

Noname manuscript No.
(will be inserted by the editor)

A difference-of-convex approach for split feasibility with applications to matrix factorizations and outlier detection

Chen Chen · Ting Kei Pong · Lulin Tan ·
Liaoyuan Zeng

February 23, 2020

Abstract The split feasibility problem is to find an element in the intersection of a closed set C and the linear preimage of another closed set D , assuming the projections onto C and D are easy to compute. This class of problems arises naturally in many contemporary applications such as compressed sensing. While the sets C and D are typically assumed to be convex in the literature, in this paper, we allow both sets to be possibly nonconvex. We observe that, in this setting, the split feasibility problem can be formulated as an optimization problem with a difference-of-convex objective so that standard majorization-minimization type algorithms can be applied. Here we focus on the nonmonotone proximal gradient algorithm with majorization studied in [22, Appendix A]. We show that, when this algorithm is applied to a split feasibility problem, the sequence generated clusters at a stationary point of the problem under mild assumptions. We also study local convergence property of the sequence under suitable assumptions on the closed sets involved. Finally, we perform numerical experiments to illustrate

Ting Kei Pong's research was supported partly by Hong Kong Research Grants Council PolyU153085/16p.

Lulin Tan's research was supported partly by Natural Science Foundation of China (No. 11601162) and Natural Science Foundation of Guangdong Province, China (No. 2017A030310167).

Chen Chen
School of Mathematical Sciences, South China Normal University, Guangzhou 510631, China
E-mail: chenchen@m.scnu.edu.cn

Ting Kei Pong
Department of Applied Mathematics, The Hong Kong Polytechnic University, Hong Kong, People's Republic of China
E-mail: tk.pong@polyu.edu.hk

Lulin Tan
School of Mathematical Sciences, South China Normal University, Guangzhou 510631, China
E-mail: lulin_9@hotmail.com

Liaoyuan Zeng
Department of Applied Mathematics, The Hong Kong Polytechnic University, Hong Kong, People's Republic of China
E-mail: lyzeng@polyu.edu.hk

the efficiency of our approach on solving split feasibility problems that arise in completely positive matrix factorization, (uniformly) sparse matrix factorization, and outlier detection.

1 Introduction

The split feasibility problem aims at finding an element common to a closed set C and the linear preimage of another closed set D , under the assumption that the projections onto C and D can be computed efficiently. This latter assumption is satisfied by a large class of closed convex sets (to which the projection is unique) including some simple polyhedral sets, and many widely used nonconvex sets such as the set of s -sparse vectors (see, for example, [24, Proposition 3.1]), the simplex with additional cardinality constraints [18], the set of orthogonal matrices [1, Proposition 7] and the set of matrices of rank at most r [15], etc. The split feasibility problem was first introduced in [11], and has found various applications, such as compressed sensing, signal processing, image reconstruction and intensity modulated therapy; see, for example, [10, 12, 23, 33] and references therein.

Although the split feasibility problem can be seen as a special case of the classical feasibility problem that finds a point in the intersection of two closed sets, a direct application of algorithms for feasibility problems such as the alternating projection method and the Douglas-Rachford splitting method may not be desirable. This is because, in a split feasibility problem, we only assume that the projections onto C and D are easy to compute; in particular, it can be difficult to project onto the linear preimage of D , rendering a direct application of classical methods for feasibility problems inefficient. Specialized algorithms have thus been proposed for solving split feasibility problems, using only projections onto C and D as well as applications of the linear map and its adjoint. However, most existing work on split feasibility problems focuses on the convex settings, i.e., they assume C and D are also convex; see, for example [10–13, 23, 29, 31–35]. This does not cover contemporary applications that involve nonconvex constraints.

In this paper, we consider the split feasibility problem in a possibly nonconvex setting, i.e., we allow the sets C and D to be possibly nonconvex. We propose an algorithm for solving it and analyze its global and local convergence properties. The algorithm we propose can be viewed as a generalization of the classical CQ algorithm, which was proposed in [10] for convex split feasibility problems. Indeed, as we will discuss in more detail in Section 3, the split feasibility problem can be reformulated into a special possibly nonconvex optimization problem with a difference-of-convex objective function so that standard majorization-minimization type algorithms can be employed. Our main algorithm is an adaptation of the majorization-minimization type algorithm proposed in [22, Appendix A] to solve this special optimization problem. When C and D are both convex and a constant stepsize strategy is adopted, our proposed algorithm reduces to the classical CQ algorithm.

Since we are solving the split feasibility problem via solving a nonconvex optimization problem, one cannot expect to obtain a global minimizer in general. Instead, we define a new concept of stationary point for split feasibility problem in Definition 3 below and show that, under mild assumptions, any cluster point of the sequence generated by our algorithm is a *stationary point* of the split feasi-

bility problem. The whole sequence generated is further shown to be convergent under additional assumptions such as the Kurdyka-Lojasiewicz property [3–5] and Lipschitz differentiability at the limit point. Furthermore, we also analyze local convergence rate, based on the Kurdyka-Lojasiewicz exponent [3, 4, 20] and a generalization of the concept of linearly regular intersection: the concept of linearly regular intersection was proposed in [19] for studying local convergence rate of the alternating projection method for feasibility problems.

Finally, we perform numerical experiments to illustrate the efficiency of our method for solving split feasibility problems. Specifically, we perform numerical experiments on the completely positive matrix factorization problem, the (uniformly) sparse matrix factorization problem and an outlier detection problem. In particular, for the completely positive matrix factorization problem, we follow the approach in [17] to reformulate the completely positive matrix factorization problem into a nonconvex split feasibility problem. Our numerical results show that our method always outperforms [17, Algorithm 2] in terms of both CPU time and solution quality.

The rest of the paper is organized as follows. In Section 2, we introduce notation and some preliminary results. The mathematical formulation of the split feasibility problem and our main algorithm for solving it are described in Section 3. In Section 4, we study subsequential convergence of the sequence generated by our algorithm. The global (sequential) convergence and the local convergence rate of the sequence generated by our algorithm are studied in Section 5. Finally, in Sections 6, 7 and 8, we discuss how our algorithm can be applied to solving the completely positive matrix factorization problem, the (uniformly) sparse matrix factorization problem and an outlier detection problem, respectively, and perform numerical experiments to study the performance of our algorithm.

2 Notation and preliminaries

In this paper, we let \mathbb{R}^n denote the n -dimensional Euclidean space. For a vector $x \in \mathbb{R}^n$, we denote its Euclidean norm, ℓ_1 norm and ℓ_∞ norm by $\|x\|$, $\|x\|_1$ and $\|x\|_\infty$ respectively. We also let $B(x, r)$ denote the closed ball centered at x with radius r , i.e., $B(x, r) = \{u \in \mathbb{R}^n : \|u - x\| \leq r\}$.

An extended real valued function $f : \mathbb{R}^n \rightarrow (-\infty, \infty]$ is said to be proper if $\text{dom} f := \{x \in \mathbb{R}^n : f(x) < \infty\} \neq \emptyset$. Such a function is said to be closed if it is lower semicontinuous. For a proper closed function f , the regular subdifferential and the (limiting) subdifferential of f at an $\bar{x} \in \text{dom} f$ are respectively defined as [30, Definition 8.3]

$$\hat{\partial}f(\bar{x}) := \left\{ v \in \mathbb{R}^n : \liminf_{x \rightarrow \bar{x}, x \neq \bar{x}} \frac{f(x) - f(\bar{x}) - \langle v, x - \bar{x} \rangle}{\|x - \bar{x}\|} \geq 0 \right\},$$

and

$$\partial f(\bar{x}) := \{v \in \mathbb{R}^n : \exists x^t \xrightarrow{f} \bar{x} \text{ and } v^t \rightarrow v \text{ with } v^t \in \hat{\partial}f(x^t) \text{ for each } t\},$$

where $x^t \xrightarrow{f} \bar{x}$ means both $f(x^t) \rightarrow f(\bar{x})$ and $x^t \rightarrow \bar{x}$. By convention, we also set $\hat{\partial}f(x) = \partial f(x) = \emptyset$ if $x \notin \text{dom} f$. We let $\text{dom} \partial f$ denote the domain of subdifferential, which is defined as $\text{dom} \partial f := \{x \in \mathbb{R}^n : \partial f(x) \neq \emptyset\}$. It is known in [30, Exercise 8.8]

that if f is continuously differentiable at $x \in \mathbb{R}^n$, then $\partial f(x) = \{\nabla f(x)\}$. In addition, if f is proper convex, then ∂f coincides with the notion of subdifferential in convex analysis; see [30, Proposition 8.12].

For a nonempty closed set $C \subseteq \mathbb{R}^n$, we let C^∞ denote the horizon cone of C , which is defined in [30, Definition 3.3] as

$$C^\infty := \{x \in \mathbb{R}^n : \exists x^t \in C \text{ and } \beta_t \downarrow 0 \text{ with } \beta_t x^t \rightarrow x\}.$$

It is known that C is compact if and only if $C^\infty = \{0\}$; see [30, Theorem 3.5]. We also let δ_C denote the indicator function of C , which is zero in C and equals infinity otherwise. The distance from x to C is denoted by $d(x, C) := \inf_{u \in C} \|x - u\|$, and we use $\text{Proj}_C(x)$ to denote the set of projections of x onto C , which is defined as

$$\text{Proj}_C(x) := \underset{y \in C}{\text{Argmin}} \|y - x\|,$$

where Argmin denotes the set of minimizers. The set of projections onto the nonempty closed set C is always nonempty, and reduces to a singleton set if C is in addition convex. The regular normal cone and the (limiting) normal cone of a nonempty closed set C at an $x \in C$ are defined by $\hat{N}_C(x) := \hat{\partial}\delta_C(x)$ and $N_C(x) := \partial\delta_C(x)$ respectively. These notions of normal cones are closely related to projections. Indeed, if $x \in \text{Proj}_C(y)$, then we have from [30, Example 6.16] and [30, Proposition 6.5] that

$$y - x \in \hat{N}_C(x) \subseteq N_C(x). \quad (1)$$

Finally, following [30, Definition 6.4] and [30, Definition 7.25], we say that a nonempty closed set C is (Clarke) regular at an $x \in C$ if $N_C(x) = \hat{N}_C(x)$, and a proper closed function f is regular at an $x \in \text{dom } f$ if its epigraph $\text{epi } f := \{(x, t) \in \mathbb{R}^n \times \mathbb{R} : f(x) \leq t\}$ is regular at $(x, f(x))$.

We next recall the Kurdyka-Łojasiewicz (KL) property [4]. This property has been used extensively in recent years for analyzing the rate of convergence of various first-order methods, especially in a nonconvex setting; see, for example, [3–5].

Definition 1 (KL property) We say that a proper closed function f satisfies the KL property at $\bar{x} \in \text{dom}\partial f$ if there exist a neighborhood U of \bar{x} , $s \in (0, \infty]$ and a continuous concave function $\psi : [0, s) \rightarrow \mathbb{R}_+$ with $\psi(0) = 0$ such that:

- (i) ψ is continuously differentiable on $(0, s)$ with $\psi' > 0$;
- (ii) for all $x \in U$ with $f(\bar{x}) < f(x) < f(\bar{x}) + s$, one has

$$\psi'(f(x) - f(\bar{x}))d(0, \partial f(x)) \geq 1.$$

A proper closed function f satisfying the KL property at all points in $\text{dom}\partial f$ is called a KL function.

Functions satisfying the KL property arise naturally in many applications. In particular, it is known that any proper closed semialgebraic function is a KL function; see [4, 8] for more examples. Moreover, for proper closed semialgebraic functions, the ψ in Definition 1 can actually be chosen as $\psi(a) = ca^{1-\theta}$ for some $c > 0$ and $\theta \in [0, 1)$; see [4, Section 4.3] and references therein. This exponent θ is important in estimating the rate of convergence of sequences generated by various first-order methods; see, for example, [3, 4, 20].

Definition 2 (KL exponent) Let $\theta \in [0, 1)$ and f be a proper closed function. We say that f satisfies the KL property at $\bar{x} \in \text{dom}\partial f$ with exponent θ if there exist $c, \epsilon > 0$ and $s \in (0, +\infty]$ such that

$$d(0, \partial f(x)) \geq c(f(x) - f(\bar{x}))^\theta$$

whenever $\|x - \bar{x}\| \leq \epsilon$ and $0 < f(x) - f(\bar{x}) < s$. If f satisfies the KL property with exponent θ at every $x \in \text{dom}\partial f$, then we say that f is a KL function with exponent θ .

3 Problem statement and difference-of-convex reformulation

In this section, we give the formal mathematical definition of the split feasibility problem and describe the basic ideas leading to our solution strategy. Precisely, the split feasibility problem [11] is stated as follows: Given a matrix $A \in \mathbb{R}^{m \times n}$, and two nonempty closed sets $C \subseteq \mathbb{R}^n$ and $D \subseteq \mathbb{R}^m$,

$$\text{Find } x \in \mathbb{R}^n \text{ s.t. } Ax \in D \text{ and } x \in C; \quad (2)$$

here, we assume that an element of $\text{Proj}_C(x)$ and $\text{Proj}_D(x)$ can be computed efficiently for any given x . The above problem arises in various contemporary applications. For instance, the *noiseless* compressed sensing problem was modeled as an instance of (2) in [23, Section 6], where D is the singleton set containing the noiseless measurement and C is the ℓ_1 norm ball of suitable radius; notice that $\text{Proj}_C(x)$ and $\text{Proj}_D(x)$ can be computed efficiently for this choice of C and D .

We would like to point out that (2) can also be viewed as a special case of the classical feasibility problem, where one attempts to find a point in the intersection of two closed sets: in this case, $A^{-1}D$ and C . However, classical algorithms for such a feasibility problem typically involve $\text{Proj}_{A^{-1}D}(x)$, which can be hard to compute even though $\text{Proj}_D(x)$ can be computed efficiently. Thus, specialized algorithms have been designed for solving (2), making use of only projections onto C and D as well as multiplications by the matrix A and its transpose. Almost all algorithms proposed for solving (2) were for the *convex setting*, i.e., when C and D are both in addition convex. One classical algorithm is the so-called CQ algorithm proposed in [10, Algorithm 1.1], which takes the following form: given $x^0 \in \mathbb{R}^n$ and $\gamma \in (0, \frac{2}{\lambda_{\max}(A^T A)})$, update

$$x^{t+1} = \text{Proj}_C \left(x^t - \gamma A^T [Ax^t - \text{Proj}_D(Ax^t)] \right). \quad (3)$$

After the proposal of the CQ algorithm, many other algorithms for solving the split feasibility problem (2) in the convex setting have been proposed; we refer the interested readers to [29, 34, 35] for more detail.

In this paper, we consider the split feasibility problem (2) in a possibly nonconvex setting, i.e., we allow the sets C and D to be possibly nonconvex. Our approach is based on a (standard) reformulation of (2) into the following optimization problem:

$$\min_x F(x) := \frac{1}{2} d^2(Ax, D) + \delta_C(x). \quad (4)$$

Indeed, it is easy to see that (2) is solved if and only if (4) has an optimal solution with the optimal value being zero. Thus, in order to solve (2), it suffices to solve (4).

In the case when C and D are both convex, problem (4) is a convex optimization problem and the function $x \mapsto \frac{1}{2}d^2(Ax, D)$ is smooth with Lipschitz gradient whose modulus is $\lambda_{\max}(A^T A)$. Thus, one can apply first-order methods such as the proximal gradient algorithm and its variant for solving (4) efficiently; in particular, in each iteration of these algorithms, one only needs to compute the projections onto D (for evaluating the gradient of $\frac{1}{2}d^2(A \cdot, D)$) and C as well as multiplications by A and A^T , which can be done efficiently. Notice that the classical CQ algorithm (3) is just an application of the standard proximal gradient algorithm to (4) in the convex setting; see, for example, the introduction of [23].

However, in the general case when C and D can be both nonconvex, the squared distance function in problem (4) is nonsmooth in general, and the proximal gradient algorithm cannot be applied. Fortunately, it is known that the squared distance function can be written as the difference of two convex functions [2]: specifically, for any $u \in \mathbb{R}^m$, we have

$$\frac{1}{2}d^2(u, D) = \frac{1}{2}\|u\|^2 - \sup_{y \in D} \left\{ \langle u, y \rangle - \frac{1}{2}\|y\|^2 \right\}.$$

Now, notice that the function $u \mapsto \sup_{y \in D} \{ \langle u, y \rangle - \frac{1}{2}\|y\|^2 \}$, as the supremum of affine functions and being finite valued, is convex continuous. Thus, we can write F in (4) as

$$F(x) = \underbrace{\frac{1}{2}\|Ax\|^2}_{h(x)} + \underbrace{\delta_C(x)}_{P(x)} - \underbrace{\sup_{y \in D} \left\{ \langle Ax, y \rangle - \frac{1}{2}\|y\|^2 \right\}}_{g(x)}, \quad (5)$$

where h is quadratic, P is proper closed and g is convex continuous. Moreover, under a mild additional assumption, we can show in the next proposition that for any $x^0 \in C$, the set $\{x : F(x) \leq F(x^0)\}$ is bounded.

Lemma 1 *Consider the split feasibility problem (2) with $C^\infty \cap A^{-1}D^\infty = \{0\}$. Then for any $x^0 \in C$, the set $\{x : F(x) \leq F(x^0)\}$ is bounded, where F is defined in (4).*

Proof Suppose to the contrary that the set $\{x : F(x) \leq F(x^0)\}$ is unbounded. Then there exists a sequence $\{x^t\}$ such that $F(x^t) \leq F(x^0)$ for all t and $\lim_{t \rightarrow \infty} \|x^t\| = +\infty$. By passing to a subsequence if necessary, we may assume without loss of generality that $\|x^t\| \neq 0$ for all t and that $\frac{x^t}{\|x^t\|} \rightarrow y^*$ for some y^* . Then it holds that $\|y^*\| = 1$ and we also have $y^* \in C^\infty$ from the definition of the horizon cone. Next, for each t , we have

$$\frac{1}{2}d^2(Ax^t, D) = F(x^t) \leq F(x^0) = \frac{1}{2}d^2(Ax^0, D).$$

Hence there exists $\{d^t\} \subseteq D$ such that for each t ,

$$\|Ax^t - d^t\| \leq d(Ax^0, D).$$

Since $\|x^t\| \neq 0$ for all t , dividing both sides of the above inequality by $\|x^t\|$, we have for all t that

$$\left\| \frac{Ax^t}{\|x^t\|} - \frac{d^t}{\|x^t\|} \right\| \leq \frac{d(Ax^0, D)}{\|x^t\|}.$$

Passing to the limit as $t \rightarrow \infty$ in the above inequality and noting that $\frac{x^t}{\|x^t\|} \rightarrow y^*$, we deduce further that $\frac{d^t}{\|x^t\|} \rightarrow Ay^*$. Thus, we have $Ay^* \in D^\infty$, i.e., $y^* \in A^{-1}D^\infty$, according to the definition of the horizon cone. Since it also holds that $y^* \in C^\infty$ and $\|y^*\| = 1$, we have arrived at a contradiction. This completes the proof. \square

Remark 1 The condition $C^\infty \cap A^{-1}D^\infty = \{0\}$ used in Lemma 1 holds in the following cases.

- (i) The set C is compact: in this case, we can deduce from [6, Proposition 2.1.2] that $C^\infty = \{0\}$ and hence $C^\infty \cap A^{-1}D^\infty = \{0\}$.
- (ii) The sets C, D are closed convex with $C \cap A^{-1}D$ being nonempty and bounded: in this case, we have

$$C^\infty \cap A^{-1}D^\infty = C^\infty \cap (A^{-1}D)^\infty = (C \cap A^{-1}D)^\infty = \{0\},$$

where the first equality follows from [6, Proposition 2.1.11] and the fact that $A^{-1}D \neq \emptyset$, the second equality follows from [6, Proposition 2.1.9] and the fact that $C \cap A^{-1}D \neq \emptyset$, and the last equality follows from [6, Proposition 2.1.2].

Consequently, thanks to Lemma 1, under the additional assumption that $C^\infty \cap A^{-1}D^\infty = \{0\}$, the function $F = h + P - g$ given in (5) satisfies all the assumptions necessary for applying the so-called $\text{NPG}_{\text{major}}$ in [22, Appendix A]. In addition, since

$$A^T \text{Proj}_D(Ax) \subseteq \partial g(x), \quad (6)$$

one can choose in **Step 1a)** of the $\text{NPG}_{\text{major}}$ any $\eta^t \in \text{Proj}_D(Ax^t)$ so that $\zeta^t := A^T \eta^t \in \partial g(x^t)$. Moreover, using the definition of h and P in (5), the subproblem of $\text{NPG}_{\text{major}}$ in [22, Eq (45)] becomes

$$u \in \text{Proj}_C \left(x^t - \frac{A^T(Ax^t - \eta^t)}{L_t} \right).$$

Having these in mind, we are now ready to present our algorithm $\text{SpFeas}_{\text{DC}_s}$ as Algorithm 1 below for solving (2), which is basically an application of the $\text{NPG}_{\text{major}}$ in [22, Appendix A] to (4).

Algorithm 1 $\text{SpFeas}_{\text{DC}_s}$ for (2)

Step 0. Choose $x^0 \in C$, $L_{\max} \geq L_{\min} > 0$, $\tau > 1$, $c > 0$ and an integer $M \geq 0$. Set $t = 0$.

Step 1. Choose any $L_t^0 \in [L_{\min}, L_{\max}]$ and set $L_t = L_t^0$.

1a) Pick any $\eta^t \in \text{Proj}_D(Ax^t)$ and find

$$u \in \text{Proj}_C \left(x^t - \frac{A^T(Ax^t - \eta^t)}{L_t} \right).$$

1b) If

$$d^2(Au, D) \leq \max_{[t-M]_+ \leq i \leq t} d^2(Ax^i, D) - c\|u - x^t\|^2 \quad (7)$$

is satisfied, go to **Step 2)**.

1c) Set $L_t \leftarrow \tau L_t$ and go to **Step 1a)**.

Step 2. If a termination criterion is not met, set $\bar{L}_t = L_t$, $x^{t+1} = u$, $t = t + 1$. Go to **Step 1**.

Notice that each iteration of $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ only involves projections onto C and D as well as multiplications by the matrix A and its transpose, which can be performed efficiently. Moreover, when $C^\infty \cap A^{-1}D^\infty = \{0\}$, it can be shown as in [22, Proposition 1] that the linesearch criterion in **Step 1b**) must be satisfied after finitely many inner iterations (independent of t), and as in [22, Proposition 2] that successive changes of the sequence $\{x^t\}$ go to zero. In summary, we have the following convergence result as an immediate corollary of [22, Proposition 1] and [22, Proposition 2].

Lemma 2 *Consider the split feasibility problem (2) with $C^\infty \cap A^{-1}D^\infty = \{0\}$ and let $\{x^t\}$ and $\{\bar{L}_t\}$ be generated by $\text{SpFeas}_{\text{DC}_{\text{ls}}}$. Then it holds that $\sup_t \bar{L}_t < \infty$ and $\lim_{t \rightarrow \infty} \|x^{t+1} - x^t\| = 0$.*

Using Lemma 2, it is routine to show that, when $C^\infty \cap A^{-1}D^\infty = \{0\}$, any accumulation point \bar{x} of $\{x^t\}$ generated by $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ satisfies

$$0 \in \partial h(\bar{x}) + \partial P(\bar{x}) - \partial g(\bar{x}), \quad (8)$$

where h , P and g are given in (5). However, in view of the structure of g , it is not trivial to completely characterize the set ∂g so as to relate (8) to the original split feasibility problem (2). In the next section, we will look at another characterization of the set of accumulation points of $\{x^t\}$ that is more closely tied with (2).

4 Subsequential convergence of $\text{SpFeas}_{\text{DC}_{\text{ls}}}$

In this section, we characterize the set of accumulation points of the sequence $\{x^t\}$ generated by $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ under the assumption $C^\infty \cap A^{-1}D^\infty = \{0\}$. We start with the following proposition, which concerns the subdifferential of F in (4).

Proposition 1 *Consider the split feasibility problem (2) and let F be defined in (4). Then for every $x \in C$, we have*

$$\partial F(x) \subseteq A^T Ax - A^T \text{Proj}_D(Ax) + N_C(x). \quad (9)$$

If in addition C is regular at some $\bar{x} \in C$ and the function $y \mapsto d_D(y)$ is regular at $A\bar{x} \in \mathbb{R}^m$, then we have

$$\partial F(\bar{x}) = A^T A\bar{x} - A^T \text{Proj}_D(A\bar{x}) + N_C(\bar{x}).$$

Remark 2 In view of [30, Example 8.53] and [30, Corollary 8.11], we see that the function $y \mapsto d_D(y)$ is regular at $A\bar{x}$ under one of the following conditions:

- (i) $A\bar{x} \notin D$ and $\text{Proj}_D(A\bar{x})$ is a singleton set;
- (ii) $A\bar{x} \in D$ and D is regular at $A\bar{x}$.

Proof Notice from [30, Example 8.53] and [25, Theorem 1.110(ii)] that for any $y \in \mathbb{R}^m$, we have

$$\partial \left(\frac{1}{2} d^2(\cdot, D) \right) (y) = d(y, D) \partial(d(\cdot, D))(y) = y - \text{Proj}_D(y). \quad (10)$$

This together with [30, Corollary 10.9] and [30, Theorem 10.6] gives (9).

Now, assume in addition that C is regular at some $\bar{x} \in C$ and $y \mapsto d_D(y)$ is regular at $A\bar{x} \in \mathbb{R}^m$. Then at $\bar{y} := A\bar{x}$, we have

$$\begin{aligned} \partial \left(\frac{1}{2} d^2(\cdot, D) \right) (\bar{y}) &\stackrel{(a)}{=} d(\bar{y}, D) \partial (d(\cdot, D)) (\bar{y}) \\ &\stackrel{(b)}{=} d(\bar{y}, D) \hat{\partial} (d(\cdot, D)) (\bar{y}) \stackrel{(c)}{\subseteq} \hat{\partial} \left(\frac{1}{2} d^2(\cdot, D) \right) (\bar{y}) \subseteq \partial \left(\frac{1}{2} d^2(\cdot, D) \right) (\bar{y}) \end{aligned}$$

where (a) follows from [25, Theorem 1.110 (ii)], (b) follows from the assumption that $y \mapsto d_D(y)$ is regular at \bar{y} and [30, Corollary 8.11] and (c) can be verified directly from the definition. Thus,

$$\partial \left(\frac{1}{2} d^2(\cdot, D) \right) (\bar{y}) = \hat{\partial} \left(\frac{1}{2} d^2(\cdot, D) \right) (\bar{y}).$$

This together with [30, Corollary 8.11] implies that the function $y \mapsto \frac{1}{2} d^2(y, D)$ is regular at \bar{y} . Using this together with [30, Theorem 10.6], we deduce further that the function $x \mapsto \frac{1}{2} d^2(Ax, D)$ is regular at \bar{x} and

$$\partial \left(\frac{1}{2} d^2(A\cdot, D) \right) (\bar{x}) = A^T \partial \left(\frac{1}{2} d^2(\cdot, D) \right) (A\bar{x}) = A^T (A\bar{x} - \text{Proj}_D(A\bar{x})),$$

where the last equality follows from (10). Finally, since C is regular at \bar{x} , we have from [30, Example 7.28] that δ_C is regular at \bar{x} . The desired conclusion now follows from [30, Corollary 10.9]. \square

Notice that if x^* solves (2), then it also solves (4). According to [30, Theorem 10.1], we must then have $0 \in \partial F(x^*)$. Motivated by this observation and Proposition 1, we make the following definition.

Definition 3 (Stationary points of (2)) For the split feasibility problem (2), we say that x^* is a stationary point of this problem if

$$0 \in A^T Ax^* - A^T \text{Proj}_D(Ax^*) + N_C(x^*).$$

Based on [30, Theorem 10.1] and Proposition 1, we see that if x^* solves (2) (and hence (4)), then it is a stationary point of (2). Moreover, if C is regular at $x^* \in C$ and $y \mapsto d_D(y)$ is regular at $Ax^* \in \mathbb{R}^m$, then x^* being stationary for problem (2) is the same as x^* being a stationary point of the function F defined in (4), in the sense that $0 \in \partial F(x^*)$. Finally, in view of (6), we deduce that if x^* is a stationary point of (2) in the sense of Definition 3, then it satisfies (8) in place of \bar{x} . This shows that the notion of stationarity of (2) defined in Definition 3 is at least as strong as the condition (8).

We next show that the sequence generated by $\text{SpFeas}_{D_{C_{\text{ls}}}}$ clusters at a stationary point of (2).

Theorem 1 (Subsequential convergence of $\text{SpFeas}_{D_{C_{\text{ls}}}$) Consider the split feasibility problem (2) with $C^\infty \cap A^{-1}D^\infty = \{0\}$ and let $\{x^t\}$ be the sequence generated by $\text{SpFeas}_{D_{C_{\text{ls}}}}$. Then the following statements hold:

- (i) The sequence $\{x^t\}$ is bounded.
- (ii) Any accumulation point of $\{x^t\}$ is a stationary point of (2).

Proof The boundedness of $\{x^t\}$ follows from Lemma 1 and (7).

Next, let x^* be an accumulation point of $\{x^t\}$, which exists because the sequence is bounded. Then there exists a convergent subsequence $\{x^{t_j}\}$ such that $\lim_{j \rightarrow \infty} x^{t_j} = x^*$. Clearly, $x^* \in C$ because C is closed. Now, using (1) and the definition of x^{t+1} as a projection of $x^t - \bar{L}_t^{-1}[A^T(Ax^t - \eta^t)]$ onto C , we have for each j that

$$0 \in \bar{L}_{t_j}(x^{t_j+1} - x^{t_j}) + A^T Ax^{t_j} - A^T \eta^{t_j} + N_C(x^{t_j+1}); \quad (11)$$

moreover, $\{\bar{L}_{t_j}\}$ is bounded thanks to Lemma 2.

On the other hand, notice from the definition of η^{t_j} as projection that $F(x^{t_j}) = \frac{1}{2}\|Ax^{t_j} - \eta^{t_j}\|^2$ for all j , where F is defined in (4). Since $F(x^{t_j}) \leq F(x^0)$ for all j according to (7) and the sequence $\{x^t\}$ is bounded according to (i), we deduce further that $\{\eta^{t_j}\}$ is bounded. By passing to a further subsequence if necessary, we may assume without loss of generality that $\eta^{t_j} \rightarrow \eta^*$ for some η^* . Since $\eta^{t_j} \in \text{Proj}_D(Ax^{t_j})$ and D is closed, we have $\eta^* \in D$. Also, we have

$$\frac{1}{2}d^2(Ax^*, D) = \lim_{j \rightarrow \infty} \frac{1}{2}d^2(Ax^{t_j}, D) = \lim_{j \rightarrow \infty} \frac{1}{2}\|Ax^{t_j} - \eta^{t_j}\|^2 = \frac{1}{2}\|Ax^* - \eta^*\|^2.$$

This shows that $\eta^* \in \text{Proj}_D(Ax^*)$. Now, passing to the limit as $j \rightarrow \infty$ in (11) and invoking Lemma 2 and the closedness of $x \mapsto N_C(x)$ at $x^* \in C$, we obtain

$$0 \in A^T Ax^* - A^T \eta^* + N_C(x^*),$$

showing that x^* is a stationary point of (2). This completes the proof. \square

The algorithm $\text{SpFeas}_{\text{DC}_s}$ involves a linesearch subroutine in each iteration. We next discuss a variant that does not require any linesearch procedure, which is presented in Algorithm 2 below as $\text{SpFeas}_{\text{DC}}$, where

$$r_C := \begin{cases} 2 & \text{if } C \text{ is convex,} \\ 1 & \text{otherwise.} \end{cases} \quad (12)$$

We will argue that $\text{SpFeas}_{\text{DC}}$ is a special instance of $\text{SpFeas}_{\text{DC}_s}$.

Algorithm 2 $\text{SpFeas}_{\text{DC}}$ for (2).

Step 0. Choose $x^0 \in C$ and $L > \frac{\lambda_{\max}(A^T A)}{r_C}$.

Step 1. For each $t = 0, 1, 2, \dots$, pick any $\eta^t \in \text{Proj}_D(Ax^t)$ and set

$$x^{t+1} \in \text{Proj}_C \left(x^t - \frac{A^T(Ax^t - \eta^t)}{L} \right).$$

Step 2. If a termination criterion is not met, go to **Step 1**.

To this end, we first prove the following proposition.

Proposition 2 Consider the split feasibility problem (2) and let F be defined as in (4). Let $L > 0$, $x \in C$, $\eta \in \text{Proj}_D(Ax)$ and set

$$u \in \text{Proj}_C \left(x - \frac{A^T(Ax - \eta)}{L} \right). \quad (13)$$

Then

$$F(u) \leq F(x) - \frac{r_C L - \lambda_{\max}(A^T A)}{2} \|u - x\|^2, \quad (14)$$

where r_C is defined as in (12).

Proof Using the decomposition of F in (5), we have for any $u \in C$ that

$$\begin{aligned} F(u) &= h(u) + P(u) - g(u) = \frac{1}{2} \|Au\|^2 + \delta_C(u) - g(u) \\ &\leq \frac{1}{2} \|Ax\|^2 + (Ax)^T (Au - Ax) + \frac{\lambda_{\max}(A^T A)}{2} \|u - x\|^2 - \eta^T (Au - Ax) - g(x) \\ &= F(x) + (Ax - \eta)^T (Au - Ax) + \frac{\lambda_{\max}(A^T A)}{2} \|u - x\|^2, \end{aligned} \quad (15)$$

where the inequality holds because of the Taylor's inequality applied to $w \mapsto \frac{1}{2} \|Aw\|^2$ and the fact that $A^T \eta \in \partial g(x)$ (see (6)).

Now, suppose that C is not convex. Then we see from (13) that

$$\left\| u - x + \frac{A^T(Ax - \eta)}{L} \right\|^2 \leq \left\| \frac{A^T(Ax - \eta)}{L} \right\|^2,$$

because $x \in C$. Thus, $\frac{L}{2} \|u - x\|^2 + (u - x)^T A^T (Ax - \eta) \leq 0$. This together with (15) shows that (14) holds (with $r_C = 1$) when C is nonconvex.

On the other hand, if C is convex, then the function $\rho(y) := (A^T Ax - A^T \eta)^T (y - x) + \frac{L}{2} \|y - x\|^2 + \delta_C(y)$ is a strongly convex function with modulus L . Moreover, we see from (13) that u is the unique minimizer of ρ . Thus, we have $\rho(x) - \rho(u) \geq \frac{L}{2} \|x - u\|^2$, which gives

$$L \|u - x\|^2 + (u - x)^T A^T (Ax - \eta) \leq 0.$$

This together with (15) shows that (14) holds (with $r_C = 2$) when C is convex. This completes the proof. \square

We can now argue that $\text{SpFeas}_{\text{DC}}$ is a special instance of $\text{SpFeas}_{\text{DC}_{\text{ls}}}$. To this end, pick an $x^0 \in C$ and suppose that an $L > \frac{\lambda_{\max}(A^T A)}{r_C}$ is chosen. Let $c > 0$ be such that

$$L = \frac{c + \lambda_{\max}(A^T A)}{r_C} > 0. \quad (16)$$

If we use this c in $\text{SpFeas}_{\text{DC}_{\text{ls}}}$, set $L_{\max} = L_{\min} = L$ and fix any nonnegative integer M and any $\tau > 1$, then, according to Proposition 2, the linesearch condition in (7) is always satisfied with $L_t = L_t^0 = L$. Hence, $\text{SpFeas}_{\text{DC}}$ initialized at $x^0 \in C$ with the chosen $L > \frac{\lambda_{\max}(A^T A)}{r_C}$ generates the same sequence as $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ initialized at $x^0 \in C$ with c chosen as in (16) and $L_{\max} = L_{\min} = L$. We thus have the following immediate corollary concerning the convergence of $\text{SpFeas}_{\text{DC}}$.

Corollary 1 (Subsequential convergence of $\text{SpFeas}_{\text{DC}}$) Consider the split feasibility problem (2) with $C^\infty \cap A^{-1}D^\infty = \{0\}$ and let $\{x^t\}$ be the sequence generated by $\text{SpFeas}_{\text{DC}}$. Then the following statements hold:

- (i) The sequence $\{x^t\}$ is bounded.
- (ii) Any accumulation point of $\{x^t\}$ is a stationary point of (2).

On passing, we note that $\text{SpFeas}_{\text{DC}}$ reduces to the classical CQ algorithm (3) when C and D are both convex.¹ Thus, our algorithm $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ is a generalization of the classical CQ algorithm for solving (2) in the general nonconvex setting.

5 Sequential convergence based on KL property

5.1 Global convergence

In this section, we establish the convergence of the whole sequence generated by $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ with $M = 0$ under the KL property and some mild assumptions. The KL property has been used extensively in recent years for establishing global convergence of the sequence generated by various first-order methods; see, for example, [3–5]. Our proof for the next theorem follows closely the arguments in [5] and is routine. We include its proof in the appendix for the ease of readers.

Theorem 2 (Global convergence of $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ with $M = 0$) Consider the split feasibility problem (2) with $C^\infty \cap A^{-1}D^\infty = \{0\}$ and let the function F in (4) be a KL function. Let $\{x^t\}$ be the sequence generated by $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ with $M = 0$. Then the sequence $\{x^t\}$ is bounded. Let x^* be an accumulation point of $\{x^t\}$ and suppose that the function $y \mapsto \frac{1}{2}d^2(y, D)$ is continuously differentiable at Ax^* with locally Lipschitz gradient. Then the whole sequence $\{x^t\}$ is convergent.

Theorem 2 requires the function F in (4) to have the KL property. This particular condition is not very restrictive. Indeed, according to [4, Section 4.3], the function F satisfies the KL property when C and D are both in addition semialgebraic. On the other hand, since D is possibly nonconvex, the function $y \mapsto \frac{1}{2}d^2(y, D)$ may not be differentiable in general at Ax^* , where x^* is an accumulation point of $\{x^t\}$, as required by Theorem 2. In the next two propositions, we give simple sufficient conditions for $y \mapsto \frac{1}{2}d^2(y, D)$ to be continuously differentiable at Ax^* with locally Lipschitz gradient. The first proposition concerns prox-regularity and is an immediate consequence of [19, Proposition 8.1]. Recall from [28, Theorem 1.3] that a closed set Ω is said to be prox-regular at an $\bar{x} \in \Omega$ if there exists $\epsilon > 0$ such that $\text{Proj}_\Omega(x)$ is single-valued for every $x \in B(\bar{x}, \epsilon)$.

Proposition 3 Consider the split feasibility problem (2). If $x^* \in C \cap A^{-1}D$ and D is prox-regular at Ax^* , then $y \mapsto \frac{1}{2}d^2(y, D)$ is continuously differentiable at Ax^* with locally Lipschitz gradient.

Proof The desired result follows directly from [19, Proposition 8.1]. \square

¹ This is because in this case, we have $r_C = 2$ and hence $\text{SpFeas}_{\text{DC}}$ corresponds to (3) with $\gamma = \frac{1}{L}$.

Proposition 4 *Consider the split feasibility problem (2) and suppose that $D = \bigcup_{i=1}^m D_i$, where each D_i , $i = 1, \dots, m$, is closed and convex. If $I(Ax^*) := \{i : d(Ax^*, D) = d(Ax^*, D_i)\}$ is a singleton set, then $y \mapsto \frac{1}{2}d^2(y, D)$ is continuously differentiable at Ax^* with locally Lipschitz gradient.*

Proof It is easy to see that $\frac{1}{2}d^2(y, D) = \min_{1 \leq i \leq m} \frac{1}{2}d^2(y, D_i)$ for all $y \in \mathbb{R}^m$. Moreover, for each i , the function $y \mapsto \frac{1}{2}d^2(y, D_i)$ is continuously differentiable with Lipschitz gradient because D_i is closed and convex. Write $y^* = Ax^*$ for notational simplicity and note that $I(y^*) = \{i_0\}$ for some $i_0 \in \{1, \dots, m\}$ by assumption.

From the definition of $I(y^*)$ we have $\min_{i \notin I(y^*)} \frac{1}{2}d^2(y^*, D_i) > \frac{1}{2}d^2(y^*, D)$. By continuity, it then holds that for all y sufficiently close to y^* , we have

$$\min_{i \notin I(y^*)} \frac{1}{2}d^2(y, D_i) > \frac{1}{2}d^2(y, D).$$

Hence $I(y) = \{i_0\}$ for all y sufficiently close to y^* . Thus, it holds that $\frac{1}{2}d^2(y, D) = \frac{1}{2}d^2(y, D_{i_0})$ locally around y^* . Consequently, the function $y \mapsto \frac{1}{2}d^2(y, D)$ is continuously differentiable at y^* with locally Lipschitz gradient. This completes the proof. \square

5.2 Local convergence behavior

In this section, we study the local convergence rate of the sequence $\{x^t\}$ generated by $\text{SpFeas}_{\text{DC}_s}$ with $M = 0$. Local convergence rates of various first-order methods have been widely studied recently and they are usually analyzed based on the so-called KL exponent of a certain potential function; see, for example, [3, 4, 20]. Here, our analysis uses the F in (4) as the potential function and makes use of the assumption that F is a KL function with exponent $\theta \in [0, 1)$. We first show in the proposition below that this latter assumption holds when C and D are subanalytic sets (see [8, Definition 2.1 (ii)] for the definition of subanalytic sets).

Proposition 5 *Consider the split feasibility problem (2) and let F be defined in (4). If C and D are subanalytic sets, then F is a KL function with exponent $\theta \in [0, 1)$.*

Proof First of all, since D is subanalytic, we see from p3 and p5 of [16, page 597] that the squared distance function $x \mapsto \frac{1}{2}d^2(Ax, D)$ is a subanalytic function, which means that its graph, given by $\{(x, \frac{1}{2}d^2(Ax, D)) : x \in \mathbb{R}^n\}$, is a subanalytic set. Next, notice that the graph of F in (4) is given by

$$\{(x, \frac{1}{2}d^2(Ax, D)) : x \in C\} = \{(x, \frac{1}{2}d^2(Ax, D)) : x \in \mathbb{R}^n\} \cap (C \times \mathbb{R}).$$

In addition, the set $C \times \mathbb{R}$ is subanalytic as C is subanalytic, thanks to p2 of [16, page 597]. In view of these, we deduce from p1 of [16, page 597] that the graph of F is also subanalytic, which means that F is a subanalytic function. Now, since $\text{dom } F$ is closed and F is continuous relative to $\text{dom } F$, using [8, Theorem 3.1], we conclude that F is a KL function with exponent $\theta \in [0, 1)$. This completes the proof.

We now state in the following theorem our local convergence result for $\text{SpFeas}_{\text{DC}_s}$ with $M = 0$ based on the KL exponent θ of the F in (4). The proof is standard and follows a similar line of arguments as in [3, Theorem 2], is thus omitted for brevity.

Theorem 3 (Local convergence rate) Consider the split feasibility problem (2) with $C^\infty \cap A^{-1}D^\infty = \{0\}$ and let the function F in (4) be a KL function with exponent $\theta \in [0, 1)$. Let $\{x^t\}$ be the sequence generated by $\text{SpFeas}_{DC_{\text{is}}}$ with $M = 0$ and let x^* be its accumulation point. If $y \mapsto \frac{1}{2}d^2(y, D)$ is continuously differentiable at Ax^* with locally Lipschitz gradient, then the following statements hold:

- (i) If $\theta = 0$, then there exists $\bar{t} \geq 0$ such that $x^t = x^*$ whenever $t \geq \bar{t}$;
- (ii) If $\theta \in (0, \frac{1}{2}]$, then there exist $\bar{t} \geq 0$, $\bar{d} > 0$ and $\sigma \in (0, 1)$ such that $\|x^t - x^*\| \leq \bar{d}\sigma^t$ for all $t \geq \bar{t}$;
- (iii) If $\theta \in (\frac{1}{2}, 1)$, then there exist $\bar{t} \geq 0$ and $\bar{d} > 0$ such that $\|x^t - x^*\| \leq \bar{d}t^{-\frac{1-\theta}{2\theta-1}}$ for all $t \geq \bar{t}$.

From Theorem 3, we know that if the function F in (4) satisfies the KL property with exponent $\frac{1}{2}$ and a certain differentiability assumption holds at an accumulation point of $\{x^t\}$, then the sequence $\{x^t\}$ generated is locally linearly convergent. We next give sufficient conditions on C and D in (2) so that the F in (4) satisfies the KL property with exponent $\frac{1}{2}$. Our first result concerns polyhedrality.

Theorem 4 Consider the split feasibility problem (2) and let F be defined in (4). If C and D are both unions of polyhedral sets, then F is a KL function with exponent $\frac{1}{2}$.

Proof Let $D = \bigcup_{i=1}^k D_i$ and $C = \bigcup_{j=1}^\ell C_j$, where D_1, \dots, D_k and C_1, \dots, C_ℓ are all polyhedral sets. Then one can see that

$$F(x) = \min_{\substack{1 \leq i \leq k \\ 1 \leq j \leq \ell}} \underbrace{\frac{1}{2}d^2(Ax, D_i) + \delta_{C_j}(x)}_{F_{i,j}(x)}. \quad (17)$$

Since D_i and C_j are polyhedral for each i and j , we conclude from [30, Exercise 10.22] and [30, Example 12.31] that $F_{i,j}$ is piecewise linear-quadratic for each i and j . Using the definition of piecewise linear-quadratic function, we can further rewrite $F_{i,j}$ as follows:

$$F_{i,j}(x) = \min_{1 \leq \nu \leq l_{i,j}} \left\{ \frac{1}{2}x^T G_{i,j,\nu} x + \alpha_{i,j,\nu}^T x + \beta_{i,j,\nu} + \delta_{P_{i,j,\nu}}(x) \right\}, \quad (18)$$

where $G_{i,j,\nu} \in \mathbb{R}^{n \times n}$ is symmetric, $\alpha_{i,j,\nu} \in \mathbb{R}^n$, $\beta_{i,j,\nu} \in \mathbb{R}$, and $P_{i,j,\nu}$ is polyhedral for each $\nu \in \{1, \dots, l_{i,j}\}$. The desired conclusion now follows from (17), (18) and [20, Corollary 5.2]. This completes the proof. \square

Our next result concerns a certain kind of regularity condition, defined as follows.

Definition 4 (Linearly regular intersection with respect to A) Consider the split feasibility problem (2). We say that the pair of sets $\{C, A^{-1}D\}$ has a linearly regular intersection with respect to A at a point $x^* \in C \cap A^{-1}D$ if the following implication holds:

$$\begin{aligned} A^T w^* + v^* = 0 \text{ for some } w^* \in N_D(Ax^*) \text{ and } v^* \in N_C(x^*), \\ \implies w^* = 0 \text{ and } v^* = 0. \end{aligned} \quad (19)$$

The concept of linearly regular intersection defined above for split feasibility problems is a generalization of the corresponding property for classical feasibility problems. Recall from [19, Section 2] that the pair of nonempty closed sets $\{C_1, C_2\}$ has linearly regular intersection at a point $x^* \in C_1 \cap C_2$ if the following implication holds:

$$\begin{aligned} v_1 + v_2 = 0 \text{ for some } v_1 \in N_{C_1}(x^*) \text{ and } v_2 \in N_{C_2}(x^*), \\ \implies v_1 = v_2 = 0. \end{aligned}$$

It was proved in [19, Theorem 5.16] that if $\{C_1, C_2\}$ has linearly regular intersection at an $x^* \in C_1 \cap C_2$ and at least one of these two sets is super-regular at x^* (see [19, Definition 4.3]), then the sequence generated by the alternating projection algorithm for finding a point in $C_1 \cap C_2$ is *locally linear convergent* as long as the algorithm was initialized sufficiently close to x^* . Here, we will show in Theorem 5 below that (19) has a similar implication on split feasibility problem (2): under (19), the function F defined in (4) has the KL property with exponent $\frac{1}{2}$ at the point $x^* \in C \cap A^{-1}D$.

We start with an auxiliary lemma.

Lemma 3 *Consider the split feasibility problem (2). Let $x^* \in C \cap A^{-1}D$ and suppose that the pair of sets $\{C, A^{-1}D\}$ has a linearly regular intersection with respect to A at x^* , i.e., (19) holds. Then the following statements hold:*

(i) *There exist $\gamma_1 > 0$ and $\epsilon_1 > 0$ such that*

$$\|A^T(Ax - \eta)\| \geq \gamma_1 \|Ax - \eta\| \quad (20)$$

whenever $x \in B(x^, \epsilon_1)$ and $\eta \in \text{Proj}_D(Ax)$.*

(ii) *There exist $\gamma_2 \in [0, 1)$ and $\epsilon_2 > 0$ such that*

$$v^T A^T(Ax - \eta) \geq -\gamma_2 \|A^T(Ax - \eta)\| \|v\| \quad (21)$$

whenever $x \in B(x^, \epsilon_2) \cap C$, $\eta \in \text{Proj}_D(Ax)$ and $v \in N_C(x)$.*

Proof We first prove (i). Suppose to the contrary that (i) does not hold. Then there exist $\{x^t\}$ and $\{\eta^t\}$ satisfying $x^t \rightarrow x^*$, $\eta^t \in \text{Proj}_D(Ax^t)$ and

$$\|A^T(Ax^t - \eta^t)\| < \frac{1}{t} \|Ax^t - \eta^t\| \quad (22)$$

for all $t \geq 1$. In particular, we have $Ax^t - \eta^t \neq 0$ for all $t \geq 1$. Moreover, by passing to a subsequence if necessary, we may assume without loss of generality that

$$\frac{Ax^t - \eta^t}{\|Ax^t - \eta^t\|} \rightarrow q^* \quad (23)$$

for some q^* with $\|q^*\| = 1$.

Now, since $\eta^t \in \text{Proj}_D(Ax^t)$ and $Ax^* \in D$, we have for all $t \geq 1$ that

$$\|\eta^t - Ax^t\| \leq \|Ax^t - Ax^*\|.$$

This together with $x^t \rightarrow x^*$ implies that $\eta^t \rightarrow Ax^*$. In addition, the relation $\eta^t \in \text{Proj}_D(Ax^t)$ together with (1) implies that for all $t \geq 1$,

$$Ax^t - \eta^t \in N_D(\eta^t).$$

Combining this with (23), the fact that $\eta^t \rightarrow Ax^*$ and the closedness of the normal cone mapping yields $q^* \in N_D(Ax^*)$. Next, divide both sides of (22) by $\|Ax^t - \eta^t\|$ and pass to the limit, we obtain $A^T q^* = 0$. This together with $q^* \in N_D(Ax^*)$ and (19) gives $q^* = 0$, which is a contradiction. This proves (i).

We now prove (ii). Suppose to the contrary that (ii) does not hold. Then there exist $\{x^t\} \subseteq C$, $\{\gamma_2^t\}$, $\{\eta^t\}$ and $\{v^t\}$ satisfying $x^t \rightarrow x^*$, $\gamma_2^t \uparrow 1$, $\eta^t \in \text{Proj}_D(Ax^t)$, $v^t \in N_C(x^t)$ and

$$v^{tT} A^T (Ax^t - \eta^t) < -\gamma_2^t \|A^T (Ax^t - \eta^t)\| \|v^t\| \quad (24)$$

for all $t \geq 1$. This implies in particular that $A^T (Ax^t - \eta^t) \neq 0$ and $v^t \neq 0$ for all $t \geq 1$. Also, observe from (i) that for all sufficiently large t , we have

$$\|A^T (Ax^t - \eta^t)\| \geq \gamma_1 \|Ax^t - \eta^t\|.$$

Thus, by passing to subsequences if necessary, we may assume without loss of generality that

$$\frac{Ax^t - \eta^t}{\|A^T (Ax^t - \eta^t)\|} \rightarrow p^* \text{ and } \frac{v^t}{\|v^t\|} \rightarrow q^* \quad (25)$$

for some p^* and q^* so that $\|A^T p^*\| = \|q^*\| = 1$.

Next, observe from $\eta^t \in \text{Proj}_D(Ax^t)$, $x^t \rightarrow x^*$ and $Ax^* \in D$ that

$$\limsup_{t \rightarrow \infty} \|\eta^t - Ax^t\| \leq \limsup_{t \rightarrow \infty} \|Ax^t - Ax^*\| = 0.$$

Hence, $\eta^t \rightarrow Ax^*$. In addition, the relation $\eta^t \in \text{Proj}_D(Ax^t)$ together with (1) shows that for all $t \geq 1$,

$$Ax^t - \eta^t \in N_D(\eta^t).$$

This together with (25), the fact that $\eta^t \rightarrow Ax^*$ and the closedness of the normal cone mapping gives $p^* \in N_D(Ax^*)$. Similarly, the relation $v^t \in N_C(x^t)$, (25), the fact that $x^t \rightarrow x^*$ and the closedness of normal cone mapping imply $q^* \in N_C(x^*)$. Now, divide both sides of (24) by $\|A^T (Ax^t - \eta^t)\| \|v^t\|$ and pass to the limit, we see that $(A^T p^*)^T q^* \leq -1$. Hence

$$\|A^T p^* + q^*\|^2 = \|A^T p^*\|^2 + \|q^*\|^2 + 2(A^T p^*)^T q^* = 2 + 2(A^T p^*)^T q^* \leq 0,$$

where the second equality holds because of (25). The above display shows that $A^T p^* + q^* = 0$. This together with $p^* \in N_D(Ax^*)$, $q^* \in N_C(x^*)$ and (19) gives $A^T p^* = q^* = 0$, which is a contradiction. This completes the proof. \square

We are now ready to show that under (19), the function F defined in (4) has the alleged KL property.

Theorem 5 (KL exponent under linear regularity) *Consider the split feasibility problem (2). Let $x^* \in C \cap A^{-1}D$ and suppose that the pair of sets $\{C, A^{-1}D\}$ has a linearly regular intersection with respect to A at x^* , i.e., (19) holds. Then the function F defined in (4) has the KL property with exponent $\frac{1}{2}$ at x^* .*

Proof Let $\epsilon = \min\{\epsilon_1, \epsilon_2\}$, where ϵ_1 and ϵ_2 are as in Lemma 3(i) and (ii) respectively. When $x \in B(x^*, \epsilon) \cap C$, $\eta \in \text{Proj}_D(Ax)$ and $v \in N_C(x)$, we have

$$\begin{aligned} \|A^T(Ax - \eta) + v\|^2 &= \|A^T(Ax - \eta)\|^2 + \|v\|^2 + 2v^T A^T(Ax - \eta) \\ &\geq \|A^T(Ax - \eta)\|^2 + \|v\|^2 - 2\gamma_2 \|A^T(Ax - \eta)\| \|v\| \quad (26) \\ &\geq (1 - \gamma_2)(\|A^T(Ax - \eta)\|^2 + \|v\|^2), \end{aligned}$$

where the first inequality follows from Lemma 3(ii). Thus, whenever $x \in B(x^*, \epsilon) \cap C$, we have

$$\begin{aligned} d(0, \partial F(x)) &\stackrel{(a)}{\geq} \inf_{\eta \in \text{Proj}_D(Ax), v \in N_C(x)} \sqrt{\|A^T(Ax - \eta) + v\|^2} \\ &\stackrel{(b)}{\geq} \sqrt{1 - \gamma_2} \inf_{\eta \in \text{Proj}_D(Ax), v \in N_C(x)} \sqrt{\|A^T(Ax - \eta)\|^2 + \|v\|^2} \\ &\geq \sqrt{1 - \gamma_2} \inf_{\eta \in \text{Proj}_D(Ax)} \|A^T(Ax - \eta)\| \stackrel{(c)}{\geq} \gamma_1 \sqrt{1 - \gamma_2} \inf_{\eta \in \text{Proj}_D(Ax)} \|Ax - \eta\| \\ &= \gamma_1 \sqrt{2(1 - \gamma_2)} (F(x) - F(x^*))^{\frac{1}{2}}, \end{aligned}$$

where (a) follows from Proposition 1, (b) follows from (26) and (c) follows from Lemma 3(i). This completes the proof. \square

6 Factorizing completely positive matrices

In this section, we consider the problem of factorizing completely positive matrices. Recall that a symmetric matrix $G \in \mathbb{R}^{n \times n}$ is completely positive if there exists a $B \in \mathbb{R}_+^{n \times r}$ for some $r \geq 1$ such that $G = BB^T$. It is known that determining whether a given matrix is completely positive is NP-hard; see, for example, [14] and references therein.

Given a completely positive matrix G , the factorization problem aims at finding a $B \in \mathbb{R}_+^{n \times r}$ for some $r \geq 1$ so that $G = BB^T$. In [17], this factorization problem was reformulated as a feasibility problem. Precisely, given a completely positive matrix $G \in \mathbb{R}^{n \times n}$, the authors in [17] started with an initial factorization $G = BB^T$ for some $B \in \mathbb{R}^{n \times r}$; here, B may not be entrywise nonnegative and $r \geq n$. They then rewrite the factorization as $G = (BQ)(BQ)^T$ for some orthogonal matrix $Q \in \mathbb{R}^{r \times r}$. If r is chosen to be at least as large as the completely positive rank of G (see [17, Definition 2.2]), then the completely positive matrix factorization problem is equivalent to finding an orthogonal matrix Q so that BQ is entrywise nonnegative, i.e.,

$$\text{Find } Q \in \mathbb{R}^{r \times r} \text{ s.t. } Q \in \mathcal{P} \text{ and } Q \in C, \quad (27)$$

where C is the set of $r \times r$ orthogonal matrices, and $\mathcal{P} := \{Q \in \mathbb{R}^{r \times r} : BQ \in \mathbb{R}_+^{n \times r}\}$. We would like to point out that the completely positive rank of G is generally hard to compute (see [7]) and we refer the readers to [9, Theorem 4.1] for upper bounds of completely positive rank. These upper bounds are in the order of n^2 for large n . Instead of using these bounds as r , in our experiments, as a heuristic, we choose r in the order of n and we will specify our choices later.

In [17], the authors considered two algorithms for solving (27):

1. the classical alternating projection method, which can be inefficient because $\text{Proj}_{\mathcal{P}}$ is in general difficult to compute;
2. the modified alternating projection algorithm (see [17, Algorithm 2]), which only requires computing projections onto C and the nonnegative orthant $\mathbb{R}_+^{n \times r}$, as well as multiplications by B and its Moore-Penrose inverse B^\dagger . This algorithm is described in Algorithm 3 below.

It was discussed in [17, Section 5] that the modified alternating projection algorithm is more efficient empirically than the classical alternating projection algorithm for solving (27).

Algorithm 3 The modified alternating projection algorithm in [17]

Step 0. Choose $r \geq n$ and $B \in \mathbb{R}^{n \times r}$ so that $G = BB^T$. Then select a $Q^0 \in C$. Set $t = 0$.

Step 1. Compute $W^t = \text{Proj}_{\mathbb{R}_+^{n \times r}}(BQ^t)$ and find

$$Q^{t+1} \in \text{Proj}_C[B^\dagger W^t + (I - B^\dagger B)Q^t].$$

Step 2. If a termination criterion is not met, set $t = t + 1$. Go to **Step 1**.

Here, we consider an alternative approach for solving (27). Indeed, one can observe immediately that (27) can be reformulated as the following split feasibility problem:

$$\text{Find } Q \in \mathbb{R}^{r \times r} \text{ s.t. } BQ \in D \text{ and } Q \in C, \quad (28)$$

where $D = \mathbb{R}_+^{n \times r}$ and C is the set of $r \times r$ orthogonal matrices. Note that the projections onto C and D have closed form solutions; see, for example, [17, Lemma 4.1] for the closed form formula of Proj_C . Moreover, we have $C^\infty = \{0\}$ because the set of $r \times r$ orthogonal matrices is bounded. Thus, we can apply $\text{SpFeas}_{DC_{\text{ls}}}$ to solving (28), and any accumulation point of the sequence generated is a stationary point of the split feasibility problem (28) according to Theorem 1.²

6.1 Numerical experiments for completely positive matrix factorization

In this section, we compare $\text{SpFeas}_{DC_{\text{ls}}}$ and the modified alternating projection algorithm (i.e., Algorithm 3) for solving (28) (or equivalently, (27)). All codes are written in Matlab, and the experiments are performed in Matlab 2019b on a 64-bit PC with an Intel(R) Core(TM) i7-6700 CPU (3.40GHz) and 32GB of RAM.

We first discuss the implementation details of the algorithms. In $\text{SpFeas}_{DC_{\text{ls}}}$, we set $M = 4$, $\tau = 2$, $c = 10^{-4}$, $L_{\max} = 10^8$ and $L_{\min} = 10^{-8}$. Moreover, we set $L_0^0 = 1$, and when $t \geq 1$:

$$L_t^0 = \begin{cases} \min\{\max\{\frac{\text{tr}([Y^t]^T S^t)}{\|S^t\|_F^2}, L_{\min}\}, L_{\max}\} & \text{if } \text{tr}([Y^t]^T S^t) \geq 10^{-16} \\ \min\{\max\{\frac{L_{t-1}}{1.1}, L_{\min}\}, L_{\max}\} & \text{otherwise.} \end{cases}$$

² We note that in this case D is convex and hence $Q \mapsto \frac{1}{2}d^2(BQ, D)$ is smooth. Our algorithm $\text{SpFeas}_{DC_{\text{ls}}}$ reduces to the standard gradient projection algorithm with nonmonotone linesearch.

where $S^t = Q^t - Q^{t-1}$ and $Y^t = B^T[BQ^t - \eta^t] - B^T[BQ^{t-1} - \eta^{t-1}]$, with $\eta^t \in \text{Proj}_D(BQ^t)$ chosen in **Step 1a** of $\text{SpFeas}_{\text{DC}_{\text{ls}}}$. We terminate it when $\text{iter} > 5000$ or $\min\{BQ^t\}_{ij} \geq -10^{-16}$ or $\bar{L}_t > 10^{10}$. On the other hand, for the modified alternating projection algorithm (i.e., Algorithm 3), we terminate it when $\text{iter} > 5000$ or $\min\{BQ^t\}_{ij} \geq -10^{-15}$. We will describe their initializations later.

Both algorithms require a choice of $r \geq n$ and an initial factorization $G = BB^T$. In our experiments below, we follow the approach in [17, Section 3] to generate the B . Specifically, given a completely positive matrix $G \in \mathbb{R}^{n \times n}$, we compute the Cholesky decomposition of G such that $G = LL^T$ for some lower triangular matrix L , if successful, and set $\bar{B} = L$. On the other hand, if the Cholesky decomposition fails, we compute the eigenvalue decomposition of G such that $G = U\Sigma_G U^T$ for some orthogonal matrix U and diagonal matrix Σ_G , and set $\bar{B} = U\Sigma_G^{\frac{1}{2}}U^T$. Then we define B as follows:

$$B := \left[\bar{\mathbf{b}}_1, \dots, \bar{\mathbf{b}}_{j-1}, \underbrace{\frac{1}{\sqrt{m}}\bar{\mathbf{b}}_j, \bar{\mathbf{b}}_{j+1}, \dots, \bar{\mathbf{b}}_n, \frac{1}{\sqrt{m}}\bar{\mathbf{b}}_j, \dots, \frac{1}{\sqrt{m}}\bar{\mathbf{b}}_j}_{m-1 \text{ columns}} \right], \quad (29)$$

where $\bar{\mathbf{b}}_j$ is the column of \bar{B} with the least number of negative entries, and $m = r - n + 1$.

We perform two experiments comparing $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ and Algorithm 3. In our first experiment, we consider randomly generated completely positive matrices as in [17, Section 7.8]. We generate a random $n \times n$ completely positive matrix G using the following MATLAB code:

```
G_0 = abs(randn(n,2*n)); G = G_0*G_0';
```

We set $(n, r) = (n, 1.5n) = (10i, 15i)$ for $i = 1, 2, 3, 4, 10, 20, 30, 40$ in Table 1 and 2, and set $(n, r) = (n, 3n + 1) = (10i, 30i + 1)$ for $i = 1, 2, 3, 4, 10, 20$ in Table 3. For each i , we randomly generate 50 completely positive matrices G as described above. We generate B as in (29) and consider two possible ways of initializing the algorithms:

- We initialize both algorithms at $Q^0 = I$ for solving the corresponding (28).
- We initialize both algorithms at the same random initial point, where we first generate an $r \times r$ matrix \tilde{Q} with i.i.d. standard Gaussian entries and then pick any $Q^0 \in \text{Proj}_C(\tilde{Q})$.

We first present the computational results with $Q^0 = I$ in Table 1, where we report the largest and smallest function values ($\frac{1}{2}d^2(BQ^t, D)$) at termination, the average number of iterations among successful instances (iter_s) and the average number of iterations among failed instances (iter_f).³ We also report the average CPU time (in seconds) among successful instances (CPU_s) and the average CPU time among failed instances (CPU_f). The success rate is also listed. We can see from Table 1 that $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ significantly outperforms Algorithm 3, with $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ being able to solve all instances and being much faster.

³ We say that the instance is solved by the algorithm successfully if the algorithm is terminated with the desired accuracy achieved, i.e., $\min\{BQ^t\}_{ij} \geq -10^{-16}$ for $\text{SpFeas}_{\text{DC}_{\text{ls}}}$, and $\min\{BQ^t\}_{ij} \geq -10^{-15}$ for Algorithm 3.

Table 1 Comparing $\text{SpFeas}_{\text{DC}_{1s}}$ and Algorithm 3 on factorizing random completely positive matrices when $Q^0 = I$.

n	r	$\text{SpFeas}_{\text{DC}_{1s}}$						
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f
10	15	100	2e-33	0e+00	5	-	0.0010	-
20	30	100	0e+00	0e+00	8	-	0.0014	-
30	45	100	0e+00	0e+00	10	-	0.0050	-
40	60	100	4e-33	0e+00	11	-	0.0077	-
100	150	100	0e+00	0e+00	18	-	0.0622	-
200	300	100	3e-33	0e+00	190	-	2.2804	-
300	450	100	5e-33	0e+00	486	-	14.0715	-
400	600	100	1e-34	0e+00	731	-	43.5873	-
n	r	Algorithm 3						
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f
10	15	0	1e+00	9e-03	-	5001	-	0.1195
20	30	0	4e+00	1e-01	-	5001	-	0.5320
30	45	0	6e+00	2e-01	-	5001	-	1.6316
40	60	0	6e+00	6e-01	-	5001	-	2.3411
100	150	0	1e+01	2e+00	-	5001	-	13.9835
200	300	0	3e+01	1e+01	-	5001	-	57.6484
300	450	0	3e+02	4e+01	-	5001	-	147.4416
400	600	0	3e+02	9e+01	-	5001	-	293.6904

Then, in Table 2 and Table 3, we present the computational results with random initial points, and set $r = 1.5n$ and $r = 3n + 1$ respectively.⁴ Here, for each random instance, we run the algorithms on the same set of random initial points, where we use at most 100 initial points when $n \leq 50$, and at most 10 initial points otherwise,⁵ and declare a success once the random instance is solved by the algorithm successfully.³ We report the largest and smallest function values ($\frac{1}{2}d^2(BQ^t, D)$) at termination, the average total number of iterations among successful instances (iter_s) and failed instances (iter_f), the average total CPU time (in seconds) among successful instances (CPU_s) and failed instances (CPU_f), and the success rate. We also report the average number of random initial points used among instances that are successfully solved (InitNo_s). We can see from Table 2 and Table 3 that $\text{SpFeas}_{\text{DC}_{1s}}$ significantly outperforms Algorithm 3, with $\text{SpFeas}_{\text{DC}_{1s}}$ being able to solve all instances using only *one* random initial point, and being much faster. Moreover, by comparing these two tables, we can see that $\text{SpFeas}_{\text{DC}_{1s}}$ performs similarly for the two choices of r , while the performance of Algorithm 3 is sensitive to the choice of r .

Next, as in [17, Section 7.6], we perform a second experiment to study the performance of the algorithms in factorizing completely positive matrices that are close to the boundary of the completely positive cone. Specifically, as in [17,

⁴ We do not present results with $(n, 3n + 1) = (10i, 30i + 1)$ for $i = 30, 40$ because they take too much CPU time.

⁵ This choice follows the one used in [17, Table 2].

Table 2 Comparing SpFeas_{DC_{ls}} and Algorithm 3 on factorizing random completely positive matrices with $r = 1.5n$ and random initializations.

n	r	SpFeas _{DC_{ls}}							
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f	InitNo _s
10	15	100	0e+00	0e+00	14	-	0.0013	-	1.0
20	30	100	0e+00	0e+00	18	-	0.0026	-	1.0
30	45	100	0e+00	0e+00	25	-	0.0114	-	1.0
40	60	100	0e+00	0e+00	37	-	0.0212	-	1.0
100	150	100	0e+00	0e+00	112	-	0.3440	-	1.0
200	300	100	4e-33	0e+00	243	-	3.0006	-	1.0
300	450	100	4e-33	0e+00	463	-	14.7035	-	1.0
400	600	100	3e-33	0e+00	711	-	44.6827	-	1.0
n	r	Algorithm 3							
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f	InitNo _s
10	15	100	2e-30	2e-33	5232	-	0.1474	-	1.9
20	30	100	3e-30	9e-32	4468	-	0.4483	-	1.5
30	45	100	2e-30	2e-33	5496	-	2.0830	-	1.7
40	60	100	2e-30	2e-32	7497	-	3.4731	-	1.9
100	150	2	2e-06	1e-31	44693	50010	99.1045	117.2377	9.0
200	300	0	4e-05	2e-28	-	50010	-	546.9136	-
300	450	0	2e-07	2e-28	-	50010	-	1413.3341	-
400	600	0	7e-04	1e-27	-	50010	-	2809.6050	-

Table 3 Comparing SpFeas_{DC_{ls}} and Algorithm 3 on factorizing random completely positive matrices with $r = 3n + 1$ and random initializations.

n	r	SpFeas _{DC_{ls}}							
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f	InitNo _s
10	31	100	4e-35	0e+00	13	-	0.0030	-	1.0
20	61	100	0e+00	0e+00	14	-	0.0070	-	1.0
30	91	100	0e+00	0e+00	16	-	0.0168	-	1.0
40	121	100	0e+00	0e+00	19	-	0.0303	-	1.0
100	301	100	1e-35	0e+00	93	-	1.0123	-	1.0
200	601	100	5e-33	0e+00	209	-	11.0123	-	1.0
n	r	Algorithm 3							
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f	InitNo _s
10	31	100	4e-30	3e-31	1237	-	0.1126	-	1.1
20	61	100	3e-30	2e-32	753	-	0.2883	-	1.0
30	91	100	4e-30	2e-31	2065	-	1.6742	-	1.0
40	121	100	5e-30	2e-31	4739	-	6.0358	-	1.4
100	301	0	3e-27	2e-29	-	50010	-	458.5531	-
200	601	0	6e-27	2e-28	-	50010	-	2300.6188	-

Example 7.3], we consider

$$G := \begin{bmatrix} 8 & 5 & 1 & 1 & 5 \\ 5 & 8 & 5 & 1 & 1 \\ 1 & 5 & 8 & 5 & 1 \\ 1 & 1 & 5 & 8 & 5 \\ 5 & 1 & 1 & 5 & 8 \end{bmatrix}, \quad P := \begin{bmatrix} 2 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 1 & 2 \end{bmatrix},$$

and define

$$G_\lambda = \lambda G + (1 - \lambda)P. \quad (30)$$

We apply the two algorithms to factorizing G_λ with different values of λ . Moreover, for the two algorithms, we consider random initializations: we first generate an $r \times r$ matrix \tilde{Q} with i.i.d. standard Gaussian entries and then pick any $Q^0 \in \text{Proj}_C(\tilde{Q})$.

In our experiments below, we consider λ as listed in Table 4 and set $r = 12$: This choice of r was also used in [17, Section 7.6]. For each λ , we consider 100 random initializations as described above, and apply the two algorithms to factorizing G_λ from these initial points. Our computational results are presented in Table 4, where we report the largest and smallest function values ($\frac{1}{2}d^2(BQ^t, D)$) at termination, the average number of iterations among successful instances (iter_s) and the average number of iterations among failed instances (iter_f).⁶ We also report the average CPU time (in seconds) among successful instances (CPU_s) as well as the average CPU time among failed instances (CPU_f). We can see from Table 4 that $\text{SpFeas}_{\text{DCIs}}$ again significantly outperforms Algorithm 3. Moreover, the success rates for both algorithms decrease when λ increases.

Table 4 Comparing $\text{SpFeas}_{\text{DCIs}}$ and Algorithm 3 on factorizing G_λ in (30).

λ	r	$\text{SpFeas}_{\text{DCIs}}$						
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f
0.00	12	100	0e+00	0e+00	8	-	0.0011	-
0.20	12	100	0e+00	0e+00	27	-	0.0011	-
0.40	12	100	0e+00	0e+00	87	-	0.0031	-
0.60	12	100	0e+00	0e+00	333	-	0.0116	-
0.80	12	99	3e-21	0e+00	1229	5001	0.0399	0.1510
0.90	12	82	2e-17	0e+00	2547	5001	0.0786	0.1529
0.95	12	29	7e-13	0e+00	2897	5001	0.0907	0.1611
0.96	12	9	4e-11	0e+00	2994	5001	0.0951	0.1656
0.97	12	4	7e-12	0e+00	2892	5001	0.0911	0.1709
0.98	12	5	2e-10	0e+00	3356	5001	0.1057	0.1814
0.99	12	1	8e-07	0e+00	3421	5001	0.1243	0.2047
λ	r	Algorithm 3						
		success (%)	fval _{max}	fval _{min}	iter _s	iter _f	CPU _s	CPU _f
0.00	12	99	5e-17	2e-31	155	5001	0.0047	0.1158
0.20	12	91	2e-10	2e-32	476	5001	0.0114	0.0955
0.40	12	77	5e-02	2e-31	703	5001	0.0161	0.0937
0.60	12	45	2e-01	3e-32	871	5001	0.0198	0.0877
0.80	12	36	4e-01	7e-32	1240	5001	0.0268	0.0889
0.90	12	16	5e-01	4e-31	2899	5001	0.0558	0.0893
0.95	12	26	6e-01	4e-31	3106	5001	0.0590	0.0879
0.96	12	31	6e-01	2e-31	2842	5001	0.0547	0.0895
0.97	12	36	6e-01	2e-31	2980	5001	0.0577	0.0870
0.98	12	0	7e-01	8e-17	-	5001	-	0.0921
0.99	12	0	7e-01	1e-04	-	5001	-	0.0922

7 Sparse matrix factorization

Given a matrix G , the sparse matrix factorization problem consists in factorizing G (approximately) as the product of several sparse matrices. This problem is closely related to deep learning, sparse encoding and dictionary learning; see [26] and references therein. In this section, we consider a special instance of the sparse

⁶ As in the previous experiment, we say that the instance is solved by the algorithm successfully if the algorithm is terminated with the desired accuracy achieved, i.e, $\min\{BQ^t\}_{ij} \geq -10^{-16}$ for $\text{SpFeas}_{\text{DCIs}}$, and $\min\{BQ^t\}_{ij} \geq -10^{-15}$ for Algorithm 3.

matrix factorization problem. Specifically, given a positive semidefinite matrix $G \in \mathbb{R}^{n \times n}$, we would like to find a *sparse* matrix $P \in \mathbb{R}^{n \times n}$ so that $G = PP^T$. In addition, we require the *columns* of P to be uniformly sparse: this ensures the cost of the multiplication $\mathbf{p}_j^T x$ remains more or less the same for each j , where \mathbf{p}_j is the j th column of P and x is an n -dimensional vector. More precisely, our problem is described as follows:

$$\text{Find } P \in \mathbb{R}^{n \times n} \text{ s.t. } \max_{j=1, \dots, n} \|\mathbf{p}_j\|_0 \leq s \text{ and } G = PP^T, \quad (31)$$

where $G \in \mathbb{R}^{n \times n}$ is a given positive semidefinite matrix, and $\|v\|_0$ is the number of nonzero entries of the vector v .

To solve (31), we mimic the approach described in Section 6 and reformulate it as a split feasibility problem. In detail, starting with an initial factorization $G = BB^T$ for some $B \in \mathbb{R}^{n \times n}$, one can see that (31) can be equivalently reformulated as the following split feasibility problem:

$$\text{Find } Q \in \mathbb{R}^{n \times n} \text{ s.t. } BQ \in D \text{ and } Q \in C, \quad (32)$$

where C is the set of $n \times n$ orthogonal matrices and $D = \{U \in \mathbb{R}^{n \times n} : \|\mathbf{u}_j\|_0 \leq s \text{ for each } i = 1, \dots, n\}$, with \mathbf{u}_j being the j th column of $U \in \mathbb{R}^{n \times n}$. It is easy to see that if Q^* is a solution of (32), then BQ^* solves (31).

Note that the projections onto C and D have closed form solutions; see [17, Lemma 4.1] and [24, Proposition 3.1] for the closed form formula of Proj_C and Proj_D , respectively. In addition, the boundedness of C implies that $C^\infty = \{0\}$. Therefore, we can employ SpFeas_{DC_s} and SpFeas_{DC} to solve (32) according to the discussions in Section 3, and it follows from Theorem 1 and Corollary 1 that any accumulation point of the sequence generated is a stationary point of the split feasibility problem (32).

7.1 Numerical experiments for sparse matrix factorization

In this section, we perform numerical experiments to compare the performances of SpFeas_{DC_s} and SpFeas_{DC} on solving (32). All codes are written in Matlab, and the experiments are performed in Matlab 2019b on a 64-bit PC with an Intel(R) Core(TM) i7-6700 CPU (3.40GHz) and 32GB of RAM.

We first discuss the implementation details of the algorithms. In SpFeas_{DC_s} , we set $M = 4$, $\tau = 2$, $c = 10^{-4}$, $L_{\max} = 10^8$, $L_{\min} = 10^{-8}$. Moreover, we set $L_0^0 = 1$, and when $t \geq 1$:

$$L_t^0 = \begin{cases} \min\{\max\{\frac{\text{tr}([Y^t]^T S^t)}{\|S^t\|_F^2}, L_{\min}\}, L_{\max}\} & \text{if } \text{tr}([Y^t]^T S^t) \geq 10^{-12} \\ \min\{\max\{\frac{L_{t-1}}{2.5}, L_{\min}\}, L_{\max}\} & \text{otherwise.} \end{cases}$$

where $S^t = Q^t - Q^{t-1}$ and $Y^t = B^T[BQ^t - \eta^t] - B^T[BQ^{t-1} - \eta^{t-1}]$, with $\eta^t \in \text{Proj}_D(BQ^t)$ chosen in **Step 1a** of SpFeas_{DC_s} . On the other hand, for SpFeas_{DC} , we set $L = \lambda_{\max}(B^T B) + 10^{-4}$. We initialize both algorithms at the identity matrix. We terminate SpFeas_{DC_s} when $d(BQ^t, D) < 10^{-9}$, or $\text{iter} > 10000$, or $L_t > 10^{10}$, while SpFeas_{DC} is terminated when $d(BQ^t, D) < 10^{-9}$ or $\text{iter} > 10000$.

We compare the above algorithms on randomly generated positive semidefinite matrices that admit uniformly sparse factorizations. We first generate a random

matrix $P_0 \in \mathbb{R}^{n \times n}$ with i.i.d. standard Gaussian entries. We then project P_0 onto D to obtain \tilde{P}_0 and form a positive semidefinite matrix G by $G = \tilde{P}_0 \tilde{P}_0^T$. We then use the above algorithms to solve the corresponding (32) with $B := G^{\frac{1}{2}}$.

In our experiment below, we set $(n, r) = (100i, 0.1j)$ for $i = 1, 2, 3, 4$ and $j = 6, 7, 8, 9$ and set $s = nr$. For each i and j , we generate 20 random instances as described above. We present the computational results in Table 5, where we report the value $d(BQ^t, D)$ at termination, the number of iterations (iter), and the CPU time in seconds (CPU), averaged over the 20 random instances. One can see that $\text{SpFeas}_{\text{DC}_{1s}}$ notably outperforms $\text{SpFeas}_{\text{DC}}$ in terms of both CPU times and the terminating function values. Moreover, both algorithms become faster when s increases.

Table 5 Comparing $\text{SpFeas}_{\text{DC}_{1s}}$ and $\text{SpFeas}_{\text{DC}}$ on solving (32) with $s = nr$.

n	r	$\text{SpFeas}_{\text{DC}_{1s}}$			$\text{SpFeas}_{\text{DC}}$		
		$d(BQ^t, D)$	iter	CPU	$d(BQ^t, D)$	iter	CPU
100	0.6	1e-09	4014	14.7	3e-04	10001	17.8
100	0.7	1e-09	774	2.7	1e-09	7270	12.5
100	0.8	9e-10	245	0.8	1e-09	2940	5.0
100	0.9	7e-10	141	0.4	1e-09	1381	2.4
200	0.6	1e-09	3800	54.0	3e-04	10001	70.1
200	0.7	1e-09	761	10.4	1e-09	7526	50.7
200	0.8	9e-10	241	3.0	1e-09	3122	21.0
200	0.9	7e-10	146	1.8	1e-09	1477	9.9
300	0.6	1e-09	4067	136.4	4e-04	10001	155.2
300	0.7	1e-09	734	22.8	1e-09	7636	116.6
300	0.8	9e-10	274	8.2	1e-09	3168	48.6
300	0.9	7e-10	149	4.1	1e-09	1509	22.2
400	0.6	1e-09	3736	231.8	6e-04	10001	295.9
400	0.7	1e-09	626	37.1	1e-09	7711	222.5
400	0.8	9e-10	246	13.8	1e-09	3201	92.4
400	0.9	7e-10	145	7.7	1e-09	1528	43.9

8 An outlier detection problem

In this section, we look at an outlier detection problem. Specifically, we consider the problem of finding an s -sparse solution of a linear system $Ax = b$ where some of the b_i 's are wrongly recorded. This class of problem arises in applications such as compressed sensing, where signals may be contaminated by the so-called electromyographic noise, resulting in extreme measurements [27].

Here, we approach this problem by considering the following split feasibility problem, which assumes prior knowledge of s and the number of incorrect b_i 's:

$$\text{Find } x \in \mathbb{R}^n \text{ s.t. } x \in C, Ax \in D, \quad (33)$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $C = \{x \in \mathbb{R}^n : \|x\|_0 \leq s, \|x\|_\infty \leq 10^8\}$, and $D = \{y \in \mathbb{R}^m : \|y\|_0 \leq r\} + b$, with r being an upper estimate of the number of incorrect b_i 's (outliers).

Note that the projections onto C and D have closed form solutions; see, for example, [24, Proposition 3.1]. Moreover, since C is bounded, we have $C^\infty = \{0\}$.

Thus, we can apply $\text{SpFeas}_{\text{DC}_s}$ and $\text{SpFeas}_{\text{DC}}$ to solving (33) according to the discussions in Section 3. Moreover, any accumulation point of the sequence generated is a stationary point of the split feasibility problem (33).

8.1 Numerical experiments for outlier detection

In this section, we perform numerical experiments to study the behavior of $\text{SpFeas}_{\text{DC}_s}$ and $\text{SpFeas}_{\text{DC}}$ on the outlier detection problem (33). All codes are written in Matlab, and the experiments are performed in Matlab 2019b on a 64-bit PC with an Intel(R) Core(TM) i7-6700 CPU (3.40GHz) and 32GB of RAM.

We first discuss the implementation details of the algorithms. In $\text{SpFeas}_{\text{DC}_s}$, we set $M = 4$, $\tau = 2$, $c = 10^{-4}$, $L_{\max} = 10^8$ and $L_{\min} = 10^{-8}$. We also set $L_0^0 = 1$, and when $t \geq 1$,

$$L_t^0 = \begin{cases} \min\{\max\{\frac{s^{tT}y^t}{\|s^t\|^2}, L_{\min}\}, L_{\max}\} & \text{if } s^{tT}y^t \geq 10^{-12} \\ \min\{\max\{\frac{L_{t-1}}{2}, L_{\min}\}, L_{\max}\} & \text{otherwise.} \end{cases}$$

where $s^t = x^t - x^{t-1}$, $y^t = A^T[Ax^t - \eta^t] - A^T[Ax^{t-1} - \eta^{t-1}]$, with η^t defined in **Step 1a)** of the algorithm. We initialize $\text{SpFeas}_{\text{DC}_s}$ at $x^0 = 0$ and terminate it when

$$\frac{\sqrt{\left(\sqrt{\lambda_{\max}(A^T A)}\|A(x^t - x^{t-1})\| + \bar{L}_{t-1}\|x^t - x^{t-1}\|\right)^2 + \|x^t - x^{t-1}\|^2}}{\max\{1, \|x^t\|\}} < 10^{-8};$$

following the discussions in [21, Section 6], this guarantees $d(0, \partial \Xi(x^t, \eta^{t-1})) < 10^{-8} \max\{1, \|x^t\|\}$, where $\Xi(x, \eta) := h(x) + P(x) - \eta^T x + g^*(\eta)$, with h , P and g given in (5), and g^* is the convex conjugate of g .⁷ On the other hand, for $\text{SpFeas}_{\text{DC}}$, we set $L = \lambda_{\max}(A^T A) + 10^{-4}$. We initialize this algorithm at $x^0 = 0$ and terminate it when

$$\frac{\sqrt{\left(\sqrt{\lambda_{\max}(A^T A)}\|A(x^t - x^{t-1})\| + L\|x^t - x^{t-1}\|\right)^2 + \|x^t - x^{t-1}\|^2}}{\max\{1, \|x^t\|\}} < 10^{-8},$$

or when the number of iterations reaches 3000.

We compare the above algorithms on randomly generated instances. We first generate an $m \times n$ matrix A with i.i.d. standard Gaussian entries, and normalize it to have unit column norms. We next generate an s -sparse vector $w \in \mathbb{R}^n$ with i.i.d. standard Gaussian entries at uniformly randomly chosen positions. We then set

$$b_i = \begin{cases} (Aw)_i \\ (Aw)_i + 10 \cdot \text{sign}(\bar{n}_{i-m+r}) & i = m - r + 1, \dots, m. \end{cases}$$

where $\bar{n} \in \mathbb{R}^r$ has i.i.d. standard Gaussian entries.

In our experiment below, we set (n, m, s, r) as listed in Table 6. For each quadruple (n, m, s, r) , we generate 20 random instances as described above. Our

⁷ As discussed in [21, Section 6], this termination criterion is motivated by the fact that (8) holds if and only if $0 \in \partial \Xi(\bar{x}, \bar{\eta})$ for some $\bar{\eta}$.

computational results are presented in Table 6, where we report the value $d(Ax^t, D)$ at termination, the number of iterations (iter), and the CPU time in seconds (CPU), averaged over the 20 random instances. One can observe that $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ significantly outperforms $\text{SpFeas}_{\text{DC}}$ in terms of both CPU times and the terminating function values.

Table 6 Comparing $\text{SpFeas}_{\text{DC}_{\text{ls}}}$ and $\text{SpFeas}_{\text{DC}}$ on solving (33).

n	m	s	r	$\text{SpFeas}_{\text{DC}_{\text{ls}}}$			$\text{SpFeas}_{\text{DC}}$		
				$d(Ax^t, D)$	iter	CPU	$d(Ax^t, D)$	iter	CPU
10000	2000	500	100	2e-08	94	1.3	2e-01	1876	24.9
12000	2400	600	120	3e-08	103	2.3	7e-04	1894	38.3
14000	2800	700	140	2e-08	97	2.8	2e-01	1799	49.0
16000	3200	800	160	3e-08	104	4.0	5e-07	1866	65.4
18000	3600	900	180	4e-08	107	5.4	5e-07	1896	85.7
20000	4000	1000	200	3e-08	100	5.9	6e-04	1937	106.1
22000	4400	1100	220	4e-08	93	6.3	5e-07	1847	122.4
24000	4800	1200	240	2e-08	93	7.5	6e-07	1845	145.0
26000	5200	1300	260	4e-08	92	8.8	6e-07	1863	172.5
28000	5600	1400	280	5e-08	96	10.7	6e-07	1941	208.7
30000	6000	1500	300	4e-08	91	11.5	6e-07	1828	223.2

9 Conclusion and future work

In this paper, we considered the split feasibility problem, which is to find an element in the intersection of a closed set C and the linear preimage of another closed set D . We reformulated this problem as an optimization problem with a DC objective, and applied the nonmonotone proximal gradient algorithm with majorization in [22, Appendix A] for solving it. We established global convergence and studied local convergence rate of the sequence generated by our algorithm, under mild assumptions. Our numerical experiments demonstrate that our algorithm performs well on solving split feasibility problems that arise from completely positive matrix factorization, sparse matrix factorization and outlier detection.

There are several avenues for future research. For instance, as suggested by one of the referees, a possible future research direction is to extend our approach in Section 7 to find sparse matrix factorization for *rectangular* matrices. In this case, the matrix to be factorized is not necessarily symmetric, and hence, one cannot apply [17, Lemma 2.5] to reformulate this factorization problem into a split feasibility problem involving the set of orthogonal matrices, as in Section 7. However, note that one can prove the following analogue of [17, Lemma 2.5]:

Lemma 4 *If $X, \tilde{X} \in \mathbb{R}^{m \times r}$ and $Y, \tilde{Y} \in \mathbb{R}^{r \times n}$ are matrices of full rank and satisfy $XY = \tilde{X}\tilde{Y}$, then there exists an invertible matrix $Q \in \mathbb{R}^{r \times r}$ so that*

$$X = \tilde{X}Q \quad \text{and} \quad Y = Q^{-1}\tilde{Y}.$$

Thus, if a matrix $G \in \mathbb{R}^{m \times n}$ can be factorized as the product of two full-rank matrices $A \in \mathbb{R}^{m \times r}$ and $B \in \mathbb{R}^{r \times n}$, other possible factorizations of G into full-rank matrices of the same sizes can be obtained by multiplying A from the right by an

invertible matrix Q , and multiplying B from the left by Q^{-1} . Now, given an initial factorization AB of G with $A \in \mathbb{R}^{m \times r}$ and $B \in \mathbb{R}^{r \times n}$ having full rank, we can reformulate the sparse factorization problem on G as the following split feasibility problem:

$$\text{Find } Q, P \in \mathbb{R}^{r \times r} \text{ s.t. } (AQ, PB) \in D := D_s^m \times D_s^n \text{ and } (Q, P) \in C, \quad (34)$$

where $C := \{(Q, P) : QP = I\}$,

$$D_s^m := \left\{ U \in \mathbb{R}^{m \times r} : \|u_i\|_0 \leq s \text{ for each } i = 1, \dots, r \right\},$$

$$D_s^n := \left\{ V \in \mathbb{R}^{r \times n} : \|v_j\|_0 \leq s \text{ for each } j = 1, \dots, r \right\},$$

with u_i, v_j being the i th column and the j th row of U and V , respectively. One difficulty in using this formulation is that the projection onto the set C may not be easy to compute. Finding efficient ways to project onto C is an interesting future research question.

A Proof of Theorem 2

Proof The boundedness of $\{x^t\}$ follows from Theorem 1(i). We now prove convergence of the whole sequence. By assumption, x^* is an accumulation point of $\{x^t\}$ so that the function

$$\kappa(u) := \frac{1}{2}d^2(u, D)$$

is continuously differentiable at Ax^* with locally Lipschitz gradient. Then we have from [30, Example 8.53], [25, Theorem 1.110(ii)] and the chain rule that

$$\nabla(\kappa \circ A)(x^*) = A^T(Ax^* - \text{Proj}_D(Ax^*)).$$

Using this and the fact that x^* is a stationary point of the split feasibility problem (2) (see Theorem 1(ii)), we deduce further that

$$\begin{aligned} 0 &\in A^T(Ax^* - \text{Proj}_D(Ax^*)) + N_C(x^*) \\ &= \nabla(\kappa \circ A)(x^*) + N_C(x^*) = \partial F(x^*), \end{aligned}$$

where the last equality follows from [30, Exercise 8.8(c)]. In particular, it holds that $x^* \in \text{dom} \partial F$.

Since F is a KL function and $x^* \in \text{dom} \partial F$, there exist $\epsilon > 0$ and a continuous concave function ψ as in Definition 1 so that

$$\psi'(F(x) - F(x^*)) \cdot d(0, \partial F(x)) \geq 1 \quad (35)$$

whenever $\|x - x^*\| \leq \epsilon$ and $F(x^*) < F(x) < F(x^*) + \epsilon$. Moreover, by shrinking ϵ if necessary, we may assume without loss of generality that $\nabla \kappa$ is globally Lipschitz in $\{Ax : x \in B(x^*, \epsilon)\}$ with Lipschitz modulus τ .

Next, observe from (7) with $M = 0$ that $\{F(x^t)\}$ is nonincreasing. Since F is also nonnegative, we deduce that the limit $\lim_{t \rightarrow \infty} F(x^t)$ exists. In addition, notice that F is continuous in its closed domain and x^* is an accumulation point of $\{x^t\}$. Thus, we conclude that $\lim_{t \rightarrow \infty} F(x^t) = F(x^*)$.

Now, if $F(x^{t_0}) = F(x^*)$ for some $t_0 \geq 0$, then we see from (7) with $M = 0$ and $\lim_{t \rightarrow \infty} F(x^t) = F(x^*)$ that $x^{t+1} = x^t$ for all $t \geq t_0$, which implies that the sequence $\{x^t\}$ converges (finitely). Thus, from now on, we focus on the case that $F(x^t) > F(x^*)$ for all $t \geq 0$.

In this case, note from Lemma 2 that there exists $N_0 > 1$ so that $\|x^t - x^{t-1}\| \leq \frac{\epsilon}{2}$ whenever $t \geq N_0$. Also, using Lemma 2, the definition of accumulation point and the fact that $\lim_{t \rightarrow \infty} F(x^t) = F(x^*)$, there exists $N_1 \geq N_0$ so that

- (i) $\|x^{N_1} - x^*\| \leq \frac{\epsilon}{2}$ and $F(x^*) < F(x^{N_1}) < F(x^*) + \epsilon$.
- (ii) $\|x^{N_1} - x^*\| + \|x^{N_1} - x^{N_1-1}\| + C_1 \psi(F(x^{N_1}) - F(x^*)) \leq \frac{\epsilon}{2}$,

where $C_1 := \frac{2(\tau\lambda_{\max}(A^T A) + \beta)}{c}$, c is as in (7), τ is the Lipschitz continuity modulus of $\nabla\kappa$ on $\{Ax : x \in B(x^*, \epsilon)\}$, $\beta = \sup_t \bar{L}_t$ with \bar{L}_t defined in **Step 2** of $\text{SpFeas}_{\text{DC}_{\mathbb{S}}}$, and β is finite according to Lemma 2.

We claim that if $t \geq N_1$ and $\|x^t - x^*\| \leq \epsilon/2$, then

$$2\|x^{t+1} - x^t\| \leq \|x^t - x^{t-1}\| + C_1 [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))]. \quad (36)$$

To this end, note that since $x^t \in B(x^*, \epsilon/2)$ and $t \geq N_1 \geq N_0$, we have $\|x^t - x^{t-1}\| \leq \frac{\epsilon}{2}$ and hence $\|x^{t-1} - x^*\| \leq \epsilon$. Thus, κ is continuously differentiable at Ax^{t-1} and Ax^t . Moreover, we see from [30, Example 8.53] and [25, Theorem 1.110(ii)] (see also (10)) that $\nabla(\kappa \circ A)(x^{t-1}) = A^T(Ax^{t-1} - \text{Proj}_D(Ax^{t-1}))$. Using this and the definition of x^t , we deduce that

$$x^t \in \text{Proj}_C \left(x^{t-1} - \frac{\nabla(\kappa \circ A)(x^{t-1})}{\bar{L}_{t-1}} \right).$$

Thus, according to (1),

$$v^t := \bar{L}_{t-1}(x^{t-1} - x^t) - \nabla(\kappa \circ A)(x^{t-1}) \in N_C(x^t).$$

Moreover, using the definition of v^t , we have

$$\begin{aligned} \|v^t + \nabla(\kappa \circ A)(x^t)\| &\leq \|\nabla(\kappa \circ A)(x^t) - \nabla(\kappa \circ A)(x^{t-1})\| + \bar{L}_{t-1}\|x^t - x^{t-1}\| \\ &\leq (\tau\lambda_{\max}(A^T A) + \beta)\|x^t - x^{t-1}\|, \end{aligned} \quad (37)$$

where the second inequality holds because $\beta = \sup_t \bar{L}_t$ and $\nabla\kappa$ is globally Lipschitz in $\{Ax : x \in B(x^*, \epsilon)\}$ with Lipschitz modulus τ . Since $v^t + \nabla(\kappa \circ A)(x^t) \in N_C(x^t) + \nabla(\kappa \circ A)(x^t) = \partial F(x^t)$, we obtain from (37) that

$$d(0, \partial F(x^t)) \leq (\tau\lambda_{\max}(A^T A) + \beta)\|x^t - x^{t-1}\|.$$

Making use of this, the concavity of ψ and (7) with $M = 0$, we see further that

$$\begin{aligned} &(\tau\lambda_{\max}(A^T A) + \beta)\|x^t - x^{t-1}\| \cdot [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))] \\ &\geq d(0, \partial F(x^t)) \cdot [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))] \\ &\geq d(0, \partial F(x^t)) \cdot \psi'(F(x^t) - F(x^*)) \cdot [F(x^t) - F(x^{t+1})] \\ &\geq \frac{c}{2}\|x^{t+1} - x^t\|^2, \end{aligned}$$

where the last inequality follows from (7) with $M = 0$, (35), and the facts that $\|x^t - x^*\| \leq \epsilon/2$ and that $F(x^*) < F(x^t) \leq F(x^{N_1}) < F(x^*) + \epsilon$ (since $t \geq N_1$). Dividing both sides of the above inequality by $\frac{c}{2}$, taking square root, using the relation $\sqrt{ab} \leq \frac{a+b}{2}$ for any nonnegative numbers a and b and invoking the definition of C_1 , we obtain further that

$$\begin{aligned} \|x^{t+1} - x^t\| &\leq \sqrt{\|x^t - x^{t-1}\| \cdot C_1 [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))]} \\ &\leq \frac{1}{2} (\|x^t - x^{t-1}\| + C_1 [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))]), \end{aligned}$$

from which (36) follows immediately.

Next, we show by induction that $x^t \in B(x^*, \epsilon/2)$ whenever $t \geq N_1$. The case $t = N_1$ follows from construction. Suppose that $x^t \in B(x^*, \epsilon/2)$ whenever $t = N_1, \dots, N_1 + k - 1$ for some

$k \geq 1$. Then

$$\begin{aligned}
\|x^{N_1+k} - x^*\| &\leq \|x^{N_1} - x^*\| + \sum_{t=N_1}^{N_1+k-1} \|x^{t+1} - x^t\| \\
&\stackrel{(a)}{\leq} \|x^{N_1} - x^*\| \\
&\quad + \sum_{t=N_1}^{N_1+k-1} (\|x^t - x^{t-1}\| - \|x^{t+1} - x^t\| + C_1 [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))]) \\
&\leq \|x^{N_1} - x^*\| + \|x^{N_1} - x^{N_1-1}\| + C_1 \psi(F(x^{N_1}) - F(x^*)) \stackrel{(b)}{\leq} \frac{\epsilon}{2},
\end{aligned}$$

where (a) follows from the induction hypothesis and (36), and (b) follows from the definition of N_1 . Thus, $x^t \in B(x^*, \epsilon/2)$ whenever $t \geq N_1$ by induction.

Since $x^t \in B(x^*, \epsilon/2)$ whenever $t \geq N_1$, we can sum both sides of (36) from N_1 to ∞ and obtain

$$\begin{aligned}
&\sum_{t=N_1}^{\infty} \|x^{t+1} - x^t\| \\
&\leq \sum_{t=N_1}^{\infty} \left(\|x^t - x^{t-1}\| - \|x^{t+1} - x^t\| + C_1 [\psi(F(x^t) - F(x^*)) - \psi(F(x^{t+1}) - F(x^*))] \right) \\
&\leq \|x^{N_1} - x^{N_1-1}\| + C_1 \psi(F(x^{N_1}) - F(x^*)) < \infty.
\end{aligned}$$

Thus, the sequence $\{x^t\}$ is Cauchy and is hence convergent. \square

References

1. P. A. Absil and J. Malick. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22:135–158, 2012.
2. E. Asplund. Differentiability of the metric projection in finite-dimensional Euclidean space. *Proceedings of the American Mathematical Society*, 38:218–219, 1973.
3. H. Attouch and J. Bolte. On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Mathematical Programming*, 116:5–16, 2009.
4. H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35:438–457, 2010.
5. H. Attouch, J. Bolte, and B. F. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Mathematical Programming*, 137:91–129, 2013.
6. A. Auslender and M. Teboulle. *Asymptotic Cones and Functions in Optimization and Variational Inequalities*. Springer, 2003.
7. A. Berman, M. Dür, and N. Shaked-Monderer. Open problems in the theory of completely positive and copositive matrices. *Electronic Journal of Linear Algebra*, 29:46–58, 2015.
8. J. Bolte, A. Daniilidis, and A. Lewis. The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. *SIAM Journal on Optimization*, 17:1205–1223, 2007.
9. I. M. Bomze, P. J. C. Dickinson, G. Still. The structure of completely positive matrices according to their CP-rank and CP-plus-rank. *Linear Algebra and its Applications*, 482:191–206, 2015.
10. C. Byrne. Iterative oblique projection onto convex sets and the split feasibility problem. *Inverse Problems*, 18:441–453, 2002.
11. Y. Censor and T. Elfving. A multiprojection algorithm using Bregman projections in a product space. *Numerical Algorithms*, 8:221–239, 1994.
12. Y. Censor, T. Elfving, N. Kopf, and T. Bortfeld. The multiple-sets split feasibility problem and its applications for inverse problems. *Inverse Problems*, 21:2071–2084, 2005.

13. Y. Censor, A. Motova, and A. Segal. Perturbed projections and subgradient projections for the multiple-sets split feasibility problem. *Journal of Mathematical Analysis and Applications*, 327:1244–1256, 2007.
14. P. J. C. Dickinson and L. Gijben. On the computational complexity of membership problems for the completely positive cone and its dual. *Computational Optimization and Applications*, 57:403–415, 2014.
15. C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.
16. F. Facchinei and J.-S. Pang. *Finite-dimensional Variational Inequalities and Complementarity Problems*, Vol I. Springer, 2003.
17. P. Groetzner and M. Dür. A factorization method for completely positive matrices. *Linear Algebra and its Applications*, 591:1–24, 2020.
18. A. Kyrillidis, S. Becker, V. Cevher, and C. Koch. Sparse projections onto the simplex. In *Proceedings of the 30th International Conference on Machine Learning*, pages 280–288, 2013.
19. A. S. Lewis, D. R. Luke, and J. Malick. Local linear convergence for alternating and averaged nonconvex projections. *Foundations of Computational Mathematics*, 9:485–513, 2009.
20. G. Li and T. K. Pong. Calculus of the exponent of Kurdyka-Lojasiewicz inequality and its applications to linear convergence of first-order methods. *Foundations of Computational Mathematics*, 18:1199–1232, 2018.
21. T. Liu, T. K. Pong, and A. Takeda. A refined convergence analysis of pDCA _{ϵ} with applications to simultaneous sparse recovery and outlier detection. To appear in *Computational Optimization and Applications*. DOI:10.1007/s10589-019-00067-z.
22. T. Liu, T. K. Pong, and A. Takeda. A successive difference-of-convex approximation method for a class of nonconvex nonsmooth optimization problems. To appear in *Mathematical Programming*. DOI:10.1007/s10107-018-1327-8.
23. G. López, V. Martín-Márquez, F. Wang, and H.-K. Xu. Solving the split feasibility problem without prior knowledge of matrix norms. *Inverse Problems*, 28:085004, 2012.
24. Z. Lu and Y. Zhang. Sparse approximation via penalty decomposition methods. *SIAM Journal on Optimization*, 23:2448–2478, 2013.
25. B. S. Mordukhovich. *Variational Analysis and Generalized Differentiation, I: Basic Theory*. Springer, 2006.
26. B. Neyshabur and R. Panigrahy. Sparse matrix factorization. Available at <https://arxiv.org/abs/1311.3315>, 2014.
27. L. F. Polania, R. E. Carrillo, M. Blanco-Velasco, and K. E. Barner. Compressive sensing for ECG signals in the presence of electromyographic noise. In *Proceedings of the 38th Annual Northeast Bioengineering Conference*, pages 295–296, 2012.
28. R. A. Poliquin, R. T. Rockafellar, and L. Thibault. Local differentiability of distance functions. *Transactions of the American Mathematical Society*, 352:5231–5249, 2000.
29. B. Qu and N. Xiu. A note on the CQ algorithm for the split feasibility problem. *Inverse Problems*, 21:1655–1665, 2005.
30. R. T. Rockafellar and R. J-B. Wets. *Variational Analysis*. Springer, 1998.
31. Y. Shehu and O. S. Iyiola. Strong convergence result for proximal split feasibility problem in Hilbert spaces. *Optimization*, 66:2275–2290, 2017.
32. Z. Wang, Q. Yang, and Y. Yang. The relaxed inexact projection methods for the split feasibility problem. *Applied Mathematics and Computation*, 217:5347–5359, 2011.
33. J. Xu, E. C. Chi, M. Yang, and K. Lange. A majorization-minimization algorithm for split feasibility problems. *Computational Optimization and Applications*, 71:795–828, 2018.
34. Q. Yang. The relaxed CQ algorithm solving the split feasibility problem. *Inverse Problems*, 20:1261–1266, 2004.
35. J. Zhao and Q. Yang. Several solution methods for the split feasibility problem. *Inverse Problems*, 21:1791–1799, 2005.