A Block Coordinate Descent Method for Regularized Multiconvex Optimization with Applications to Nonnegative Tensor Factorization and Completion

Yangyang Xu† and Wotao Yin†

Abstract. This paper considers regularized block multiconvex optimization, where the feasible set and objective function are generally nonconvex but convex in each block of variables. It also accepts nonconvex blocks and requires these blocks to be updated by proximal minimization. We review some interesting applications and propose a generalized block coordinate descent method. Under certain conditions, we show that any limit point satisfies the Nash equilibrium conditions. Furthermore, we establish global convergence and estimate the asymptotic convergence rate of the method by assuming a property based on the Kurdyka–Lojasiewicz inequality. The proposed algorithms are tested on nonnegative matrix and tensor factorization, as well as matrix and tensor recovery from incomplete observations. The tests include synthetic data and hyperspectral data, as well as image sets from the CBCL and ORL databases. Compared to the existing state-of-the-art algorithms, the proposed algorithms demonstrate superior performance in both speed and solution quality. The MATLAB code of nonnegative matrix/tensor decomposition and completion, along with a few demos, are accessible from the authors’ homepages.

Key words. block multiconvex, block coordinate descent, Kurdyka–Lojasiewicz inequality, Nash equilibrium, nonnegative matrix and tensor factorization, matrix completion, tensor completion, proximal gradient method

AMS subject classifications. 49M20, 65B05, 90C26, 90C30, 90C52

DOI. 10.1137/120887795

1. Introduction. In this paper, we consider the optimization problem

\[
\min_{\mathbf{x} \in \mathcal{X}} F(\mathbf{x}_1, \ldots, \mathbf{x}_s) \equiv f(\mathbf{x}_1, \ldots, \mathbf{x}_s) + \sum_{i=1}^{s} r_i(\mathbf{x}_i),
\]

where variable \( \mathbf{x} \) is decomposed into \( s \) blocks \( \mathbf{x}_1, \ldots, \mathbf{x}_s \), the set \( \mathcal{X} \) of feasible points is assumed to be a closed and block multiconvex subset of \( \mathbb{R}^n \), \( f \) is assumed to be a differentiable and block multiconvex function, and \( r_i, i = 1, \ldots, s \), are extended-value convex functions. Set \( \mathcal{X} \) and function \( f \) can be nonconvex over \( \mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_s) \).

We call a set \( \mathcal{X} \) block multiconvex if its projection to each block of variables is convex; namely, for each \( i \) and fixed \( (s - 1) \) blocks \( \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_s \), the set

\[
\mathcal{X}_i(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_s) \triangleq \{ \mathbf{x}_i \in \mathbb{R}^{n_i} : (\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_s) \in \mathcal{X} \}
\]
is convex. We call a function \( f \) block multiconvex if, for each \( i \), \( f \) is a convex function of \( x_i \) while all the other blocks are fixed. Therefore, when all but one block are fixed, (1.1) over the free block is a convex problem. (Later, using the proximal update (1.3b), we allow \( f \) to be nonconvex over a block.)

Extended value means \( r_i(x_i) = \infty \) if \( x_i \notin \text{dom}(r_i) \), \( i = 1, \ldots, s \). In particular, \( r_i \) (or a part of \( r_i \)) can be indicator functions of convex sets. We use \( x \in \mathcal{X} \) to model joint constraints, and \( r_1, \ldots, r_s \) to include individual constraints of \( x_1, \ldots, x_s \) when they are present. In addition, \( r_i \) can include nonsmooth functions.

Our main interest is the block coordinate descent (BCD) method of Gauss–Seidel type, which minimizes \( F \) cyclically over each of \( x_1, \ldots, x_s \) while fixing the remaining blocks at their last updated values. Let \( x_i^k \) denote the value of \( x_i \) after its \( k \)th update, and let

\[
f_i^k(x_i) \triangleq f(x_1^k, \ldots, x_{i-1}^k, x_i, x_{i+1}^k, \ldots, x_s^k) \quad \forall i, \forall k.
\]

At each step, we consider three different updates:

\[
(1.3a) \quad \text{Original: } x_i^k = \arg\min_{x_i \in \mathcal{X}_i^k} f_i^k(x_i) + r_i(x_i);
\]

\[
(1.3b) \quad \text{Proximal: } x_i^k = \arg\min_{x_i \in \mathcal{X}_i^k} f_i^k(x_i) + \frac{L_{i-1}^{k-1}}{2} \| x_i - x_i^{k-1} \|^2 + r_i(x_i); \quad \text{and}
\]

\[
(1.3c) \quad \text{Prox-linear: } x_i^k = \arg\min_{x_i \in \mathcal{X}_i^k} \langle \nabla_i^k, x_i - x_i^{k-1} \rangle + \frac{L_{i-1}^{k-1}}{2} \| x_i - x_i^{k-1} \|^2 + r_i(x_i),
\]

where \( \| \cdot \| \) denotes the \( \ell_2 \)-norm, \( L_{i-1}^{k-1} > 0 \),

\[
\mathcal{X}_i^k = \mathcal{X}(x_1^k, \ldots, x_{i-1}^k, x_{i+1}^k, \ldots, x_s^k),
\]

and, in the last type of update (1.3c),

\[
(1.4) \quad x_i^{k-1} = x_i^{k-1} + \omega_i^{k-1}(x_i^{k-1} - x_i^{k-2})
\]

denotes an extrapolated point, \( \omega_i^{k-1} \geq 0 \) is the extrapolation weight, and \( \nabla_i^k = \nabla f_i^k(x_i^{k-1}) \) is the block-partial gradient of \( f \) at \( x_i^{k-1} \). We consider extrapolation (1.4) for update (1.3c) since it significantly accelerates the convergence of BCD in our applications. The framework of BCD is given in Algorithm 1, which allows each \( x_i \) to be updated by (1.3a), (1.3b), or (1.3c).

When \( \mathcal{X} \) and \( f \) are block multiconvex, all three subproblems in (1.3) are convex. In general, the three updates generate different sequences and thus can cause BCD to converge to different solutions. We found that, in many tests, applying (1.3c) on all or some blocks gives solutions of lower objective values, possibly because its local prox-linear approximation helps avoid the small regions around certain local minima. In addition, it is generally more time-consuming to compute (1.3a) and (1.3b) than (1.3c), though each time the former two tend to make larger objective decreases than (1.3c) applied without extrapolation. We consider all three updates since they fit different applications, and also different blocks in the same application, yet their convergence can be analyzed in a unified framework.
Algorithm 1. Block coordinate descent (BCD) method for solving (1.1).

**Initialization:** choose two initial points \((x_1^{-1}, \ldots, x_s^{-1}) = (x_0^1, \ldots, x_0^s)\)

for \(k = 1, 2, \ldots\) do
  for \(i = 1, 2, \ldots, s\) do
    \(x^k_i \leftarrow (1.3a), (1.3b), \text{ or } (1.3c).\)
  end for
  if stopping criterion is satisfied then
    return \((x^k_1, \ldots, x^k_s)\).
  end if
end for

To ensure the convergence of Algorithm 1, for every block \(i\) to which \((1.3a)\) is applied, we require \(f^k_i(x_i)\) to be strongly convex, and for every block \(i\) to which \((1.3c)\) is applied, we require \(\nabla f^k_i(x_i)\) to be Lipschitz continuous. The parameter \(L^k_i\) in both \((1.3b)\) and \((1.3c)\) can be fixed for all \(k\). For generality and faster convergence, we allow it to change during the iterations. Use of \((1.3b)\) only requires \(L^k_i\) to be uniformly lower bounded from zero and uniformly upper bounded. In fact, \(f^k_i\) in \((1.3)\) can be nonconvex, and our proof still goes through. Update \((1.3b)\) is a good replacement for \((1.3a)\) if \(f^k_i\) is not strongly convex. Use of \((1.3c)\) requires more conditions on \(L^k_i\); see Lemmas 2.2 and 2.6. Update \((1.3c)\) is relatively easy to solve and often allows closed form solutions. For block \(i\), \((1.3c)\) is preferred over \((1.3a)\) and \((1.3b)\) when they are expensive to solve and \(f^k_i\) has Lipschitz continuous gradients. Overall, the three choices cover a large number of cases.

Original subproblem \((1.3a)\) is the most used form in BCD and has been extensively studied. It dates back to methods in [52] for solving equation systems and to works [24, 70, 5, 61], which analyze the method assuming \(F\) to be convex (or quasi-convex or hemivariate) and differentiable, and to have bounded level sets except for certain classes of convex functions. When \(F\) is nonconvex, BCD may cycle and stagnate [56]. However, subsequence convergence can be obtained for special cases such as quadratic function [48], strict pseudoconvexity in each of \((s - 2)\) blocks [22], and unique minimizer per block [47, p. 195]. If \(F\) is nondifferentiable, BCD can get stuck at a nonstationary point; see [5, p. 94]. However, subsequence convergence can be obtained if the nondifferentiable part is separable; see works [23, 50, 65, 66] for results on different forms of \(F\). In our objective function, \(f\) is differentiable and possibly nonconvex, and the nonsmooth part consists of block-separable functions \(r_i\).

Proximal subproblem \((1.3b)\) has been used with BCD in [22]. For \(X = \mathbb{R}^n\), the authors’ work shows that every limit point is a critical point. Recently, this method was revisited in [4] for only two blocks and was shown to converge globally via the Kurdyka–Lojasiewicz (KL) inequality.

Prox-linear subproblem \((1.3c)\) with extrapolation is new but very similar to the update in the block coordinate gradient descent (BCGD) method of [67], which identifies a block descent direction by gradient projection and then performs an Armijo-type line search. The paper [67] does not use extrapolation \((1.4)\). This work considers more general functions \(f\) which are smooth but not necessarily multiconvex, and it does not consider joint constraints. Through private communication, we learned that the recent report [57] provides a unified convergence analysis of coordinatewise successive minimization methods for nonsmooth nonconvex opti-
lishes the square summable result both synthetic and real data in both speed and solution quality. It is demonstrated to be superior to the state-of-the-art algorithms on complete observations. If $r_i$ that make (1.3c) easier to compute than (1.3a) and (1.3b). For instance, if $r_i = \delta d_i$, the indicator function of convex set $D_i$ (equivalent to $x_i \in D_i$), (1.3c) reduces to $x_i^k = P_{A_i \cap D_i}(x_i^{k-1} - \hat{g}_i^{k-1}/L_i^{k-1})$, where $P_{A_i \cap D_i}$ is the projection to set $A_i \cap D_i$. If $r_i(x_i) = \lambda_i \|x_i\|_1$ and $A_i = \mathbb{R}^m$, (1.3c) reduces to $x_i^k = S_{\lambda_i L_i^{k-1}}(x_i^{k-1} - \hat{g}_i^{k-1}/L_i^{k-1})$, where $S_{\lambda}(\cdot)$ is soft-thresholding, defined componentwise as $S_{\nu}(t) = \text{sign}(t) \max(|t| - \nu, 0)$. More examples arise in joint/group $\ell_1$ and nuclear norm minimization, total variation, etc.

1.1. Contributions. We propose Algorithm 1 and establish its global convergence and asymptotic rate of convergence. The algorithm is applied to two classes of problems: (i) nonnegative matrix/tensor factorization, and (ii) nonnegative matrix/tensor completion from incomplete observations. It is demonstrated to be superior to the state-of-the-art algorithms on both synthetic and real data in both speed and solution quality.

Our convergence analysis takes two steps. Under certain assumptions, the first step establishes the square summable result $\sum_k \|x^k - x^{k+1}\|^2 < \infty$ and obtains subsequence convergence to Nash equilibrium points, as well as global convergence to a single Nash point if the sequence is bounded and the Nash points are isolated. The second step, which is motivated by [4], assumes the KL inequality [13, 14] and improves the result to $\sum_k \|x^k - x^{k+1}\| < \infty$, which gives the algorithm’s global convergence, as well as asymptotic rates of convergence. The classes of functions that obey the KL inequality are reviewed. Despite the popularity of BCD, very few works establish global convergence without the (quasi-)convexity assumption on $F$; the authors of [48, 67] have obtained global convergence by assuming a local Lipschitzian error bound and the isolation of the isocost surfaces of $F$. Some interesting problems satisfy their assumptions; however, it appears that our assumptions are met by more problems and are easier to verify. Their and our assumptions do not imply each other, though there are problems satisfying both.

1.2. Applications. A large number of practical problems can be formulated in the form of (1.1) such as convex problems: (group) Lasso [64, 75] or the basis pursuit (denoising) [15], low-rank matrix recovery [58], hybrid Huberized support vector machine [69], and so on. We give some nonconvex examples as follows.

Blind source separation and sparse dictionary learning. Let $s_1, \ldots, s_n \in \mathbb{R}^{1 \times p}$ be a set of source signals. Given $m$ sensor signals $x_i = \sum_{j=1}^n a_{ij} s_j + \eta_i$, $i = 1, \ldots, m$, where $A = [a_{ij}]_{m \times n} \in \mathbb{R}^{m \times n}$ is an unknown mixing matrix and $\eta_i$ is noise, blind source separation (BSS) [27] aims to estimate both $A$ and $S = [s_1^\top, \ldots, s_n^\top]^\top$. It has found applications in many areas such as artifact removal [26] and image processing [28]. Two classical approaches for BSS are principal component analysis (PCA) [62] and independent component analysis (ICA) [18]. If $m < n$ and no prior information on $A$ and $S$ is given, these methods will fail. Assuming that $s_1, \ldots, s_n$ are sparse under some dictionary $B \in \mathbb{R}^{T \times p}$, namely, $s_i = y_i B$ and $y_i \in \mathbb{R}^{1 \times T}$ is sparse for $i = 1, \ldots, n$ [79, 12], use the sparse BSS model

\begin{equation}
\min_{A, Y} \frac{\lambda}{2} \|AY - X\|_F^2 + r(Y) \quad \text{subject to } A \in D,
\end{equation}
where \( Y = [y_1^T, \ldots, y_n^T]^T \in \mathbb{R}^{n \times T} \), \( r(Y) \) is a sparsity regularizer such as \( r(Y) = \|Y\|_1 \), \( D \) is a convex set to control the scale of \( A \) such as \( \|A\|_F \leq 1 \), and \( \lambda \) is a balancing parameter. Note that model (1.5) is block multiconvex but nonconvex jointly with respect to \( A \) and \( Y \). A similar model which appears in cosmic microwave background analysis \cite{10} solves

\[
\min_{A,Y} \lambda \frac{1}{2} \text{trace} \left( (AYB - X)^\top C^{-1}(AYB - X) \right) + r(Y) \quad \text{subject to } A \in D
\]

for a certain covariance matrix \( C \). Algorithms for (sparse) BSS include an online learning algorithm \cite{2}, feature extraction method \cite{43}, feature sign algorithm \cite{40}, and so on.

Model (1.5) with \( B = I \) also arises in sparse dictionary training \cite{1, 49}, where the goal is to build a dictionary \( A \) that sparsely represents the signals in \( X \).

**Nonnegative matrix factorization.** Nonnegative matrix factorization (NMF) was first proposed by Paatero and his coworkers in the area of environmental science \cite{53}. The later popularity of NMF can be partially attributed to the publication of \cite{38} in *Nature*. It has been widely applied in data mining such as text mining \cite{55} and image mining \cite{41}, dimension reduction and clustering \cite{16, 73}, and hyperspectral endmember extraction, as well as spectral data analysis \cite{54}. A widely used model for (regularized) NMF is

\[
\min_{X \geq 0, Y \geq 0} \frac{1}{2} \|XY - M\|_F^2 + \alpha r_1(X) + \beta r_2(Y),
\]

where \( M \) is the input nonnegative matrix, \( r_1, r_2 \) are some regularizers promoting solution structures, and \( \alpha, \beta \) are weight parameters. Two early popular algorithms for NMF are the projected alternating least squares (ALS) method \cite{53} and the multiplicative updating method \cite{39}. Due to the biconvexity of the objective in (1.7), a series of alternating nonnegative least squares (ANLS) methods have been proposed such as \cite{42, 30, 32}; they are BCDs with update (1.3a). Recently, the classic alternating direction method (ADM) has been applied in \cite{78}. We compare the proposed algorithms to some existing methods for NMF in section 4.

Similar models also arise in low-rank matrix recovery, such as the one considered in \cite{58},

\[
\min_{X, Y} \frac{1}{2} \|A(XY) - b\|^2 + \alpha \|X\|_F^2 + \beta \|Y\|_F^2,
\]

where \( A \) is a linear operator. The method of multipliers is employed in \cite{58} to solve (1.8) with no convergence guarantees. Since the objective of (1.8) is coercive and real analytic, our algorithm is guaranteed to produce a sequence of points that globally converge to a critical point; see Theorems 2.8 and 2.9.

**Nonnegative tensor factorization.** Nonnegative tensor factorization (NTF) is a generalization of NMF to multidimensional arrays. One commonly used model for NTF is based on CANDECOMP/PARAFAC tensor decomposition \cite{71}

\[
\min_{A_1, \ldots, A_N \geq 0} \frac{1}{2} \|\mathcal{M} - A_1 \circ A_2 \circ \cdots \circ A_N\|_F^2 + \sum_{n=1}^N \lambda_n r_n(A_n),
\]
and another one is based on Tucker decomposition [34]

\[(1.10) \quad \min_{\mathbf{G}, \mathbf{A}_1, \ldots, \mathbf{A}_N \geq 0} \frac{1}{2} \| \mathbf{M} - \mathbf{G} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 \cdots \times_N \mathbf{A}_N \|^2_F + \lambda r(\mathbf{G}) + \sum_{n=1}^{N} \lambda_n r_n(\mathbf{A}_n), \]

where \(\mathbf{M}\) is a given nonnegative tensor; \(r, r_1, \ldots, r_N\) are regularizers; \(\lambda, \lambda_1, \ldots, \lambda_N\) are weight parameters; and “\(\times\)” and “\(\times_n\)” represent outer product and tensor-matrix multiplication, respectively. (The necessary background of tensors is reviewed in section 3.) Most algorithms for solving NMF have been directly extended to NTF. For example, the multiplicative update in [53] is extended to solving (1.9) in [63]. The ANLS methods in [30, 32] are extended to solving (1.9) in [31, 33]. Algorithms for solving (1.10) also include the columnwise coordinate descent method [44] and the ALS method [21]. More about NTF algorithms can be found in [76].

1.3. Organization. The rest of the paper is organized as follows. Section 2 studies the convergence of Algorithm 1. In section 3, Algorithm 1 is applied to both the nonnegative matrix/tensor factorization problem and the completion problem. The numerical results are presented in section 4. Finally, section 5 concludes the paper.

2. Convergence analysis. In this section, we analyze the convergence of Algorithm 1 under the following assumptions.

Assumption 1. \(F\) is continuous in \(\text{dom}(F)\) and \(\inf_{\mathbf{x} \in \text{dom}(F)} F(\mathbf{x}) > -\infty\). Problem (1.1) has a Nash point (see (2.3) for definition).

Assumption 2. Each block \(i\) is updated by the same scheme among (1.3a)–(1.3c) for all \(k\). Let \(I_1, I_2,\) and \(I_3\) denote the set of blocks updated by (1.3a), (1.3b), and (1.3c), respectively. In addition, there exist constants \(0 < \ell_i \leq L_i < \infty\), \(i = 1, \ldots, s\), such that

1. for \(i \in I_1\), \(f_i^k\) is strongly convex with modulus \(\ell_i \leq L_i^k \leq L_i\), namely,

\[(2.1) \quad f_i^k(\mathbf{u}) - f_i^k(\mathbf{v}) \geq \langle \nabla f_i^k(\mathbf{v}), \mathbf{u} - \mathbf{v} \rangle + \frac{L_i^k - 1}{2} \| \mathbf{u} - \mathbf{v} \|^2 \quad \forall \mathbf{u}, \mathbf{v} \in \lambda_i^k; \]

2. for \(i \in I_2\), parameters \(L_i^{k-1}\) obey \(\ell_i \leq L_i^{k-1} \leq L_i\);
3. for \(i \in I_3\), \(\nabla f_i^k\) is Lipschitz continuous, and parameters \(L_i^{k-1}\) obey \(\ell_i \leq L_i^{k-1} \leq L_i\) and

\[(2.2) \quad f_i^k(\mathbf{x}_i^k) \leq f_i^k(\mathbf{x}_i^{k-1}) + \langle \nabla f_i^k(\mathbf{x}_i^{k-1}), \mathbf{x}_i^k - \mathbf{x}_i^{k-1} \rangle + \frac{L_i^{k-1}}{2} \| \mathbf{x}_i^k - \mathbf{x}_i^{k-1} \|^2. \]

Remark 2.1. The same notation \(L_i^{k-1}\) is used in all three schemes for the simplicity of unified convergence analysis, but we want to emphasize that it has different meanings in the three different schemes. For \(i \in I_1\), \(L_i^{k-1}\) is determined by the objective and the current values of all other blocks, while for \(i \in I_2 \cup I_3\) we have some freedom to choose \(L_i^{k-1}\). For \(i \in I_2\), \(L_i^{k-1}\) can be simply fixed to a positive constant or selected by a predetermined rule to be uniformly lower bounded from zero and upper bounded. For \(i \in I_3\), \(L_i^{k-1}\) is selected to satisfy (2.2). Taking \(L_i^{k-1}\) as the Lipschitz constant of \(\nabla f_i^k\) can satisfy (2.2). However, we allow smaller \(L_i^{k-1}\), which can speed up the algorithm.
In addition, we want to emphasize that we make different assumptions on the three different schemes. The use of (1.3a) requires block strong convexity with modulus uniformly away from zero and upper bounded, and the use of (1.3c) requires a block Lipschitz continuous gradient. The use of (1.3b) requires neither strong convexity nor Lipschitz continuity. Even the block convexity is unnecessary for (1.3b), and our proof still goes through. Each assumption on the corresponding scheme guarantees sufficient decrease of the objective and makes it square summable; see Lemma 2.2, which plays the key role in our convergence analysis.

For our analysis below, we need the Nash equilibrium condition of (1.1): for \( i = 1, \ldots, s \),

\[
F(\bar{x}_1, \ldots, \bar{x}_{i-1}, \bar{x}_i, \bar{x}_{i+1}, \ldots, \bar{x}_s) \leq F(\bar{x}_1, \ldots, \bar{x}_{i-1}, x_i, \bar{x}_{i+1}, \ldots, \bar{x}_s) \quad \forall x_i \in \bar{X}_i,
\]

or, equivalently,

\[
\langle \nabla_{x_i} f(x) + p_i, x_i - \bar{x}_i \rangle \geq 0 \quad \forall x_i \in \bar{X}_i \text{ and for some } p_i \in \partial r_i(x_i),
\]

where \( \bar{X}_i = \bar{x}_i'(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_s) \) and \( \partial r(x_i) \) is the limiting subdifferential (e.g., see [60]) of \( r \) at \( x_i \). We call \( x \) a Nash point or block coordinatewise minimizer. Let \( \mathcal{N} \) be the set of all Nash points, which we assume to be nonempty.

**Remark 2.2.** As shown in [4], it holds that

\[
\partial F(x) = \{ \nabla_{x_1} f(x) + \partial r_1(x_1) \} \times \cdots \times \{ \nabla_{x_s} f(x) + \partial r_s(x_s) \}.
\]

Therefore, if \( \mathcal{X} = \mathbb{R}^n \) or \( \bar{x} \) is an interior point of \( \mathcal{X} \), (2.4) reduces to the first-order optimality condition \( 0 \in \partial F(\bar{x}) \), and \( \bar{x} \) is a critical point (or stationary point) of (1.1). In general, the condition (2.4) is weaker than the first-order optimality condition. For problem (1.1), a critical point must be a Nash point, but a Nash point is not necessarily a critical point. An example can be found in section 4 of [72].

2.1. Preliminary result. The analysis in this subsection follows the following steps. First, we show sufficient descent at each step (inequality (2.8)), from which we establish the square summable result (Lemma 2.2). Next, the square summable result is exploited to show that any limit point is a Nash point in Theorem 2.3. Finally, with the additional assumptions of isolated Nash points and bounded \{\( x^k \)\}, global convergence is obtained in Corollary 2.4. The first step is essential, while the last two use rather standard arguments. We begin with the following lemma, which is similar to Lemma 2.3 of [8]. Since the proof in [8] does not consider constraints, we include a slightly changed proof for completeness.

**Lemma 2.1.** Let \( \xi_1(u) \) and \( \xi_2(u) \) be two convex functions defined on the convex set \( \mathcal{U} \), and let \( \xi_1(u) \) be differentiable. Let \( \xi(u) = \xi_1(u) + \xi_2(u) \) and \( u^* = \arg\min_{u \in \mathcal{U}} \langle \nabla \xi_1(v), u - v \rangle + \frac{L}{2} \| u - v \|^2 + \xi_2(u) \). If

\[
\xi_1(u^*) \leq \xi_1(v) + \langle \nabla \xi_1(v), u^* - v \rangle + \frac{L}{2} \| u^* - v \|^2,
\]

then we have

\[
\xi(u) - \xi(u^*) \geq \frac{L}{2} \| u^* - v \|^2 + L \langle v - u, u^* - v \rangle \quad \text{for any } u \in \mathcal{U}.
\]
Proof. Since \( u^* = \arg\min_{u \in U} (\nabla \xi_1(v), u - v) + \frac{L}{2}\|u - v\|^2 + \xi_2(u) \), the first-order optimality condition holds; i.e.,

\[
(2.7) \quad \langle \nabla \xi_1(v) + L(u^* - v) + g, u - u^* \rangle \geq 0 \quad \text{for any } u \in U,
\]

for some \( g \in \partial \xi_2(u^*) \). For any \( u \in U \), we have

\[
\xi(u) - \xi(u^*) \\
\geq \xi(u) - (\xi_1(v) + \langle \nabla \xi_1(v), u^* - v \rangle + \frac{L}{2}\|u^* - v\|^2) - \xi_2(u^*) \\
= \xi_1(u) - \xi_1(v) - \langle \nabla \xi_1(v), u - v \rangle + \langle \nabla \xi_1(v), u - u^* \rangle + \xi_2(u) - \xi_2(u^*) - \frac{L}{2}\|u^* - v\|^2 \\
\geq \xi_2(u) - \xi_2(u^*) - \langle g, u - u^* \rangle - L\|u^* - v, u - u^* \rangle - \frac{L}{2}\|u^* - v\|^2 \\
\geq -L\|u^* - v, u - u^* \rangle - \frac{L}{2}\|u^* - v\|^2 \\
= \frac{L}{2}\|u^* - v\|^2 + L\langle v - u, u^* - v \rangle,
\]

where the first inequality uses (2.5), the second inequality is obtained from the convexity of \( \xi_1 \) and (2.7), and the last inequality uses the convexity of \( \xi_2 \) and the fact that \( g \in \partial \xi_2(u^*) \). This completes the proof. 

Based on this lemma, we can show our key lemma below.

**Lemma 2.2 (square summable \( \|x^k - x^{k+1}\| \)).** Under Assumptions 1 and 2, let \( \{x^k\} \) be the sequence generated by Algorithm 1 with \( 0 \leq \omega_i^{k-1} \leq \delta_i \sqrt{L_i^{k-2}/L_i^{k-1}} \) for \( \delta_i < 1 \) uniformly over all \( i \in I_3 \) and \( k \). Then \( \sum_{k=0}^{\infty} \|x^k - x^{k+1}\|^2 < \infty \).

**Proof.** For \( i \in I_3 \), we have inequality (2.2) and use Lemma 2.1 by letting \( F_i^k \triangleq f_i^k + r_i \) and taking \( \xi_1 = f_i^k, \xi_2 = r_i, v = \hat{x}_{i}^{k-1}, \) and \( u = x_i^{k-1} \) in (2.6) to have

\[
F_i^k(x_i^{k-1}) - F_i^k(x_i^{k}) \geq \frac{L_i^{k-1}}{2}\|x_i^{k-1} - x_i^{k}\|^2 + L_i^{k-1}\langle \hat{x}_{i}^{k-1} - x_i^{k-1}, x_i^{k} - \hat{x}_{i}^{k-1} \rangle \\
= \frac{L_i^{k-1}}{2}\|x_i^{k-1} - x_i^{k}\|^2 - \frac{L_i^{k-1}}{2}(\omega_i^{k-1})^2\|x_i^{k-2} - x_i^{k-1}\|^2 \\
\geq \frac{L_i^{k-1}}{2}\|x_i^{k-1} - x_i^{k}\|^2 - \frac{L_i^{k-2}}{2}\delta_i^2\|x_i^{k-2} - x_i^{k-1}\|^2.
\]

(2.8)

For \( i \in I_1 \cup I_2 \) we have \( F_i^k(x_i^{k-1}) - F_i^k(x_i^{k}) \geq \frac{L_i^{k-1}}{2}\|x_i^{k-1} - x_i^{k}\|^2 \), and thus inequality (2.8) still holds. Therefore,

\[
F(x^{k-1}) - F(x^k) = \sum_{i=1}^{s} \left( F_i^k(x_i^{k-1}) - F_i^k(x_i^{k}) \right) \\
= \sum_{i=1}^{s} \left( \frac{L_i^{k-1}}{2}\|x_i^{k-1} - x_i^{k}\|^2 - \frac{L_i^{k-2}}{2}\delta_i^2\|x_i^{k-2} - x_i^{k-1}\|^2 \right).
\]
Summing the above inequality over \( k \) from 1 to \( K \), we have
\[
F(x^0) - F(x^K) \geq \sum_{k=1}^{K} \sum_{i=1}^{s} \left( \frac{L_{i}^{k-1}}{2} \|x_{i}^{k-1} - x_{i}^{k}\|^{2} - \frac{L_{i}^{k-2}}{2} \delta_{i}^{2} \|x_{i}^{k-2} - x_{i}^{k-1}\|^{2} \right)
\]
\[\geq \sum_{k=1}^{K} \sum_{i=1}^{s} \left( 1 - \frac{\delta_{i}^{2}}{2} \right) L_{i}^{k-1} \|x_{i}^{k-1} - x_{i}^{k}\|^{2} \geq \sum_{k=1}^{K} \left( 1 - \frac{\delta_{i}^{2}}{2} \right) L_{i} \|x^{k-1} - x^{k}\|^{2}.\]

Since \( F \) is lower bounded, taking \( K \to \infty \) completes the proof.

Now, we can establish the following preliminary convergence result.

**Theorem 2.3 (limit point is Nash point).** Define the difference measure of two sets \( X, Y \) by
\[
\text{diff}(X, Y) = \max \left( \sup_{x \in X} \inf_{y \in Y} \|x - y\|, \sup_{y \in Y} \inf_{x \in X} \|x - y\| \right).
\]
Assume that the set map \( X_i(\cdot) \) defined in (1.2) continuously changes; namely, \( x^{k'}, x \in X \), and \( x^{k'} \to x \) imply
\[
\lim_{k \to \infty} \text{diff} \left( X_i(x^{k'}, \ldots, x_{i-1}^{k'}, x_{i+1}^{k'}, \ldots, x_s^{k'}), X_i(x, \ldots, x_{i-1}, x_{i+1}, \ldots, x_s) \right) = 0 \quad \forall i.
\]
Then if the assumptions of Lemma 2.2 hold, any limit point of \( \{x^k\} \) is a Nash point, namely, satisfying the Nash equilibrium condition (2.4).

**Proof.** Let \( \bar{x} \) be a limit point of \( \{x^k\} \), and let \( \{x^{k_j}\} \) be the subsequence converging to \( \bar{x} \). The closedness of \( X \) implies \( \bar{x} \in X \). Since \( \{L_{i}^{k}\} \) is bounded, passing another sequence if necessary, we have \( L_{i}^{k_j} \to \bar{L}_{i} \) for \( i = 1, \ldots, s \) as \( j \to \infty \). Lemma 2.2 implies that \( \|x^{k+1} - x^k\| \to 0 \), so \( \{x^{k_j+1}\} \) also converges to \( \bar{x} \).

For \( i \in I_1 \), we have
\[
F_{i}^{k_{j}+1}(x_{i}^{k_{j}+1}) \leq F_{i}^{k_{j}+1}(x_{i}) \quad \forall x_{i} \in X_{i}^{k_{j}+1}.
\]
Letting \( j \to \infty \), we can show (2.3) by the continuity of \( F \), and the set map \( X_i(\cdot) \) by the following arguments. For any block \( i_0 \in I_1 \) and any \( y_{i_0} \in X_{i_0} \), since
\[
\lim_{j} \text{diff} \left( X_{i_0}^{k_{j}+1}, x_{i_0} \right) = 0,
\]
there is \( y_{i_0}^{j} \in X_{i_0}^{k_{j}+1} \) such that \( \lim_{j} y_{i_0}^{j} = y_{i_0} \). From the continuity of \( F \), we have
\[
\lim_{j \to \infty} F(x_{1}^{k_{j}+1}, \ldots, x_{i_0-1}^{k_{j}+1}, y_{i_0}^{j}, x_{i_0+1}^{k_{j}}, \ldots, x_{s}^{k_{j}}) = F(x_{1}, \ldots, x_{i_0-1}, y_{i_0}, x_{i_0+1}, \ldots, x_{s}).
\]
Note that (2.9) implies
\[
F(x_{1}^{k_{j}+1}, \ldots, x_{i_0-1}^{k_{j}+1}, x_{i_0}^{k_{j}+1}, \ldots, x_{s}^{k_{j}}) \leq F(x_{1}^{k_{j}+1}, \ldots, x_{i_0-1}^{k_{j}+1}, y_{i_0}^{j}, x_{i_0+1}^{k_{j}}, \ldots, x_{s}^{k_{j}}).
\]
Letting \( j \to \infty \) and using (2.10), we get
\[
F(x_{1}, \ldots, x_{i_0-1}, x_{i_0}, x_{i_0+1}, \ldots, x_{s}) \leq F(x_{1}, \ldots, x_{i_0-1}, y_{i_0}, x_{i_0+1}, \ldots, x_{s}).
\]
Hence, (2.3) holds. Similarly, for $i \in I_2$, we have
\[
F(\tilde{x}_1, \ldots, \tilde{x}_{i-1}, \tilde{x}_i, \tilde{x}_{i+1}, \ldots, \tilde{x}_s) \leq F(\tilde{x}_1, \ldots, \tilde{x}_{i-1}, \tilde{x}_i, \tilde{x}_{i+1}, \ldots, \tilde{x}_s) + \frac{\tilde{L}_i}{2} \|\tilde{x}_i - \bar{\tilde{x}}_i\|^2 \quad \forall \tilde{x}_i \in \mathcal{X}_i;
\]
namely,
\[
\bar{\tilde{x}}_i = \arg\min_{\tilde{x}_i \in \mathcal{X}_i} F(\tilde{x}_1, \ldots, \tilde{x}_{i-1}, \tilde{x}_i, \tilde{x}_{i+1}, \ldots, \tilde{x}_s) + \frac{\tilde{L}_i}{2} \|\tilde{x}_i - \bar{\tilde{x}}_i\|^2.
\]
Thus, $\bar{\tilde{x}}_i$ satisfies the first-order optimality condition of (2.11), which is precisely (2.4). For $i \in I_3$, we have
\[
x_i^{k+3} = \arg\min_{x_i \in \mathcal{X}_i} \langle \tilde{f}_i^{k+3}(\tilde{x}_i), x_i - \bar{\tilde{x}}_i \rangle + \frac{\tilde{L}_i}{2} \|x_i - \bar{\tilde{x}}_i\|^2 + r_i(x_i).
\]
The convex proximal minimization is continuous in the sense that the output $x_i^{k+3}$ depends continuously on the input $\bar{\tilde{x}}_i^{k+3}$ [59]. Letting $j \to \infty$, from $x_i^{k+3} \to \bar{x}_i$ and $\bar{\tilde{x}}_i^{k+3} \to \bar{x}_i$, we get
\[
x_i = \arg\min_{x_i \in \mathcal{X}_i} \langle \tilde{f}_i(x), x_i - \bar{x}_i \rangle + \frac{\tilde{L}_i}{2} \|x_i - \bar{x}_i\|^2 + r_i(x_i).
\]
Hence, $x_i$ satisfies the first-order optimality condition of (2.12), which is precisely (2.4). This completes the proof.

**Remark 2.3.** If $\mathcal{X}$ is convex, then the set map $\mathcal{X}(-)$ is continuous; see Theorem 4.32 in [60]. A special case is $\mathcal{X} = \mathbb{R}^n$; namely, there are no joint constraints.

**Corollary 2.4 (global convergence given isolated Nash points).** Under the assumptions of Theorem 2.3, if $\{x^k\}$ is bounded, we have $\text{dist}(x^k, \mathcal{N}) \to 0$. If, further, $\mathcal{N}$ contains uniquely isolated points, namely, there is $\eta > 0$ such that $\|x - y\| \geq \eta$ for any distinct points $x, y \in \mathcal{N}$, then $\{x^k\}$ converges to a point in $\mathcal{N}$.

**Proof.** Suppose $\text{dist}(x^k, \mathcal{N})$ does not converge to 0. Then there exist $\varepsilon > 0$ and a subsequence $\{x^{k_j}\}$ such that $\text{dist}(x^{k_j}, \mathcal{N}) \geq \varepsilon$ for all $j$. However, the boundedness of $\{x^{k_j}\}$ implies that it must have a limit point $\bar{x} \in \mathcal{N}$ according to Theorem 2.3, which is a contradiction.

From $\text{dist}(x^0, \mathcal{N}) \to 0$, it follows that there is an integer $K_1 > 0$ such that $x^k \in \cup_{y \in \mathcal{N}} B(y, \frac{\varepsilon}{2})$ for all $k \geq K_1$, where $B(y, \frac{\varepsilon}{2}) = \{x \in \mathcal{X} : \|x - y\| < \frac{\varepsilon}{2}\}$. In addition, Lemma 2.2 implies that there exists another integer $K_2 > 0$ such that $\|x^k - x^{k+1}\| < \frac{\varepsilon}{2}$ for all $k \geq K_2$. Take $K = \max(K_1, K_2)$ and assume $x^K \in B(\bar{x}, \frac{\varepsilon}{2})$ for some $\bar{x} \in \mathcal{N}$. We claim that for any $y \in \mathcal{N}$ and $y \neq \bar{x}$, $\|x^k - y\| > \frac{\varepsilon}{2}$ holds for all $k \geq K$. This claim can be shown by induction on $k \geq K$. If some $x^k \in B(\bar{x}, \frac{\varepsilon}{2})$, then $\|x^{k+1} - \bar{x}\| \leq \|x^k - x^{k+1}\| + \|x^k - \bar{x}\| < \frac{\varepsilon}{2}$, and
\[
\|x^{k+1} - y\| \geq \|x - y\| - \|x^{k+1} - x\| > \frac{\eta}{3} \quad \text{for any } x \neq y \in \mathcal{N}.
\]
Therefore, $x^K \in B(\bar{x}, \frac{\varepsilon}{2})$ for all $k \geq K$ since $x^k \in \cup_{y \in \mathcal{N}} B(y, \frac{\varepsilon}{2})$, and thus $\{x^k\}$ has the unique limit point $\bar{x}$, which means that $x^K \to \bar{x}$.

**Remark 2.4.** The boundedness of $\{x^k\}$ is guaranteed if the level set $\{x \in \mathcal{X} : F(x) \leq F(x^0)\}$ is bounded. On the other hand, the isolation assumption is difficult to verify, or even fails to hold, for many functions. This motivates another approach below for global convergence.
2.2. Kurdyka–Łojasiewicz inequality. Before proceeding with our analysis, let us briefly review the KL inequality, which is central to the global convergence analysis in the next subsection.

Definition 2.5. A function \( \psi(x) \) satisfies the Kurdyka–Łojasiewicz (KL) property at point \( \bar{x} \in \text{dom}(\partial \psi) \) if there exists \( \theta \in [0,1) \) such that

\[
(2.13) \quad \frac{|\psi(x) - \psi(\bar{x})|^\theta}{\text{dist}(0, \partial \psi(x))}
\]

is bounded around \( \bar{x} \) under the following notational conventions: \( 0^0 = 1, \infty/\infty = 0/0 = 0 \). In other words, in a certain neighborhood \( \mathcal{U} \) of \( \bar{x} \), there exists \( \phi(s) = cs^{1-\theta} \) for some \( c > 0 \) and \( \theta \in [0,1) \) such that the KL inequality holds:

\[
(2.14) \quad \phi'(|\psi(x) - \psi(\bar{x})|)\text{dist}(0, \partial \psi(x)) \geq 1 \quad \text{for any } x \in \mathcal{U} \cap \text{dom}(\partial \psi) \text{ and } \psi(x) \neq \psi(\bar{x}),
\]

where \( \text{dom}(\partial \psi) \triangleq \{ x : \partial \psi(x) \neq \emptyset \} \) and \( \text{dist}(0, \partial \psi(x)) \triangleq \min\{ ||y|| : y \in \partial \psi(x) \} \).

This property was introduced by Łojasiewicz [46] on real analytic functions, for which (2.13) is bounded around any critical point \( \bar{x} \) for \( \theta \in \left[ \frac{1}{2}, 1 \right) \). Kurdyka extended this property to functions on the \( \sigma \)-minimal structure in [36]. Recently, the KL inequality was extended to nonsmooth subanalytic functions [13].

The KL inequality (2.14) is usually weaker than the condition of isolated Nash points used in Corollary 2.4. In (2.14), we require \( \psi(x) \neq \psi(\bar{x}) \), so a point obeying the KL inequality need not be an isolated Nash point. The function \( \psi(x,y) = (xy - 1)^2 \) is an example, where \( (x,y) = (1,1) \) is a minimizer meeting the KL inequality but is not an isolated Nash point. While it is not trivial to check the conditions in the definition, we summarize a few large classes of functions that satisfy the KL inequality.

Real analytic functions. A smooth function \( \varphi(t) \) on \( \mathbb{R} \) is analytic if \( (\frac{\varphi^{(k)}(t)}{k!})^n \) is bounded for all \( k \) and on any compact set \( \mathcal{D} \subset \mathbb{R} \). One can verify whether a real function \( \psi(x) \) on \( \mathbb{R}^n \) is analytic by checking the analyticity of \( \varphi(t) \triangleq \psi(x + ty) \) for any \( x, y \in \mathbb{R}^n \). For example, any polynomial function is real analytic, such as \( \|Ax - b\|^2 \) and the first terms in the objectives of (1.9) and (1.10). In addition, it is not difficult to verify that the nonconvex function \( L_q(x, \varepsilon, \lambda) = \|x^1 - \varepsilon^1\|^{q/2} + \frac{1}{\|x\|}\|Ax - b\|^2 \) with \( 0 < q < 1 \) considered in [37] for sparse vector recovery is a real analytic function (the first term is the \( \varepsilon \)-smoothed \( l_q \)-seminorm). The logistic loss function \( \psi(t) = \log(1 + e^{-t}) \) is also analytic. Therefore, all the above functions satisfy the KL inequality with \( \theta \in [\frac{1}{2}, 1) \) in (2.13).

Locally strongly convex functions. A function \( \psi(x) \) is strongly convex in a neighborhood \( \mathcal{D} \) with constant \( \mu \) if

\[
\psi(y) \geq \psi(x) + \langle \gamma(x), y - x \rangle + \frac{\mu}{2}||x - y||^2 \quad \forall \gamma(x) \in \partial \psi(x) \text{ and for any } x, y \in \mathcal{D}.
\]

According to the definition and using the Cauchy–Schwarz inequality, we have

\[
\psi(y) - \psi(x) \geq \langle \gamma(x), y - x \rangle + \frac{\mu}{2}||x - y||^2 \geq -\frac{1}{\mu}||\gamma(x)||^2 \quad \forall \gamma(x) \in \partial \psi(x),
\]
where the last inequality is obtained by minimizing the middle term over \( y \). Hence, \( \mu(\psi(x) - \psi(y)) \leq (\text{dist}(0, \partial \psi(x)))^2 \), and \( \psi \) satisfies the KL inequality (2.14) at any point \( y \in D \) with \( \phi(s) = \frac{s}{\mu} \sqrt{s} \) and \( \mathcal{U} = D \cap \{ x : \psi(x) \geq \psi(y) \} \). For example, the logistic loss function \( \psi(t) = \log(1 + e^{-t}) \) is strongly convex in any bounded set \( D \).

**Semialgebraic functions.** A set \( D \subseteq \mathbb{R}^n \) is called semialgebraic [11] if it can be represented as

\[
D = \bigcup_{i=1}^s \bigcap_{j=1}^t \{ x \in \mathbb{R}^n : p_{ij}(x) = 0, q_{ij}(x) > 0 \},
\]

where \( p_{ij}, q_{ij} \) are real polynomial functions for \( 1 \leq i \leq s, 1 \leq j \leq t \). A function \( \psi \) is called semialgebraic if its graph \( \text{Gr}(\psi) \triangleq \{(x, \psi(x)) : x \in \text{dom}(\psi)\} \) is a semialgebraic set.

Semialgebraic functions are subanalytic, so they satisfy the KL inequality according to [13, 14]. We list some known elementary properties of semialgebraic sets and functions below, as they help identify semialgebraic functions.

1. If a set \( D \) is semialgebraic, so is its closure \( \text{cl}(D) \).
2. If \( D_1 \) and \( D_2 \) are both semialgebraic, so are \( D_1 \cap D_2, D_1 \cup D_2, \) and \( \mathbb{R}^n \setminus D_1 \).
3. Indicator functions of semialgebraic sets are semialgebraic.
4. Finite sums and products of semialgebraic functions are semialgebraic.
5. The composition of semialgebraic functions is semialgebraic.

From items 1 and 2, any polyhedral set is semialgebraic, such as the nonnegative orthant \( \mathbb{R}^n_+ = \{ x \in \mathbb{R}^n : x_i \geq 0 \ \forall i \} \). Hence, the indicator function \( \delta_{\mathbb{R}^n_+} \) is a semialgebraic function. The absolute value function \( \varphi(t) = |t| \) is also semialgebraic since its graph is \( \text{cl}(D) \), where

\[
D = \{(t, s) : t + s = 0, -t > 0 \} \cup \{(t, s) : t - s = 0, t > 0 \}.
\]

Hence, the \( \ell_1 \)-norm \( \|x\|_1 \) is semialgebraic since it is the finite sum of absolute functions. In addition, the sup-norm \( \|x\|_\infty \) is semialgebraic, which can be shown by observing

\[
\text{Graph}(\|x\|_\infty) = \{(x, t) : t = \max_j |x_j| = \bigcup_i \{(x, t) : |x_i| = t, |x_j| \leq t \ \forall j \neq i \} \}.
\]

Further, the Euclidean norm \( \|x\| \) is shown to be semialgebraic in [11]. According to item 5, \( \|Ax - b\|_1, \|Ax - b\|_\infty, \) and \( \|Ax - b\| \) are all semialgebraic functions.

**Sum of real analytic and semialgebraic functions.** Both real analytic and semialgebraic functions are subanalytic. According to [11], if \( \psi_1 \) and \( \psi_2 \) are both subanalytic and \( \psi_1 \) maps bounded sets to bounded sets, then \( \psi_1 + \psi_2 \) is also subanalytic. Since real analytic functions map bounded sets to bounded sets, the sum of a real analytic function and a semialgebraic function is subanalytic, so the sum satisfies the KL property. For example, the sparse logistic regression function

\[
\psi(x, b) = \frac{1}{n} \sum_{i=1}^n \log \left( 1 + \exp \left( -c_i (a_i^\top x + b) \right) \right) + \lambda \|x\|_1
\]

is subanalytic and satisfies the KL inequality.
2.3. Global convergence and rate. If \( \{x^k\} \) is bounded, then Theorem 2.3 guarantees that there exists one subsequence converging to a Nash point of (1.1). In this subsection, we assume \( \mathcal{X} = \mathbb{R}^n \) and strengthen this result for problems with \( F \) obeying the KL inequality. Our analysis here was motivated by [4], which applies the inequality to establish the global convergence of the alternating proximal point method—the special case of BCD with two blocks and using only update (1.3b).

We make the following modification to Algorithm 1.

(M1) Whenever \( F(x^k) \geq F(x^{k-1}) \), we redo the \( k \)th iteration with \( \hat{x}^{k-1} = x^{k-1} \) (i.e., no extrapolation) for all \( i \in I_3 \).

Remark 2.5. From the proof of Lemma 2.2, we can see that this modification makes \( F(x^k) \) strictly less than \( F(x^{k-1}) \) as long as \( x^k \neq x^{k-1} \). To show this, observe that the proof of Lemma 2.2 implies that updates (1.3a) and (1.3b) both make the objective decrease by at least \( \frac{L^k_{i-2}}{2} \|x^k - x^{k-1}\|^2 \), and update (1.3c) also makes the objective decrease by at least \( \frac{L^k_{i-2}}{2} \|x^1 - x^0\|^2 \) when \( x^i = x^{k-1} \), namely, when the modification step (M1) occurs. Hence, if \( x^{k_0} = x^{k_0-1} \) for some \( k_0 \), then \( F(x^k) = F(x^{k_0}) \) and \( x^k = x^{k_0} \) for all \( k \geq k_0 \).

In what follows, we use the notations \( F_k = F(x^k) \) and \( \hat{F} = F(\hat{x}) \). First, we establish the following preconvergence result, the proof of which is given in Appendix A.

Lemma 2.6. Under Assumptions 1 and 2, let \( \{x^k\} \) be the sequence of Algorithm 1 with (M1) and its parameters satisfying \( \omega^k_i \leq \min(1, \delta, \sqrt{I^i_{k-1}/L^k_i}), \delta, \omega < 1 \), for all \( i \in I_3 \) and \( k \).

Assume the following:

1. \( \nabla f \) is Lipschitz continuous on any bounded set;
2. \( F \) satisfies the KL inequality (2.14) at \( \bar{x} \);
3. \( x_0 \) is sufficiently close to \( \bar{x} \), and \( F_k > F \) for \( k \geq 0 \).

Then there is some \( B \subset U \cap \text{dom}(\partial \psi) \) with \( \psi = F \) in (2.14) such that \( \{x^k\} \subset B \) and \( x^k \) converges to a point in \( B \).

Remark 2.6. In the lemma, the required closeness of \( x_0 \) to \( \bar{x} \) depends on \( U, \phi \), and \( \psi = F \) in (2.14) (see the inequality in (A.1)).

The following corollary is a straightforward application of Lemma 2.6.

Corollary 2.7. Under the assumptions of Lemma 2.6, \( \{x^k\} \) converges to a global minimizer of (1.1) if the initial point \( x^0 \) is sufficiently close to any global minimizer \( \bar{x} \).

Proof. Suppose \( F(x^{k_0}) = F(\bar{x}) \) at some \( k_0 \). Then \( x^k = x^{k_0} \) for all \( k \geq k_0 \), according to Remark 2.5. Now consider \( F(x^k) > F(\bar{x}) \) for all \( k \geq 0 \), and thus Lemma 2.6 implies that \( x^k \) converges to some critical point \( x^* \) if \( x^0 \) is sufficiently close to \( \bar{x} \), where \( x^0, x^*, \bar{x} \in \mathcal{B} \). If \( F(x^*) > F(\bar{x}) \), then the KL inequality (2.14) indicates \( \phi'(F(x^* - F(\bar{x})) \geq 1 \), which is impossible since \( 0 \in \partial F(x^*) \).

Next, we give the convergence result of Algorithm 1.

Theorem 2.8 (global convergence). Under the assumptions of Lemma 2.6 and the fact that \( \{x^k\} \) has a finite limit point \( \bar{x} \) where \( F \) satisfies the KL inequality (2.14), the sequence \( \{x^k\} \) converges to \( \bar{x} \), which is a critical point of (1.1).

Proof. Note that \( F(x^k) \) is monotonically nonincreasing and converges to \( F(\bar{x}) \). If \( F(x^{k_0}) = F(\bar{x}) \) at some \( k_0 \), then \( x^k = x^{k_0} = \bar{x} \) for all \( k \geq k_0 \), according to Remark 2.5. It remains to consider \( F(x^k) > F(\bar{x}) \) for all \( k \geq 0 \). Since \( \bar{x} \) is a limit point and \( F(x^k) \to F(\bar{x}) \), there must exist an integer \( k_0 \) such that \( x^{k_0} \) is sufficiently close to \( \bar{x} \) as required in Lemma 2.6 (see
the inequality in (A.1). Hence, the entire sequence \( \{x^k\} \) converges according to Lemma 2.6. Since \( \bar{x} \) is a limit point of \( \{x^k\} \), we have \( x^k \to \bar{x} \).

We can also estimate the rate of convergence, and the proof is given in Appendix A.

**Theorem 2.9 (convergence rate).** Assume the assumptions of Lemma 2.6, and suppose that \( x^k \) converges to a critical point \( \bar{x} \), at which \( F \) satisfies the KL inequality with \( \phi(s) = cs^{1-\theta} \) for \( c > 0 \) and \( \theta \in [0, 1) \). Then the following hold:

1. If \( \theta = 0 \), \( x^k \) converges in finitely many iterations.
2. If \( \theta \in (0, \frac{1}{2}] \), \( \|x^k - \bar{x}\| \leq C\tau^k \) for all \( k \geq k_0 \), for certain \( k_0 > 0 \), \( C > 0 \), \( \tau \in [0, 1) \).
3. If \( \theta \in (\frac{1}{2}, 1) \), \( \|x^k - \bar{x}\| \leq Ck^{-(1-\theta)/(2\theta-1)} \) for all \( k \geq k_0 \), for certain \( k_0 > 0 \), \( C > 0 \).

Parts 1, 2, and 3 correspond to finite convergence, linear convergence, and sublinear convergence, respectively.

3. Factorization and completion of nonnegative matrices and tensors. In this section, we apply Algorithm 1 with modification (M1) to the factorization and completion of nonnegative matrices and tensors. Since a matrix is a two-way tensor, we present the algorithm for tensors. We first give an overview of a tensor and its two popular factorizations.

3.1. Overview of tensor. A tensor is a multidimensional array. For example, a vector is a first-order tensor, and a matrix is a second-order tensor. The order of a tensor is the number of dimensions, also called way or mode. For an \( N \)-way tensor \( \mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \), we let its \((i_1, i_2, \ldots, i_N)\)th element be denoted by \( x_{i_1i_2\cdots i_N} \). Below we list some concepts related to a tensor. For more details about a tensor, the reader is referred to the review paper [35].

1. Fiber: A fiber of a tensor \( \mathbf{X} \) is a vector obtained by fixing all indices of \( \mathbf{X} \) except one. For example, a row of a matrix is a mode-2 fiber (the first index is fixed), and a column is a mode-1 fiber (the second index is fixed). We use \( x_{i_1\cdots i_{n-1}j_{n+1}\cdots i_N} \) to denote a mode-\( n \) fiber of an \( N \)-th order tensor \( \mathbf{X} \).

2. Slice: A slice of a tensor \( \mathbf{X} \) is a matrix obtained by fixing all indices of \( \mathbf{X} \) except two. Take a third-order tensor \( \mathbf{X} \), for example. \( X_{ij:} \), \( X_{:j:} \), and \( X_{:k:} \) denote horizontal, lateral, and frontal slices, respectively, of \( \mathbf{X} \).

3. Matricization: The mode-\( n \) matricization of a tensor \( \mathbf{X} \) is a matrix whose columns are the mode-\( n \) fibers of \( \mathbf{X} \) in the lexicographical order. We let \( X_{(n)} \) denote the mode-\( n \) matricization of \( \mathbf{X} \).

4. Tensor-matrix product: The mode-\( n \) product of a tensor \( \mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N} \) with a matrix \( \mathbf{A} \in \mathbb{R}^{J_1 \times I_1} \) is a tensor of size \( I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N \) defined as

\[
(\mathbf{X} \times_n \mathbf{A})_{i_1\cdots i_{n-1}j_{n+1}\cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1i_2\cdots i_{n-1}j_{n+1}\cdots i_N} a_{j_{n+1}i_n}.
\]

In addition, we briefly review the matrix Kronecker, Khatri–Rao, and Hadamard products below, which we use to derive tensor-related computations.

The Kronecker product of matrices \( \mathbf{A} \in \mathbb{R}^{m \times n} \) and \( \mathbf{B} \in \mathbb{R}^{p \times q} \) is an \( mp \times nq \) matrix defined by \( \mathbf{A} \otimes \mathbf{B} = [a_{ij} \mathbf{B}]_{mp \times nq} \). The Khatri–Rao product of matrices \( \mathbf{A} \in \mathbb{R}^{m \times n} \) and \( \mathbf{B} \in \mathbb{R}^{p \times q} \) is an \( mp \times q \) matrix \( \mathbf{A} \odot \mathbf{B} = [a_1 \otimes b_1, a_2 \otimes b_2, \ldots, a_q \otimes b_q] \), where \( a_i, b_j \) are the \( i \)th columns of \( \mathbf{A} \) and \( \mathbf{B} \), respectively. The Hadamard product of matrices \( \mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n} \) is the componentwise product defined by \( \mathbf{A} \circ \mathbf{B} = [a_{ij}b_{ij}]_{m \times n} \).
Two important tensor decompositions are the CANDECOMP/PARAFAC (CP) [29] and Tucker [68] decompositions. The former decomposes a tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ in the form of $\mathbf{X} = \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N$, where $\mathbf{A}_n \in \mathbb{R}^{I_n \times r}$, $n = 1, \ldots, N$, are factor matrices, $r$ is the tensor rank of $\mathbf{X}$, and the outer product $\circ$ is defined as

$$
\mathbf{x}_{i_1 i_2 \cdots i_N} = \sum_{j=1}^{r} a_{i_1 j}^{(1)} a_{i_2 j}^{(2)} \cdots a_{i_N j}^{(N)} \quad \text{for } i_n \in [I_n], \quad n = 1, \ldots, N,
$$

where $a_{i j}^{(n)}$ is the $(i, j)$th element of $\mathbf{A}_n$ and $[I] \triangleq \{1, 2, \ldots, I\}$. The latter Tucker decomposition decomposes a tensor $\mathbf{X}$ in the form of $\mathbf{X} = \mathbf{G} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 \cdots \times_N \mathbf{A}_N$, where $\mathbf{G} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is called the core tensor and $\mathbf{A}_n \in \mathbb{R}^{I_n \times J_n}$, $n = 1, \ldots, N$, are factor matrices.

### 3.2. An algorithm for nonnegative tensor factorization

One can obtain a nonnegative CP decomposition of a nonnegative tensor $\mathbf{M} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ by solving

$$
\min_{\mathbf{A}} \frac{1}{2} \| \mathbf{M} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N \|_F^2 \quad \text{subject to } \mathbf{A}_n \in \mathbb{R}^{I_n \times r}, \quad n = 1, \ldots, N,
$$

where $r$ is a specified order and the Frobenius norm of a tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is defined as $\| \mathbf{X} \|_F = \sqrt{\sum_{i_1, i_2, \ldots, i_N} x_{i_1 i_2 \cdots i_N}^2}$. Similar models based on the CP decomposition can be found in [31, 19, 33]. One can obtain a nonnegative Tucker decomposition of $\mathbf{M}$ by solving

$$
\min_{\mathbf{G}, \mathbf{A}_n} \frac{1}{2} \| \mathbf{M} - \mathbf{G} \times_1 \mathbf{A}_1 \times_2 \mathbf{A}_2 \cdots \times_N \mathbf{A}_N \|_F^2 \quad \text{subject to } \mathbf{G} \in \mathbb{R}^{I_1 \times \cdots \times J_N}, \quad \mathbf{A}_n \in \mathbb{R}^{I_n \times J_n} \quad \forall n,
$$

as in [34, 51, 44]. Usually, it is computationally expensive to update $\mathbf{G}$. Since applying Algorithm 1 to problem (3.3) involves many computing details, we focus on applying Algorithm 1 with update (1.3c) to problem (3.2).

Let $\mathbf{A} = (\mathbf{A}_1, \ldots, \mathbf{A}_N)$, and let

$$
F(\mathbf{A}) = F(\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N) = \frac{1}{2} \| \mathbf{M} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N \|_F^2
$$

be the objective of (3.2). Consider updating $\mathbf{A}_n$ at iteration $k$. Using the fact that if $\mathbf{X} = \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N$, then $\mathbf{X}_{(n)} = \mathbf{A}_n (\mathbf{A}_N \circ \cdots \circ \mathbf{A}_{n+1} \circ \mathbf{A}_{n-1} \circ \cdots \circ \mathbf{A}_1)^\top$, we have

$$
F(\mathbf{A}) = \frac{1}{2} \| \mathbf{M}_{(n)} - \mathbf{A}_n (\mathbf{A}_N \circ \cdots \circ \mathbf{A}_{n+1} \circ \mathbf{A}_{n-1} \circ \cdots \circ \mathbf{A}_1)^\top \|_F^2
$$

and

$$
\nabla_{\mathbf{A}_n} F = \left( \mathbf{A}_n (\mathbf{A}_N \circ \cdots \circ \mathbf{A}_{n+1} \circ \mathbf{A}_{n-1} \circ \cdots \circ \mathbf{A}_1)^\top - \mathbf{M}_{(n)} \right)
$$

\[ = (\mathbf{A}_N \circ \cdots \circ \mathbf{A}_{n+1} \circ \mathbf{A}_{n-1} \circ \cdots \circ \mathbf{A}_1). \]

Let

$$
\mathbf{B}_{n}^{k-1} = \mathbf{A}_N^{k-1} \circ \cdots \circ \mathbf{A}_{n+1}^{k-1} \circ \mathbf{A}_{n-1}^{k-1} \circ \cdots \circ \mathbf{A}_1^{k}.
$$
We take

\[ L_n^{k-1} = \|(B_n^{k-1})^T B_n^{k-1}\|, \quad \omega_n^{k-1} = \min \left( \hat{\omega}_{k-1}, \delta_{\omega} \sqrt{\frac{L_n^{k-1}}{L_n^{k-1}}} \right), \]

where \(|A|\) is the spectral norm of \(A\), \(\delta_{\omega} < 1\) is preselected, and \(\hat{\omega}_{k-1} = \frac{t_{k-1} - 1}{t_k}\) with

\[ t_0 = 1, \quad t_k = \frac{1}{2} \left( 1 + \sqrt{1 + 4t_{k-1}^2} \right). \]

In addition, let \(\hat{A}_n^{k-1} = A_n^{k-1} + \omega_n^{k-1}(A_n^{k-1} - A_n^{k-2})\), and let

\[ \hat{G}_n^{k-1} = \left( \hat{A}_n^{k-1}(B_n^{k-1})^T - M_n \right) B_n^{k-1} \]

be the gradient. Then we derive update (1.3c):

\[ A_n^k = \arg\min_{A_n \geq 0} \left( \hat{G}_n^{k-1}, A_n - \hat{A}_n^{k-1} \right) + \frac{L_n^{k-1}}{2} \|A_n - \hat{A}_n^{k-1}\|_F^2, \]

which can be written in the closed form

\[ A_n^k = \max \left( 0, \hat{A}_n^{k-1} - \hat{G}_n^{k-1}/L_n^{k-1} \right). \]

At the end of iteration \(k\), we check whether \(F(A^k) \geq F(A^{k-1})\). If so, we reupdate \(A_n^k\) by (3.6) with \(\hat{A}_n^{k-1} = A_n^{k-1}\) for \(n = 1, \ldots, N\).

**Remark 3.1.** In (3.6), \(\hat{G}_n^{k-1}\) is most expensive to compute. To efficiently compute it, we write \(\hat{G}_n^{k-1} = \hat{A}_n^{k-1}(B_n^{k-1})^T B_n^{k-1} - M_n B_n^{k-1}\). Using \((A \odot B)^T (A \odot B) = (A^T A) * (B^T B)\), we compute \((B_n^{k-1})^T B_n^{k-1}\) by

\[ (B_n^{k-1})^T B_n^{k-1} = \left( (A_1^k)^T A_1^k \right) * \cdots * \left( (A_{n-1}^k)^T A_{n-1}^k \right) * \left( (A_{n+1}^k)^T A_{n+1}^k \right) * \cdots * \left( (A_N^k)^T A_N^k \right). \]

Then, \(M_n B_n^{k-1}\) can be obtained by the *matricized-tensor-times-Khatri–Rao-product* [6].

Algorithm 2 summarizes how to apply Algorithm 1 with update (1.3c) to problem (3.2).

**Remark 3.2.** When \(N = 2\), \(M\) becomes a matrix, and Algorithm 2 solves NMF.

### 3.3. Convergence results

Since problem (3.2) is a special case of problem (1.1), the convergence results in section 2 apply to Algorithm 2. Let \(D_n = \mathbb{R}_{+}^{r_1 \times r_n}\), and let \(\delta_{D_n}(\cdot)\) be the indicator function on \(D_n\) for \(n = 1, \ldots, N\). Then (3.2) is equivalent to

\[ \min_{A_1, \ldots, A_N} Q(A) = F(A) + \sum_{n=1}^N \delta_{D_n}(A_n). \]

According to the discussion in section 2.2, \(Q\) is a semialgebraic function and satisfies the KL property (2.13) at any feasible point. Furthermore, we get \(\theta \neq 0\) in (2.13) for \(Q\) at any critical point. By writing the first-order optimality conditions of (3.7), one can find that if
Algorithm 2. Alternating proximal gradient (APG) method for solving (3.2).

1: Input: Nonnegative N-way tensor $\mathbf{M}$ and rank $r$.
2: Output: Nonnegative factors $\mathbf{A}_1, \ldots, \mathbf{A}_N$.
3: Initialization: Choose a positive number $\delta_0 < 1$ and randomize $\mathbf{A}^{-1}_n = \mathbf{A}_{0n}^0$, $n = 1, \ldots, N$, as nonnegative matrices of appropriate sizes.
4: for $k = 1, 2, \ldots$ do
5:     for $n = 1, 2, \ldots, N$ do
6:         Compute $L_n^{k-1}$, and set $\omega_n^{k-1}$ according to (3.5).
7:         Let $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1} + \omega_n^{k-1}(\mathbf{A}_n^{k-1} - \mathbf{A}_n^{k-2})$.
8:         Update $\mathbf{A}_n^k$ according to (3.6).
9:     end for
10:    if $F(\mathbf{A}^k) \geq F(\mathbf{A}^{k-1})$ then
11:        Reupdate $\mathbf{A}_n^k$ according to (3.6) with $\hat{\mathbf{A}}_n^{k-1} = \mathbf{A}_n^{k-1}$, $n = 1, \ldots, N$.
12:     end if
13:     if stopping criterion is satisfied then
14:            Return $\mathbf{A}_1^k, \ldots, \mathbf{A}_N^k$.
15:    end if
16: end for

($\mathbf{A}_1, \ldots, \mathbf{A}_N$) is a critical point, then so is $(t\hat{\mathbf{A}}_1, 1 \frac{1}{t} \hat{\mathbf{A}}_2, \hat{\mathbf{A}}_3, \ldots, \mathbf{A}_N)$ for any $t > 0$. Therefore, from Theorems 2.8 and 2.9 and the above discussions, we have the following theorem.

Theorem 3.1. Let $\{\mathbf{A}^k\}$ be the sequence generated by Algorithm 2. Assume that $\{\mathbf{A}^k\}$ is bounded and there is a positive constant $\ell$ such that $\ell \leq L_n^k$ for all $k$ and $n$. Then $\{\mathbf{A}^k\}$ converges to a critical point $\hat{\mathbf{A}}$, and the asymptotic convergence rates in parts 2 and 3 of Theorem 2.9 apply.

Remark 3.3. The boundedness of $\{\mathbf{A}^k\}$ guarantees that $L_n^k$ is upper bounded. A simple way to make $\{\mathbf{A}^k\}$ bounded is to scale $(\mathbf{A}_1, \ldots, \mathbf{A}_N)$ so that $\|\mathbf{A}_1\|_F = \cdots = \|\mathbf{A}_N\|_F$ after each iteration. The existence of a positive $\ell$ can be satisfied if one changes $L_n^k$ to $\max(L_n^k, L_{\min})$ for a positive constant $L_{\min}$.

3.4. An algorithm for nonnegative tensor completion. Algorithm 2 can be easily modified for solving the nonnegative tensor completion problem

\begin{equation}
\min_{\mathbf{A}_1, \ldots, \mathbf{A}_N \geq 0} \frac{1}{2} \|\mathcal{P}_\Omega(\mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N)\|_F^2,
\end{equation}

where $\Omega \subset [I_1] \times [I_2] \times \cdots \times [I_N]$ is the index set of the observed entries of $\mathbf{M}$, and $\mathcal{P}_\Omega(\mathbf{X})$ keeps the entries of $\mathbf{X}$ in $\Omega$ and sets the remaining ones to zero. Nonnegative matrix completion (corresponding to $N = 2$) has been proposed in [74], where it is demonstrated that a low-rank and nonnegative matrix can be recovered from a small set of its entries by taking advantage of both low-rankness and nonnegative factors. To solve (3.8), we transform it into the equivalent problem

\begin{equation}
\mathbf{X}, \mathbf{A}_{n = 1, \ldots, N} \min \quad G(\mathbf{X}, \mathbf{A}) = \frac{1}{2} \|\mathbf{X} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \cdots \circ \mathbf{A}_N\|_F^2,
\end{equation}

subject to $\mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{M})$.

Our algorithm shall cycle through the decision variables $\mathbf{A}_1, \ldots, \mathbf{A}_N$ and $\mathbf{X}$. To save space, we describe a modification to Algorithm 2. At the $k$th iteration of Algorithm 2, we use
\( \mathcal{M} = \mathcal{X}^{k-1} \) wherever \( \mathcal{M} \) is referred to. Specifically, we use \( \mathcal{M} = \mathcal{X}^{k-1} \) for the computation of \( G_n^{k-1} \) in line 8 and for the evaluation of \( F \) in line 10 of Algorithm 2. After line 12, we perform update (1.3a) on \( \mathcal{X} \) as

\[
\mathcal{X}^k = \mathcal{P}_{\Omega}(\mathcal{M}) + \mathcal{P}_{\Omega^c}(\mathcal{A}_1 \circ \cdots \circ \mathcal{A}_N^k),
\]

where \( \Omega^c \) is the complement of \( \Omega \). Note that for a fixed \( \mathcal{A} \), \( G(\mathcal{A}, \mathcal{X}) \) is a strongly convex function of \( \mathcal{X} \) with modulus 1; namely, the condition in item 1 of Assumption 2 is satisfied. Hence, according to Theorem 2.8, the convergence result for Algorithm 2 still holds for this algorithm with extra update (3.10).

4. Numerical results. In this section, we test Algorithm 2 for nonnegative matrix and three-way tensor factorization, as well as their completion. In our implementations, we choose \( \delta = 0.9999 \). The algorithm is terminated whenever \( \frac{F_k - F_{k+1}}{F_k} \leq \text{tol} \) holds for three iterations in a row or \( \frac{F_k}{\|M\|_F} \leq \text{tol} \) is met, where \( F_k \) is the objective value after iteration \( k \) and \( \text{tol} \) is specified below. We compare

- APG-MF: nonnegative matrix factorization (NMF) by Algorithm 2 in section 3.2;
- APG-TF: nonnegative tensor factorization (NTF) by Algorithm 2 in section 3.2;
- APG-MC: nonnegative matrix completion (NMC) by modified Algorithm 2 in section 3.4;
- APG-TC: nonnegative tensor completion (NTC) by modified Algorithm 2 in section 3.4.

All the tests were performed on a laptop with an i7-620m CPU and 3GB RAM, running 32-bit Windows 7 and MATLAB 2010b with Tensor Toolbox, version 2.5 [7].

4.1. Nonnegative matrix factorization. We choose to compare the most popular and recent algorithms. The first two are the alternating least squares method (ALS-MF) [53, 9] and multiplicative updating method (Mult-MF) [39], which are available as the MATLAB function \( \text{nnmf} \) with specifiers \( \text{als} \) and \( \text{mult} \), respectively. The recent ANLS method Blockpivot-MF is compared since it outperforms all other compared ANLS methods in both speed and solution quality [32]. Another compared algorithm is the recent ADM-based method ADM-MF [78]. Although both Blockpivot-MF and ADM-MF are superior in performance to ALS-MF and Mult-MF, we include the latter in the first two tests due to their popularity.

We set \( \text{tol} = 10^{-4} \) for all of the compared algorithms except ADM-MF, for which we set \( \text{tol} = 10^{-5} \) since it is a dual algorithm and \( 10^{-4} \) is too loose. The maximum number of iterations (\( \text{maxit} \)) is set to 2000 for all algorithms. The same random starting points are used for all the algorithms except Mult-MF. Since Mult-MF is very sensitive to initial points, we set the initial point by running Mult-MF 10 iterations for five independent times and choose the best one. All of the other parameters for ALS-MF, Mult-MF, Blockpivot-MF, and ADM-MF are set to their default values.

4.1.1. Synthetic data. Each matrix in this test is exactly low-rank and can be written in the form of \( \mathcal{M} = \mathcal{L} \mathcal{R} \), where \( \mathcal{L} \) and \( \mathcal{R} \) are generated by MATLAB commands \( \max(0, \text{randn}(m,q)) \) and \( \text{rand}(q,n) \), respectively. It is worth mentioning that generating \( \mathcal{R} \) by \( \text{rand}(q,n) \) makes the problems more difficult than \( \max(0, \text{randn}(q,n)) \) or \( \text{abs}(\text{randn}(q,n)) \). The algorithms are compared with fixed \( n = 1000 \), \( m \) chosen from \{200, 500, 1000\}, and \( q \) cho-
Table 1
Comparison on nonnegative random $m \times n$ matrices for $n = 1000$; bold values are large error or slow time.

<table>
<thead>
<tr>
<th>$m$</th>
<th>$r$</th>
<th>APG-MF† (prop’d)</th>
<th>ADM-MF</th>
<th>Blockpivot-MF</th>
<th>ALS-MF</th>
<th>Mult-MF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>200</td>
<td>10</td>
<td>9.98e-5</td>
<td>0.72</td>
<td>2.24e-3</td>
<td>1.04</td>
<td>5.36e-4</td>
</tr>
<tr>
<td>200</td>
<td>20</td>
<td>9.97e-5</td>
<td>2.09</td>
<td>3.02e-3</td>
<td>2.80</td>
<td>1.02e-3</td>
</tr>
<tr>
<td>200</td>
<td>30</td>
<td>9.97e-5</td>
<td>4.72</td>
<td>4.55e-3</td>
<td>5.70</td>
<td>1.75e-3</td>
</tr>
<tr>
<td>500</td>
<td>10</td>
<td>9.98e-5</td>
<td>1.61</td>
<td>2.26e-3</td>
<td>2.39</td>
<td>5.11e-4</td>
</tr>
<tr>
<td>500</td>
<td>20</td>
<td>9.98e-5</td>
<td>3.66</td>
<td>2.82e-3</td>
<td>4.38</td>
<td>5.36e-4</td>
</tr>
<tr>
<td>500</td>
<td>30</td>
<td>9.98e-5</td>
<td>7.75</td>
<td>3.51e-3</td>
<td>8.34</td>
<td>5.75e-4</td>
</tr>
<tr>
<td>1000</td>
<td>10</td>
<td>9.98e-5</td>
<td>2.86</td>
<td>2.11e-3</td>
<td>3.44</td>
<td>4.99e-4</td>
</tr>
<tr>
<td>1000</td>
<td>20</td>
<td>9.98e-5</td>
<td>7.44</td>
<td>2.82e-3</td>
<td>7.19</td>
<td>5.46e-4</td>
</tr>
<tr>
<td>1000</td>
<td>30</td>
<td>9.98e-5</td>
<td>12.7</td>
<td>3.01e-3</td>
<td>12.8</td>
<td>5.76e-4</td>
</tr>
</tbody>
</table>

† The relerr values of APG-MF are nearly the same due to the use of the same stopping tolerance.

Table 2
Comparison on 2000 selected images from the CBCL face database; bold values are large error or slow time.

<table>
<thead>
<tr>
<th>$r$</th>
<th>APG-MF (prop’d)</th>
<th>ADM-MF</th>
<th>Blockpivot-MF</th>
<th>ALS-MF</th>
<th>Mult-MF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>30</td>
<td>1.91e-1</td>
<td>3.68</td>
<td>1.92e-1</td>
<td>7.33</td>
<td>1.90e-1</td>
</tr>
<tr>
<td>60</td>
<td>1.42e-1</td>
<td>12.5</td>
<td>1.43e-1</td>
<td>19.5</td>
<td>1.40e-1</td>
</tr>
<tr>
<td>90</td>
<td>1.13e-1</td>
<td>26.7</td>
<td>1.15e-1</td>
<td>34.2</td>
<td>1.12e-1</td>
</tr>
</tbody>
</table>

sen from \(\{10, 20, 30\}\). The parameter $r$ is set to $q$ in (3.2). We use relative error $\text{relerr} = \|A_1 A_2 - M\|_F / \|M\|_F$ and CPU time (in seconds) to measure performance. Table 1 lists the average results of 20 independent trials. From the table, we can see that APG-MF outperforms all the other algorithms in both CPU time and solution quality.

4.1.2. Image data. In this subsection, we compare APG-MF (proposed), ADM-MF, Blockpivot-MF, ALS-MF, and Mult-MF on the CBCL and ORL image databases used in [25, 42]. There are 6977 face images in the training set of CBCL, each having $19 \times 19$ pixels. Multiple images of each face are taken with varying illuminations and facial expressions. The first 2000 images are used for our test. We vectorize every image and obtain a matrix $M$ of size $361 \times 2000$. Rank $r$ is chosen from \(\{30, 60, 90\}\). The average results of 10 independent trials are given in Table 2. We can see that APG-MF outperforms ADM-MF in both speed and solution quality. APG-MF is as accurate as Blockpivot-MF but runs much faster. ALS-MF and Mult-MF fail this test, and ALS-MF stagnates at solutions of low quality at the very beginning. Due to the poor performance of ALS-MF and Mult-MF, we compare only APG-MF, ADM-MF, and Blockpivot-MF in the remaining tests.

The ORL database has 400 images divided into 40 groups. Each image has $112 \times 92$ pixels, and each group has 10 images of one face taken from 10 different directions and with different expressions. All the images are used for our test. We vectorize each image and obtain a matrix $M$ of size $10304 \times 400$. As in the previous test, we choose $r$ from \(\{30, 60, 90\}\). The average results of 10 independent trials are listed in Table 3. From the results, we can see
Table 3
Comparison on the images from the ORL face database; **bold** values are **slow time**.

<table>
<thead>
<tr>
<th></th>
<th>APG-MF (proposed)</th>
<th>ADM-MF</th>
<th>Blockpivot-MF</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>30</td>
<td>1.67e-1</td>
<td>15.8</td>
<td>1.71e-1</td>
</tr>
<tr>
<td>60</td>
<td>1.41e-1</td>
<td>42.7</td>
<td>1.45e-1</td>
</tr>
<tr>
<td>90</td>
<td>1.26e-1</td>
<td>76.4</td>
<td>1.30e-1</td>
</tr>
</tbody>
</table>

Figure 1. **Hyperspectral data** of $150 \times 150 \times 163$: Four selected slices are shown.

Table 4
Comparison on hyperspectral data of size $150 \times 150 \times 163$; **bold** values are **large error** or **slow time**.

<table>
<thead>
<tr>
<th></th>
<th>APG-MF (proposed)</th>
<th>ADM-MF</th>
<th>Blockpivot-MF</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>20</td>
<td>1.18e-2</td>
<td>34.2</td>
<td><strong>2.34e-2</strong></td>
</tr>
<tr>
<td>30</td>
<td>9.07e-3</td>
<td>63.2</td>
<td><strong>2.02e-2</strong></td>
</tr>
<tr>
<td>40</td>
<td>7.56e-3</td>
<td>86.2</td>
<td><strong>1.78e-2</strong></td>
</tr>
<tr>
<td>50</td>
<td>6.45e-3</td>
<td>120</td>
<td><strong>1.58e-2</strong></td>
</tr>
</tbody>
</table>

again that APG-MF is better than ADM-MF in both speed and solution quality, and that in far less time APG-MF achieves relative errors comparable to those of Blockpivot-MF.

4.1.3. **Hyperspectral data.** It has been shown in [54] that NMF can be applied to spectral data analysis. In [54], a regularized NMF model is also considered with penalty terms $\alpha \|A_1\|_F^2$ and $\beta \|A_2\|_F^2$ added in the objective of (3.2). The parameters $\alpha$ and $\beta$ can be tuned for specific purposes in practice. Here, we focus on the original NMF model to show the effectiveness of our algorithm. However, our method can be easily modified for solving the regularized NMF model. In this test, we use a $150 \times 150 \times 163$ hyperspectral cube to test the compared algorithms. Each slice of the cube is reshaped as a column vector, and a $22500 \times 163$ matrix $M$ is obtained. In addition, the cube is scaled to have a unit maximum element. Four selected slices before scaling are shown in Figure 1, corresponding to the 1st, 50th, 100th, and 150th columns of $M$. The dimension $r$ is chosen from $\{20, 30, 40, 50\}$, and Table 4 lists the average results of 10 independent trials. We can see from the table that APG-MF is superior to ADM-MF and Blockpivot-MF in both speed and solution quality.

4.1.4. **Nonnegative matrix completion.** In this subsection, we compare APG-MC and the ADM-based algorithm (ADM-MC) proposed in [74] on the hyperspectral data used in the previous test. It is demonstrated in [74] that ADM-MC outperforms other matrix completion solvers such as APGL and LMaFit on recovering nonnegative matrices because ADM-MC takes advantage of data nonnegativity, while the latter two do not. We fix the dimension
Table 5
Comparison on hyperspectral data at stopping time $T = 50, 100$ (sec); bold values are large error.

<table>
<thead>
<tr>
<th>Smpl. rate</th>
<th>APG-MC (prop'd) PSNR MSE</th>
<th>ADM-MC PSNR MSE</th>
<th>Smpl. rate</th>
<th>APG-MC (prop'd) PSNR MSE</th>
<th>ADM-MC PSNR MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.20</td>
<td>32.30 5.89e-4</td>
<td>28.72 1.35e-3</td>
<td>0.20</td>
<td>32.57 5.54e-4</td>
<td>28.80 1.33e-3</td>
</tr>
<tr>
<td>0.30</td>
<td>40.65 8.62e-5</td>
<td>33.58 4.64e-4</td>
<td>0.30</td>
<td>41.19 7.61e-5</td>
<td>33.69 4.52e-4</td>
</tr>
<tr>
<td>0.40</td>
<td>45.77 2.66e-5</td>
<td>38.52 1.46e-4</td>
<td>0.40</td>
<td>46.03 2.50e-5</td>
<td>38.69 1.41e-4</td>
</tr>
</tbody>
</table>

Table 6
Comparison on synthetic three-way tensors; bold values are large error or slow time.

<table>
<thead>
<tr>
<th>Problem setting</th>
<th>APG-TF (prop'd) relerr Time</th>
<th>AS-TF relerr Time</th>
<th>Blockpivot-TF relerr Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_1 N_2 N_3 q$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80 80 80 10</td>
<td>8.76e-005 0.44</td>
<td>7.89e-005 0.86</td>
<td>8.62e-005 0.82</td>
</tr>
<tr>
<td>80 80 80 20</td>
<td>9.47e-005 1.26</td>
<td>1.97e-004 1.45</td>
<td>1.77e-004 1.21</td>
</tr>
<tr>
<td>80 80 80 30</td>
<td>9.65e-005 2.83</td>
<td>2.05e-004 2.13</td>
<td>2.07e-004 1.95</td>
</tr>
<tr>
<td>50 50 500 10</td>
<td>9.15e-005 1.27</td>
<td>1.07e-004 1.91</td>
<td>9.54e-005 1.87</td>
</tr>
<tr>
<td>50 50 500 20</td>
<td>9.44e-005 3.42</td>
<td>1.86e-004 3.17</td>
<td>1.77e-004 3.47</td>
</tr>
<tr>
<td>50 50 500 30</td>
<td>9.74e-005 7.11</td>
<td>1.89e-004 5.04</td>
<td>1.88e-004 4.54</td>
</tr>
</tbody>
</table>

$r = 40$ in (3.8) and choose sample ratio $\Omega \triangleq \frac{|\Omega|}{mn}$ from $\{0.20, 0.30, 0.40\}$, where the samples in $\Omega$ are chosen at random. The parameter $\delta_\omega$ for APG-MC is set to 1, and all the parameters for ADM-MC are set to their default values. To make the comparison consistent, we let both of the algorithms run to a maximum time (sec) $T = 50, 100$, and we employ peak signal-to-noise ratio (PSNR) and mean squared error (MSE) to measure the performance of the two algorithms. Table 5 lists the average results of 10 independent trials. From the table, we can see that APG-MC is significantly better than ADM-MC in all cases.

4.2. Nonnegative three-way tensor factorization. To the best of our knowledge, all the existing algorithms for NTFs are extensions of those for NMF, including the multiplicative updating method [71], hierarchical ALS algorithm [19], alternating Poisson regression algorithm [17], and ANLS methods [31, 33]. We compare APG-TF with two ANLS methods, AS-TF [31] and Blockpivot-TF [33], which are also proposed based on the CP decomposition and are superior to many other algorithms. We set $tol = 10^{-4}$ and $maxit = 2000$ for all the compared algorithms. All the other parameters for Blockpivot-TF and AS-TF are set to their default values.

4.2.1. Synthetic data. We compare APG-TF, Blockpivot-TF, and AS-TF on randomly generated three-way tensors. Each tensor is $\mathbf{M} = \mathbf{L} \circ \mathbf{C} \circ \mathbf{R}$, where $\mathbf{L}, \mathbf{C}$ are generated by MATLAB commands $\max(0, \text{rand}(N_1, q))$ and $\max(0, \text{rand}(N_2, q))$, respectively, and $\mathbf{R}$ by $\text{rand}(N_3, q)$. The algorithms are compared with two sets of $(N_1, N_2, N_3)$ and $r = q = 10, 20, 30$. The relative error $\text{rele}r = \|\mathbf{M} - \mathbf{A}_1 \circ \mathbf{A}_2 \circ \mathbf{A}_3\|_F / \|\mathbf{M}\|_F$ and CPU time (sec) measure the performance of the algorithms. The average results of 10 independent runs are shown in Table 6, from which we can see that all the algorithms give similar results.

4.2.2. Image test. NMF does not utilize the spatial redundancy, and the matrix decomposition is not unique. Also, NMF factors tend to form the invariant parts of all images
as ghosts, while NTF factors can correctly resolve all the parts [63]. We compare APG-TF, Blockpivot-TF, and AS-TF on two nonnegative three-way tensors in [63]. Each slice of the tensors corresponds to an image. The first tensor is $19 \times 19 \times 2000$ and is formed from 2000 images in the CBCL database used in section 4.1.2. The average performances of 10 independent runs with $r = 40, 50, 60$ are shown in Table 7. Another tensor has the size of $32 \times 32 \times 256$ and is formed with the 256 images in the Swimmer dataset [20]. The results of 10 independent runs with $r = 40, 50, 60$ are listed in Table 8. Both tests show that APG-TF is consistently faster than Blockpivot-TF and AS-TF. In particular, APG-TF is much faster than Blockpivot-TF and AS-TF with better solution quality in the second test.

### 4.2.3. Hyperspectral data.
NTF is employed in [77] for hyperspectral unmixing. It is demonstrated that the cubic data can be highly compressed, and NTF is efficient in identifying the material signatures. We compare APG-TF with Blockpivot-TF and AS-TF on the $150 \times 150 \times 163$ hyperspectral cube, which is used in section 4.1.3. For consistency, we let them run to a maximum time $T$ and compare the relative errors. The parameter $r$ is chosen from $\{30, 40, 50, 60\}$. The relative errors corresponding to $T = 10, 25, 50, 100$ are shown in Table 9, as the average of 10 independent trials. We can see from the table that APG-TF achieves the same accuracy much earlier than Blockpivot-TF and AS-TF.

### 4.2.4. Nonnegative tensor completion.
Recently, Liu et al. [45] proposed tensor completion based on minimizing tensor $n$-rank, the matrix rank of mode-$n$ matricization of a tensor.

---

### Table 7
Comparison results on CBCL database; bold values are slow time.

<table>
<thead>
<tr>
<th>$r$</th>
<th>APG-TF (proposed)</th>
<th>AS-TF</th>
<th>Blockpivot-TF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>$40$</td>
<td>1.85e-001</td>
<td>9.95e+000</td>
<td>1.86e-001</td>
</tr>
<tr>
<td>$50$</td>
<td>1.68e-001</td>
<td>1.65e+001</td>
<td>1.68e-001</td>
</tr>
<tr>
<td>$60$</td>
<td>1.53e-001</td>
<td>2.13e+001</td>
<td>1.56e-001</td>
</tr>
</tbody>
</table>

### Table 8
Comparison results on Swimmer database; bold values are large error or slow time.

<table>
<thead>
<tr>
<th>$r$</th>
<th>APG-TF (proposed)</th>
<th>AS-TF</th>
<th>Blockpivot-TF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>$40$</td>
<td>2.43e-001</td>
<td>2.01e+000</td>
<td>2.71e-001</td>
</tr>
<tr>
<td>$50$</td>
<td>1.45e-001</td>
<td>3.21e+000</td>
<td>2.00e-001</td>
</tr>
<tr>
<td>$60$</td>
<td>3.16e-002</td>
<td>6.91e+000</td>
<td>1.10e-001</td>
</tr>
</tbody>
</table>

### Table 9
Relative errors on hyperspectral data.

<table>
<thead>
<tr>
<th>$r \setminus T$</th>
<th>APG-TF (proposed)</th>
<th>AS-TF</th>
<th>Blockpivot-TF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$10$</td>
<td>$25$</td>
<td>$50$</td>
</tr>
<tr>
<td>$30$</td>
<td>2.56e-1</td>
<td>2.68e-1</td>
<td>2.53e-1</td>
</tr>
<tr>
<td>$40$</td>
<td>2.32e-1</td>
<td>2.37e-1</td>
<td>2.26e-1</td>
</tr>
<tr>
<td>$50$</td>
<td>2.14e-1</td>
<td>2.07e-1</td>
<td>2.04e-1</td>
</tr>
<tr>
<td>$60$</td>
<td>2.00e-1</td>
<td>1.91e-1</td>
<td>1.87e-1</td>
</tr>
</tbody>
</table>
Table 10
Comparison results on synthetic NTC; **bold** values are bad or slow.

<table>
<thead>
<tr>
<th>Problem setting</th>
<th>APG-TC (prop’d) $r = q$</th>
<th>APG-TC (prop’d) $r = \lceil 1.25q \rceil$</th>
<th>FaLRTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_1$ $N_2$ $N_3$</td>
<td>relerr</td>
<td>Time</td>
<td>relerr</td>
</tr>
<tr>
<td>80 80 80 10</td>
<td>2.02e-4</td>
<td>4.09</td>
<td>6.08e-4</td>
</tr>
<tr>
<td>80 80 80 10</td>
<td>1.18e-4</td>
<td>2.52</td>
<td>3.29e-4</td>
</tr>
<tr>
<td>80 80 80 10</td>
<td>9.54e-5</td>
<td>2.22</td>
<td>2.45e-4</td>
</tr>
<tr>
<td>80 80 80 20</td>
<td>1.50e-4</td>
<td>9.55</td>
<td>4.84e-4</td>
</tr>
<tr>
<td>80 80 80 20</td>
<td>1.15e-4</td>
<td>6.08</td>
<td>2.64e-4</td>
</tr>
<tr>
<td>80 80 80 20</td>
<td>9.65e-5</td>
<td>3.04</td>
<td>1.72e-4</td>
</tr>
<tr>
<td>80 80 80 30</td>
<td>3.14e-3</td>
<td>16.4</td>
<td>4.23e-4</td>
</tr>
<tr>
<td>80 80 80 30</td>
<td>1.04e-4</td>
<td>11.2</td>
<td>1.94e-4</td>
</tr>
<tr>
<td>80 80 80 30</td>
<td>1.14e-4</td>
<td>9.91</td>
<td>1.47e-4</td>
</tr>
<tr>
<td>50 50 500 10</td>
<td>2.76e-4</td>
<td>11.6</td>
<td>4.69e-4</td>
</tr>
<tr>
<td>50 50 500 10</td>
<td>9.81e-5</td>
<td>6.24</td>
<td>2.12e-4</td>
</tr>
<tr>
<td>50 50 500 10</td>
<td>9.51e-5</td>
<td>5.34</td>
<td>1.74e-4</td>
</tr>
<tr>
<td>50 50 500 20</td>
<td>1.80e-4</td>
<td>24.5</td>
<td>3.50e-4</td>
</tr>
<tr>
<td>50 50 500 20</td>
<td>3.95e-3</td>
<td>13.4</td>
<td>1.59e-4</td>
</tr>
<tr>
<td>50 50 500 20</td>
<td>5.32e-3</td>
<td>11.8</td>
<td>1.15e-4</td>
</tr>
<tr>
<td>50 50 500 30</td>
<td>7.09e-3</td>
<td>39.0</td>
<td>5.08e-4</td>
</tr>
<tr>
<td>50 50 500 30</td>
<td>1.03e-4</td>
<td>25.4</td>
<td>1.26e-4</td>
</tr>
<tr>
<td>50 50 500 30</td>
<td>3.28e-3</td>
<td>23.0</td>
<td>1.03e-4</td>
</tr>
</tbody>
</table>

Using the matrix nuclear norm instead of matrix rank, they solve the convex program

\[
\min_{\mathbf{X}} \sum_{n=1}^{N} \alpha_n \|\mathbf{X}_{(n)}\|_* \quad \text{subject to } P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{M}),
\]

where $\alpha_n$ are prespecified weights satisfying $\sum_n \alpha_n = 1$ and $\|\mathbf{A}\|_*$ is the nuclear norm of $\mathbf{A}$ defined as the sum of singular values of $\mathbf{A}$. Meanwhile, they proposed some algorithms to solve (4.1) or its relaxed versions, including simple low-rank tensor completion (SiLRTC), fast low-rank tensor completion (FaLRTC), and high accuracy low-rank tensor completion (HaLRTC). We compare APG-TC with FaLRTC on synthetic three-way tensors since FaLRTC is more efficient and stable than SiLRTC and HaLRTC. Each tensor is generated similarly as in section 4.2.1. Rank $q$ is chosen from $\{10, 20, 30\}$, and sampling ratio $\text{SR} = |\Omega|/(N_1 N_2 N_3)$ from $\{0.10, 0.30, 0.50\}$. For APG-TC, we use $r = q$ and $r = \lceil 1.25q \rceil$ in (3.8). We set $\text{tol} = 10^{-4}$ and $\text{maxit} = 2000$ for both algorithms. The weights $\alpha_n$ in (4.1) are set to $\alpha_n = \frac{1}{3}, n = 1, 2, 3$, and the smoothing parameters for FaLRTC are set to $\mu_n = \frac{5}{N_n}, n = 1, 2, 3$. Other parameters of FaLRTC are set to their default values. The average results of 10 independent trials are shown in Table 10. We can see that APG-TC produces much more accurate solutions within less time.

4.3. Summary. Although our test results are obtained with a given set of parameters, it is clear from the results that, compared to the existing algorithms, the proposed algorithms can return solutions of similar or better quality in less time. Tuning the parameters of the compared algorithms can hardly obtain much improvement in both solution quality and time.
We believe that the superior performance of the proposed algorithms is due to the use of prox-linear steps, which are not only easy to compute but also, as a local approximate, help avoid the small regions around certain local minima.

5. Conclusions. We have proposed a block coordinate descent method with three choices of update schemes for multiconvex optimization. The diverse update schemes fit different applications. We established subsequence and global convergence guarantees, as well as asymptotic rate of convergence, under certain assumptions. Numerical results on both synthetic and real image data illustrate the high efficiency of the proposed algorithms.

Appendix A. Proofs of Lemma 2.6 and Theorem 2.9.

A.1. Proof of Lemma 2.6. Without loss of generality, we assume \( F = 0 \). Otherwise, we can consider \( F - \bar{F} \). Let \( B(\bar{x},\rho) \triangleq \{ x : \| x - \bar{x} \| \leq \rho \} \subset \mathcal{U} \) for some \( \rho > 0 \), where \( \mathcal{U} \) is the neighborhood of \( \bar{x} \) in (2.14) with \( \psi = F \), and let \( L_G \) be the global Lipschitz constant for \( \nabla x_i f(x) \), \( i = 1, \ldots, s \), within \( B(\bar{x}, \sqrt{11} \rho) \), namely,

\[
\| \nabla x_i f(x) - \nabla x_i f(y) \| \leq L_G \| x - y \|, \quad i = 1, \ldots, s,
\]

for any \( x, y \in B(\bar{x}, \sqrt{11} \rho) \).

The proof will follow two steps. The first step will show the following claim.

Claim A.1. Let \( \ell = \min_i \ell_i \), \( L = \max_i L_i \), and

\[
C_1 = \frac{9(L + sL_G)}{2\ell(1 - \delta \omega)^2}, \quad C_2 = 2\sqrt{\frac{2}{\ell}} + \left( \frac{3\delta \omega}{1 - \delta \omega} \sqrt{\frac{L}{\ell} + 3} \right) \sqrt{\frac{2}{\ell} + \frac{2L\delta^2 \omega}{\ell^2}},
\]

where \( \ell_i, L_i \) are the constants in Assumption 2. If \( F_k > \bar{F} \) and

\[
C_1 \phi(F_0 - \bar{F}) + C_2 \sqrt{F_0 - \bar{F}} + \| x_0 - \bar{x} \| < \rho,
\]

then

\[
\| x_k - x_{k+1} \| \leq C_1 \phi(F_N - \bar{F}) + \| x_{N-1} - x_{N-2} \| + \left( \frac{3\delta \omega}{1 - \delta \omega} \sqrt{\frac{L}{\ell} + 2} \right) \| x^N - x^{N-1} \| \quad \forall N \geq 2,
\]

where \( C_1 \) is specified in Claim A.1.

Note that (A.1) quantifies how close to \( \bar{x} \) the initial point \( x_0 \) is required to be. The second step will establish the next claim.

Claim A.2.

\[
\sum_{k=N}^{\infty} \| x_{k+1} - x_k \| \leq C_1 \phi(F_N - \bar{F}) + \| x_{N-1} - x_{N-2} \| + \left( \frac{3\delta \omega}{1 - \delta \omega} \sqrt{\frac{L}{\ell} + 2} \right) \| x^N - x^{N-1} \| \quad \forall N \geq 2,
\]

where \( C_1 \) is specified in Claim A.1.

Note (A.3) implies that \( \{ x_k \} \) is a Cauchy sequence, and thus \( x_k \) converges. Hence, if (A.2) and (A.3) both hold, then letting \( B = B(\bar{x}, \rho) \) will prove the results of Lemma 2.6.
Proof of Claim A.1. We will prove $x^k \in B(\bar{x}, \rho)$ by induction on $k$. Obviously, $x^0 \in B(\bar{x}, \rho)$ from (A.1). Hence, (A.2) holds for $k = 0$.

For $k = 1$, we have from (2.8) that

$$F_0 \geq F_0 - F_1 \geq \sum_{i=1}^{s} \frac{L_0^i}{2} \|x_i^0 - x_i^1\|^2 \geq \frac{\ell}{2} \|x^0 - x^1\|^2.$$ 

Hence, $\|x^0 - x^1\| \leq \sqrt{\frac{2}{\ell} F_0}$, and

$$\|x^1 - \bar{x}\| \leq \|x^0 - x^1\| + \|x^0 - \bar{x}\| \leq \sqrt{\frac{2}{\ell} F_0} + \|x^0 - \bar{x}\|,$$

which indicates $x^1 \in B(\bar{x}, \rho)$.

For $k = 2$, we have from (2.8) that

$$F_0 \geq F_1 - F_2 \geq \sum_{i=1}^{s} \frac{L_1^i}{2} \|x_i^1 - x_i^2\|^2 - \sum_{i=1}^{s} \frac{L_0^i}{2} \delta_{\omega}^2 \|x_i^0 - x_i^1\|^2.$$ 

Note that $L_0^i \leq L$ and $L_1^i \geq \ell$ for $i = 1, \ldots, s$. Thus, it follows from the above inequality that

$$\frac{\ell}{2} \|x^1 - x^2\|^2 \leq \sum_{i=1}^{s} \frac{L_1^i}{2} \|x_i^1 - x_i^2\|^2 \leq F_0 + \frac{L}{2} \delta_{\omega}^2 \|x^0 - x^1\|^2 \leq \left(1 + \frac{L}{\ell} \delta_{\omega}^2\right) F_0,$$

which implies $\|x^1 - x^2\| \leq \sqrt{\frac{2}{\ell} + \frac{2L\delta_{\omega}^2}{\ell^2}} \sqrt{F_0}$. Therefore,

$$\|x^2 - \bar{x}\| \leq \|x^1 - x^2\| + \|x^1 - \bar{x}\| \leq \left(\sqrt{\frac{2}{\ell}} + \sqrt{\frac{2L\delta_{\omega}^2}{\ell^2}}\right) \sqrt{F_0} + \|x^0 - \bar{x}\|,$$

and thus $x^2 \in B(\bar{x}, \rho)$.

Suppose $x^k \in B(\bar{x}, \rho)$ for $0 \leq k \leq K$. We proceed to show that $x^{K+1} \in B(\bar{x}, \rho)$. For $k \leq K$, note that

$$\partial F(x^k) = \left\{\partial r_1(x^k_i) + \nabla_{x_i} f(x^k)\right\} \times \cdots \times \left\{\partial r_s(x^k_s) + \nabla_{x_s} f(x^k)\right\}$$

and

$$-\nabla f_i^k(x_i^k) + \nabla_{x_i} f(x^k) \in \partial r_i(x_i^k) + \nabla_{x_i} f(x^k), \quad i \in I_1,$$

$$-L_i^{-1}(x_i^k - x_i^{k-1}) - \nabla f_i^k(x_i^k) + \nabla_{x_i} f(x^k) \in \partial r_i(x_i^k) + \nabla_{x_i} f(x^k), \quad i \in I_2,$$

$$-L_i^{-1}(x_i^k - x_i^{k-1}) - \nabla f_i^k(x_i^{k-1}) + \nabla_{x_i} f(x^k) \in \partial r_i(x_i^k) + \nabla_{x_i} f(x^k), \quad i \in I_3,$$

so (for $i \in I_1 \cup I_2$, regard $x_i^{k-1} = x_i^k - x_i^{k-1}$ in $x_i^k - x_i^{k-1}$ and $x_i^{k-1} = x_i^k$ in $\nabla f_i^k(x_i^{k-1}) - \nabla_{x_i} f(x^k)$, respectively)

$$\text{dist}(0, \partial F(x^k))$$

$$\leq \left\|\left(L_1^{-1}(x_1^k - x_1^{k-1}), \ldots, L_s^{-1}(x_s^k - x_s^{k-1})\right)\right\| + \sum_{i=1}^{s} \left\|\nabla f_i^k(x_i^{k-1}) - \nabla_{x_i} f(x^k)\right\|.$$
For the first term on the right-hand side of (A.5), plugging in $x_i^{k-1}$ and recalling $L_i^{k-1} \leq L$ for $i = 1, \ldots, s$, we can easily get

\[(A.6) \quad \left\| \left( L_1^{k-1}(x_1^k - x_1^{k-1}), \ldots, L_s^{k-1}(x_s^k - x_s^{k-1}) \right) \right\| \leq L \left( \| x^k - x^{k-1} \| + \| x^{k-1} - x^{k-2} \| \right).\]

For the second term on the right-hand side of (A.5), it is not difficult to verify

\[\left( x_1^k, \ldots, x_i^{k-1}, x_i^k, \ldots, x_s^{k-1} \right) \in B(x, \sqrt{t\rho}).\]

In addition, note that

\[\nabla f_i^k(x_i^{k-1}) = \nabla x_i f \left( x_1^k, \ldots, x_i^{k-1}, x_i^k, \ldots, x_s^{k-1} \right).\]

Hence,

\[\sum_{i=1}^{s} \left\| \nabla x_i f_i^k(x_i^{k-1}) - \nabla x_i f(x^k) \right\| \leq \sum_{i=1}^{s} L_G \left\| \left( x_1^k, \ldots, x_i^{k-1}, x_i^k, \ldots, x_s^{k-1} \right) - x^k \right\| \]

\[\leq sL_G \left( \| x^k - x^{k-1} \| + \| x^{k-1} - x^{k-2} \| \right) .\]

Combining (A.5), (A.6), and (A.7) gives

\[\text{dist}(0, \partial f(x^k)) \leq (L + sL_G) \left( \| x^k - x^{k-1} \| + \| x^{k-1} - x^{k-2} \| \right),\]

which together with the KL inequality (2.14) implies

\[(A.8) \quad \phi'(F_k) \geq (L + sL_G)^{-1} \left( \| x^k - x^{k-1} \| + \| x^{k-1} - x^{k-2} \| \right)^{-1} .\]

Note that $\phi$ is concave and $\phi'(F_k) > 0$. Thus it follows from (2.8) and (A.8) that

\[\phi(F_k) - \phi(F_{k+1}) \geq \phi'(F_k)(F_k - F_{k+1}) \]

\[\geq \frac{\sum_{i=1}^{s} L_i^k \| x_i^k - x_i^{k+1} \|^2 - L_i^{k-1} \delta^2 \| x_i^{k-1} - x_i^k \|^2}{2(L + sL_G) \left( \| x^k - x^{k-1} \| + \| x^{k-1} - x^{k-2} \| \right)},\]

or, equivalently,

\[(A.9) \quad \left\| \left( \sqrt{L_1^k}(x_1^k - x_1^{k+1}), \ldots, \sqrt{L_s^k}(x_s^k - x_s^{k+1}) \right) \right\|^2 \]

\[\leq \delta^2 \left\| \left( \sqrt{L_1^{k-1}}(x_1^{k-1} - x_1^k), \ldots, \sqrt{L_s^{k-1}}(x_s^{k-1} - x_s^k) \right) \right\|^2 \]

\[+ 2(L + sL_G) \left( \| x^k - x^{k-1} \| + \| x^{k-1} - x^{k-2} \| \right) \phi(F_k) - \phi(F_{k+1}).\]
Using inequalities $a^2 + b^2 \leq (a + b)^2$ and $ab \leq ta^2 + \frac{b^2}{4t}$ for $t > 0$, we get from (A.9) that

\[
\left\| \left( \sqrt{L^k_1(x^k_1 - x^{k+1}_1)}, \ldots, \sqrt{L^k_s(x^k_s - x^{k+1}_s)} \right) \right\| 
\leq \delta_w \left\| \left( \sqrt{L^{-1}_1(x^{k-1}_1 - x^k_1)}, \ldots, \sqrt{L^{-1}_s(x^{k-1}_s - x^k_s)} \right) \right\|
\]

\[
+ \left( \|x^k - x^{k-1}\| + \|x^{k-1} - x^{k-2}\| \right) \frac{1}{2} \left( 2(L + sL_G)(\phi(F_k) - \phi(F_{k+1})) \right)^{\frac{1}{2}} \]

(A.10) \leq \delta_w \left\| \left( \sqrt{L^{-1}_1(x^{k-1}_1 - x^k_1)}, \ldots, \sqrt{L^{-1}_s(x^{k-1}_s - x^k_s)} \right) \right\|
\]

\[
+ \frac{(1 - \delta_w)\sqrt{\ell}}{3} \left( \|x^k - x^{k-1}\| + \|x^{k-1} - x^{k-2}\| \right) + \frac{3(L + sL_G)}{2\sqrt{\ell}(1 - \delta_w)}(\phi(F_k) - \phi(F_{k+1})).
\]

Summing up (A.10) over $k$ from 2 to $K$ and doing some eliminations gives

\[
\left\| \left( \sqrt{L^k_1(x^k_1 - x^{K+1}_1)}, \ldots, \sqrt{L^k_s(x^k_s - x^{K+1}_s)} \right) \right\|
\leq \delta_w \left\| \left( \sqrt{L^{-1}_1(x^{k-1}_1 - x^k_1)}, \ldots, \sqrt{L^{-1}_s(x^{k-1}_s - x^k_s)} \right) \right\|
\]

\[
+ \frac{(1 - \delta_w)\sqrt{\ell}}{3} \sum_{k=2}^{K} \left( \|x^k - x^{k-1}\| + \|x^{k-1} - x^{k-2}\| \right)
\]

\[
+ \frac{3(L + sL_G)}{2\sqrt{\ell}(1 - \delta_w)}(\phi(F_2) - \phi(F_{K+1})).
\]

Note that $\ell \leq L^k_j \leq L$ for all $i, k$. We have from the above inequality that

\[
\sum_{k=2}^{K} (1 - \delta_w)\sqrt{\ell}\|x^k - x^{k+1}\|
\leq \delta_w \sqrt{\ell}\|x^1 - x^2\| + \frac{(1 - \delta_w)\sqrt{\ell}}{3} \sum_{k=2}^{K} \left( \|x^k - x^{k-1}\| + \|x^{k-1} - x^{k-2}\| \right)
\]

\[
+ \frac{3(L + sL_G)}{2\sqrt{\ell}(1 - \delta_w)}(\phi(F_2) - \phi(F_{K+1})),
\]

which indicates that

\[
\sum_{k=2}^{K} \|x^k - x^{k+1}\|
\leq \left( \frac{3\delta_w}{1 - \delta_w}\sqrt{\frac{L}{\ell}} + 2 \right) \|x^1 - x^2\| + \|x^0 - x^1\| + \frac{9(L + sL_G)}{2\sqrt{\ell}(1 - \delta_w)}(\phi(F_2) - \phi(F_{K+1})).
\]

(A.11)

Recalling that $\|x^0 - x^1\| \leq \sqrt{\frac{\ell}{2}}F_0$, $\|x^1 - x^2\| \leq \sqrt{\frac{\ell}{2}} + \frac{2L\sqrt{\ell}}{1 - \delta_w} \sqrt{F_0}$ and using (A.4), we have from
(A.11) that
\[ \|x^{K+1} - \bar{x}\| \]
\[ \leq \sum_{k=2}^{K} \|x^k - x^{k+1}\| + \|x^2 - \bar{x}\| \]
\[ \leq \left( \frac{3\delta_\omega}{1 - \delta_\omega} \sqrt{\frac{L}{\ell} + 2} \right) \|x^1 - x^2\| + \|x^0 - x^1\| + \frac{9(L + sL G)}{2(1 - \delta_\omega)^2} \phi(F_0) + \|x^2 - \bar{x}\| \]
\[ \leq \frac{9(L + sL G)}{2(1 - \delta_\omega)^2} \phi(F_0) + \left( 2 \sqrt{\frac{2}{\ell}} + \left( \frac{3\delta_\omega}{1 - \delta_\omega} \sqrt{\frac{L}{\ell} + 3} \right) \sqrt{\frac{2}{\ell} + \frac{2L\delta_\omega^2}{\ell^2}} \right) \sqrt{F_0 + \|x^0 - \bar{x}\|}, \]
where we have used \( \phi(F_0) \geq \phi(F_k) \geq 0 \) in the second inequality. Hence, \( x^{K+1} \in B(\bar{x}, \rho) \), and this completes the proof of Claim A.1.

**Proof of Claim A.2.** Note that (A.10) holds for all \( k \geq 0 \). Summing it over \( k \) from \( N \) to \( T \) and in the same way that we derive (A.11), we have
\[ \sum_{k=N}^{T} \|x^k - x^{k+1}\| \]
\[ \leq \|x^{N-2} - x^{N-1}\| + \left( \frac{3\delta_\omega}{1 - \delta_\omega} \sqrt{\frac{L}{\ell} + 2} \right) \|x^{N-1} - x^N\| + \frac{9(L + sL G)}{2(1 - \delta_\omega)^2} \phi(F_N) - \phi(F_{T+1}). \]
Letting \( T \to \infty \) completes the proof of Claim A.2.

**A.2. Proof of Theorem 2.9.** If \( \theta = 0 \), we must have \( F(x^{k_0}) = F(\bar{x}) \) for some \( k_0 \). Otherwise, \( F(x^k) > F(\bar{x}) \) for all sufficiently large \( k \). The KL inequality gives \( c \cdot \text{dist}(0, \partial F(x^k)) \geq 1 \) for all \( k \geq 0 \), which is impossible since \( x^k \to \bar{x} \) and \( 0 \in \partial F(\bar{x}) \). The finite convergence now follows from the fact that \( F(x^{k_0}) = F(\bar{x}) \) implies \( x^k = x^{k_0} = \bar{x} \) for all \( k \geq k_0 \).

For \( \theta \in (0, 1) \), we assume \( F(x^k) \geq F(\bar{x}) = 0 \) and use the same notation as in the proof of Lemma 2.6. Define \( S_k = \sum_{i=k}^{\infty} \|x^i - x^{i+1}\| \). Then (A.3) can be written as
\[ S_k \leq C_1 \phi(F_k) + \left( \frac{3\delta_\omega}{1 - \delta_\omega} \sqrt{\frac{L}{\ell} + 2} \right) (S_{k-1} - S_k) + S_{k-2} - S_{k-1} \quad \text{for } k \geq 2, \]
which implies that
\[ S_k \leq C_1 \phi(F_k) + \left( \frac{3\delta_\omega}{1 - \delta_\omega} \sqrt{\frac{L}{\ell} + 2} \right) (S_{k-2} - S_k) \quad \text{for } k \geq 2, \]
since \( S_{k-2} - S_{k-1} \geq 0 \). Using \( \phi(s) = cs^{1-\theta} \), we have from (A.8) for sufficiently large \( k \) that
\[ c(1 - \theta)(F_k)^{-\theta} \geq (L + sL G)^{-1} \left( \|x^k - x^{k-1}\| + \|x^{k-1} - x^{k-2}\| \right)^{-1}, \]
or, equivalently, \( (F_k)^{\theta} \leq c(1 - \theta)(L + sL G)(S_{k-2} - S_k) \). Then,
\[ \phi(F_k) = c(F_k)^{1-\theta} \leq c(1 - \theta)(L + sL G)(S_{k-2} - S_k)^{\frac{1-\theta}{\theta}}. \]
Letting $C_3 = C_1 c(1 - \theta)(L + sL_G)\frac{1-\theta}{2}$ and $C_4 = \frac{3s}{1-\theta} \sqrt{\frac{\tau}{c}} + 2$, we have from (A.12) and (A.13) that

\[(A.14) \quad S_k \leq C_3 (S_{k-2} - S_k) \frac{1-\theta}{2} + C_4 (S_{k-2} - S_k).\]

When $\theta \in (0, \frac{1}{2}]$, i.e., $\frac{1-\theta}{2} \geq 1$, (A.14) implies that $S_k \leq (C_3 + C_4)(S_{k-2} - S_k)$ for sufficiently large $k$ since $S_{k-2} - S_k \to 0$, and thus $S_k \leq \frac{C_3 + C_4}{1+C_3+C_4} S_{k-2}$. Note that $\|x^k - \bar{x}\| \leq S_k$. Therefore, item 2 holds with $\tau = \sqrt{\frac{C_3 + C_4}{1+C_3+C_4}} < 1$ and sufficiently large $C$.

When $\theta \in (\frac{1}{2}, 1)$, i.e., $\frac{1-\theta}{2} < 1$, we get

\[(A.15) \quad S_{N\nu}^\nu + S_{N-1}^{\nu\nu} - S_{K+1}^{\nu\nu} - S_K^{\nu\nu} \geq \mu(N - K)\]

for $\nu = \frac{1-2\theta}{1-\theta} < 0$, some constant $\mu > 0$, and any $N > K$ with sufficiently large $K$ by the same argument as in the proof of Theorem 2 of [3]. Note that $S_N \leq S_{N-1}$ and $\nu < 0$. Hence, (A.15) implies that

$S_N \leq \left(\frac{1}{2} \left( S_{K+1}^{\nu\nu} + S_K^{\nu\nu} + \mu(N - K) \right) \right)^{\frac{1}{\nu}} \leq CN^{-\frac{1-\theta}{2\nu-1}}$

for sufficiently large $C$ and $N$. This completes the proof. \[\square\]

**Acknowledgments.** The authors would like to thank three anonymous referees for their careful reviews and helpful comments. Also, they would like to thank Zhi-Quan (Tom) Luo for very helpful discussions and for sharing his manuscript [57], and Michael Ulbrich and Zaiwen Wen for carefully reading and pointing out mistakes in our proofs.

**REFERENCES**


