is the *soft shrinkage* operator [21], and so, for all $i = 1, \dots, d$,

$$(x_{n+1})_i = \operatorname{sign}((w_n)_i) \max\{0, |(w_n)_i| - \gamma\tau_n\}.$$

For this test case, we compare our proposed Algorithm 1 with the following algorithms:

- Alternating direction method of multipliers (ADMM) proposed in [2];
- Generalized proximal point algorithm (GPPA) proposed in [11];
- Proximal difference-of-convex algorithm with extrapolation (pDCAe) in [14].

Note that the ADMM algorithm uses the $L_1 - L_2$ proximal operator which was first proposed in [2]. For ADMM, we have $f(x) = \gamma\|x\|_1 - \gamma\|x\|$, $h(x) = \varphi(Ax)$, and $g \equiv 0$. For GPPA and pDCAe, we let $f(x) = \gamma\|x\|_1$, $h(x) = \varphi(Ax)$, and $g(x) = \gamma\|x\|$. The parameters of ADMM and pDCA are derived from [2, 14]. The step sizes for GPPA and pDCAe are $0.8/\lambda_{\max}(A^T A)$ and $1/\lambda_{\max}(A^T A)$, respectively, where $\lambda_{\max}(M)$ is the maximum eigenvalue of a symmetric matrix M . We set $\gamma = 0.1$ and run all algorithms, initialized at the origin, for a maximum of 3000 iterations. Note

that $\beta = 0$ (since g is convex) and $\ell = 1$ (since $\nabla\varphi(z) = z - b$). For our proposed algorithm, $\delta = 5 \times 10^{-25}$, $\bar{\lambda} = 0.1$, $\bar{\mu} = 0.01$, $\tau_n = 1 / (2\delta + \ell\|A\|^2 (2\bar{\lambda} + 1) + 2\bar{\mu})$ with $\|A\|$ being spectral norm, and

$$\lambda_n = \bar{\lambda} \frac{\kappa_{n-1} - 1}{\kappa_n}, \quad \mu_n = \bar{\mu} \tau_n \frac{\kappa_{n-1} - 1}{\kappa_n},$$

where $\kappa_{-1} = \kappa_0 = 1$ and $\kappa_{n+1} = \frac{1 + \sqrt{1 + 4\kappa_n^2}}{2}$. Here, we adopt the well-known restarting techniques (see, for example, [17, Chapter 10]) and reset $\kappa_{n-1} = \kappa_n = 1$ every 50 iterations. Note that this technique has been utilized in several existing studies such as [14, 23]. We generate the vector b based on the same method as in [2]. In generating the matrix A , we use both randomly generated Gaussian matrices and discrete cosine transform (DCT) matrices. For each case, we consider different matrix sizes of $m \times d$ with sparsity level s as given in Table 1. For the ground truth sparse vector x_g , a random index set is generated and non-zero elements are drawn following the standard normal distribution. The stopping condition for all algorithms is $\frac{\|x_{n+1} - x_n\|}{\|x_n\|} < 10^{-8}$.

In Table 2, we report the CPU time, the number of iteration, and the function values at termination, the error to the ground truth at termination, averaged over 30 random instances. It can be observed that since Step 2 involves the calculation of matrix multiplication, the CPU time is significantly increased with the increasing dimension of the matrices. In addition, in terms of running time, objective function values, the number of iterations used, and the error with respect to the ground truth solution (defined as $\frac{\|x_{n+1} - x_g\|}{\|x_g\|}$), our proposed algorithm outperforms ADMM and GPPA in all test cases. Our algorithm also appears to be comparable to pDCAe. Note that our algorithm can be applied to a more general framework than pDCAe (see the next numerical experiment with the Lorentzian norm loss function for an illustration).

Next, we consider the case of Lorentzian norm loss function by letting $\varphi(z) = \|z - b\|_{LL_{2,1}}$. Lorentzian norm can be useful in robust sparse signal reconstruction [6]. In this case, the optimization problem (7) becomes

$$\min_{x \in \mathbb{R}^d} (\|Ax - b\|_{LL_{2,1}} + \gamma(\|x\|_1 - \|x\|)). \tag{9}$$

Table 1 Test cases for Case Study 4.1

Matrix type	Case	m	d	s
Gaussian	1	180	640	20
	2	360	1280	40
	3	720	2560	80
	4	2880	10240	320
	5	180	640	20
DCT	6	360	1280	40
	7	720	2560	80
	8	2880	10240	320

Table 2 Results of 30 random generated instances for 8 test cases - least square loss function

Case	CPU time (seconds)				Iteration				Error vs ground truth			
	ADMM	GPPA	pDCAe	Proposed	ADMM	GPPA	pDCAe	Proposed	ADMM	GPPA	pDCAe	Proposed
1	0.15	0.02	0.02	0.02	1803	487	274	406	5.739E-04	3.702E-07	3.505E-07	2.987E-07
2	0.42	0.15	0.12	0.14	1583	449	292	325	2.268E-04	3.340E-07	1.095E-07	2.316E-07
3	2.92	1.20	0.87	0.84	1471	417	300	298	2.059E-04	3.039E-07	6.742E-08	2.132E-07
4	61.06	17.77	15.54	14.13	1415	380	311	279	1.893E-04	2.717E-07	4.937E-08	1.913E-07
5	0.05	0.01	0.01	0.01	612	157	121	112	7.222E-05	9.690E-08	8.409E-08	8.026E-08
6	0.17	0.07	0.06	0.06	627	186	128	112	7.095E-05	2.345E-05	5.180E-08	6.383E-08
7	1.19	0.50	0.36	0.31	634	170	131	113	7.116E-05	1.777E-06	4.240E-08	6.303E-08
8	29.92	8.11	7.49	6.47	721	181	155	133	6.908E-05	3.984E-05	2.742E-08	6.092E-08

The bold entries indicate the better values compared to the remaining values

Table 3 Results of 30 random generated instances for 8 test cases - Lorentzian norm loss function

Case	CPU time (seconds)		Iteration		Error vs ground truth	
	GPPA	Proposed	GPPA	Proposed	GPPA	Proposed
1	0.36	0.30	2104	1720	2.865E-03	2.863E-03
2	2.94	2.44	2282	1870	4.043E-03	3.132E-03
3	17.15	14.06	2369	1936	3.168E-03	3.166E-03
4	277.46	225.60	2438	1993	3.356E-03	3.354E-03
5	0.36	0.30	2148	1765	1.425E-03	1.416E-03
6	3.00	2.47	2347	1922	2.134E-03	1.169E-03
7	16.85	13.59	2334	1908	1.213E-03	1.205E-03
8	260.61	220.55	2440	2064	2.269E-03	2.261E-03

The bold entries indicate the better values compared to the remaining values

We note that

$$\nabla\varphi(z) = \left(\frac{2(z_1 - b_1)}{1 + |z_1 - b_1|^2}, \dots, \frac{2(z_m - b_m)}{1 + |z_m - b_m|^2} \right)^T.$$

is Lipschitz continuous with modulus $\ell = 2$. Since the loss function is now nonconvex and the pDCAe algorithm in [14] requires a convex loss function, pDCAe is not applicable in this case. Moreover, the ADMM algorithm in [2] is also not directly applicable due to the presence of the Lorentzian norm. Therefore, we compare our method with the GPPA only. For GPPA, we let $h(x) = \varphi(Ax)$. The step size for GPPA is $\tau = 0.8/(2\lambda_{\max}(A^T A))$. For this case, we set $\gamma = 0.001$ and run the GPPA and our proposed algorithm, which are both initialized at the origin, for a maximum of 4000 iterations. The remaining parameters of our algorithm are set to the same values as before. We also use 30 random instances of the previous 8 test cases. The results are presented in Table 3. It can be seen from Table 3 that the proposed algorithm outperforms GPPA in this case.

4.2 Optimal power flow considering photovoltaic systems placement

Optimal power flow (OPF) is a well-known problem in power system engineering [38]. The integration of many distributed energy resources (DERs) such as photovoltaic systems, has become increasingly popular in modern smart grid [39], leading to the needs of developing more complicated OPF models considering the DERs. Metaheuristic algorithms are popular in solving OPF, and they have also been applied to solve the OPF with DERs integration [40, 41]. However, the drawbacks of the metaheuristic algorithms are that the convergence proof cannot be established, and their performances are not consistent [42]. Difference-of-convex programming has also been successfully applied to solve the OPF problem in [43], although DERs are not considered. Motivated by the aforementioned results, in this work we try to apply our proposed algorithm to solve the OPF in a low voltage network, which includes optimizing the placement of photovoltaic (PV) systems. We formulate two models

which are based on the Direct Current OPF (DC OPF) [44], and Alternating Current OPF (AC OPF) [45]. To the best of the authors’ knowledge, this is the first time a proximal algorithm is used to solve an DER-integrated OPF with a difference-of-convex formulation, considering PV systems placement. The objective function aims at minimizing the cost of the conventional generator, which is a diesel generator in this case study, while maximizing the PV-penetration, which is defined as the ratio of the power generated by the PV systems divided by the total demand. The network considered in this case study is illustrated in Fig. 1, which consists of 14 buses. This case study is taken from a real low voltage network in Victoria, Australia. Currently, there are loads at buses 1, 3, 4, 6, 8, 9, 13, and 14. There are 6 PV systems at buses 1, 2, 4, 5, 7, and 8 with a capacity of 800 kW. A 5000 kW diesel generator is connected to bus 11. All of the parameters and decision variables in this case study are presented in Table 6. The cost of the current situation (before optimization is performed) is based on the cost of active power withdrawn from the generator, plus the installation cost of the PV systems. To determine this initial cost, the amount of active power generated by the generator is determined via DlgSILENT Power Factory 2021. After that, the cost of active power is calculated by the expression $\sum_{i \in M} (a(P_i^G)^2 + bP_i^G + c)$, plus the installation cost of the six PV systems.

We first formulate the OPF problem with PV, which is based on the DC OPF, as follows

$$\min \left(\sum_{i \in N} CX_i + \sum_{i \in M} (a(P_i^G)^2 + bP_i^G + c) - \frac{\sum_{i \in N} P_i^{PV}}{\sum_{i \in N} D_i} \right) \tag{10a}$$

$$\text{subject to } P_{ij} = b_{ij}(\theta_i - \theta_j), \quad \forall i, j \in N \tag{10b}$$

$$\theta_{11} = 0 \tag{10c}$$

$$\sum_{j \in N, j \neq i} P_{ij} = P_i^{PV} + P_i^G - D_i, \quad \forall i \in M \tag{10d}$$

$$\sum_{j \in N, j \neq i} P_{ij} = P_i^{PV} - D_i, \quad \forall i \in N \setminus M \tag{10e}$$

$$\frac{\sum_{i \in N} P_i^{PV}}{\sum_{i \in N} D_i} \geq 0.5 \tag{10f}$$

$$|P_{ij}| \leq \bar{P}, \quad \forall i, j \in N \tag{10g}$$

$$0 \leq P_i^{PV} \leq X_i \bar{P}^{PV}, \quad \forall i \in N \tag{10h}$$

$$0 \leq P_i^G \leq \bar{P}^G, \quad \theta_i \in [0, 2\pi], \quad \forall i \in M \tag{10i}$$

$$X_i \in \{0, 1\}, \quad \forall i \in N. \tag{10j}$$

We see that for any $i \in N$, if $X_i \in [0, 1]$, then $X_i - X_i^2 = X_i(1 - X_i) \geq 0$. Therefore,

$$\begin{aligned} & (\forall i \in \{1, \dots, N\}, \quad X_i \in \{0, 1\}) \\ & \iff (\forall i \in \{1, \dots, N\}, \quad X_i \in [0, 1] \text{ and } \sum_{i \in N} (X_i^2 - X_i) \geq 0). \end{aligned}$$

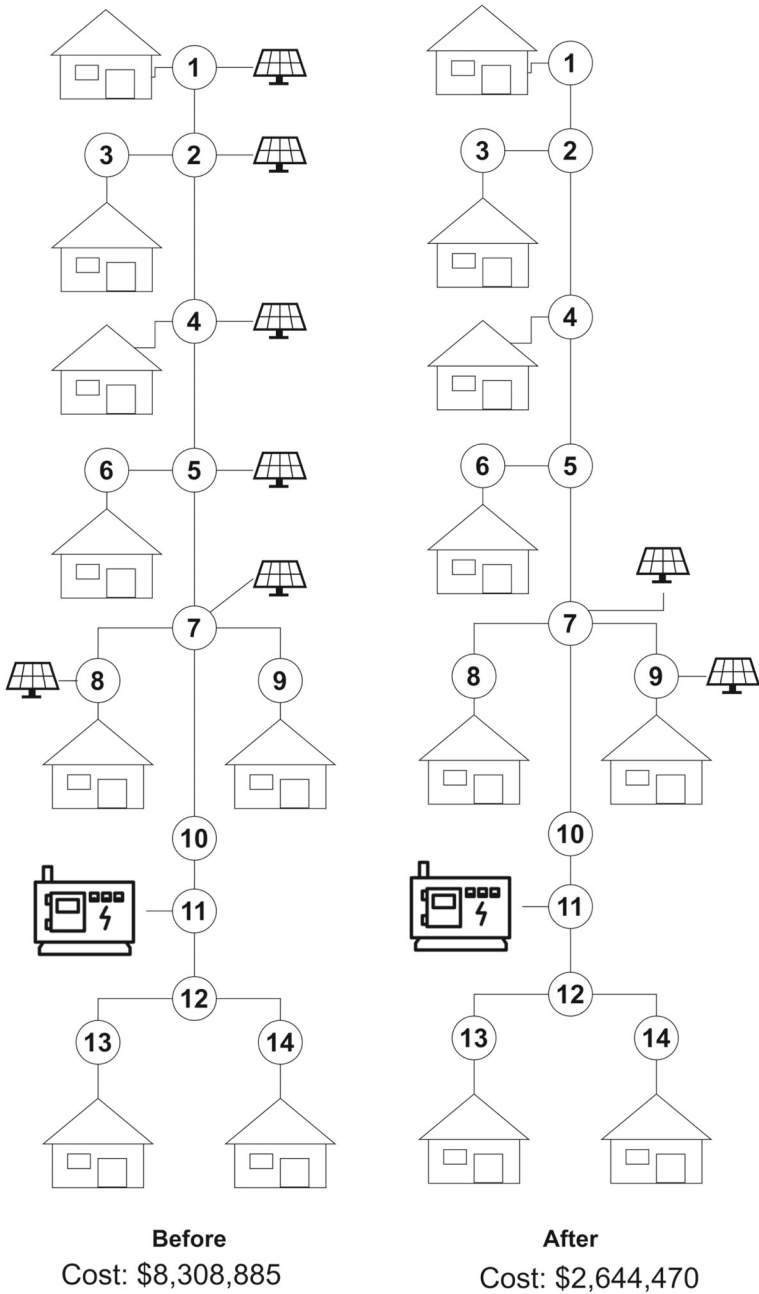


Fig. 1 Best solution found in Case Study 4.2, together with the total cost

Taking into account of the above equivalence, a plausible alternative optimization model for the OPF problem with PV is as follows a

$$\min \left(\sum_{i \in N} CX_i + \sum_{i \in M} \left(a(P_i^G)^2 + bP_i^G + c \right) - \frac{\sum_{i \in N} P_i^{PV}}{\sum_{i \in N} D_i} - \gamma \sum_{i \in N} (X_i^2 - X_i) \right) \tag{11a}$$

subject to (10b) \rightarrow (10i) (11b)

$$X_i \in [0, 1], \quad \forall i \in N. \tag{11c}$$

The objective function (11a) aims at minimizing the installation cost and the generation cost of the diesel generator and maximizing the PV penetration, which is defined as $\frac{\sum_{i \in N} P_i^{PV}}{\sum_{i \in N} D_i}$ [46], the parameter $\gamma > 0$ which serves as a Lagrangian multiplier for the discrete constraints $X_i \in \{0, 1\}$.

With this reformulation, the objective function (11a) now becomes a difference-of-convex function. Constraint (10b) describes the relationship between the power flow from one bus to another and their corresponding phasor angles, constraint (10c) defines the voltage angle at the slack bus, which is the bus connected to the diesel generator, constraints (10d) and (10e) define the power flow in and out of any buses, constraint (10f) ensures that the PV penetration rate is at least 50 percent, constraint (10g) defines the transmission limits of the transmission lines, and constraint (10h) makes sure that the solar power only exists at a bus when there is a PV system at that bus. Finally, constraint (10i) defines the boundaries of the remaining decision variables. All of the constraints form the feasible set S . This problem takes the form of (P) with $f = \iota_S, h = \sum_{i \in N} CX_i + \sum_{i \in M} (a(P_i^G)^2 + bP_i^G + c) - \frac{\sum_{i \in N} P_i^{PV}}{\sum_{i \in N} D_i}$, and $g = \gamma \sum_{i \in N} (X_i^2 - X_i)$. By Remark 3.4 (ii) and Theorem 3.3, in this case the proposed algorithm converges with a linear rate. The update of x_{n+1} in Algorithm 1 becomes

$$x_{n+1} = \underset{x \in S}{\operatorname{argmin}} \|x - v_n + \tau_n \nabla h(u_n) - \tau_n \nabla g(x_n)\|^2.$$

Here,

$$x = [P_1^{PV}, \dots, P_{14}^{PV}, P_{11}^G, X_1, \dots, X_{14}, \theta_1, \dots, \theta_{14}, P_{1,1}, \dots, P_{1,14}, \dots, P_{14,1}, \dots, P_{14,14}]^T.$$

This step is solved by MATLAB’s `quadprog` command. Noting that $\beta = 0$ (since g is convex) and $\ell = 2a$, the parameters are set as follows: $\delta = 5 \times 10^{-25}, \bar{\lambda} = 0.1, \bar{\mu} = 0.01, \tau_n = 1 / (2\delta + \ell (2\bar{\lambda} + 1) + 2\bar{\mu}), \mu_n = \bar{\mu}\tau_n$, and λ_n is chosen in the same way as in Section 4.1. The performance of the proposed algorithm is compared with the the GPPA, and the pDCAe, as illustrated in Table 4. We use the step size $\tau_n = 0.8/\ell$ for GPPA, and $\tau_n = 1/\ell$ for pDCAe. The maximum number of iteration is 1000, and the stopping condition is the same as the one used in Section 4.1.

We test all algorithms for 30 times, at each time we use a random starting point between the upper bound and the lower bound of the variables. The mean objective

Table 4 Comparison of GPPA, pDCAe, and the proposed algorithm on 30 runs of the DC OPF model

Algorithm	GPPA	pDCAe	Proposed
Mean objective function value	3.724581	3.719692	3.706267
Best objective function value	1.920925	1.920924	1.920922
Mean iteration number	3	4	5
Mean CPU time (seconds)	0.08	0.11	0.12

The bold entries indicate the better values compared to the remaining values

function values, and the best objective function values found by all algorithms are reported in Table 4. Although the proposed algorithm, on average, needs more iterations than the remaining ones, it can find a better solution. The mean objective function value found by our algorithm is also better than the ones found by the other algorithms.

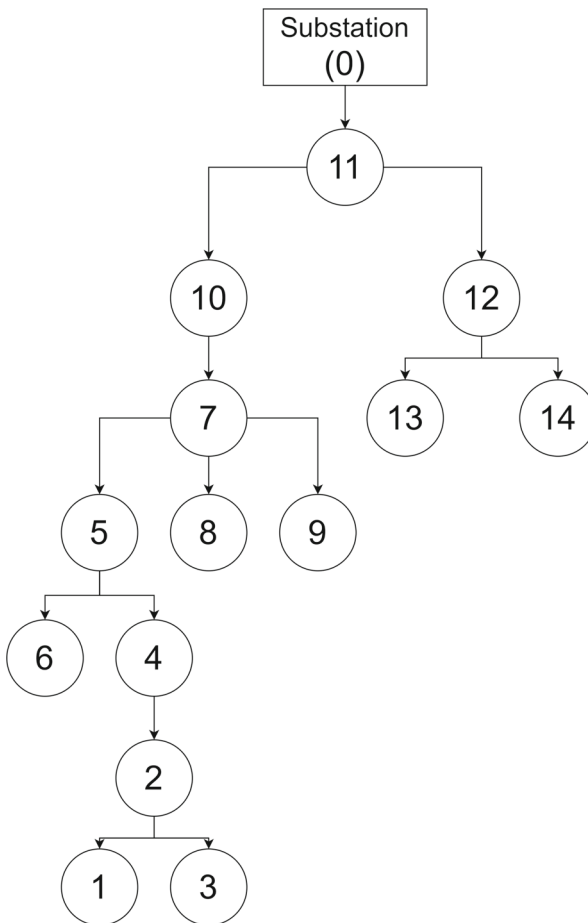


Fig. 2 Directed graph representation of the network

Our algorithm is also comparable to the GPPA and the pDCAe in terms of average CPU time.

The details of the best solution found by our algorithm are shown in Fig. 1.

Now we consider the case of AC OPF model. The formulation is based on the *branch flow model* given in [45]. Firstly, the network is treated as a directed graph, as shown in Fig. 2.

We denote a directed link by (i, j) or $i \rightarrow j$ if it points from bus i to bus j , and the set of all directed links by E . Next, the formulation is given as follows,

$$\min \left(\sum_{i \in N} CX_i + \sum_{i \in M} (a(P_i^G)^2 + bP_i^G + c) - \frac{\sum_{i \in N} P_i^{PV}}{\sum_{i \in N} D_i} - \gamma \sum_{i \in N} (X_i^2 - X_i) \right) \quad (12a)$$

$$\text{s.t. } \hat{I}_{ij} = |I_{ij}|^2, \quad \forall (i, j) \in E \quad (12b)$$

$$v_i = |V_i|^2, \quad \forall i \in N \quad (12c)$$

$$P_{0,11} = Q_{0,11} = 0 \quad (12d)$$

$$P_{ij} + P_j^{PV} + P_j^G - D_j = \sum_{k \in N: j \rightarrow k} P_{jk}, \quad \forall (i, j) \in E, j \in M \quad (12e)$$

$$Q_{ij} + Q_j^{PV} + Q_j^G - D_j^Q = \sum_{k \in N: j \rightarrow k} Q_{jk}, \quad \forall (i, j) \in E, j \in M \quad (12f)$$

$$P_{ij} + P_j^{PV} - r_{ij} \hat{I}_{ij} - D_j = \sum_{k \in N: j \rightarrow k} P_{jk}, \quad \forall (i, j) \in E, j \notin M \quad (12g)$$

$$Q_{ij} + Q_j^{PV} - \chi_{ij} \hat{I}_{ij} - D_j^Q = \sum_{k \in N: j \rightarrow k} Q_{jk}, \quad \forall (i, j) \in E, j \notin M \quad (12h)$$

$$\frac{\sum_{j \in N} P_j^{PV}}{\sum_{j \in N} D_j} \geq 0.5 \quad (12i)$$

$$0 \leq P_j^{PV} \leq X_j \overline{P^{PV}}, \quad \forall j \in N \quad (12j)$$

$$0 \leq Q_j^{PV} \leq X_j \overline{Q^{PV}}, \quad \forall j \in N \quad (12k)$$

$$v_j = v_i - 2(r_{ij} P_{ij} + \chi_{ij} Q_{ij}) + (r_{ij}^2 + \chi_{ij}^2) \hat{I}_{ij}, \quad \forall (i, j) \in E \quad (12l)$$

$$\hat{I}_{ij} v_i = P_{ij}^2 + Q_{ij}^2, \quad \forall (i, j) \in E \quad (12m)$$

$$\underline{V}^2 \leq v_i \leq \overline{V}^2, \quad \forall i \in N \quad (12n)$$

$$\underline{I}^2 \leq \hat{I}_{ij} \leq \overline{I}^2, \quad \forall (i, j) \in E \quad (12o)$$

$$|P_{ij}| \leq \overline{P}, \quad \forall i, j \in E \quad (12p)$$

$$|Q_{ij}| \leq \overline{Q}, \quad \forall i, j \in E \quad (12q)$$

$$0 \leq P_j^G \leq \overline{P^G}, \quad \forall j \in M \quad (12r)$$

$$0 \leq Q_j^G \leq \overline{Q^G}, \quad \forall j \in M \quad (12s)$$

$$X_j \in [0, 1], \quad \forall j \in N \quad (12t)$$

Table 5 Comparison of GPPA and the proposed algorithm on 30 runs of the AC OPF model

Algorithm	GPPA	Proposed
Mean objective function value	3.492971	3.416897
Best objective function value	1.920924	1.920923
Mean iteration number	33	20
Mean CPU time (seconds)	152.69	109.20

The bold entries indicate the better values compared to the remaining values

The main differences between the AC OPF model and the DC OPF model are that the AC OPF model has a nonconvex feasible set, and that it also accounts for the loss in the network as well as the reactive power. Consequently, AC OPF is more accurate than DC OPF in practice [47], and due to its nonconvexity, it is also more challenging to solve [48]. Constraints (12e) → (12h) define the power flow in any directed links. Constraint (12i) ensures that the PV penetration rate is at least 50 percent. Constraints (12j) and (12k) ensure that the active and reactive power from PV systems only exist at a bus if and only if there is a PV system at that bus. Constraint (12l) describes the relationship between the voltage of any two bus in a directed link. Constraint (12m) is a nonconvex constraint ensuring that the solution have physical meaning. Finally, constraints (12n) → (12t) define the boundaries of the decision variables. The update of x_{n+1} is also the same as before. For this case,

$$x = [P_{11}^G, Q_{11}^G, v_1, \dots, v_{14}, \hat{I}_{0,11}, \dots, \hat{I}_{2,1}, P_{0,11}, \dots, P_{2,1}, Q_{0,11}, \dots, Q_{2,1}, P_1^{PV}, \dots, P_{14}^{PV}, Q_1^{PV}, \dots, Q_{14}^{PV}, X_1, \dots, X_{14}]^T.$$

We also perform the same numerical experiment as in the DC OPF case. However, the pDCAe is not applicable in this case, so we compare our algorithm with the GPPA only. The parameters of GPPA and our proposed algorithm are set to the same values as those used for the DC OPF model. Due to the nonconvex constraint, MATLAB’s `fmincon` is used to solve the subproblem in Step 2 instead of `quadprog`. The results are shown in Table 5.

Table 5 shows that our proposed algorithm takes less time and fewer iterations than the GPPA to converge. The best solution found by our algorithm in this case is also the same as the one found in the DC OPF model.

It can be seen that for both DC OPF and AC OPF, two PV systems need to be installed at bus 7 and bus 9, the remaining demands can be supplied by the generator, and the demands are satisfied by the power flows. Although the mathematical model aims at maximizing the PV penetration, drawing power from the diesel generator is still more economical due to the high installation cost of the PV systems. The solution significantly reduces the cost by approximately 70% from the original situation. This can serve as a proof of concept for future research.

5 Conclusion

In this paper, we proposed an extrapolated proximal subgradient algorithm for solving a broad class of structured nonconvex and nonsmooth optimization problems. Our

general framework and the proposed algorithm impose less restrictions on the smoothness and convexity requirements than the current literature, and allow us to tackle problems with specific nonconvex loss functions and nonconvex constraints. In addition, our choice of the extrapolation parameters is flexible enough to cover the popular choices used in restarted FISTA scheme. The convergence of the whole sequence generated by our algorithm was proved via the abstract convergence framework given in [23]. We then performed numerical experiments on a least squares problem with the nonconvex $L_1 - L_2$ regularization, and on a compressed sensing problem with the nonconvex Lorentzian norm loss function. In the numerical experiments, the proposed algorithm exhibited very competitive performance, and, in various instances, outperformed the existing algorithms in terms of time and the quality of the solutions. We also applied this algorithm to solve an OPF problem considering PV placement, which serves as a proof of concept for future works.

Author Contributions All authors contributed to the manuscript and approved the submitted version.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. The research of TNP was supported by Henry Sutton PhD Scholarship Program from Federation University Australia. The research of MND benefited from the FMJH Program Gaspard Monge for optimization and operations research and their interactions with data science, and was supported by a public grant as part of the Investissement d'avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH. The research of GL was supported by Discovery Project 190100555 from the Australian Research Council.

Data Availability All data generated or analyzed during this study are included in this article. In particular, the data for Case study 4.1 were generated randomly and we explained how they were explicitly generated. The data for Case study 4.2 are available in the [Appendix](#).

Declarations

Ethics approval Not applicable.

Conflict of interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendix: Data of Case study 4.2

In Case study 4.2, we use a base power of 100 MVA, and a base voltage of 22 kV. All of the parameters are converted into Per Unit (pu) values in the calculation. Readers can refer to [49, Chapter 2] for a detailed tutorial on the Per Unit system. The active and reactive power demand are given in Table 7. The other technical parameters of the system including susceptance, resistance, and reactance of the lines are given in Tables 8, 9, and 10, respectively.

Table 6 Parameters and variables of case study 4.2

Parameters	Description	Values
N	Set of buses	{1, 2, ..., 14}
M	Set of buses that are connected to diesel generators, $M \subseteq N$	{11}
E	Set of directed links	{(0, 11), (11, 10), ..., (2, 1)}
D_i	Active power demand at bus i	See Table 7
D_i^Q	Reactive power demand at bus i	See Table 7
b_{ij}	Susceptance value of the line connecting bus i and bus j	See Table 8
r_{ij}	Resistance value of the line connecting bus i and bus j	See Table 9
X_{ij}	Reactance value of the line connecting bus i and bus j	See Table 10
C	Unit installation cost of a PV at bus i	1 (1 unit = \$1,040,000)
a, b, c	Coefficients associated with the cost of diesel generator. These coefficients for a diesel generator are derived from [50, 51]	0.246, 0.084, 0.433
P^{PV}	Active power capacity of PVs	800 kW (0.008 pu)
Q^{PV}	Reactive power capacity of PVs	300 kW (0.003 pu)
P^G	Active power capacity of diesel generator	5000 kW (0.05 pu)
Q^G	Reactive power capacity of diesel generator	3000 kW (0.03 pu)
\bar{P}, \bar{Q}	Transmission limits of lines	3000 kW (0.03 pu)
\bar{V}, \underline{V}	Voltage limits	1.05 pu, 0.95 pu
\bar{I}, \underline{I}	Current limits	2 pu, 0 pu
γ	Relaxation parameter	1

Table 6 continued

Parameters	Description	Values
Variables		
P_i^{PV}	Active power generated by a PV system at bus i , $i \in N$	
Q_i^{PV}	Reactive power generated by a PV system at bus i , $i \in N$	
P_i^G	Active power generated by diesel generator at bus i , $i \in M$	
Q_i^G	Reactive power generated by diesel generator at bus i , $i \in M$	
X_i	1 if there is a PV system needed at bus i , and 0 otherwise, $i \in N$	
V_i	Nodal voltage of bus i , $i \in N$	
I_{ij}	Current between bus i and bus j , $i, j \in N$, $i \neq j$	
θ_i	Voltage angle of bus i , $i \in N$	
P_{ij}	Active power flow between bus i and bus j , $i, j \in N$, $i \neq j$	
Q_{ij}	Reactive power flow between bus i and bus j , $i, j \in N$, $i \neq j$	

Table 7 Active and Reactive power demand

Bus	1	2	3	4	5	6	7	8	9	10	11	12	13	14
D_i	7.91E-03	0	2.81E-03	3.40E-03	0	3.05E-03	0	3.32E-03	5.90E-03	0	0	0	2.12E-03	2.64E-03
D_i^Q	1.98E-03	0	7.04E-03	8.51E-03	0	7.64E-04	0	8.32E-03	1.48E-03	0	0	0	5.32E-03	6.63E-04

Table 8 Susceptance b_{ij}

Bus	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1		-9.98E+02	9.98E+02	0	0	0	0	0	0	0	0	0	0	0
2			-2.60E+03	4.97E+02	1.11E+03	0	0	0	0	0	0	0	0	0
3				4.97E+02	-4.97E+02	0	0	0	0	0	0	0	0	0
4					1.11E+03	-4.35E+03	3.24E+03	0	0	0	0	0	0	0
5						3.24E+03	-4.79E+03	5.72E+02	9.77E+02	0	0	0	0	0
6							5.72E+02	-5.72E+02	0	0	0	0	0	0
7								9.77E+02	-4.00E+03	6.92E+02	9.26E+02	1.41E+03	0	0
8									6.92E+02	-6.92E+02	0	0	0	0
9										9.26E+02	-9.26E+02	0	0	0
10											1.41E+03	-2.27E+03	8.64E+02	0
11												8.64E+02	-3.85E+03	2.99E+03
12													2.99E+03	-7.10E+03
13														2.08E+03
14														

Table 9 Resistance r_{ij}

Bus	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	5.01E-04	0	0	0	0	0	0	0	0	0	0	0	0
2	5.01E-04	0	1.01E-03	4.51E-04	0	0	0	0	0	0	0	0	0	0
3	0	1.01E-03	0	0	0	0	0	0	0	0	0	0	0	0
4	0	4.51E-04	0	0	1.54E-04	0	0	0	0	0	0	0	0	0
5	0	0	0	1.54E-04	0	8.75E-04	5.12E-04	0	0	0	0	0	0	0
6	0	0	0	0	8.75E-04	0	0	0	0	0	0	0	0	0
7	0	0	0	0	5.12E-04	0	0	7.23E-04	5.40E-04	3.56E-04	0	0	0	0
8	0	0	0	0	0	0	7.23E-04	0	0	0	0	0	0	0
9	0	0	0	0	0	0	5.40E-04	0	0	0	0	0	0	0
10	0	0	0	0	0	0	3.56E-04	0	0	0	5.79E-04	0	0	0
11	0	0	0	0	0	0	0	0	0	5.79E-04	0	1.67E-04	0	0
12	0	0	0	0	0	0	0	0	0	0	1.67E-04	0	2.40E-04	2.46E-04
13	0	0	0	0	0	0	0	0	0	0	0	2.40E-04	0	0
14	0	0	0	0	0	0	0	0	0	0	0	2.46E-04	0	0

Table 10 Reactance X_{ij}

Bus	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	5.01E-04	0	0	0	0	0	0	0	0	0	0	0	0
2	5.01E-04	0	1.01E-03	4.51E-04	0	0	0	0	0	0	0	0	0	0
3	0	1.01E-03	0	0	0	0	0	0	0	0	0	0	0	0
4	0	4.51E-04	0	0	1.54E-04	0	0	0	0	0	0	0	0	0
5	0	0	0	1.54E-04	0	8.75E-04	5.12E-04	0	0	0	0	0	0	0
6	0	0	0	0	8.75E-04	0	0	0	0	0	0	0	0	0
7	0	0	0	0	5.12E-04	0	0	7.23E-04	5.40E-04	3.56E-04	0	0	0	0
8	0	0	0	0	0	0	7.23E-04	0	0	0	0	0	0	0
9	0	0	0	0	0	0	5.40E-04	0	0	0	0	0	0	0
10	0	0	0	0	0	0	3.56E-04	0	0	0	5.79E-04	0	0	0
11	0	0	0	0	0	0	0	0	0	5.79E-04	0	1.67E-04	0	0
12	0	0	0	0	0	0	0	0	0	0	1.67E-04	0	2.40E-04	2.46E-04
13	0	0	0	0	0	0	0	0	0	0	0	2.40E-04	0	0
14	0	0	0	0	0	0	0	0	0	0	0	2.46E-04	0	0

References



1. Cheng, Y., Pesavento, M.: Joint optimization of source power allocation and distributed relay beamforming in multiuser peer-to-peer relay networks. *IEEE Transactions on Signal Processing* **60**(6), 2962–2973 (2012)
2. Lou, Y., Yan, M.: Fast L1–L2 minimization via a proximal operator. *Journal of Scientific Computing* **74**(2), 767–785 (2017)
3. Le Thi, H.A., Pham Dinh, T.: DC programming and DCA: thirty years of developments. *Mathematical Programming* **169**(1), 5–68 (2018)
4. Wang, H., Shao, N.X.Y.: Proximal operator and optimality conditions for ramp loss svm. *Optimization Letters* **16**(3), 999–1014 (2022)
5. Xiao, Y., Wang, W.X.H.: Ramp loss based robust one-class svm. *Pattern Recognition Letters* **85**(1), 15–20 (2017)
6. Carrillo, R.E., Barner, T.C.A.K.E.: Robust sampling and reconstruction methods for sparse signals in the presence of impulsive noise. *IEEE Journal of Selected Topics in Signal Processing* **4**, 392–408 (2010)
7. Ahn, M., Pang, J., Xin, J.: Difference-of-convex learning: Directional stationarity, optimality, and sparsity. *SIAM Journal on Optimization* **27**(3), 1637–1665 (2017)
8. Antoniadis, A.: Wavelets in statistics: A review. *Journal of the Italian Statistical Society* **6**(2), 97–130 (1997)
9. Gotoh, J., Takeda, A., Tono, K.: DC formulations and algorithms for sparse optimization problems. *Mathematical Programming* **169**(1), 141–176 (2017)
10. Zhang, C.: Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics* **38**(2) (2010)
11. An, N.T., Nam, N.M.: Convergence analysis of a proximal point algorithm for minimizing differences of functions. *Optimization* **66**(1), 129–147 (2016)
12. Phan, D.N., Le, M.H., Le Thi, H.A.: Accelerated difference of convex functions algorithm and its application to sparse binary logistic regression. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence* (2018)
13. Liu, T., Takeda, A.: An inexact successive quadratic approximation method for a class of difference-of-convex optimization problems. *Computational Optimization and Applications* **82**, 141–173 (2022)
14. Wen, B., Chen, X., Pong, T.K.: A proximal difference-of-convex algorithm with extrapolation. *Computational Optimization and Applications* **69**(2), 297–324 (2017)
15. Lu, Z., Zhou, Z.: Nonmonotone enhanced proximal DC algorithms for a class of structured nonsmooth DC programming. *SIAM Journal on Optimization* **29**(4), 2725–2752 (2019)
16. Lu, Z., Zhou, Z., Sun, Z.: Enhanced proximal DC algorithms with extrapolation for a class of structured nonsmooth DC minimization. *Mathematical Programming* **176**(1), 369–401 (2018)
17. Beck, A.: *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization, vol. 25. Society for Industrial and Applied Mathematics, Philadelphia, USA (2017)
18. Polyak, B.T.: Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics* **4**(5), 1–17 (1964)
19. Nesterov, Y.: Inexact accelerated high-order proximal-point methods. *Mathematical Programming* **2021**, 1–26 (2021)
20. Nesterov, Y.: *Lectures on Convex Optimization*. Springer Optimization and Its Applications, vol. 137. Springer, Cham, Switzerland (2018)
21. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences* **2**(1), 183–202 (2009)
22. Attouch, H., Cabot, A.: Convergence rates of inertial forward-backward algorithms. *SIAM Journal on Optimization* **28**(1), 849–874 (2018)
23. Boţ, R.I., Dao, M.N., Li, G.: Extrapolated proximal subgradient algorithms for nonconvex and nonsmooth fractional programs. *Mathematics of Operations Research* **47**(3), 1707–2545 (2022)
24. Mordukhovich, B.S.: *Variational Analysis and Generalized Differentiation I*. Grundlehren der mathematischen Wissenschaften, vol. 330. Springer, Berlin, Heidelberg (2006)
25. Dao, M.N., Tam, M.K.: A Lyapunov-type approach to convergence of the Douglas-Rachford algorithm for a nonconvex setting. *Journal of Global Optimization* **73**(1), 83–112 (2019)
26. Xu, Z., Chang, X., Xu, F., Zhang, H.: L1/2 regularization: A thresholding representation theory and a fast solver. *IEEE Transactions on Neural Networks and Learning Systems* **23**(7), 1013–1027 (2012)

27. Afef, C., Émilie, C., Marc-André, D.: Proximity operators for a class of hybrid sparsity + entropy priors. Application to dosy NMR signal reconstruction. In: Proceedings of the 8th International Symposium on Signal, Image, Video and Communications (ISIVC), pp. 120–125 (2016)
28. Souza, J.C.O., Oliveira, P.R., Soubeyran, A.: Global convergence of a proximal linearized algorithm for difference of convex functions. *Optimization Letters* **10**(7), 1529–1539 (2015)
29. Rockafellar, R.T., J-B. Wets, R.: *Variational Analysis. Grundlehren der mathematischen Wissenschaften*, vol. 317. Springer, Berlin, Heidelberg (1998)
30. Kurdyka, K.: On gradients of functions definable in o-minimal structures. *Annales de l'institut Fourier* **48**(3), 769–783 (1998)
31. Lojasiewicz, S.: Une propriété topologique des sous-ensembles analytiques réels. *Les Équations aux Dérivées Partielles*, 87–89 (1963)
32. Attouch, H., Bolte, J.: On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Mathematical Programming* **116**(1–2), 5–16 (2007)
33. Li, G., Pong, T.K.: Calculus of the exponent of Kurdyka-Łojasiewicz inequality and its applications to linear convergence of first-order methods. *Foundations of Computational Mathematics* **18**(5), 1199–1232 (2017)
34. Boţ, R.I., Dao, M.N., Li, G.: Inertial proximal block coordinate method for a class of nonsmooth sum-of-ratios optimization problems. *SIAM Journal on Optimization* **33**(2), 361–393 (2023)
35. Bolte, J., Daniilidis, A., Lewis, A., Shiota, M.: Clarke subgradients of stratifiable functions. *SIAM Journal on Optimization* **18**(2), 556–572 (2007)
36. Nikolova, M.: Analysis of the recovery of edges in images and signals by minimizing nonconvex regularized least-squares. *Multiscale Modeling & Simulation* **4**(3), 960–991 (2005)
37. Kim, S., Koh, K., Lustig, M., Boyd, S., Gorinevsky, D.: An interior-point method for large-scale-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing* **1**(4), 606–617 (2007)
38. Abdi, H., Beigvand, S.D., Scala, M.L.: A review of optimal power flow studies applied to smart grids and microgrids. *Renewable and Sustainable Energy Reviews* **71**, 742–766 (2017)
39. Wankhede, S.K., Paliwal, P., Kirar, M.K.: Increasing penetration of DERs in smart grid framework: A state-of-the-art review on challenges, mitigation techniques and role of smart inverters. *Journal of Circuits, Systems and Computers* **29**(16), 2030014 (2020)
40. Shaheen, M.A.M., Hasanien, H.M., Mekhamer, S.F., Talaat, H.E.A.: Optimal power flow of power systems including distributed generation units using sunflower optimization algorithm. *IEEE Access* **7**, 109289–109300 (2019)
41. Khaled, U., Eltamaly, A.M., Beroual, A.: Optimal power flow using particle swarm optimization of renewable hybrid distributed generation. *Energies* **10**(7), 1013 (2017)
42. Ezugwu, A.E., Adeleke, O.J., Akinyelu, A.A., Viriri, S.: A conceptual comparison of several meta-heuristic algorithms on continuous optimisation problems. *Neural Computing and Applications* **32**(10), 6207–6251 (2019)
43. Merkli, S., Domahidi, A., Jerez, J.L., Morari, M., Smith, R.S.: Fast AC power flow optimization using difference of convex functions programming. *IEEE Transactions on Power Systems* **33**(1), 363–372 (2018)
44. Kargarian, A., Mohammadi, J., Guo, J., Chakrabarti, S., Barati, M., Hug, G., Kar, S., Baldick, R.: Toward distributed/decentralized DC optimal power flow implementation in future electric power systems. *IEEE Transactions on Smart Grid* **9**(4), 2574–2594 (2018)
45. Farivar, M., Low, S.H.: Branch flow model: Relaxations and convexification-part I. *IEEE Transactions on Power Systems* **28**(3), 2554–2564 (2013)
46. Hoke, A., Butler, R., Hambrick, J., Kroposki, B.: Steady-state analysis of maximum photovoltaic penetration levels on typical distribution feeders. *IEEE Transactions on Sustainable Energy* **4**(2), 350–357 (2013)
47. Frank, S., Rebenack, S.: An introduction to optimal power flow: Theory, formulation, and examples. *IIE Transactions* **48**(12), 1172–1197 (2016)
48. Low, S.H.: Convex relaxation of optimal power flow-part I: Formulations and equivalence. *IEEE Transactions on Control of Network Systems* **1**(1), 15–27 (2014)
49. Weedy, B.M., Cory, B.J., Jenkins, N., Ekanayake, J.B., Strbac, G.: *Electric Power Systems*, 5th edn. Wiley-Blackwell, Hoboken, NJ (2012)
50. Kusakana, K.: Optimal scheduled power flow for distributed photovoltaic/wind/diesel generators with battery storage system. *IET Renewable Power Generation* **9**(8), 916–924 (2015)

51. Fodhil, F., Hamidat, A., Nadjemi, O.: Potential, optimization and sensitivity analysis of photovoltaic-diesel-battery hybrid energy system for rural electrification in algeria. *Energy* **169**, 613–624 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Tan Nhat Pham^{1,3} · Minh N. Dao²  · Rakibuzzaman Shah³ ·
Nargiz Sultanova¹ · Guoyin Li⁴  · Syed Islam³

Tan Nhat Pham
pntan.iac@gmail.com

Rakibuzzaman Shah
m.shah@federation.edu.au

Nargiz Sultanova
n.sultanova@federation.edu.au

Guoyin Li
g.li@unsw.edu.au

Syed Islam
s.islam@federation.edu.au

- ¹ Centre for Smart Analytics, Federation University Australia, Ballarat, VIC 3353, Australia
- ² School of Science, RMIT University, Melbourne, VIC 3000, Australia
- ³ Centre for New Energy Transition Research, Federation University Australia, Ballarat, VIC 3353, Australia
- ⁴ Department of Applied Mathematics, University of New South Wales, Sydney, NSW 2052, Australia