Supplemental Material for *an efficient penalty decomposition algorithm for minimization over sparse symmetric sets*

Ahmad Mousavi
Department of Mathematics and Statistics, American University
Washington, DC, USA
email: mousavi@american.edu

Morteza Kimiaei
Fakultät für Mathematik, Universität Wien
Oskar-Morgenstern-Platz 1, A-1090 Wien, Austria
email: kimiaeim83@univie.ac.at
WWW: `http://www.mat.univie.ac.at/~kimiaei`

Saman Babaie–Kafaki
Faculty of Engineering, Free University of Bozen–Bolzano
Bolzano 39100, Italy
email: saman.babaiekafaki@unibz.it

Vyacheslav Kungurtsev
Department of Computer Science, Czech Technical University
Karlovo Namesti 13, 121 35 Prague 2, Czech Republic
email: vyacheslav.kungurtsev@fel.cvut.cz

January 17, 2026

Contents

1 Introduction

The supplementary document provides several technical results and algorithmic details that support the main paper [8]:

Section 2 lists explicit stationarity conditions for common convex sets, obtained as direct consequences of the general variational inequality.

Section 3 reviews basic feasibility and L-stationarity under symmetry assumptions, summarizes key results of Beck and Hallak [3], and includes a practical criterion for verifying basic feasibility using a single super support set.

Section 4 contains the proof of Lemma 1 [8], establishing the cone–continuity property for convex symmetric sets, which is a key technical ingredient in our convergence analysis and allows us to pass from approximate to limiting stationarity in symmetric sparse optimization problems.

Section 5 provides the full proof of Theorem 1 [8], organized into six steps and relying on standard calculus rules for Fréchet normal cones.

Section 6 compares basic feasibility with the different CC-stationarity notions and clarifies their position within the stationarity hierarchy.

Section 7 describes practical enhancements of our algorithm, called PD-QN, including an improved line search and three diagonal Hessian approximations.

Section 8 recalls sparse projection algorithms for symmetric sets and explains their role in the analysis and in the generation of a challenging benchmark suite of 30 cardinality-constrained problems with dimensions ranging from $n = 10$ to $n = 500$. Finally, Section 9 reports additional numerical comparisons among variants of PD-QN.

For ease of reference, we restate the optimization model that appears throughout this supplementary material. We consider the cardinality–constrained problem

$$\min_{x \in C \cap C_s} f(x), \tag{CCOP}$$

where

$$C_s := \{x \in \mathbb{R}^n \mid \|x\|_0 \le s\}, \tag{1}$$

and $C \subseteq \mathbb{R}^n$ is a closed and convex set that encodes additional structure. Typical choices for C include the full space, the nonnegative orthant, the simplex, ℓ_p balls, and box constraints. Many of these sets possess symmetry with respect to coordinate permutations or sign changes, and this symmetry plays a central role in the sparse projection properties and optimality conditions developed in the sections that follow.

Throughout this supplemental material, let $g(x) := \nabla f(x)$ and $[n] := \{1, 2, \dots, n\}$. Let us also recall the following definitions from the main paper. A permutation $\pi \in \tilde{\mathfrak{S}}_n$ is called a **sorting permutation** of a vector $x \in \mathbb{R}^n$ if its entries are rearranged in non-increasing order, that is,

$$x_{\pi(1)} \ge x_{\pi(2)} \ge \cdots \ge x_{\pi(n)}.$$

Here, $\tilde{\mathfrak{S}}_n$ denotes the sorting permutation group over the index set $[n]$. For any permutation $\pi \in \tilde{\mathfrak{S}}_n$, we define

$$S^{\pi}_{[j_1, j_2]} = \begin{cases} \{\pi(j_1), \pi(j_1+1), \dots, \pi(j_2)\}, & \text{if } 0 < j_1 \le j_2 \le n, \\ \emptyset, & \text{otherwise.} \end{cases} \tag{2}$$

2 Optimality Conditions for Common Convex Sets

Remark 2.1 (Explicit stationarity conditions for common convex sets) *For many standard convex sets $C \subseteq \mathbb{R}^n$, the general stationarity inequality specializes in familiar optimality rules:*

- ***Full space*** $\mathbb{R}^n$***:*** $g(x^*) = 0$.
- ***Nonnegative orthant*** $\mathbb{R}^n_+$***:*** $g_i(x^*) = 0$ *if* $x_i^* > 0$, $g_i(x^*) \ge 0$ *if* $x_i^* = 0$.
- ***Unit simplex*** Δ_n***:*** *There exists* $\mu \in \mathbb{R}$ *such that*

$$g_i(x^*) = \mu \ \text{ if } x_i^* > 0, \qquad g_i(x^*) \ge \mu \ \text{ if } x_i^* = 0.$$

- ***Unit-sum set*** Δ'_n***:*** *There exists* $\mu \in \mathbb{R}$ *such that* $g_i(x^*) = \mu$, *for all* $i \in [n]$.
- ℓ_2***-ball*** $B_2^n[0,1]$***:***

$$\begin{cases} g(x^*) = 0, & \|x^*\|_2 < 1, \\ g(x^*) + \lambda x^* = 0, \ \lambda \ge 0, & \|x^*\|_2 = 1. \end{cases}$$

- ***Box constraints*** $[l,u]^n$***:*** *For each* $i \in [n]$,

$$g_i(x^*) = \begin{cases} 0, & l < x_i^* < u, \\ \geq 0, & x_i^* = l, \\ \leq 0, & x_i^* = u. \end{cases}$$

3 Supplementary Optimality Conditions

***L*-Stationarity under Symmetry Assumptions:** We now present an explicit characterization of L-stationarity under the assumption that the feasible set $C \subseteq \mathbb{R}^n$ is either a nonnegative type-1 symmetric set or a type-2 symmetric set. It was shown in [3, Theorem 5.4], a vector $x^* \in C_{\mathrm{s}} \cap C$ is an L-stationary point of problem (`CCOP`) if and only if x^* is a `BF` point, and the following inequality holds

$$p\left(Lx_i^* - g_i(x^*)\right) \geq p\left(-g_j(x^*)\right), \qquad \text{for all } i \in I_1(x^*),\ j \in I_0(x^*),$$

where $I_1(x^*)$ denotes the support of x^* and $I_0(x^*)$ its complement and p is the symmetry function $p : \mathbb{R}^n \to \mathbb{R}^n$ is defined as

$$p(x) = \begin{cases} x, & \text{if } C \text{ is a nonnegative type-1 symmetric set,} \\ |x|, & \text{if } C \text{ is a type-2 symmetric set.} \end{cases} \tag{3}$$

Here, $|x|$ denotes the component-wise absolute value of x (this function is used to define a common sorting permutation $\pi \in \tilde{\mathfrak{S}}(p(x))$ for both cases).

Characterization of `BF` Points with Incomplete Support: It was shown in [3, Lemma 5.2] that any L-stationary point is necessarily a `BF` point. Moreover, it was shown that the converse also holds when the point does not have full support and the set C is either a nonnegative type-1 symmetric set or a type-2 symmetric set. This implies that, in such cases, verifying whether a point in $C_{\mathrm{s}} \cap C$ is `BF` actually is equivalent to checking L-stationarity. Consequently, there is no need to examine all possible super support sets to verify basic feasibility.

Let $C \subseteq \mathbb{R}^n$ be either a nonnegative type-1 symmetric set or a type-2 symmetric set, and let $x \in C_{\mathrm{s}} \cap C$ be such that $|I_1(x)| < s$, i.e., the support of x is incomplete. Let $\mathcal{L} \subseteq [n]$ be a super support set of x, and suppose that for some $L > 0$,

$$x_{\mathcal{L}} = P_{C_{\mathcal{L}}}\left(x_{\mathcal{L}} - \frac{1}{L} g_{\mathcal{L}}(x)\right). \tag{27}$$

Then, it has been shown in [3, Lemma 5.4] that the following inequality holds:

$$p\left(x_i - \frac{1}{L} g_i(x)\right) \geq p\left(x_j - \frac{1}{L} g_j(x)\right), \qquad \text{for all } i \in I_1(x),\ j \in \mathcal{L} \cap I_0(x).$$

Equivalence of `BF` and *L*-Stationarity for Incomplete Support: Let $C \subseteq \mathbb{R}^n$ be either a nonnegative type-1 symmetric set or a type-2 symmetric set, and let $x \in C_{\mathrm{s}} \cap C$ be such that $|I_1(x)| < s$. Then, it has been shown in [3, Theorem 5.5] that the following statements are equivalent: (a) x is a `BF` point of problem (`CCOP`); and, (b) x is an L-stationary point of (`CCOP`) over $C_{\mathrm{s}} \cap C$ for any $L > 0$.

Verifying Basic Feasibility via a Single Super Support Set. Assume C is either a nonnegative type-1 symmetric set or a type-2 symmetric set. Let π be any sorting permutation for the vector $-p(-g(x))$. Choose the smallest i such that $\left|I_1(x) \cup S^{\pi}_{[i,n]}\right| = s$ and define $\mathcal{L} := I_1(x) \cup S^{\pi}_{[i,n]}$. Here, $S^{\pi}_{[i,n]}$ is from (2). If for some $L > 0$,

$$x_{\mathcal{L}} = P_{C_{\mathcal{L}}}\left(x_{\mathcal{L}} - \tfrac{1}{L} g_{\mathcal{L}}(x)\right),$$

then x is a `BF` point ([3, Theorem 5.6]).

4 Proof of Lemma 1 of the main paper [8]

Here and throughout, by a "face determined by a fixed support pattern" we mean the subset of C consisting of points with identical zero–nonzero structure; this need not be a face in the classical convex–analytic sense.

Let G be the finite group of linear isometries associated with the symmetry of C: $G = \mathfrak{S}_n$ for type–1 symmetry, and

$$G = \{D_\varepsilon P : \varepsilon \in \{\pm 1\}^n,\ P \in \mathfrak{S}_n\}$$

for type–2 symmetry, where $D_\varepsilon = \operatorname{diag}(\varepsilon_1, \ldots, \varepsilon_n)$. Then, $hC = C$ for all $h \in G$.

Let $C \subseteq \mathbb{R}^n$ be a locally closed set and $h : \mathbb{R}^n \to \mathbb{R}^n$ be a linear isometry. An isometry satisfies $h^* = h^{-1}$, where h^* is the adjoint operator. We want to show that

$$N^F_C(hx) = hN^F_C(x) \qquad \forall\, h \in G, \forall\, x \in C.$$

Here, N^F_C is a regular (Fréchet) normal cone, see [7, Definition 1.1]. To do that, we follow the following steps:
Step 1. The Transformation Rule for Regular Normals. According to the calculus rules for generalized differentiation in [7, Theorem 1.19], regular normal cones exhibit specific transformation properties under strictly differentiable mappings. If $\Phi : \mathbb{R}^n \to \mathbb{R}^n$ is a diffeomorphism (which is strictly differentiable and local-bi-Lipschitzian), the regular normal cone to the image set $\Phi(C)$ at $\Phi(\bar{x})$ satisfies:

$$N^F_{\Phi(C)}(\Phi(\bar{x})) = [\nabla\Phi(\bar{x})^*]^{-1}\, N^F_C(\bar{x}).$$

Step 2. Specialization to Linear Isometries. In our case, the mapping is $\Phi(x) = hx$. Because h is linear, its Jacobian matrix is constant and equal to h itself: $\nabla\Phi(x) = h$. Substituting this into the transformation rule, we obtain:

$$N^F_{hC}(h\bar{x}) = (h^*)^{-1} N^F_C(\bar{x}).$$

Step 3. Utilizing the Isometry Property. By the definition of a linear isometry in Euclidean space, the adjoint h^* is equal to the inverse h^{-1}. Therefore:

$$(h^*)^{-1} = (h^{-1})^{-1} = h.$$

Substituting this back into the equation yields:

$$N^F_{hC}(h\bar{x}) = hN^F_C(\bar{x}).$$

Step 4. Symmetry of the Set C. By the definition of the symmetry group G, the set C is invariant under the action of any $h \in G$, meaning $hC = C$. Consequently, the left side of the equation simplifies, giving us the final equivariance property:

$$N^F_C(hx) = hN^F_C(x) \qquad \forall\, h \in G, \forall\, x \in C.$$

Fix a support pattern $\mathcal{L} \subseteq [n]$ and consider the face

$$F_{\mathcal{L}} := C \cap \{x : I_1(x) = \mathcal{L}\}.$$

If $x^k \to x$ with all $x^k, x \in F_{\mathcal{L}}$, then for any $h \in G$ we also have $hx^k \to hx$ and $hF_{\mathcal{L}} = F_{\mathcal{L}}$, because h preserves zero components and acts by permutations/sign flips on the active coordinates.

Let $v^k \in N^F_C(x^k)$ with $v^k \to v$. For each k, choose $h_k \in G$ such that $h_k x^k$ lies in a fixed fundamental domain of the G-action on $F_{\mathcal{L}}$. Since G is finite, we may pass to a subsequence for which $h_k = h$ is constant. Then

$$hx^k \to hx, \qquad hv^k \to hv,$$

and

$$hv^k \in N^F_C(hx^k).$$

By the closedness of the Fréchet normal cone (Rockafellar–Wets [10, Proposition 6.6]), this implies

$$hv \in N^F_C(hx).$$

Applying h^{-1} gives

$$v \in h^{-1}N^F_C(hx) = N^F_C(x),$$

using equivariance again. Thus

$$\limsup_{k\to\infty} N^F_C(x^k) \subseteq N^F_C(x),$$

so N^F_C is outer semicontinuous on $F_{\mathcal{L}}$. This establishes cone–continuity on every support face, i.e., `CCP` in the sense of [5].

5 Proof of Theorem 1 of the main paper [8]

We now recall two standard calculus rules for Fréchet normal cones:
(i) **Product rule.** If $A \subset \mathbb{R}^m$ and $B \subset \mathbb{R}^p$ are closed sets, then for any $(a, b) \in A \times B$,

$$N^{\mathrm{F}}_{A\times B}(a, b) \;=\; N^{\mathrm{F}}_A(a) \times N^{\mathrm{F}}_B(b).$$

(ii) **Intersection rule.** If $A, B \subset \mathbb{R}^m$ are closed sets, then for any $u \in A \cap B$,

$$N^{\mathrm{F}}_{A\cap B}(u) \;\subset\; N^{\mathrm{F}}_A(u) + N^{\mathrm{F}}_B(u).$$

The proof can be done in six steps **S1**–**S6**:
S1 (Local strictness via a proximal term). By [5, Theorem 2.2(c)], since $\widehat{x}$ is a local minimizer of the original problem, there exists $\widehat{y} \in \{0, 1\}^n$ such that $(\widehat{x}, \widehat{y})$ is a local minimizer of the relaxation problem

$$\mathcal{R}_C \;:=\; \min f(x) \quad \text{s.t.} \quad x \in \mathcal{F}_C := \{(x, y) : x \in C,\ n - \mathbf{e}^\top y \le s,\ (x, y) \in W\},$$

where $\mathbf{e} \in \mathbb{R}^n$ is the vector of ones and

$$W := Q^n = \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^n :\ x \circ y = 0\}, \quad \text{with} \ \ Q := \{(a, b) \in \mathbb{R}^2 :\ ab = 0\}.$$

Here $\circ$ denotes the Hadamard product. Since the feasible set $\mathcal{F}_C$ is closed and $(\widehat{x}, \widehat{y})$ is a local minimizer of $\mathcal{R}_C$ over $\mathcal{F}_C$, there exists $\varepsilon > 0$ such that $(\widehat{x}, \widehat{y})$ uniquely minimizes

$$\min \ f(x) + \tfrac{1}{2}\,\|(x, y) - (\widehat{x}, \widehat{y})\|^2 \quad \text{s.t.} \quad (x, y) \in \mathcal{F}_C \cap \mathbb{B}_\varepsilon((\widehat{x}, \widehat{y})). \tag{4}$$

Here, $\mathbb{B}_\varepsilon(\widehat{x}, \widehat{y})$ is the closed Euclidean ball of radius ε centered at $(\widehat{x}, \widehat{y})$ in $\mathbb{R}^{2n}$.

S2 (Localized penalization of inequalities). For $t \in \mathbb{R}$ write $t_+ := \max\{t, 0\}$ and define the hinge-square penalty

$$\pi(y) \;:=\; \tfrac{1}{2}\Big((n - \mathbf{e}^\top y - s)_+^2 + \sum_{i=1}^{n}(y_i - 1)_+^2\Big)$$

for the inequality constraints in y. Since each $\widehat{y}_i \in \{0, 1\}$, $(\widehat{y}_i - 1)_+ = \max\{\widehat{y}_i - 1, 0\} = 0$ for all i, hence $\sum_{i=1}^n (\widehat{y}_i - 1)_+^2 = 0$. Moreover,

$$n - \mathbf{e}^\mathsf{T}\widehat{y} \;=\; \#\{i : \widehat{y}_i = 0\} \;=\; \#\{i : \widehat{x}_i \ne 0\} \;=\; \|\widehat{x}\|_0 \;\le\; s,$$

so $(n - \mathbf{e}^\mathsf{T}\widehat{y} - s)_+ = 0$. Plugging these into the definition of π gives $\pi(\widehat{y}) = \frac{1}{2}(0 + 0) = 0$. Let $S := (C \times \mathbb{R}^n) \cap W$. Then, for a sequence $\{\alpha_k\} \subseteq \mathbb{R}_+$ with $\alpha_k \uparrow \infty$, consider the compact localized problems

$$\min_{(x,y)\in S\cap\mathbb{B}_\varepsilon((\widehat{x},\widehat{y}))} \ \Phi_k(x, y) \;:=\; f(x) + \tfrac{1}{2}\,\|(x, y) - (\widehat{x}, \widehat{y})\|^2 + \alpha_k\,\pi(y). \tag{5}$$

Since the objective in (5) is C^1 and $S \cap \mathbb{B}_\varepsilon((\widehat{x}, \widehat{y}))$ is nonempty and compact, (5) admits a global minimizer (x^k, y^k) for each $k \in \mathbb{N}$. By compactness, $(x^k, y^k) \to (\overline{x}, \overline{y}) \in S \cap \mathbb{B}_\varepsilon((\widehat{x}, \widehat{y}))$. Because $(\widehat{x}, \widehat{y})$ is feasible for (5) with $\pi(\widehat{y}) = 0$ and (x^k, y^k) is optimal,

$$f(x^k) + \tfrac{1}{2} \left\| (x^k, y^k) - (\widehat{x}, \widehat{y}) \right\|^2 + \alpha_k \pi(y^k) \ \leq \ f(\widehat{x}).$$

Dividing by α_k and letting $k \to \infty$ gives $\pi(y^k) \to 0$. Passing to lim sup in the above inequality yields

$$f(\overline{x}) + \tfrac{1}{2} \left\| (\overline{x}, \overline{y}) - (\widehat{x}, \widehat{y}) \right\|^2 \ \leq \ f(\widehat{x}).$$

Moreover, $\pi(y^k) \to 0$ implies $(\overline{x}, \overline{y})$ satisfies the y-inequalities, hence $(\overline{x}, \overline{y})$ is feasible for (4). Since (4) has the unique minimizer $(\widehat{x}, \widehat{y})$, we conclude $(\overline{x}, \overline{y}) = (\widehat{x}, \widehat{y})$.

S3 (First-order condition on S). Since each (x^k, y^k) is a local (indeed global) minimizer of Φ_k over the closed set S,

$$-\nabla \Phi_k(x^k, y^k) \ \in \ N_S^{\mathrm{F}}(x^k, y^k). \tag{6}$$

The gradient with respect to x and y at (x^k, y^k) is computed as

$$\nabla_x \Phi_k(x^k, y^k) \ = \ g(x^k) + (x^k - \widehat{x}), \qquad \nabla_y \Phi_k(x^k, y^k) \ = \ (y^k - \widehat{y}) + \alpha_k \nabla \pi(y^k).$$

S4 (Decomposing normals via the calculus sum rule). In this step, we rigorously justify the decomposition of the normal cone to the intersection. Since C is a symmetric convex set with a nonempty interior, it satisfies the **CC-MFCQ** at every feasible point. This ensures that the intersection of the constraints is regular, allowing us to invoke the **basic calculus sum rule** for normal cones.

Applying $A := C$ and $B := \mathbb{R}^n$ in the product rule yields

$$N_{C \times \mathbb{R}^n}^{\mathrm{F}}(x^k, y^k) = N_C^{\mathrm{F}}(x^k) \times \{0\},$$

since $N_{\mathbb{R}^n}^{\mathrm{F}}(y) = \{0\}$ for all $y \in \mathbb{R}^n$. Next, applying the intersection rule with $A := C \times \mathbb{R}^n$ and $B := W$, we obtain

$$N_{(C \times \mathbb{R}^n) \cap W}^{\mathrm{F}}(x^k, y^k) \ \subset \ N_{C \times \mathbb{R}^n}^{\mathrm{F}}(x^k, y^k) + N_W^{\mathrm{F}}(x^k, y^k).$$

Combining both relations gives

$$N_S^{\mathrm{F}}(x^k, y^k) \ \subset \ \Big(N_C^{\mathrm{F}}(x^k) \times \{0\} \Big) \ + \ N_W^{\mathrm{F}}(x^k, y^k),$$

as claimed. Hence, from (6) there exist $u^k \in N_C^{\mathrm{F}}(x^k)$ and $(a^k, b^k) \in N_W^{\mathrm{F}}(x^k, y^k)$ with

$$-\begin{bmatrix} g(x^k) + (x^k - \widehat{x}) \\ (y^k - \widehat{y}) + \alpha_k \nabla \pi(y^k) \end{bmatrix} \ = \ \begin{bmatrix} u^k \\ 0 \end{bmatrix} \ + \ \begin{bmatrix} a^k \\ b^k \end{bmatrix}.$$

In particular, for the x-components:

$$-\Big(g(x^k) + (x^k - \widehat{x}) \Big) \ = \ u^k + a^k. \tag{7}$$

Fix $i \in I_1(\widehat{x})$. Since $\widehat{x}_i \neq 0$, complementarity implies $\widehat{y}_i = 0$. Because $(x^k, y^k) \to (\widehat{x}, \widehat{y})$ and each $(x^k, y^k) \in W$ satisfies $x_i^k y_i^k = 0$, it follows that for all sufficiently large k one has $x_i^k \neq 0$ and $y_i^k = 0$. For $Q := \{(a, b) \in \mathbb{R}^2 : ab = 0\}$ one has

$$N_Q^{\mathrm{F}}(a, b) = \begin{cases} \mathbb{R} \times \{0\}, & a = 0,\ b \neq 0, \\ \{0\} \times \mathbb{R}, & a \neq 0,\ b = 0, \\ \{(0, 0)\}, & a = 0,\ b = 0. \end{cases}$$

Hence, for $W = Q^n$ and $(x^k, y^k) \in W$ we get $N_W^{\mathrm{F}}(x^k, y^k) = \prod_{i=1}^n N_Q^{\mathrm{F}}(x_i^k, y_i^k)$. In particular, if $x_i^k \neq 0$ then $y_i^k = 0$ (by complementarity), so $N_Q^{\mathrm{F}}(x_i^k, y_i^k) = \{0\} \times \mathbb{R}$ and the ith x-component of any vector in $N_Q^{\mathrm{F}}(x_i^k, y_i^k)$ is zero. Therefore,

$$(a^k)_i = 0 \quad \text{for all } i \in I_1(x^k), \quad \text{i.e., } a^k_{I_1(x^k)} = 0.$$

S5 (Building the `CC-AM` sequences). From **S4** we have, for each k, a decomposition with $u^k \in N_C^F(x^k)$ and $(a^k, b^k) \in N_W^F(x^k, y^k)$ satisfying (7). If $|I_1(x^k)| = s$ (full support), set $\gamma^k := a^k$. Then $\gamma^k_{I_1(x^k)} = 0$, hence $\gamma^k \in N_{C_s}^F(x^k)$. Otherwise, $|I_1(x^k)| < s$ (incomplete support), we set $\gamma^k := 0$. In this case $N_{C_s}^F(x^k) = \{0\}$, so again $\gamma^k \in N_{C_s}^F(x^k)$. With this choice, (7) implies

$$\nabla f(x^k) + u^k + \gamma^k = \widehat{x} - x^k \longrightarrow 0.$$

Since $x^k \to \widehat{x}$ from **S2** and $u^k \in N_C^F(x^k)$ from **S4**, the `CC-AM` conditions are satisfied.

S6 (From `CC-AM` to `CC-M` via regularity). If the cone–continuity property (`CC-AM`–regularity) holds at $\hat{x}$, then `CC-AM` implies `CC-M`; see [5, Theorem 4.7]. This regularity is automatic in several serious situations. For instance, if C is polyhedral (orthant, simplex, box constraints, etc.), then all constraint functions are affine and `CC-AM`–regularity holds by [5, Cor. 4.10(b)]. More generally, `CC-AM`–regularity follows from `CC-CPLD` [5, Cor. 4.10(a)]. In addition, by Lemma 1, every closed, convex symmetric set (type–1 or type–2) also satisfies the cone–continuity property. Hence, under any of these mild assumptions on C, `CC-AM` at $\hat{x}$ implies `CC-M`.

As a result, `CC-M` stationarity is strictly stronger than basic feasibility because it satisfies the full first-order necessary condition

$$0 \in g(\bar{x}) + N_\Omega(\bar{x}),$$

for the nonconvex feasible set $\Omega = C \cap C_{\mathrm{s}}$. Unlike `BF` points, which only guarantee stationarity over specific super-supports, `CC-M` points are variationally stationary with respect to the entire set Ω. The existence of an interior point for the symmetric set C ensures that the calculus sum rule $N_\Omega(\bar{x}) \subseteq N_C(\bar{x}) + N_{C_{\mathrm{s}}}(\bar{x})$ holds rigorously, justifying the decoupled characterization $0 \in g(\bar{x}) + u + \gamma = 0$ used throughout this work.

The relative strengths of these conditions, along with their reliance on constraint qualifications (CQs) and regularity properties like `CCP`, are summarized in Table 1.

6 Relation Between BF Points and CC–Stationarity

BF points express projected–gradient stationarity with respect to all super–supports of a given point. For the sparse problem (CCOP), every local minimizer is a BF point [3, Theorem 5.1]. However, basic feasibility does not enforce optimality of the support set itself, making BF points strictly weaker than the CC-stationarity concepts. To the best of our knowledge, current convergence results for penalty decomposition methods generally ensure convergence only to Lu–Zhang stationary points, which are even weaker than BF points and, in turn, are strictly weaker than CC-stationarity.

In the CC-stationarity framework of [5], and in the absence of additional smooth constraints, our problem satisfies the hierarchy

$$\texttt{CC-S}\text{ (strong)} \;\Rightarrow\; \texttt{CC-M}\text{ (M-stationary)} \;\Rightarrow\; \texttt{CC-AM}\text{ (app. M-stationary).}$$

CC-AM stationarity is defined in terms of sequences of feasible points and multipliers satisfying approximate first-order conditions; it is the weakest notion in this chain and holds for all local minimizers without requiring constraint qualifications. When CCP holds—which in our setting is automatic for closed, convex symmetric sets by Lemma 1 of the main paper—CC-AM stationarity implies CC-M stationarity. Consequently, the implication chain reduces to

$$\text{Local minimizer} \;\Rightarrow\; \texttt{CC-AM} \;\Rightarrow\; \texttt{CC-M}.$$

If a point additionally satisfies the multiplier sign conditions associated with CC-S stationarity, then the stronger implication

$$\texttt{CC-S} \;\Rightarrow\; \texttt{CC-M}$$

holds. The converse implications generally do not hold:

- Every CC-M stationary point is CC-AM, but there exist CC-AM points that are not CC-M.
- BF points need not be CC-AM, since basic feasibility ensures only projected–gradient stationarity over super–support sets and does not enforce the multiplier structure required for CC-stationarity.

As a result, CC-M stationarity is strictly stronger than basic feasibility because it satisfies the full first-order necessary condition

$$0 \in g(\bar{x}) + N_C(\bar{x}) + N_{C_s}(\bar{x}),$$

for the nonconvex feasible set $C \cap C_s$. In contrast, BF points guarantee only projected gradient stationarity with respect to all super-support sets and may fail to be variationally stationary or locally optimal.

As shown in Table 1, the main optimality condition definitions are consistent with their sources; we also indicate their relative strength in the stationarity hierarchy.

Table 1: Summary of optimality condition definitions and their relative strength

Condition	**Source**	**Relative Strength in Hierarchy**
Lu–Zhang	Lu & Zhang (2013)	**Weak**: Weaker than `BF`; often obtained by basic penalty methods but lacks variational rigor.
`BF`	Beck et al. (2016)	**Weak**: Ensures projected-gradient stationarity over super-supports; does not guarantee support optimality.
L-stationarity	Beck et al. (2016)	**Weak–Moderate**: Equivalent to `BF` under incomplete support with symmetry; adds a fixed-point condition.
AW	Ribeiro et al. (2022)	**Moderate**: Sequential condition holding without CQs; equivalent to `CC-AM` in the symmetric setting.
`CC-AM`	Kanzow et al. (2021)	**Moderate**: Holds for all local minimizers without CQs; implies `CC-M` under `CCP`.
`CC-M`	Kanzow et al. (2021)	**Strong**: Full first-order necessary condition using the limiting normal cone characterization.
`CC-S`	Kanzow et al. (2021)	**Strongest**: `CC-M` plus sign conditions on multipliers (strong stationarity).

7 Practical enhancements for `PD-QN`

As a member of the family of the quasi-Newton methods, `PD-QN` is capable of being equipped by some line search procedures or taking advantage of various Hessian approximations. By such developments, we can make it possible to enhance accuracy or to accelerate convergence. As a matter of fact, in high-dimensional models, we often have to sacrifice accuracy to some meaningful levels to increase the convergence speed. Motivated by these, next, we come to describe a line search technique as well as several diagonal approximations of the Hessian as possible initiatives to improve the computational performance of `PD-QN`.

7.1 `PD-QN` Equipped with an Improved Line Search

In this subsection, we describe our improved line search procedure, whose goal is to compute $x_{\ell+1}^{(j)}$ in (line 8 of) `PD-QN` for a possible (more) reduction in the model function Φ. This line search employs a backtracking technique if the line search condition for $\alpha = 1$ is not met; otherwise, it transitions into an extrapolation technique, aiming to take a longer stride toward the solution. Our improved line search only computes the model function values at the trial points, except at the accepted point, in which the main cost function value should be calculated. Hence, in each call to our new line search, one function evaluation is required, which makes its computational cost lower than that of classic line searches. Here, `PD-QN-LS` stands for `PD-QN` equipped with the improved line search.

As is known, starting from the current iterate, a line search generally deals with finding a proper step size on a directionally differentiable curve of the feasible points defined along a descent direction. Now, based on such a principal plan, by defining $\widetilde{x}(\alpha)$ as the mentioned feasible curve parameterized by $\alpha \geq 0$, our line search starts from the current point $x^{(j-1)}$ and searches for an appropriate (step size) value for α within the search path

$$\widetilde{x}(\alpha) := \left(\mathbf{H}^{(j-1)} + \alpha\rho^{(j-1)}I\right)^{-1}\left(\mathbf{H}^{(j-1)}x^{(j-1)} + \alpha\Big(\rho^{(j-1)}y - g\left(x^{(j-1)}\right)\Big)\right), \tag{8}$$

for which the tangent vector is given by

$$\widetilde{x}'(0) := \left(\mathbf{H}^{(j-1)}\right)^{-1}\left(\rho^{(j-1)}(y-x) - g\left(x^{(j-1)}\right)\right). \tag{9}$$

Since in `PD-QN` we have $\widetilde{x}(0) = x^{(j-1)}$ and $\widetilde{x}(1) = x^{(j)}_{\ell+1}$ by respectively setting $\alpha = 0$ and $\alpha = 1$ in (8), it is reasonable to first perform a simple backtracking line search along the direction

$$d^{(j)} := \widetilde{x}(1) - \widetilde{x}(0) = x^{(j)}_{\ell+1} - x^{(j-1)}, \tag{10}$$

in the sense of setting $\alpha \leftarrow \alpha/2$ until the Armijo condition

$$\Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}\left(\widetilde{x}(\alpha), y\right) \leq \Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}(x,y) + \alpha\tau\widetilde{x}'(0)^T d^{(j)} \tag{11}$$

is fulfilled, in which $\tau \in (0,1)$ is the Armijo (tuning) parameter. In addition, whenever our line search stops with $\alpha = 1$, we do an extrapolation along the direction $d^{(j)}$ to quickly leave the regions far from an optimum point. That is, in the extrapolation phase, as long as the Armijo condition (11) holds, the step size is increased as $\alpha \leftarrow 2\alpha$. So, here only the model function $\Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}\left(\widetilde{x}(\alpha), y\right)$ is successively computed, without any extra main cost function evaluations.

Proposition 7.1 (Finite Termination of the Improved Line Search) *At each inner step of Algorithm 1 presented in the main paper equipped with the improved line search* (8)–(11), *suppose that the model $x \mapsto \Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}(x,y)$ is continuously differentiable with a locally Lipschitz gradient near $x^{(j-1)}$. Then for each fixed j and y, the line search terminates after finitely many iterations:*

(i) If the Armijo condition (11) *is not satisfied at $\alpha = 1$, the backtracking phase $\alpha \leftarrow \beta\alpha$ with $\beta \in (0,1)$ produces an $\alpha > 0$ satisfying the Armijo condition after finitely many reductions.*

(ii) If the Armijo condition (11) *holds at $\alpha = 1$, the extrapolation phase $\alpha \leftarrow \beta'\alpha$ with $\beta' > 1$ produces a largest $\alpha > 0$ satisfying the Armijo condition after finitely many doublings.*

In either case, the procedure returns an $\alpha > 0$ satisfying (11) *in a finite number of steps.*

Proof. Fix j and y, and consider the search path $\widetilde{x}(\alpha)$ defined in (8). Define $\varphi(\alpha) := \Phi_{(\rho^{(j-1)},x^{(j-1)})}\Big(\widetilde{x}(\alpha), y\Big)$. Because $\Phi_{(\rho^{(j-1)},x^{(j-1)})}(\cdot, y)$ is continuously differentiable with locally Lipschitz gradient, $\varphi(\alpha)$ is continuously differentiable in a neighborhood of $\alpha = 0$.

Note that because Φ is a quadratic function in x with a positive–definite Hessian (due to $\rho > \rho_{\min} > 0$ and Assumption 2 in the main paper), the search direction computed by minimizing this model is indeed a descent direction. Therefore, by construction of the algorithm, the search direction

$$d^{(j)} := \widetilde{x}(1) - \widetilde{x}(0)$$

satisfies

$$\varphi'(0) = \nabla_x \Phi_{(\rho^{(j-1)},x^{(j-1)})}(x^{(j-1)}, y)^\top d^{(j)} < 0.$$

(i) **Backtracking phase.** Suppose the Armijo condition (11) is not satisfied at $\alpha = 1$. Since φ is differentiable at 0 with $\varphi'(0) < 0$, there exists $\bar{\alpha} > 0$ such that for all $0 < \alpha \leq \bar{\alpha}$,

$$\varphi(\alpha) \leq \varphi(0) + \alpha\tau\varphi'(0),$$

where $\tau \in (0, 1)$ is the Armijo parameter. Starting from $\alpha = 1$ and repeatedly updating $\alpha \leftarrow \beta\alpha$ with $\beta \in (0, 1)$, the step size sequence $\{\alpha_k\}$ decreases geometrically, i.e., $\alpha_k = \beta^k$. Since $\beta^k \to 0$ as $k \to \infty$, after finitely many reductions we obtain $\alpha_k \leq \bar{\alpha}$ and therefore the Armijo condition (11) is satisfied. Hence, the backtracking terminates in a finite number of steps.
(ii) **Extrapolation phase.** Suppose the Armijo condition (11) is already satisfied at $\alpha = 1$. We then test larger steps via $\alpha \leftarrow \beta'\alpha$ with some $\beta' > 1$. Because $\Phi_{(\rho^{(j-1)},x^{(j-1)})}(\cdot, y)$ is a strongly convex quadratic in x, $\varphi(\alpha)$ grows at least quadratically as $\alpha \to \infty$. Therefore, there exists a finite $\alpha^\star > 0$ such that

$$\varphi(\alpha^\star) > \varphi(0) + \alpha^\star\tau\varphi'(0),$$

meaning the Armijo condition will be violated for steps larger than $\alpha^\star$. Since the sequence $\alpha_k = (\beta')^k$ grows geometrically, after finitely many extrapolation steps we find the first k with $\alpha_k > \alpha^\star$. At that point, the algorithm terminates and accepts the last α that satisfied (11).

In both cases, the search path $\alpha \mapsto \widetilde{x}(\alpha)$ yields an $\alpha > 0$ satisfying the Armijo condition (11) after finitely many updates. Therefore, the improved line search terminates in a finite number of steps. □

Compared to the original Algorithm 1, where the x–update in line 8 was computed by

$$x_\ell^{(j-1)} = \left(\mathbf{H}^{(j-1)} + \rho^{(j-1)} I\right)^{-1} \left(\mathbf{H}^{(j-1)} x^{(j-1)} + \rho^{(j-1)} y_{\ell-1}^{(j-1)} - g\left(x^{(j-1)}\right)\right). \tag{12}$$

as the exact minimizer

$$x_\ell^{(j)} = \operatorname*{argmin}_{x \in \mathbb{R}^n} \Phi_{(\rho^{(j-1)},x^{(j-1)})}\Big(x, y_{\ell-1}^{(j-1)}\Big),$$

we now employ the line search based update $x_\ell^{(j)} = \widetilde{x}(\alpha_j)$ with α_j selected by the improved backtracking–extrapolation procedure described in (8)–(11). All main convergence results presented earlier (boundedness of iterates, subsequential convergence to BF points, primal–dual agreement, rates, and KL–based global convergence) remain valid, because the line search still provides sufficient decrease and vanishing stationarity residuals. However, the arguments that previously relied on exact minimization in line 8 have been updated to use the Armijo decrease condition. In particular:

- The closed–form solution lemma for the x–update is replaced by Proposition 7.1 to reflect the line search update. This proposition guarantees finite termination of the step computation.
- In the proofs of the inner–loop convergence (Theorem 4 of the main paper), wherever exact minimization was used, it is replaced by the sufficient decrease inequality resulting from the Armijo condition.

With these updates, the improved line search integrates seamlessly into Algorithm 1 and the convergence theory continues to hold under the revised step computation.

7.2 Diagonal Estimations of the Hessian

In our quasi-Newton penalty decomposition method, it is of vital importance to effectively update the Hessian approximation (in line 26 of `PD-QN`). A careful readout of the literature reveals a great deal of interest in devising memoryless (limited memory) approximations of the (inverse) Hessian of the cost function of the unconstrained optimization models, which frequently appear in practical fields. At the core of such studies, a meaningful plan can be observed to more directly benefit from the second-order information based on the secant equation (cf. [9]).

Here, for the model

$$\min_{x\in\mathbb{R}^n} \ \Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}(x,y), \tag{13}$$

the secant equation can be written as

$$\mathbf{H}^{(j)}p^{(j-1)} = q^{(j-1)}, \tag{14}$$

where $\mathbf{H}^{(j)} \approx \nabla_x^2\Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}\left(x^{(j)},y\right)$, with

$$q^{(j-1)} = \nabla_x\Phi_{\left(\rho^{(j-1)},x^{(j-1)}\right)}(x^{(j)},y) - \nabla_x\Phi_{\left(\rho^{(j-1)},x^{(j-2)}\right)}\left(x^{(j-1)},y\right), \tag{15}$$

and

$$p^{(j-1)} = x^{(j)} - x^{(j-1)} = \alpha_{j-1}d^{(j-1)}, \tag{16}$$

in which the step size α_{j-1} often results from an approximate line search along the direction

$$d^{(j-1)} = -{\mathbf{H}^{(j-1)}}^{-1}\nabla_x\Phi_{\left(\rho^{(j-1)},x^{(j-2)}\right)}\left(x^{(j-1)},y\right).$$

The huge growth of the dimension of the practical optimization models by the beginning of the current century pushed scholars to change their classic way of thinking toward the memoryless Hessian approximations. As a result, diagonal estimations of the Hessian have been developed, in which an approximation of the secant equation (14) is used [1, 2]. As a reasonable matter of routine to keep looking at the diversity and inclusion in the optimization tools, here we suggest modified diagonal approximations of the Hessian as well.

7.2.1 `PD-QN` with Three Limited Memory Approximations of the Hessian

In this subsection, we propose three new limited memory formulas for approximating the Hessian matrix. By forming and updating two matrices whose columns are the differences between the prior points and the differences between the prior gradients of the penalty model function, we can construct our three low-cost diagonal approximations, which have different behaviors for different practical problems.

In each iteration j of our algorithm, we need to compute the iterate difference $p^{(j)}$ (computed by (16)) of the previous points and the difference $q^{(j)}$ (defined by (15)) of the previous gradients. Then, until j is not greater than a prespecified positive integer m (as the size of the memory), we save $p^{(j)}$ as the jth column of the matrix $S^{(j)} \in \mathbb{R}^{n\times m}$, and $q^{(j)}$ as the jth column of the matrix $Y^{(j)} \in \mathbb{R}^{n\times m}$. Meanwhile, when the memory assigned to $S^{(j)}$ and $Y^{(j)}$ is full, we delete the oldest columns of $S^{(j)}$ and $Y^{(j)}$ to provide enough space for saving the newly generated pair of the vectors. As a result, we have the complete form of the matrices $S^{(j)}$ and $Y^{(j)}$ as follows:

$$S^{(j)} = \begin{pmatrix} p^{(1)} & p^{(2)} & \dots & p^{(m)} \end{pmatrix}, \qquad Y^{(j)} = \begin{pmatrix} q^{(1)} & q^{(2)} & \dots & q^{(m)} \end{pmatrix}.$$

As known, these two matrices have already been used for the construction of the limited memory quasi-Newton methods, e.g., see [4, 6]. Now, defining $A_{k:}$ as the kth row of an arbitrary matrix A, here we diagonally approximate the Hessian matrix as

$$\widehat{\mathbf{H}}^{(j)} = \operatorname{diag}\left(\widehat{h}_1^{(j)}, \widehat{h}_2^{(j)}, \dots, \widehat{h}_n^{(j)}\right),$$

whose entries are computed by the following three formulas:

$$\widehat{h}_k^{(j)} := \begin{cases} \dfrac{\|Y_{k:}^{(j)}\|_\infty}{\|S_{k:}^{(j)}\|_\infty}, & \texttt{LM1}, \\ \dfrac{\|S_{k:}^{(j)}\|_\infty}{\|Y_{k:}^{(j)}\|_\infty}, & \texttt{LM2}, \\ \max\left\{\dfrac{\|S_{k:}^{(j)}\|_\infty}{\|Y_{k:}^{(j)}\|_\infty}, \dfrac{\|Y_{k,:}^{(j)}\|_\infty}{\|S_{k:}^{(j)}\|_\infty}\right\}, & \texttt{LM3}, \end{cases} \tag{17}$$

for all $k \in [n]$. Moreover, to support well-definiteness and uniformly boundedness of the approximations, we restrict the (diagonal) entries of the matrix $\widehat{\mathbf{H}}^{(j)}$ by

$$\check{h}_k^{(j)} := \max\left\{\widehat{h}_k^{(j)}, \varrho\right\}, \qquad \text{for all } k \in [n], \tag{18}$$

for some enough small constant $\varrho > 0$. As a result, we have

$$\check{\mathbf{H}}^{(j)} = \operatorname{diag}\left(\check{h}_1^{(j)}, \check{h}_2^{(j)}, \ldots, \check{h}_n^{(j)}\right), \qquad \text{for all } j \in \mathbb{N}, \tag{19}$$

starting by $\check{\mathbf{H}}^{(0)} = \mathbf{I}$. The diagonal approximation $\check{\mathbf{H}}^{(j)}$ can be straightly employed in (line 26 of) `PD-QN`, in the framework of the Algorithm 1 given below.

Algorithm 1 Determining the diagonal update for the Hessian by three schemes

tuning parameter: the constant $\varrho > 0$.
input: the iteration number j.
Set $k = 1$.
repeat
    if $j = 0$, **then** set $\widetilde{h}_k^{(j)} = 1$; **else** compute $\widetilde{h}_k^{(j)}$ by (17). **end if**
    if $\widetilde{h}_k^{(j)} \in \{0, \mathrm{NaN}, \infty\}$, **then** set $\widetilde{h}_k^{(j)} = 1$. **end if**
    Compute $\overline{h}_k^{(j)}$ by (18).
    Set $k \leftarrow k + 1$.
until $k \leq n$.
Determine $\overline{\mathbf{H}}^{(j)}$ by (19).
output: $\overline{\mathbf{H}}^{(j)}$

7.2.2 `PD-QN-LS` with a Diagonal Estimation of the Hessian Based on a Well-conditioning Measure

In the literature of matrix computations, experts have rightly warned that some troubles are probable when doing calculations with an ill-conditioned matrix [11], a fact that has also been highlighted through experimental evidence. For symmetric positive definite matrices, discussions around the conditioning are principally centered on the distribution of the eigenvalues as the main concern. Based on the earlier efforts made in [1, 2], we can take the orthogonal matrices into account as a class of matrices that possess the least possible condition number [11].

The main feature of an orthogonal matrix (for which the condition number is equal to one) is that the inverse and the transpose of the matrix are exactly the same. This is the key issue of our approximation scheme. Meanwhile, we also need to explicitly employ the second-order information of the model function in (13), a goal that can be achieved by applying the secant equation (14). Thus, tying these two main issues together, we can shape our estimation scheme in the framework of the following model:

$$\min_{\mathbf{H}^{(j)} \in \mathbb{D}^+} \Psi\left(\mathbf{H}^{(j)}\right) = \frac{1}{2}\left\|\mathbf{H}^{(j)} p^{(j-1)} - q^{(j-1)}\right\|^2 + \frac{1}{2}\mu \|\mathbf{H}^{(j)} - \mathbf{H}^{(j)^{-1}}\|^2, \tag{20}$$

where $\mathbb{D}^+ \subset \mathbb{R}^{n\times n}$ is the set of all diagonal matrices with positive diagonal entries and $\mu > 0$ is a penalty parameter embedded to push the solution toward orthogonality, and as a result, to enforce well-conditioning. Now, since around the optimal solution (for enough large values of j) we have $p_k^{(j-1)} q_k^{(j-1)} \approx 0$, an approximate solution of the model (20) can be given by

$$\widetilde{\mathbf{H}}^{(j)} = \operatorname{diag}\left(\widetilde{h}_1^{(j)}, \widetilde{h}_2^{(j)}, \ldots, \widetilde{h}_n^{(j)}\right), \tag{21}$$

with

$$\widetilde{h}_k^{(j)} = \sqrt[4]{\frac{\mu}{p_k^{(j-1)^2} + \mu}} \in (0, 1], \qquad \text{for all } k = 1, 2, \ldots, n. \tag{22}$$

Although being positive, the values of $\left\{\widetilde{h}_k^{(j)}\right\}_{k=1}^n$ near to zero can be computationally troublesome. To prevent such an unintended consequence, and also to make the approximations uniformly bounded, here we consider a protective measure as

$$\overline{h}_k^{(j)} \stackrel{\text{def}}{=} \min\left\{\max\left\{\widetilde{h}_k^{(j)}, \varrho\right\}, \frac{1}{\varrho}\right\}, \quad \text{for all } k \in [n], \tag{23}$$

for some enough small constant $\varrho > 0$. Then, we define

$$\overline{\mathbf{H}}^{(j)} = \operatorname{diag}\left(\overline{h}_1^{(j)}, \overline{h}_2^{(j)}, \ldots, \overline{h}_n^{(j)}\right), \quad \text{for all } j \in \mathbb{N}, \tag{24}$$

starting by $\overline{\mathbf{H}}^{(0)} = \mathbf{I}$, as a bounded positive definite diagonal estimation of the Hessian. Hence, we can straightly employ the bounded Hessian approximation $\overline{\mathbf{H}}^{(j)}$ in (line 26 of) `PD-QN`, in the framework of the following algorithm.

Algorithm 2 Determining the diagonal update for the Hessian by a well-conditioning strategy

---

tuning parameters: the constants $\mu, \varrho > 0$.
input: the iteration number j.
Set $k = 1$.
repeat
  if $j = 0$, **then** set $\widetilde{h}_k^{(j)} = 1$; **else** compute $\widetilde{h}_k^{(j)}$ by (22). **end if**
  Compute $\overline{h}_k^{(j)}$ by (23).
  Set $k \leftarrow k + 1$.
until $k \leq n$.
Determine $\overline{\mathbf{H}}^{(j)}$ by (24).
output: $\overline{\mathbf{H}}^{(j)}$

---

8 Sparse Projection Over Symmetric Sets

Here, we recall Algorithms 1–4 (the sparse projection over symmetric sets) of Beck and Hallak [3] and discuss them.

As discussed in the introduction, given a closed and convex set $C \subseteq \mathbb{R}^n$ and a vector $x \in \mathbb{R}^n$, the sparse projection problem involves finding a point in the orthogonal projection of x onto the intersection $C \cap C_{\mathrm{s}}$, where C_{s} denotes the set of all vectors in $\mathbb{R}^n$ with at most s nonzero components. Formally, the sparse projection set is defined as

$$P_{C_{\mathrm{s}} \cap C}(x) = \underset{z \in C \cap C_{\mathrm{s}}}{\operatorname{argmin}} \|z - x\|^2.$$

We refer to $P_{C_{\mathrm{s}} \cap C}(x)$ as the **s-sparse projection set onto C**, and any element in this set is called an **s-sparse projection vector onto C**, or simply a **sparse projection vector**. Since the set $C \cap C_{\mathrm{s}}$ is closed, the projection set $P_{C_{\mathrm{s}} \cap C}(x)$ is nonempty for every $x \in \mathbb{R}^n$. However, due to the nonconvexity of $C_{\mathrm{s}} \cap C$, the projection may not be unique.

In the special case where $C = \mathbb{R}^n$, the sparse projection simplifies to

$$P_{C_{\mathrm{s}} \cap \mathbb{R}^n}(x) = P_{C_{\mathrm{s}}}(x),$$

which consists of all vectors obtained by retaining the s largest (in absolute value) components of x, setting all others to zero. When ties occur among components with equal magnitude, multiple valid selections are possible, and the projection set may contain more than one element.

Determining the set $P_{C_{\mathrm{s}} \cap C}(x)$, or even identifying a single vector within it, is generally challenging due to the nonconvex nature of the underlying optimization problem. Nevertheless, it was shown in [3] that under certain symmetry assumptions, this task becomes computationally tractable. As shown in [3, Lemma 4.1], $y_{\mathcal{L}} = P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$ for any super support set $\mathcal{L}$ of $y \in P_{C_{\mathrm{s}} \cap C}(x)$ with $x \in \mathbb{R}^n$. Here, $P_{C_{\mathcal{L}}}$ denotes the orthogonal projection onto the set $C_{\mathcal{L}}$, and $x_{\mathcal{L}}$ denotes the restriction of x to the index set $\mathcal{L}$. A straightforward but computationally inefficient approach for identifying a vector in the set $P_{C_{\mathrm{s}} \cap C}(x)$ involves enumerating all possible

$$\binom{n}{s}$$

super support sets $\mathcal{L} \subseteq [n]$ of cardinality s. For each candidate support set $\mathcal{L}$, one computes the projection by $P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$, where $x_{\mathcal{L}} = U_{\mathcal{L}}^T x$ is the restriction of x to the coordinates in $\mathcal{L}$, and $C_{\mathcal{L}} = \{z \in \mathbb{R}^{|\mathcal{L}|} : U_{\mathcal{L}} z \in C\}$ denotes the restriction of C to $\mathcal{L}$. The final output is the projection vector $U_{\mathcal{L}} P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$ corresponding to the support set $\mathcal{L}$ that minimizes the Euclidean distance to x, i.e.,

$$\min_{\{\mathcal{L}:|\mathcal{L}|=s\}} \left\| x - U_{\mathcal{L}} P_{C_{\mathcal{L}}}(x_{\mathcal{L}}) \right\|.$$

Given this construction, the s-sparse projection onto a symmetric type-1 set can be computed efficiently in $s+1$ steps by evaluating a candidate super support among $s+1$ possible sets. Let C be a closed and convex type-1 symmetric set, $x \in \mathbb{R}^n$, and let $\pi \in \tilde{\mathfrak{S}}(x)$ be a sorting permutation of x. Then, as shown in [3, Theorem 4.1], there exists a vector $y \in P_{C_{\mathrm{s}} \cap C}(x)$ such that

$$I_1(y) \subseteq S^{\pi}_{[1,k]} \cup S^{\pi}_{[n+k-(s-1),n]}, \qquad \text{for some } k \in \{0, 1, \ldots, s\},$$

where $I_1(y)$ denotes the support of y. This theorem implies that a super support set of a sparse projection vector can be efficiently identified by examining only $s+1$ candidate support sets. For each of such a candidate set, one computes the orthogonal projection onto the restriction of C corresponding to that index set. Among these projections, the one yielding the smallest Euclidean distance to x is selected as the optimal sparse projection vector. We now recall Algorithm 1 in [3] below to compute a projection onto a type-1 symmetric set.

Algorithm 3 Projection onto a Type-1 Symmetric Set

1: **input:** A vector $x \in \mathbb{R}^n$.
2: Find a sorting permutation $\pi \in \tilde{\mathfrak{S}}(x)$.
3: **for** $k = s, s-1, \ldots, 0$ **do**
4: Define the index set $\mathcal{L}_k = S^{\pi}_{[1,k]} \cup S^{\pi}_{[n+k-(s-1),n]}$.
5: Compute the projection $g_k = P_{C_{\mathcal{L}_k}}(x_{\mathcal{L}_k})$ and define $z_k = U_{\mathcal{L}_k} g_k$.
6: **end for**
7: Return $u = \operatorname{argmin}\{\|z - x\|^2 : z \in \{z_k\}_{k=0}^{s}\}$.
8: **output**: A vector $u \in P_{C_s \cap C}(x)$.

It is worth noting that explicitly computing a sorting permutation $\pi \in \tilde{\mathfrak{S}}(x)$, as required in line 2 of the algorithm, is not strictly necessary. In practice, it suffices to compute the index sets $\mathcal{L}_k$, which can be constructed directly in linear time by partially sorting the entries of x.

Sparse Projection onto Nonnegative Type-1 Symmetric Sets: For nonnegative type-1 symmetric sets, identifying a super support of a sparse projection vector $y \in P_{C_s \cap C}(x)$ is immediate: it suffices to select the indices corresponding to the s largest values of x. This leads to a direct and efficient projection method. Let $C \subseteq \mathbb{R}^n$ be a closed and convex nonnegative type-1 symmetric set, assume that $x \in \mathbb{R}^n$, and let $\pi \in \tilde{\mathfrak{S}}(x)$ be a sorting permutation of x. Then, as shown in [3, Theorem 4.2], there exists a vector $y \in P_{C_s \cap C}(x)$ such that the index set $S^{\pi}_{[1,s]}$ is a super support of y. We recall Algorithm 2 of [3] to compute a projection onto a nonnegative type-1 symmetric set.

Algorithm 4 Projection onto a Nonnegative Type-1 Symmetric Set

1: **input:** A vector $x \in \mathbb{R}^n$.
2: Compute the index set $\mathcal{L} = S^{\pi}_{[1,s]}$ for $\pi \in \tilde{\mathfrak{S}}(x)$.
3: Compute the projection $g = P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$ and return $u = U_{\mathcal{L}} g$.
4: **output**: A vector $u \in P_{C_s \cap C}(x)$.

Sparse Projection onto Type-2 Symmetric Sets: When the underlying set $C \subseteq \mathbb{R}^n$ is type-2 symmetric, a super support for a sparse projection vector $y \in P_{C_s \cap C}(x)$ can also be identified immediately. Specifically, it consists of the indices corresponding to the s largest components of $|x|$ in absolute value. Let $C \subseteq \mathbb{R}^n$ be a closed and convex type-2 symmetric set, and let $x \in \mathbb{R}^n$. Let $\pi \in \tilde{\mathfrak{S}}(|x|)$ be a sorting permutation of $|x|$. Then, as shown in [3, Theorem 4.3], there exists a vector $y \in P_{C_s \cap C}(x)$ such that the index set $S^{\pi}_{[1,s]}$ is a super support of y.

Let $C \subseteq \mathbb{R}^n$ be a closed and convex set that is either a nonnegative type-1 symmetric set or a type-2 symmetric set. Let $x \in \mathbb{R}^n$ and $\pi \in \tilde{\mathfrak{S}}(p(x))$, where the symmetry function

Algorithm 5 Projection onto a Type-2 Symmetric Set

input: A vector $x \in \mathbb{R}^n$.
Compute the index set $\mathcal{L} = S^{\pi}_{[1,s]}$ with a sorting permutation $\pi \in \tilde{\mathfrak{S}}(|x|)$ of $|x|$ (i.e., the component-wise absolute values of x).
Compute the projection $g = P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$ and return $u = U_{\mathcal{L}} g$.
output: A vector $u \in P_{C_s \cap C}(x)$.

$p : \mathbb{R}^n \to \mathbb{R}^n$ is defined as (3). Then, as shown in [3, Theorem 4.4], there exists a vector $y \in P_{C_s \cap C}(x)$ such that the index set $S^{\pi}_{[1,s]}$ is a super support of y. This theorem is a unified symmetric projection theorem.

We recall the recurrence-based algorithm [3, Algorithm 4] for computing a sparse projection vector on the unit-sum set.

Algorithm 6 Projection onto the Sparse Unit-Sum Set

input: A vector $x \in \mathbb{R}^n$.
Compute the index sets $S^{\pi}_{[1,s]}$ and $S^{\pi}_{[n-(s-1),n]}$ for $\pi \in \tilde{\mathfrak{S}}(x)$.
Sort the elements of x corresponding to the combined index set $S^{\pi}_{[1,s]} \cup S^{\pi}_{[n-(s-1),n]}$.
Initialize $\lambda_s = \frac{1}{s}\left(1 - \sum_{j=1}^{s} x_{\pi(j)}\right)$ and compute $f_s = \lambda_s^2 s + \sum_{j=s+1}^{n} x^2_{\pi(j)}$.
for $k = s-1, s-2, \ldots, 0$ **do**
    $\lambda_k = \lambda_{k+1} + \frac{1}{s}\left(x_{\pi(k+1)} - x_{\pi(n-(s-k))}\right)$.
    $f_k = f_{k+1} + s(\lambda_k^2 - \lambda_{k+1}^2) + x^2_{\pi(k+1)} - x^2_{\pi(n-(s-k))}$.
end for
Choose $m \in \operatorname*{argmin}_{k=0,\ldots,s} f_k$.
Obtain the projection $u \in \mathbb{R}^n$ componentwise as $u_i = x_i + \lambda_m$, if $i \in S^{\pi}_{[1,m]} \cup S^{\pi}_{[n+m-(s-1),n]}$, and $u_i = 0$, otherwise.
output: A vector $u \in P_{C_s \cap \Delta'_n}(x)$ for a unit-sum set Δ'_n.

For various simple convex sets $C \subseteq \mathbb{R}^n$, we summarize the structure of sparse projection vectors, the corresponding super support sets, and the restricted sets over which the projection is computed:

- **Full space $\mathbb{R}^n$:** The candidate projection vector is $U_{\mathcal{L}} x_{\mathcal{L}}$ with $\mathcal{L} = S^{\pi}_{[1,s]}$ for a sorting permutation $\pi \in \tilde{\mathfrak{S}}(|x|)$. The projection is computed over the restricted set $\mathbb{R}^s$.
- **Nonnegative orthant $\mathbb{R}^n_+$:** The projection vector is also $U_{\mathcal{L}} x_{\mathcal{L}}$ with $\mathcal{L} = S^{\pi}_{[1,s]}$, and $\pi \in \tilde{\mathfrak{S}}(x)$. The corresponding restricted set is $\mathbb{R}^s_+$.
- **Unit simplex Δ_n:** The projection is given by $U_{\mathcal{L}} P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$, where $\mathcal{L} = S^{\pi}_{[1,s]}$ for $\pi \in \tilde{\mathfrak{S}}(x)$, and the restricted set is Δ_s.
- **Unit-sum set Δ'_n:** For $k = 0, \ldots, s$, and $\pi \in \tilde{\mathfrak{S}}(x)$, the candidate vector is

$$U_{\mathcal{L}_k} P_{C_{\mathcal{L}_k}}(x_{\mathcal{L}_k}) \quad \text{with} \quad \mathcal{L}_k = S^{\pi}_{[1,k]} \cup S^{\pi}_{[n+k-(s-1),n]}.$$

The restricted set is Δ'_s.

- **ℓ_p-ball $B_p^n[0,1]$, $p \geq 1$:** The projection vector takes the form $U_{\mathcal{L}} P_{C_{\mathcal{L}}}(x_{\mathcal{L}})$, with $\mathcal{L} = S^{\pi}_{[1,s]}$ for $\pi \in \tilde{\mathfrak{S}}(|x|)$, and the restricted set is $B_p^s[0,1]$.

- **Box constraints $[l,u]^n$ with $l < u$:** The projection is given by $U_{\mathcal{L}_k} P_{C_{\mathcal{L}_k}}(x_{\mathcal{L}_k})$, where $\mathcal{L}_k = S^{\pi}_{[1,k]} \cup S^{\pi}_{[n+k-(s-1),n]}$ for $k = 0,\dots,s$, and $\pi \in \tilde{\mathfrak{S}}(x)$. The corresponding restricted set is $[l,u]^s$.

9 A Comparison Among Variants of Our Algorithm

We compare four variants of the proposed algorithm, namely `PD-LM1`, `PD-LM2`, `PD-LM3`, and `PD-D`, which differ in their diagonal Hessian approximation schemes (Algorithms 1–2). Following [8], our algorithm is initialized using `BFS`, a procedure specifically designed for optimization over sparse symmetric sets. We consider three choices for the maximum number of `BFS` iterations, `maxiterBFS` $\in \{250, 300, 350\}$.

Accordingly, different versions of our algorithm are denoted as `PD-LM1-a`, `PD-LM2-a`, `PD-LM3-a`, and `PD-D-a` for `maxiterBFS` $= 250$; `PD-LM1-b`, `PD-LM2-b`, `PD-LM3-b`, and `PD-D-b` for `maxiterBFS` $= 300$; and `PD-LM1-c`, `PD-LM2-c`, `PD-LM3-c`, and `PD-D-c` for `maxiterBFS` $=$ 350. We evaluate these methods using the two performance measures `nf2g` and `sec`, to identify approximate global minimizers and `CC-S` stationary points in Figures 1–6.

From Figure 1, although `PD-LM3-a` appears slightly more efficient and robust than `PD-LM1-a` in identifying approximate global minimizers, Figure 2 shows that `PD-LM1-a` is the most efficient method and is more robust than `PD-LM3-a` in computing `CC-S` stationary points. Consequently, `PD-LM1-a` is recommended.

From Figures 3 and 5, `PD-LM1-b` and `PD-LM3-b`, as well as `PD-LM1-c` and `PD-LM3-c`, exhibit the best efficiency and robustness in computing approximate global minimizers. In contrast, Figures 4 and 6 indicate that `PD-D-b` and `PD-D-c` are the most efficient and robust methods for computing `CC-S` stationary points. These observations reveal a clear trade-off between the two performance criteria, making it difficult to identify a single uniformly best-performing variant. Nevertheless, since the accurate and efficient computation of `CC-S` stationary points is our primary objective, we select `PD-D-b` and `PD-D-c` for subsequent experiments.

We finally compare `PD-LM1-a`, `PD-D-b`, and `PD-D-c` to identify the best overall version of our algorithm. From Figure 7, `PD-LM1-a` is the most efficient and robust method for computing approximate global minimizers. Moreover, Figure 8 shows that `PD-LM1-a` is the most efficient and achieves robustness comparable to the other variants in computing `CC-S` stationary points. Consequently, `PD-LM1-a` is selected as the best-performing version of the proposed algorithm.

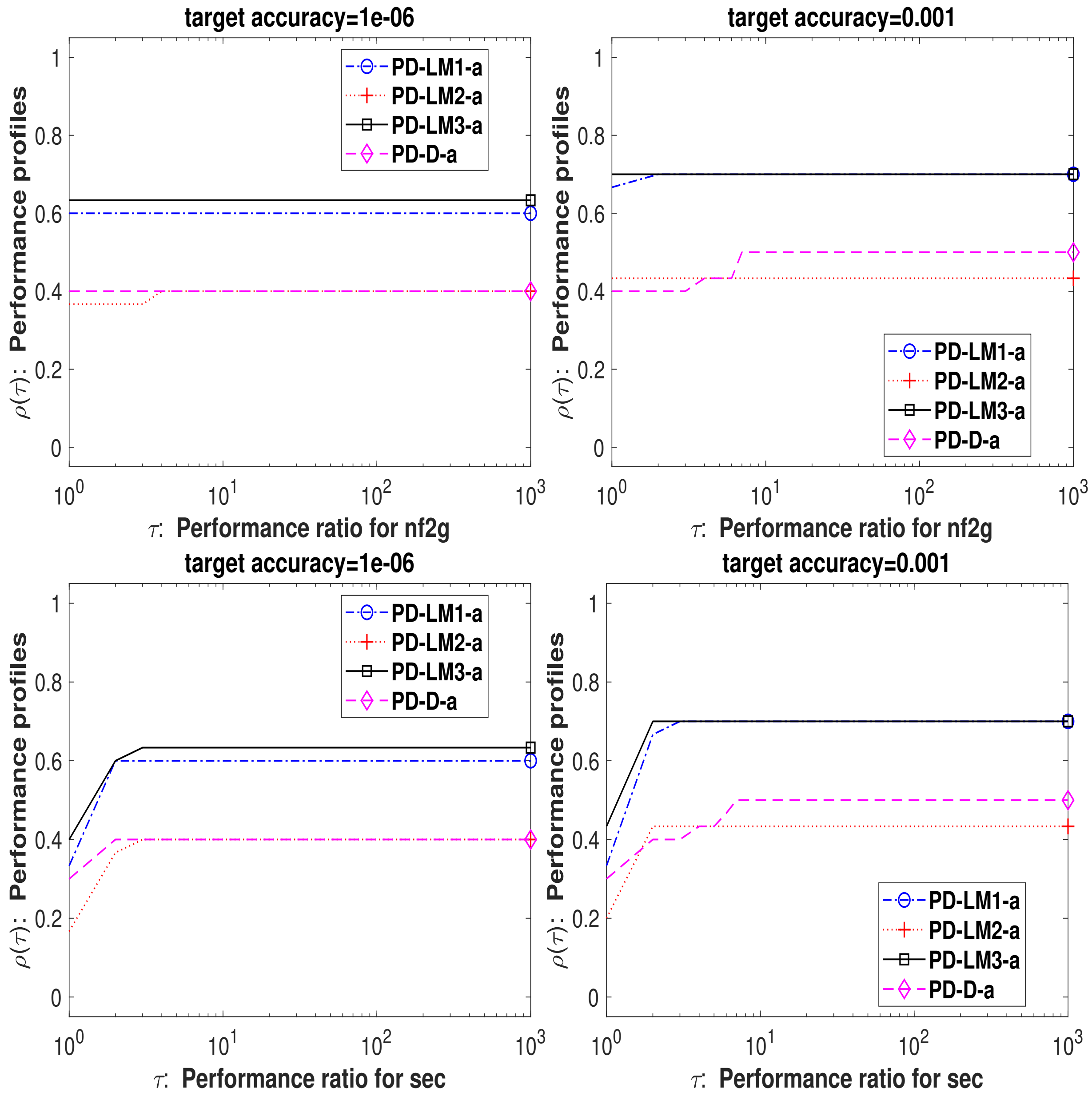

Figure 1: Performance profiles of `PD-LM1-a`, `PD-LM2-a`, `PD-LM3-a`, and `PD-D-a` in terms of `nf2g` (first row) and `sec` (second row), and with $q_{\text{sol}} \leq 10^{-6}$ (left) and $q_{\text{sol}} \leq 10^{-3}$ (right).

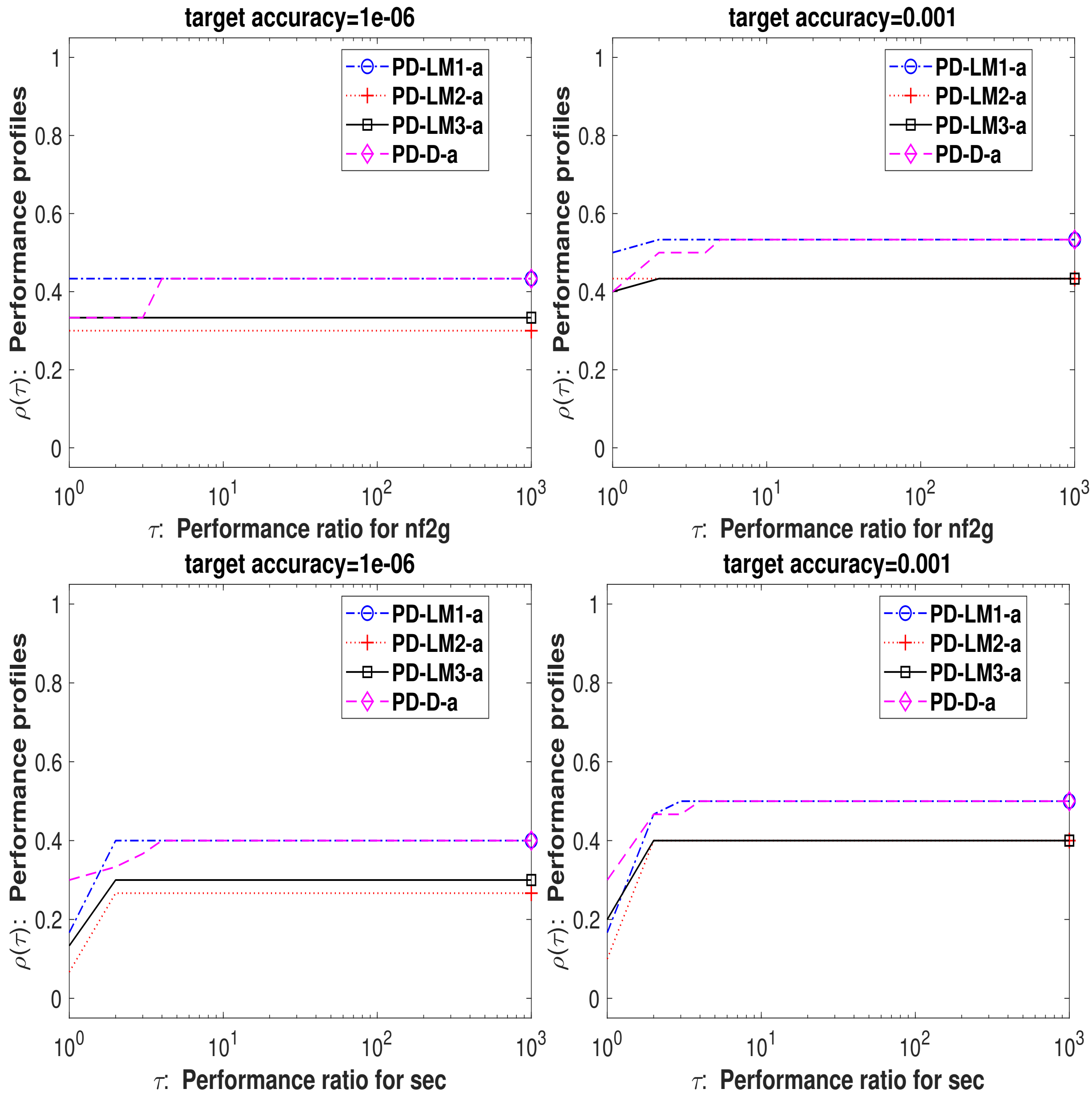

Figure 2: Performance profiles of `PD-LM1-a`, `PD-LM2-a`, `PD-LM3-a`, and `PD-D-a` in terms of `nf2g` (first row) and `sec` (second row), and with $\mathrm{rg}_S \leq 10^{-6}$ (left) and $\mathrm{rg}_S \leq 10^{-3}$ (right).

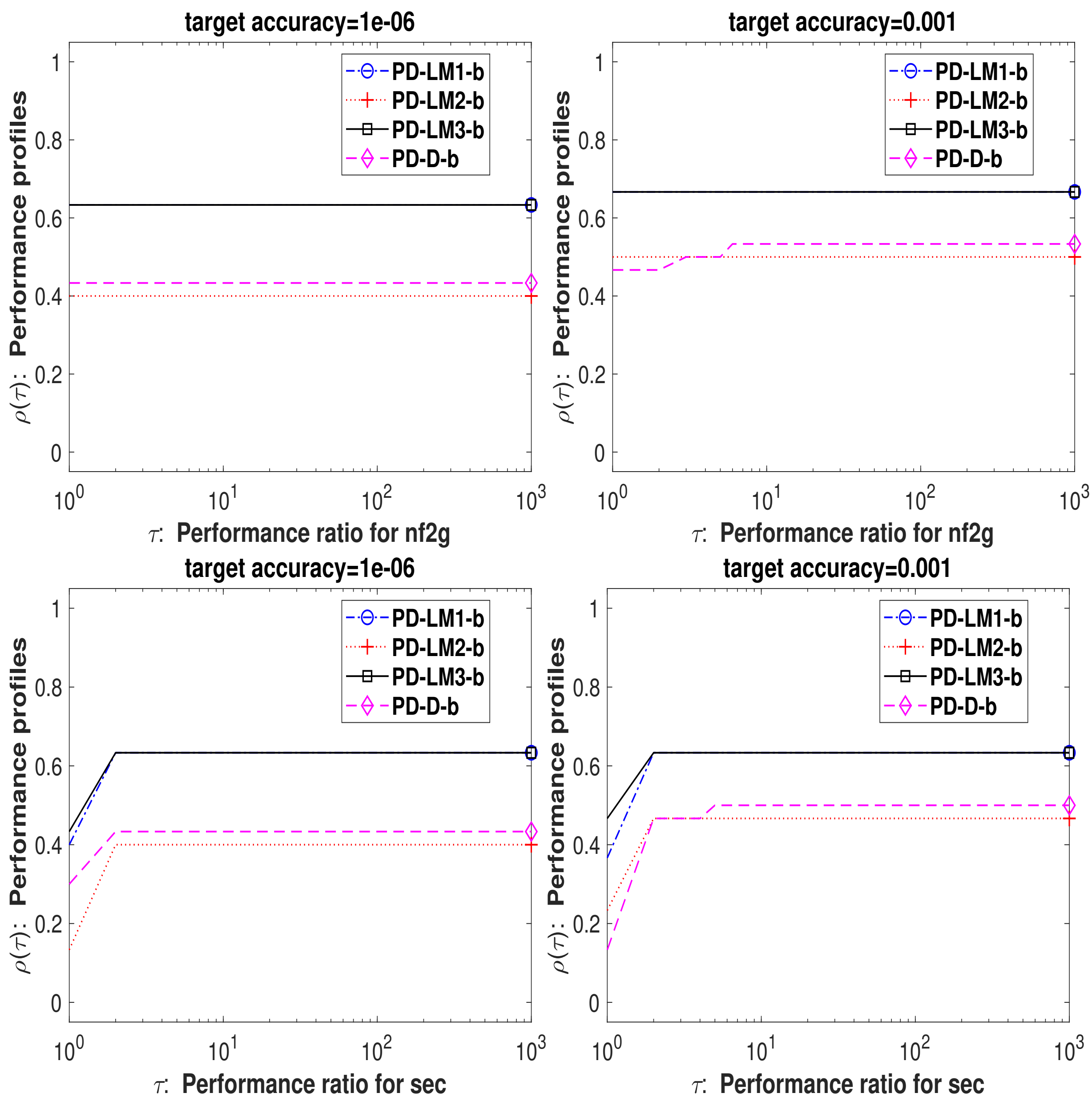

Figure 3: Performance profiles of `PD-LM1-b`, `PD-LM2-b`, `PD-LM3-b`, and `PD-D-b` in terms of `nf2g` (first row) and `sec` (second row), and with $q_{\rm sol} \leq 10^{-6}$ (left) and $q_{\rm sol} \leq 10^{-3}$ (right).

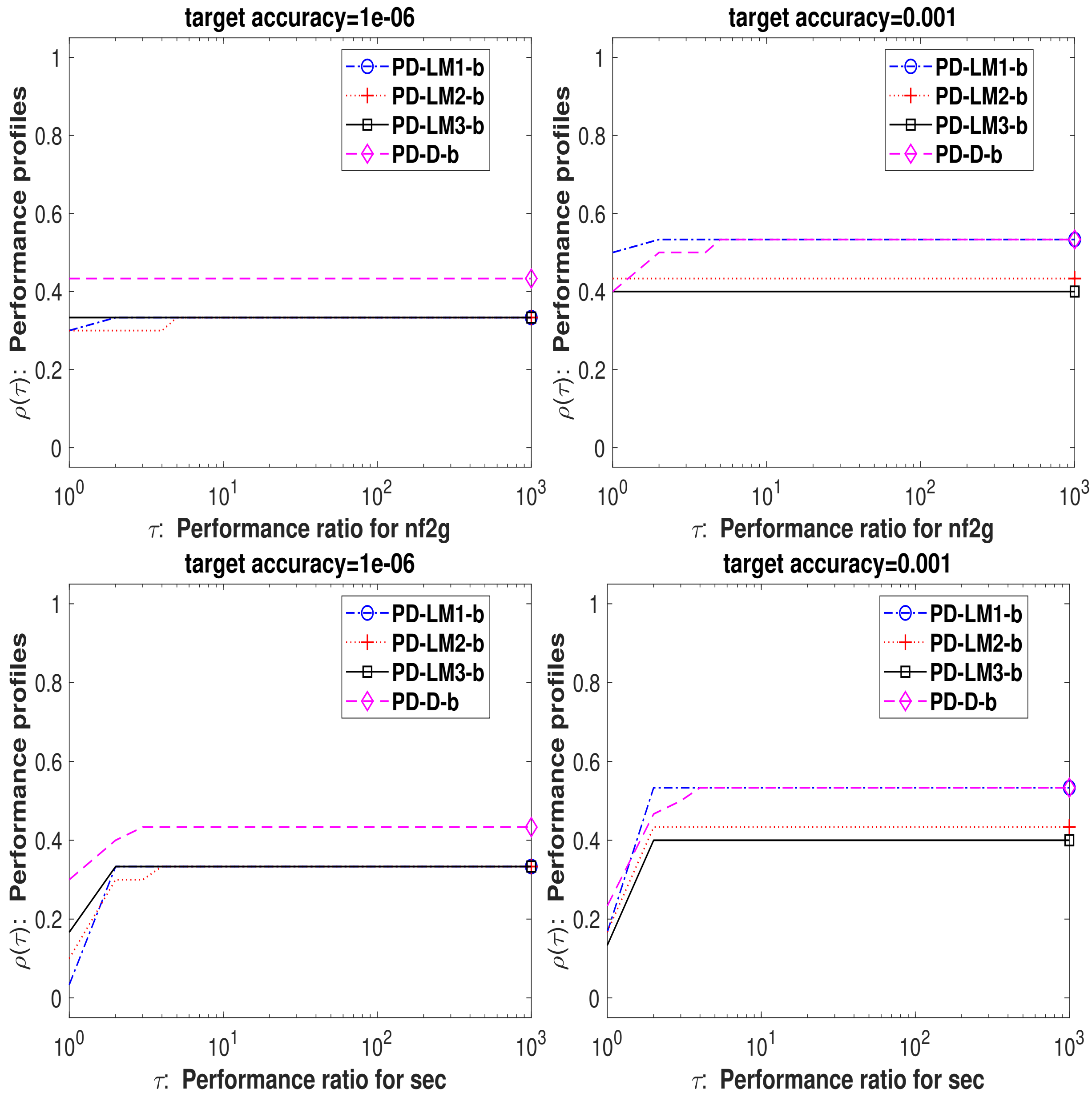

Figure 4: Performance profiles of `PD-LM1-b`, `PD-LM2-b`, `PD-LM3-b`, and `PD-D-b` in terms of `nf2g` (first row) and `sec` (second row), and with $\mathrm{rg}_S \le \mathbf{10^{-6}}$ (left) and $\mathrm{rg}_S \le \mathbf{10^{-3}}$ (right).

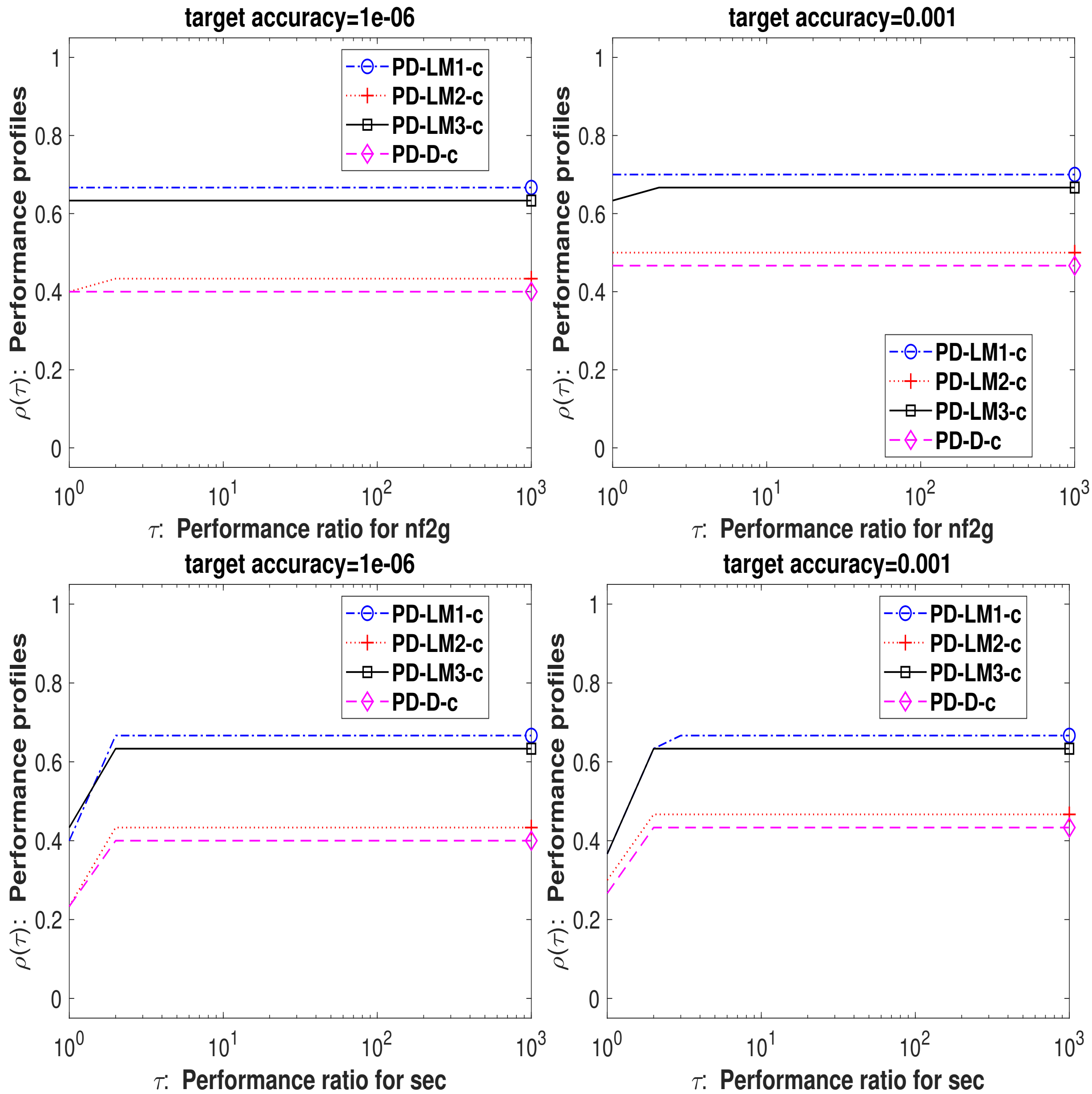

Figure 5: Performance profiles of `PD-LM1-c`, `PD-LM2-c`, `PD-LM3-c`, and `PD-D-c` in terms of `nf2g` (first row) and `sec` (second row), and with $q_{\text{sol}} \le 10^{-6}$ (left) and $q_{\text{sol}} \le 10^{-3}$ (right).

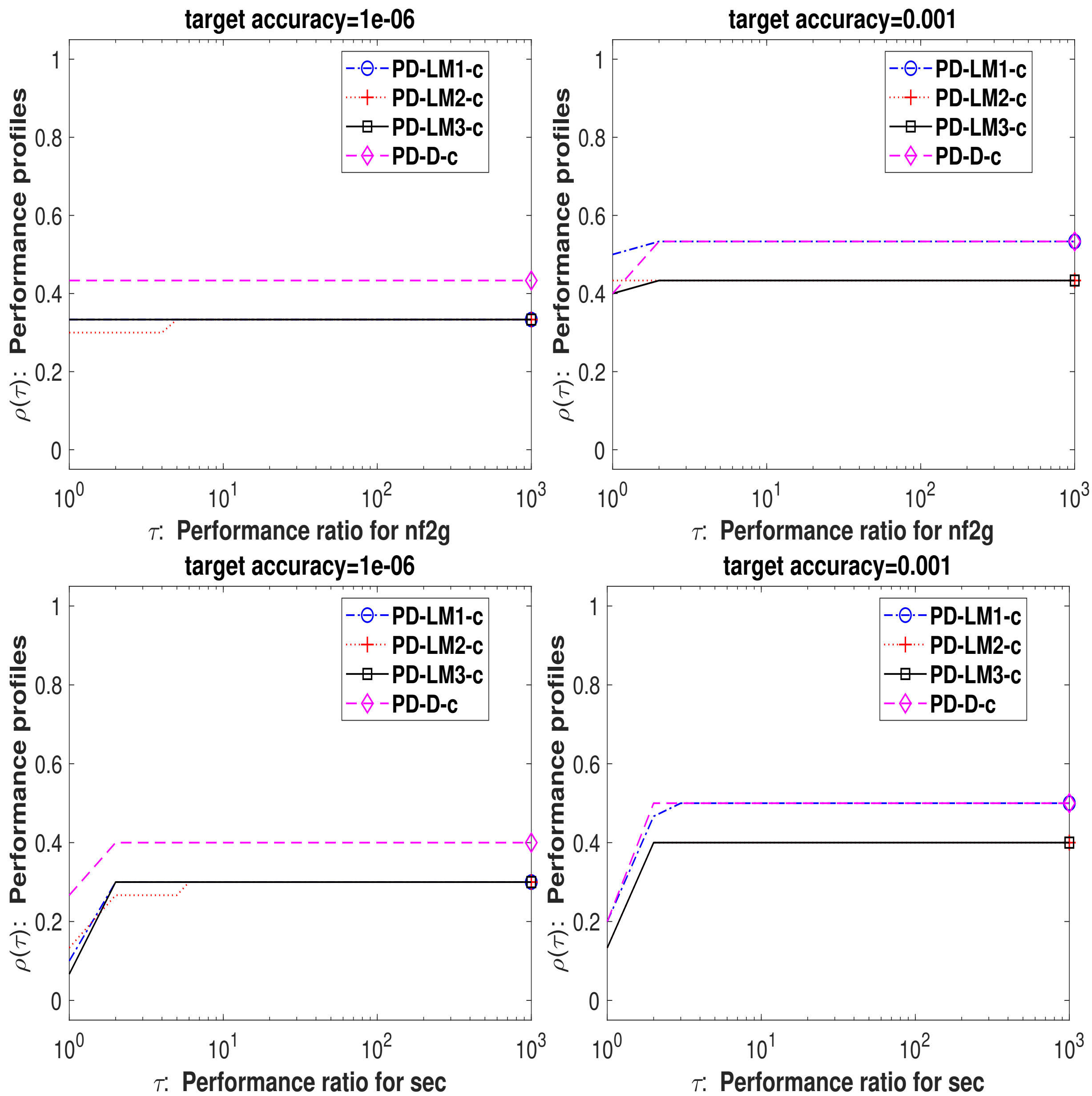

Figure 6: Performance profiles of `PD-LM1-c`, `PD-LM2-c`, `PD-LM3-c`, and `PD-D-c` in terms of `nf2g` (first row) and `sec` (second row), and with $\mathrm{rg}_S \leq \mathbf{10}^{-6}$ (left) and $\mathrm{rg}_S \leq \mathbf{10}^{-3}$ (right).

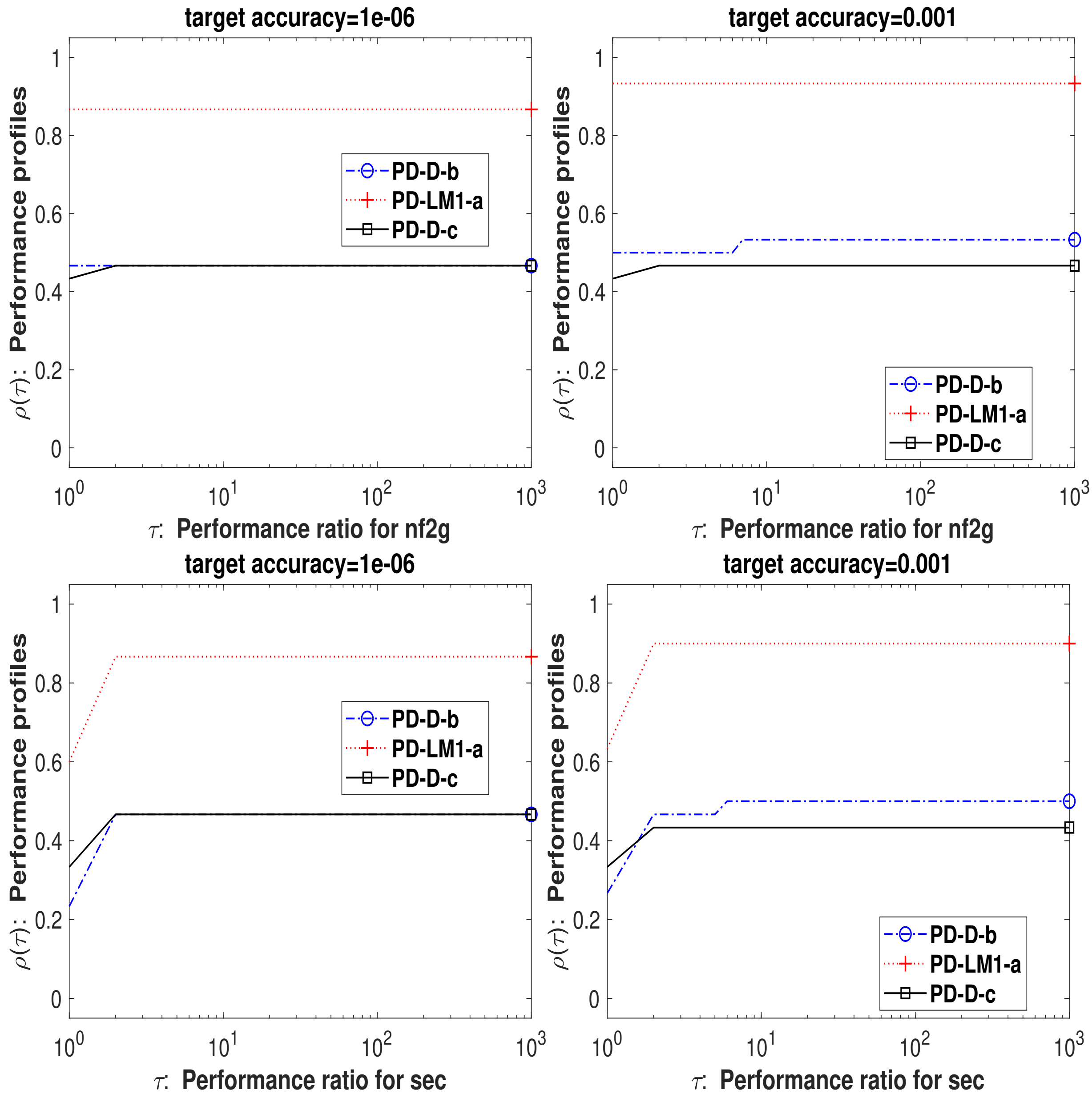

Figure 7: Performance profiles of `PD-LM1-a`, `PD-D-b`, and `PD-D-c` in terms of `nf2g` (first row) and `sec` (second row), and with $q_{\rm sol} \le 10^{-6}$ (left) and $q_{\rm sol} \le 10^{-3}$ (right).

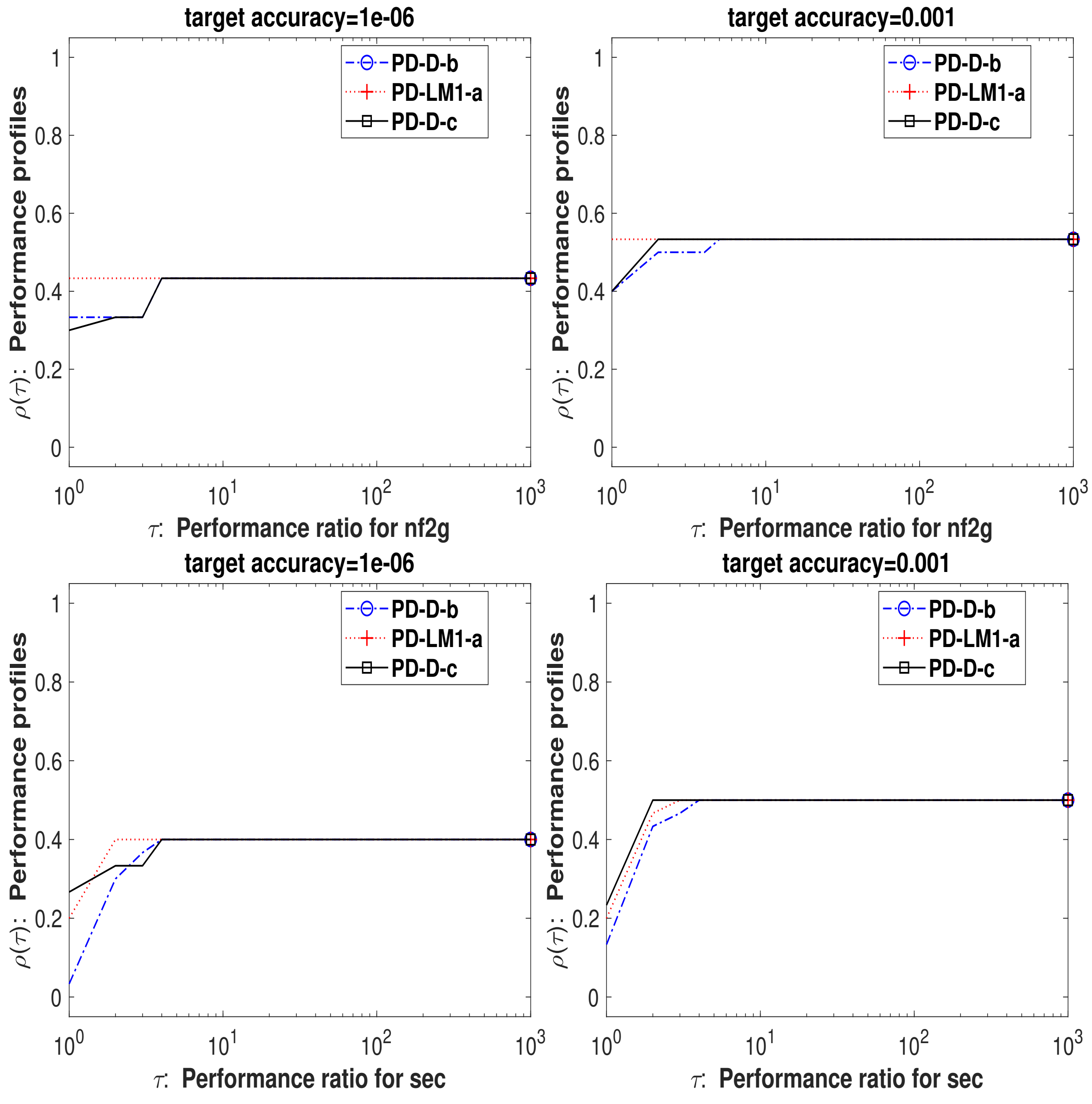

Figure 8: Performance profiles of `PD-LM1-a`, `PD-D-b`, and `PD-D-c` in terms of `nf2g` (first row) and `sec` (second row), and with $\mathrm{rg}_S \leq \mathbf{10}^{-6}$ (left) and $\mathrm{rg}_S \leq \mathbf{10}^{-3}$ (right).

References

[1] Zohre Aminifard and Saman Babaie–Kafaki. A nonmonotone ADMM-based diagonal quasi-Newton update with application to the compressive sensing problem. Mathematical Modelling and Analysis, 28(4):673–688, October 2023.

[2] Neculai Andrei. A diagonal quasi-Newton updating method based on minimizing the measure function of Byrd and Nocedal for unconstrained optimization. Optimization, 67(9):1553–1568, June 2018.

[3] Amir Beck and Nadav Hallak. On the minimization over sparse symmetric sets: Projections, optimality conditions, and algorithms. Mathematics of Operations Research, 41(1):196–223, February 2016.

[4] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific Computing, 16(5):1190–1208, September 1995.

[5] Christian Kanzow, Andreas B. Raharja, and Alexandra Schwartz. Sequential optimality conditions for cardinality-constrained optimization problems with applications. Computational Optimization and Applications, 80:185–211, July 2021.

[6] Morteza Kimiaei, Arnold Neumaier, and Behzad Azmi. LMBOPT: A limited memory method for bound-constrained optimization. Mathematical Programming Computation, 14(2):271–318, January 2022.

[7] Boris S. Mordukhovich. Variational Analysis and Generalized Differentiation I. Springer Berlin Heidelberg, 2006.

[8] Ahmad Mousavi, Morteza Kimiaei, Saman Babaie–Kafaki, and Vyacheslav Kungurtsev. An efficient penalty decomposition algorithm for minimization over sparse symmetric sets. `https://github.com/GS1400/suppMat_CC_PDQN`.

[9] Jorge Nocedal and Stephen J. Wright. Numerical Optimization. Springer, New York, 2006.

[10] R. Tyrrell Rockafellar and Roger J. B. Wets. Variational Analysis. Springer, Berlin, 1998.

[11] David S. Watkins. Fundamentals of Matrix Computations. John Wiley and Sons, New York, 2002.