Inertial Block Proximal Methods For Non-Convex Non-Smooth Optimization

Le Thi Khanh Hien [1] **Nicolas Gillis** [1] **Panagiotis Patrinos** [2]

Abstract

We propose inertial versions of block coordinate descent methods for solving non-convex non-smooth composite optimization problems. Our methods possess three main advantages compared to current state-of-the-art accelerated first-order methods: (1) they allow using two different extrapolation points to evaluate the gradients and to add the inertial force (we will empirically show that it is more efficient than using a single extrapolation point), (2) they allow to randomly picking the block of variables to update, and (3) they do not require a restarting step. We prove the subsequential convergence of the generated sequence under mild assumptions, prove the global convergence under some additional assumptions, and provide convergence rates. We deploy the proposed methods to solve non-negative matrix factorization (NMF) and show that they compete favourably with the state-of-the-art NMF algorithms. Additional experiments on non-negative approximate canonical polyadic decomposition, also known as non-negative tensor factorization, are also provided.

1. Introduction

In this paper, we consider the following non-smooth non-convex optimization problem

$$\text{minimize}_{x\in\mathbb{E}}\, F(x), \quad \text{where } F(x) := f(x)+r(x), \quad (1)$$

and $\mathbb{E} = \mathbb{E}_1 \times \ldots \times \mathbb{E}_s$ with $\mathbb{E}_i$, $i = 1, \ldots, s$, being finite dimensional real linear spaces equipped with norm $\|\cdot\|_{(i)}$ and inner product $\langle\cdot,\cdot\rangle_{(i)}$, $f : \mathbb{E} \to \mathbb{R}$ is a continuous but possibly non-smooth non-convex function, and $r(x) = \sum_{i=1}^{s} r_i(x_i)$ with $r_i : \mathbb{E}_i \to \mathbb{R} \cup \{+\infty\}$ for $i = 1, \ldots, s$ being proper and lower semi-continuous functions.

Problem (1) covers many applications including compressed sensing with non-convex "norms" (Attouch et al., 2010), sparse dictionary learning (Aharon et al., 2006; Xu & Yin, 2016), non-negative matrix factorization (NMF) (Gillis, 2014), and "l_p-norm" regularized sparse regression problems with $0 \leq p < 1$ (Blumensath & Davies, 2009; Natarajan, 1995). In this paper, we will focus on NMF which is defined as follows: given $X \in \mathbb{R}^{\mathbf{m}\times\mathbf{n}}_+$ and the integer $\mathbf{r} < \min(\mathbf{m}, \mathbf{n})$, solve

$$\min_{U,V} \frac{1}{2}\|X - UV\|_F^2 \text{ such that } U \in \mathbb{R}^{\mathbf{m}\times\mathbf{r}}_+, V \in \mathbb{R}^{\mathbf{r}\times\mathbf{n}}_+. \quad (2)$$

NMF is a key problem in data analysis and machine learning with applications in image processing, document classification, hyperspecral unmixing and audio source separation, to cite a few (Cichocki et al., 2009; Gillis, 2014; Fu et al., 2019). NMF can be written as a problem of the form (1) with $s = 2$, letting $f(U, V) = \frac{1}{2}\|X - UV\|_F^2$, and r_1 and r_2 being indicator functions $r_1(U) = I_{\mathbb{R}^{\mathbf{m}\times\mathbf{r}}_+}(U)$, and $r_2(V) = I_{\mathbb{R}^{\mathbf{r}\times\mathbf{n}}_+}(V)$. Note that $UV = \sum_{i=1}^{r} U_{:i}V_{i:}$; hence NMF can also be written as a function of $2 \times \mathbf{r}$ variables $U_{:i}$ (the columns of U) and $V_{i:}$ (the rows of V) for $i = 1, \ldots, \mathbf{r}$.

1.1. Related Works

The Gauss-Seidel iteration scheme, also known as the block coordinate descent (BCD) method, is a standard approach to solve both convex and non-convex problems in the form of (1). Starting with a given initial point $x^{(0)}$, the method generates a sequence $\{x^{(k)}\}_{k\geq 0}$ by cyclically updating one block of variables at a time while fixing the values of the other blocks. Let us denote $f_i^{(k)}(x_i) := f\big(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_s^{(k-1)}\big)$ the value of the objective function for the ith block at the kth iteration of a BCD method. Based on how the blocks are updated, BCD methods can typically be classified into three categories:

1. Classical BCD methods update (Grippo & Sciandrone, 2000; Hildreth, 1957) using exact updates:

[1]Department of Mathematics and Operational Research, University of Mons, Belgium [2]Department of Electrical Engineering (ESAT-STADIUS), KU Leuven, Belgium. Correspondence to: Le Thi Khanh Hien <thikhanhhien.le@umons.ac.be>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

$$x_i^{(k)} = \underset{x_i \in \mathbb{E}_i}{\operatorname{argmin}} f_i^{(k)}(x_i) + r_i(x_i).$$

2. Proximal BCD methods update (Auslender, 1992; Grippo & Sciandrone, 2000; Razaviyayn et al., 2013) using exact updates along with a proximal term:

$$x_i^{(k)} = \underset{x_i \in \mathbb{E}_i}{\operatorname{argmin}} f_i^{(k)}(x_i) + r_i(x_i) + \frac{1}{2\beta_i^{(k)}} \|x_i - x_i^{(k-1)}\|^2, \tag{3}$$

where $\beta_i^{(k)}$ is referred to as the stepsize.

3. Proximal gradient BCD methods update (Bolte et al., 2014; Razaviyayn et al., 2013; Tseng & Yun, 2009) using a linearization of f

$$\begin{aligned} x_i^{(k)} = \underset{x_i \in \mathbb{E}_i}{\operatorname{argmin}} & \langle \nabla f_i^{(k)}(x_i^{(k-1)}), x_i - x_i^{(k-1)} \rangle \\ & + r_i(x_i) + \frac{1}{2\beta_i^{(k)}} \|x_i - x_i^{(k-1)}\|^2. \end{aligned} \tag{4}$$

Incorporating inertial force is a popular and efficient method to accelerate the convergence of first-order methods. The inertial term was first introduced by Polyak's heavy ball method (Polyak, 1964), which adds to the new direction a momentum term equal to the difference of the two previous iterates; this is also known as extrapolation. While the gradient evaluations used in Polyak's method are not affected by the momentum, the famous accelerated gradient method of Nesterov (1983; 1998; 2004; 2005) evaluates the gradients at the points which are extrapolated. In the convex setting, these methods are proved to achieve the optimal convergence rate, while the computational cost of each iteration is essentially unchanged. In the non-convex setting, the heavy ball method was first considered by Zavriev & Kostyuk (1993) to solve an unconstrained smooth minimization problem. Two inertial proximal gradient methods were proposed by Ochs et al. (2014) and Boţ & Csetnek (2016) to solve (1) with $s = 1$. The method considered by Ochs et al. (2014), referred to as iPiano, makes use of the inertial force but does not use the extrapolated points to evaluate the gradients. iPiano was extended for $s > 1$ and analysed by Ochs (2019). Pock & Sabach (2016) proposed iPALM to solve (1) with $s = 2$. Xu & Yin (2013; 2017) proposed inertial versions of proximal BCD, cf. (4). Xu & Yin's methods need restarting steps to guarantee the decrease of the objective function. As stated by Nesterov (2004), this relaxation property for some problem classes is too expensive and may not allow optimal convergence. In another line of works, it is worth mentioning the randomized BCD methods for solving convex problems; see Fercoq & Richtarik (2015); Nesterov (2012). The analysis of this type of algorithms considers the convergence of the function values in expectation. This is out of the scope of this work.

1.2. Contribution

In this paper, we propose inertial versions for the proximal and proximal gradient BCD methods (3) and (4), for solving the non-convex non-smooth problem (1) with *multiple blocks*. For the inertial version of the proximal gradient BCD (4), two extrapolation points can be used to evaluate gradients and add the inertial force so that the corresponding scheme is more flexible and may lead to significantly better numerical performance compared with the inertial methods using a single extrapolation point; this will be confirmed with some numerical experiments (see Section 5 and the supplementary material). The idea of using two different extrapolation points was first used for iPALM to solve (1) with two blocks; however, the parameters of the implemented version of iPALM in the experiments by Pock & Sabach (2016) are chosen outside the theoretical bounds established in the paper. Our methods for solving (1) with multiple blocks allow picking deterministically or randomly the block of variables to update; it was empirically observed that randomization may lead to better solutions and/or faster convergence (Xu & Yin, 2017). Another key feature of our methods is that they do not require restarting steps. We extend our methods in the framework of Bregman divergence so that they are more general hence admit potentially more applications. To prove the convergence of the whole sequence to a critical point of F and derive its convergence rate, we combine a modification of the convergence proof recipe by Bolte et al. (2014) with the technique of using auxiliary functions (Ochs et al., 2014). By choosing appropriate parameters that guarantee the convergence, we apply the methods to NMF. We also apply it to non-negative canonical polyadic decomposition (NCPD) in the supplementary material.

2. The Proposed Methods: IBP and IBPG

Algorithm 1 describes our two proposed methods: (1) the inertial block proximal method (IBP) which is a proximal BCD method with one extrapolation point, and (2) the inertial block proximal gradient method (IBPG) which is a proximal gradient BCD method with two extrapolation points.

Algorithm 1 includes an outer loop which is indexed by k and an inner loop which is indexed by j. At the jth iteration of an inner loop, only one block is updated. Table 1 summarizes the notation used in the paper. The choice of the parameters $\alpha^{k,j}$, $\beta^{k,j}$ and $\gamma^{k,j}$ in Algorithm 1 that guarantee the convergence will be discussed in Section 3. We can observe from (7) and (8) that using two extrapolation points do not bring extra computation cost when compared with using a single extrapolation point (which happens when $\alpha_i^{(k,j)} = \gamma_i^{(k,j)}$). We make the following standard assumption throughout this paper.

Assumption 1 *For all k, all blocks are updated after the*

Table 1. Notation

Notation	Definition
$x^{(k,j)}$	x at the jth iteration within the kth outer loop
$\tilde{x}^{(k)}$	the main generated sequence (the output)
T_k	number of iterations within the kth outer loop
$f_i^{(k,j)}(x_i)$	a function of the ith block while fixing the latest updated values of the other blocks, i.e.,
$= f(x_1^{(k,j-1)}, \ldots,$	$x_{i-1}^{(k,j-1)}, x_i, x_{i+1}^{(k,j-1)}, \ldots, x_s^{(k,j-1)})$
$F_i^{(k,j)}(x_i)$	$F_i^{(k,j)}(x_i) = f_i^{(k,j)}(x_i) + r_i(x_i)$
$\bar{x}_i^{(k,m)}$	the value of block i after it has been updated m times during the kth outer loop
d_i^k	the total number of times the ith block is updated during the kth outer loop
$\bar{\alpha}_i^{(k,m)}$	the values of $\alpha_i^{(k,j)}$,
$\bar{\beta}_i^{(k,m)}$	the values of $\beta_i^{(k,j)}$,
$\bar{\gamma}_i^{(k,m)}$	and the values of $\gamma_i^{(k,j)}$ that are used in (5), (6), (7), (8), (11) and (12) to update block i from $\bar{x}_i^{(k,m-1)}$ to $\bar{x}_i^{(k,m)}$
$\{\bar{x}_i^{(k,m)}\}_{k\geq 1}$	the sequence that contains the updates of the ith block, i.e., $\{\ldots, \bar{x}_i^{(k,1)}, \ldots, \bar{x}_i^{(k,d_i^k)}, \ldots\}$

T_k iterations performed within the kth outer loop, and there exists a positive constant $\bar{T}$ such that $s \leq T_k \leq \bar{T}$.

Illustration with NMF Let us illustrate the proposed methods for NMF; see the supplementary material for the application to NCPD. We will use IBPG for NMF with 2 blocks of variables, namely U and V, and IBP with $2 \times \mathbf{r}$ blocks of variables, namely $U_{:,i}$ and $V_{i,:}$ ($1 \leq i \leq \mathbf{r}$). We choose the Frobenius norm for the proximal terms in (6) and (8). We have $\nabla_U f = UVV^T - XV^T$ and $\nabla_V f = U^TUV - U^TX$, hence the inertial proximal gradient step (8) of IBPG is a projected gradient step. If we choose $T_k = 2$ for all k then each inner loop of IBPG updates U and V once. Our algorithm also allows to choose $T_k > 2$, hence updating U or V several times before updating the other one. As explained by Gillis & Glineur (2012), repeating the update of U and V accelerates the algorithm compared to the pure cyclic update rule, because the terms VV^T and XV^T (resp. U^TU and U^TX) in the gradient of U (resp. V) do not need to be recomputed hence the second evaluation of the gradient is much cheaper; namely, $O(\mathbf{mr}^2)$ (resp. $O(\mathbf{nr}^2)$) vs. $O(\mathbf{mnr})$ operations; while $\mathbf{r} \ll \min(\mathbf{m}, \mathbf{n})$ for most applications. Regarding IBP, the inertial proximal step (6) has a closed form:

Algorithm 1 IBP and IBPG

Initialize: Choose $\tilde{x}^{(0)} = \tilde{x}^{(-1)}$. Choose a method: IBP or IBPG. Parameters are chosen as in Section 3.
for $k = 1, \ldots$ **do**
  $x^{(k,0)} = \tilde{x}^{(k-1)}$.
  for $j = 1, \ldots, T_k$ **do**
    Choose $i \in \{1, \ldots, s\}$ deterministically or randomly such that Assumption 1 is satisfied. Let y_i be the value of the ith block before it was updated to $x_i^{(k,j-1)}$.
    For **IBP**: extrapolate

$$\hat{x}_i = x_i^{(k,j-1)} + \alpha_i^{(k,j)} \left(x_i^{(k,j-1)} - y_i \right), \quad (5)$$

    and compute

$$x_i^{(k,j)} = \operatorname*{argmin}_{x_i} F_i^{(k,j)}(x_i) + \frac{1}{2\beta_i^{(k,j)}} \|x_i - \hat{x}_i\|^2. \quad (6)$$

    For **IBPG**: extrapolate

$$\begin{aligned} \hat{x}_i &= x_i^{(k,j-1)} + \alpha_i^{(k,j)} \left(x_i^{(k,j-1)} - y_i \right), \\ \grave{x}_i &= x_i^{(k,j-1)} + \gamma_i^{(k,j)} \left(x_i^{(k,j-1)} - y_i \right), \end{aligned} \quad (7)$$

    and compute

$$\begin{aligned} x_i^{(k,j)} = \operatorname*{argmin}_{x_i} & \langle \nabla f_i^{(k,j)}(\grave{x}_i), x_i - x_i^{(k,j-1)} \rangle \\ & + r_i(x_i) + \frac{1}{2\beta_i^{(k,j)}} \|x_i - \hat{x}_i\|^2. \end{aligned} \quad (8)$$

    Let $x_{i'}^{(k,j)} = x_{i'}^{(k,j-1)}$ for $i' \neq i$.
  end for
  Update $\tilde{x}^{(k)} = x^{(k,T_k)}$.
end for

$$\begin{aligned} \operatorname*{argmin}_{U_{:i} \geq 0} & \sum \frac{1}{2} \Big\| X - \sum_{q=1}^{i-1} U_{:q} V_{q:} - \sum_{q=i+1}^{r} U_{:q} V_{q:} - U_{:i} V_{i:} \Big\|^2 \\ & + \frac{1}{2\beta_i} \left\| U_{:i} - \hat{U}_{:i} \right\|^2 \\ = \max & \left(0, \frac{XV_{i:}^T - (UV)V_{i:}^T + U_{:i}V_{i:}V_{i:}^T + 1/\beta_i \hat{U}_{:i}}{V_{i:}V_{i:}^T + 1/\beta_i} \right), \end{aligned}$$

and a similar update for the rows of V can be derived by symmetry since $\|X - UV\|_F^2 = \|X^T - V^TU^T\|_F^2$. For the same reason as above, IBP should update the columns of U and the rows of V several times before doing so for the other one.

2.1. Extension to Bregman Divergence

The inertial proximal steps (6) and (8) can be generalized by replacing $\|.\|$ by a Bregman divergence. Let $H_i : \mathbb{E}_i \to \mathbb{R}$ be a strictly convex function that is continuously differentiable. The Bregman distance associated with H_i is defined as:

$$D_i(u,v) = H_i(u) - H_i(v) - \langle \nabla H_i(v), u - v \rangle, \forall u, v \in \mathbb{E}_i.$$

The squared Euclidean distance $D_i(u,v) = \frac{1}{2}\|u-v\|_2^2$ corresponds to $H_i(u) = \frac{1}{2}\|u\|_2^2$.

Definition 1 *For a given $v \in \mathbb{E}_i$, and a positive number β, the Bregman proximal map of a function ϕ is defined by*

$$\operatorname{prox}^{H_i}_{\beta,\phi}(v) := \operatorname*{argmin}_{u \in \mathbb{E}_i} \left\{ \phi(u) + \frac{1}{\beta} D_i(u,v) \right\}. \quad (9)$$

Definition 2 *For given $u_1 \in \operatorname{int}\operatorname{dom} g, u_2 \in \mathbb{E}_i$ and $\beta > 0$, the Bregman proximal gradient map of a pair of functions (ϕ, g) (g is continuously differentiable) is defined by*

$$\begin{aligned} &\operatorname{Gprox}^{H_i}_{\beta,\phi,g}(u_1,u_2) \\ &:= \operatorname*{argmin}_{u \in \mathbb{E}_i} \left\{ \phi(u) + \langle \nabla g(u_1), u \rangle + \frac{1}{\beta} D_i(u,u_2) \right\} \end{aligned} \quad (10)$$

For notation succinctness, whenever the generating function is clear from the context, we omit H_i in the notation of the corresponding Bregman proximal maps. As ϕ can be non-convex, $\operatorname{prox}_{\beta,\phi}(v)$ and $\operatorname{Gprox}_{\beta,\phi,g}(u_1,u_2)$ are set-valued maps in general. Various types of assumptions can be made to guarantee their well-definedness; see Eckstein (1993), Teboulle (1997; 2018) for the well-posedness of (9), and (Bolte et al., 2018, Lemma 3.1), (Bauschke et al., 2017, Lemma 2) for the well-posedness of (10). Note that the proximal gradient maps in (Bauschke et al., 2017; Bolte et al., 2018) use the same point for evaluating the gradient and the Bregman distance while ours allow using two different points u_1 and u_2. This modification is important for our analysis; however, it does not affect the proofs of the lemmas in those papers.

Algorithm 2 describes IBP and IBPG in the framework of Bregman divergence.

Throughout this paper, we assume the following.

Assumption 2 *(A1) The function H_i, $i = 1, \ldots, s$, is σ_i-strongly convex, continuously differentiable and ∇H_i is L_{H_i}-Lipschitz continuous.*

(A2) The proximal maps (9) and (10) are well-defined.

Note that (A1) holds if H_i satisfies $L_{H_i}\mathcal{I} \preceq \nabla^2 H_i \preceq \sigma_i \mathcal{I}$. A quadratic entropy distance is a typical example of a Bregman divergence that satisfies (A1) (Reem et al., 2019). More discussion about important properties and how to evaluate (9) and (10) are given in the supplementary material.

Algorithm 2 IBP and IBPG with Bregman divergence

Initialize: Choose $\tilde{x}^{(0)} = \tilde{x}^{(-1)}$. Choose a method: IBP or IBPG. Parameters are chosen as in Section 3.
for $k = 1, \ldots$ **do**
  $x^{(k,0)} = \tilde{x}^{(k-1)}$.
  for $j = 1, \ldots, T_k$ **do**
    Choose $i \in \{1, \ldots, s\}$ deterministically or randomly such that Assumption 1 is satisfied.
    Update of IBP: extrapolate as in (5) and compute

$$x_i^{(k,j)} \in \operatorname{prox}^{H_i}_{\beta_i^{(k,j)}, F_i^{(k,j)}}(\hat{x}_i). \quad (11)$$

    Update of IBPG: extrapolate as in (7) and compute

$$x_i^{(k,j)} \in \operatorname{Gprox}^{H_i}_{\beta_i^{(k,j)}, r_i, f_i}(\hat{x}_i, \hat{x}_i). \quad (12)$$

    Let $x_{i'}^{(k,j)} = x_{i'}^{(k,j-1)}$ for $i' \neq i$.
  end for
  Update $\tilde{x}^{(k)} = x^{(k,T_k)}$.
end for

3. Subsequential Convergence

Before providing the subsequential convergence guarantees, let us elaborate on the notation, in particular $\bar{x}_i^{(k,m)}$ and d_i^k which will be used much in the upcoming analysis, see Table 1 for a summary of the notation. The elements of the sequence $x_i^{(k,j)}$ remain unchanged during many iterations since only one block is updated within each inner loop of Algorithm 2, that is, we will have $x_i^{(k,j+1)} = x_i^{(k,j)}$ for many j's. To simplify the analysis, we introduce the subsequence $\bar{x}_i^{(k,m)}$ of $x_i^{(k,j)}$ that will only record the value of the ith block when it is actually updated. More precisely, there exists a subsequence $\{i_1, i_2, \ldots, i_{d_i^k}\}$ of $\{1, 2, \ldots, T_k\}$ such that $\bar{x}_i^{(k,m)} = x_i^{(k,i_m)}$ for all $m = 1, 2, \ldots, d_i^k$. The previous value of block i before it is updated to $\bar{x}_i^{(k,m)}$ is $\bar{x}_i^{(k,m-1)}$. We have $\bar{x}_i^{(k,0)} = \bar{x}_i^{(k-1,d_i^{k-1})} = \tilde{x}_i^{(k-1)}$ and $\bar{x}_i^{(k,d_i^k)} = \tilde{x}_i^{(k)}$. As for $x_i^{(k,j)}$, we use the notation $\bar{x}_i^{(k,-1)} = \bar{x}_i^{(k-1,d_i^{k-1}-1)}$.

3.1. Choosing Parameters

We first explain how to choose the parameters for IBP and IBPG within Algorithm 2 (note that Algorithm 1 is a special case of Algorithm 2) such that their subsequential convergence is guaranteed. Let us point out that $\bar{\alpha}_i^{(k,m)}$, $\bar{\beta}_i^{(k,m)}$, and $\bar{\gamma}_i^{(k,m)}$ are the values of $\alpha_i^{(k,j)}$, $\beta_i^{(k,j)}$ and $\gamma_i^{(k,j)}$ that are used to update block i from $\bar{x}_i^{(k,m-1)}$ to $\bar{x}_i^{(k,m)}$.

Parameters for IBP Let $0 < \nu < 1, \delta > 1$. For $m = 1, \ldots, d_i^k$ and $i = 1, \ldots, s$, denote $\theta_i^{(k,m)} = \frac{\left(L_{H_i}\bar{\alpha}_i^{(k,m)}\right)^2}{2\nu\sigma_i\bar{\beta}_i^{(k,m)}}$. Let $\theta_i^{(k,d_i^k+1)} = \theta_i^{(k+1,1)}$. We choose $\bar{\alpha}_i^{(k,m)}$ and $\bar{\beta}_i^{(k,m)}$ such that, for $m = 1, \ldots, d_i^k$,

$$\frac{(1-\nu)\sigma_i}{2\bar{\beta}_i^{(k,m)}} \geq \delta\theta_i^{(k,m+1)}. \tag{13}$$

Parameters for IBPG Considering IBPG, we need to assume that $\nabla f_i^{(k,j)}$ is $L_i^{(k,j)}$-Lipschitz continuous, with $L_i^{(k,j)} > 0$. For notational clarity, we correspondingly use $\bar{L}_i^{(k,m)}$ for $L_i^{(k,j)}$ when updating block i from $\bar{x}_i^{(k,m-1)}$ to $\bar{x}_i^{(k,m)}$. To simplify the upcoming analysis, we choose $\bar{\beta}_i^{(k,m)} = \frac{\sigma_i}{\kappa\bar{L}_i^{(k,m)}}$ with $\kappa > 1$. Let $0 < \nu < 1, \delta > 1$. Denote

$$\lambda_i^{(k,m)} = \frac{1}{2}\left(\bar{\gamma}_i^{(k,m)} + \frac{\kappa L_{H_i}\bar{\alpha}_i^{(k,m)}}{\sigma_i}\right)^2 \frac{\bar{L}_i^{(k,m)}}{\nu(\kappa-1)},$$

for $m = 1, \ldots, d_i^k$ and $i = 1, \ldots, s$. Let $\lambda_i^{(k,d_i^k+1)} = \lambda_i^{(k+1,1)}$. We choose $\bar{\alpha}_i^{(k,m)}$, $\bar{\beta}_i^{(k,m)}$ and $\bar{\gamma}_i^{(k,m)}$ such that, for $m = 1, \ldots, d_i^k$,

$$\frac{(1-\nu)(\kappa-1)\bar{L}_i^{(k,m)}}{2} \geq \delta\lambda_i^{(k,m+1)}. \tag{14}$$

We make the following standard assumption for the boundedness of the parameters; see (Xu & Yin, 2013, Assumption 2), (Bolte et al., 2014, Assumption 2).

Assumption 3 *For IBP, there exist positive numbers W_1, $\overline{\alpha}$ and $\underline{\beta}$ such that $\theta_i^{(k,m)} \geq W_1$, $\bar{\alpha}_i^{(k,m)} \leq \overline{\alpha}$ and $\underline{\beta} \leq \bar{\beta}_i^{(k,m)}$, $\forall\, k \in \mathbb{N}$, $m = 1, \ldots, d_i^k$, $i = 1, \ldots, s$.*

For IBPG, there exist positive numbers W_1, $\overline{L} > 0$, $\overline{\alpha}$ and $\overline{\gamma}$ such that $\lambda_i^{(k,m)} \geq W_1$, $\bar{L}_i^{(k,m)} \leq \overline{L}$, $\bar{\alpha}_i^{(k,m)} \leq \overline{\alpha}$ and $\bar{\gamma}_i^{(k,m)} \leq \overline{\gamma}$ for all $k \in \mathbb{N}$, $m = 1, \ldots, d_i^k$ and $i = 1, \ldots, s$.

The algorithm iPALM of Pock & Sabach (2016) is a special case of IBPG when D is the Euclidean distance, $s = 2$ and the two blocks are cyclically updated; however, our chosen parameters are different. In particular, the stepsize $\bar{\beta}_i^{(k,m)}$ of iPALM depends on the inertial parameters (Pock & Sabach, 2016, Formula 4.9), while we choose $\bar{\beta}_i^{(k,m)}$ independently of $\bar{\alpha}_i^{(k,m)}$ and $\bar{\gamma}_i^{(k,m)}$. Our parameters allow using dynamic inertial parameters (see Section 3.3). As also experimentally tested by (Pock & Sabach, 2016), choosing the inertial parameters dynamically leads to a significant improvement of the algorithm performance. The analysis by Pock & Sabach (2016) does not support this choice of parameters, while ours guarantee subsequential convergence.

3.2. Subsequential Convergence Theory

The following proposition serves as a cornerstone to prove the subsequential convergence.

Proposition 1 *Let $\{\tilde{x}^{(k)}\}$ be a sequence generated by Algorithm 2, and consider $\left\{\tilde{x}_{\text{prev}}^{(k)}\right\}$ with $\left(\tilde{x}_{\text{prev}}^{(k)}\right)_i = \bar{x}_i^{(k,d_i^k-1)}$. Suppose Assumption 1, 2 and 3 are satisfied.*

(i) We have $\sum_{k=1}^{\infty}\left\|\tilde{x}^{(k)} - \tilde{x}_{\text{prev}}^{(k)}\right\|^2 < \infty$ and $\sum_{k=1}^{\infty}\sum_{i=1}^{s}\sum_{m=1}^{d_i^k}\left\|\bar{x}_i^{(k,m)} - \bar{x}_i^{(k,m-1)}\right\|^2 < \infty$.

(ii) If there exists a limit point x^* of $\left\{\tilde{x}^{(k)}\right\}$ (that is, there exists a subsequence $\left\{\tilde{x}^{(k_n)}\right\}$ converging to x^*), then we have $\lim_{n\to\infty} r_i\left(\bar{x}_i^{(k_n,m)}\right) = r_i\left(x_i^*\right)$.

Remark 1 (Relax (13) for block-convex F) *For IBP, if F is block-wise convex then we can choose $\bar{\alpha}_i^{(k,m)}$ and $\bar{\beta}_i^{(k,m)}$ satisfying*

$$\frac{2(1-\nu)\sigma_i}{\bar{\beta}_i^{(k,m)}} \geq \delta\theta_i^{(k,m+1)}, \quad \text{for } m = 1, \ldots, d_i^k, \tag{15}$$

and Proposition 1 still holds. Compared to (13), Condition (15) allows larger values of the extrapolation parameters $\bar{\alpha}_i^{(k,m)}$ when using the same stepsize $\bar{\beta}_i^{(k,m)}$.

Remark 2 (Relax (14) for convex r_i's) *If the functions r_i's are convex (note that f is not necessary block-wise convex) then we can use a larger stepsize. Specifically, we can use $\bar{\beta}_i^{(k,m)} = \sigma_i/\bar{L}_i^{(k,m)}$ and*

$$\lambda_i^{(k,m)} = \frac{1}{2}\left(\bar{\gamma}_i^{(k,m)} + \frac{L_{H_i}\bar{\alpha}_i^{(k,m)}}{\sigma_i}\right)^2 \frac{\bar{L}_i^{(k,m)}}{\nu},$$

and choose $\bar{\alpha}_i^{(k,m)}$ and $\bar{\gamma}_i^{(k,m)}$ satisfying

$$\frac{(1-\nu)\bar{L}_i^{(k,m)}}{2} \geq \delta\lambda_i^{(k,m+1)}, \text{for } m = 1, \ldots, d_i^k, \tag{16}$$

and Proposition 1 still holds.

Remark 3 (Relax (14) for block-convex f and convex r_i's) *If the r_i's are convex and $f(x)$ is block-wise convex, then we can use larger extrapolation parameters. Specifically, we choose $H_i(x_i) = \frac{1}{2}\|x_i\|^2$ and let $\bar{\beta}_i^{(k,m)} = 1/\bar{L}_i^{(k,m)}$ and*

$$\lambda_i^{(k,m)} = \left(\left(\bar{\gamma}_i^{(k,m)}\right)^2 + \frac{\left(\bar{\gamma}_i^{(k,m)} - \bar{\alpha}_i^{(k,m)}\right)^2}{\nu}\right)\frac{\bar{L}_i^{(k,m)}}{2},$$

where $0 < \nu < 1$, and choose $\bar{\alpha}_i^{(k,m)}$ and $\bar{\gamma}_i^{(k,m)}$ satisfying

$$\frac{1-\nu}{2}\bar{L}_i^{(k,m)} \geq \delta\lambda_i^{(k,m+1)}, \text{for } m = 1, \ldots, d_i^k.$$

For these values, Proposition 1 still holds. In Section 5 we numerically show that choosing $\bar{\gamma}_i^{(k,m)} \neq \bar{\alpha}_i^{(k,m)}$ can significantly improve the performance of the algorithm.

We now state the local convergence result. The definitions of critical points can be found in the supplementary material.

Theorem 1 *Suppose Assumption 1, 2 and 3 are satisfied.*

(i) For IBP, if F is regular then every limit point of the sequence $\{\tilde{x}^{(k)}\}$ generated by Algorithm 2 is a critical point type I of F. If f is continuously differentiable then every limit point is a critical point type II of F.

(ii) For IBPG, every limit point of the sequence $\{\tilde{x}^{(k)}\}$ generated by Algorithm 2 is a critical point type II of F.

3.3. Choice of Parameters for NMF

Let us illustrate the choice of parameters for NMF. In the remainder of this paper, in the context of NMF, we will refer to IBPG as Algoritm 1 with the choice $T_k = 2$ (cyclic update of U and V), and to IBPG-A with the choice $T_k > 2$ (U and V are updated several times). IBPG-A is expected to be more efficient; see the discussion in Section 2. For IBPG and IBPG-A, we take $\bar{L}_1^{(k,m)} = \tilde{L}_1^{(k)} = \left\|(\tilde{V}^{(k-1)})^T\tilde{V}^{(k-1)}\right\|$ and $\bar{L}_2^{(k,m)} = \tilde{L}_2^{(k)} = \left\|(\tilde{U}^{(k)})^T\tilde{U}^{(k)}\right\|$ for $m \geq 1$. We take $\bar{\beta}_i^{(k,m)} = 1/\tilde{L}_i^{(k)}$, $\bar{\gamma}_i^{(k,m)} = \min\left\{\frac{\tau_k - 1}{\tau_k}, \tilde{\gamma}\sqrt{\frac{\tilde{L}_i^{(k-1)}}{\tilde{L}_i^{(k)}}}\right\}$ and $\bar{\alpha}_i^{(k,m)} = \breve{\alpha}\bar{\gamma}_i^{(k,m)}$, where $\tau_0 = 1$, $\tau_k = \frac{1}{2}(1 + \sqrt{1 + 4\tau_{k-1}^2})$, $\tilde{\gamma} = 0.99$ and $\breve{\alpha} = 1.01$. We can verify that there exists $\delta > 1$ such that $\breve{\gamma}^2\left((\breve{\alpha} - 1)^2/\nu + 1\right) < (1-\nu)/\delta$ with $\nu = 0.0099$. Hence, our choice of parameters satisfy the conditions of Remark 3.

Regarding IBP, we choose $1/\beta_i^{(k,m)} = 0.001$ and $\alpha_i^{(k,m)} = \tilde{\alpha}^{(k)} = \min(\bar{\beta}, \gamma\tilde{\alpha}^{(k-1)})$, with $\bar{\beta} = 1$, $\gamma = 1.01$ and $\tilde{\alpha}^{(1)} = 0.6$. This choice of parameters satisfies the conditions of Remark 1.

4. Global Convergence

A key tool of the upcoming global convergence (i.e., the whole sequence converges to a critical point) analysis is the Kurdyka-Łojasiewicz (KL) function defined as follows.

Definition 3 *A function $\phi(x)$ is said to have the KL property at $\bar{x} \in \operatorname{dom}\partial\,\phi$ if there exists $\eta \in (0, +\infty]$, a neighborhood U of $\bar{x}$ and a concave function $\xi : [0, \eta) \to \mathbb{R}_+$ that is continuously differentiable on $(0, \eta)$, continuous at 0, $\xi(0) = 0$, and $\xi'(s) > 0$ for all $s \in (0, \eta)$, such that for all $x \in U \cap [\phi(\bar{x}) < \phi(x) < \phi(\bar{x}) + \eta]$, we have*

$$\xi'\left(\phi(x) - \phi(\bar{x})\right)\operatorname{dist}\left(0, \partial\phi(x)\right) \geq 1. \tag{17}$$

$\operatorname{dist}\left(0, \partial\phi(x)\right) = \min\{\|y\| : y \in \partial\phi(x)\}$. If $\phi(x)$ has the KL property at each point of $\operatorname{dom}\partial\phi$ then ϕ is a KL function.

The class of KL functions is rich enough to cover many non-convex non-smooth functions found in practical applications. Some noticeable examples are real analytic functions, semi-algebraic functions, and locally strongly convex functions (Bochnak et al., 1998; Bolte et al., 2014).

4.1. Global Convergence Recipe

Attouch et al. (2010; 2013) and Bolte et al. (2014) were the first to prove the global convergence of proximal point algorithms for solving non-convex non-smooth problems. We note that a direct deployment of the methodology to our proposed algorithms is not possible since the relaxation property does not hold (that is, the objective functions are not monotonically decreasing) and our methods allow for a randomized strategy. In the following theorem, we modify the proof recipe of Bolte et al. (2014) so that it is applicable to our proposed methods.

Theorem 2 *Let $\Phi : \mathbb{R}^N \to (-\infty, +\infty]$ be a proper and lower semicontinuous function which is bounded from below. Let $\mathcal{A}$ be a generic algorithm which generates a bounded sequence $\{z^{(k)}\}$ by $z^{(0)} \in \mathbb{R}^N$, $z^{(k+1)} \in \mathcal{A}(z^{(k)})$, $k = 0, 1, \ldots$ Assume that there exist positive constants ρ_1, ρ_2 and ρ_3 and a non-negative sequence $\{\zeta_k\}_{k\in\mathbb{N}}$ such that the following conditions are satisfied*

(B1) ***Sufficient decrease property****:*

$$\rho_1\|z^{(k)} - z^{(k+1)}\|^2 \leq \rho_2\zeta_k^2 \leq \Phi(z^{(k)}) - \Phi(z^{(k+1)}),$$

$$\forall\, k = 0, 1, \ldots$$

(B2) ***Boundedness of subgradient****:*

$$\|w^{(k+1)}\| \leq \rho_3\zeta_k, w^{(k)} \in \partial\Phi(z^{(k)}), \forall k = 0, 1, \ldots$$

(B3) ***KL property****: Φ is a KL function.*

(B4) ***A continuity condition****: If a subsequence $\{z^{(k_n)}\}$ converges to $\bar{z}$ then $\Phi(z^{(k_n)}) \to \Phi(\bar{z})$ as $n \to \infty$.*

Then we have $\sum_{k=1}^{\infty}\zeta_k < \infty$ and $\{z^{(k)}\}$ converges to a critical point of Φ.

We remark that if we take $\zeta_k = \|z^{(k)} - z^{(k+1)}\|$ then Theorem 2 recovers the proof recipe of Bolte et al. (2014). The following theorem establish the convergence rate under Łojasiewicz property.

Theorem 3 *Suppose Φ is a KL function and $\xi(a)$ of Definition 3 has the form $\xi(a) = Ca^{1-\omega}$ for some $C > 0$ and $\omega \in [0, 1)$. Then we have*

(i) If $\omega = 0$ then $\{z^{(k)}\}$ converges after a finite number of steps.

(ii) If $\omega \in (0, 1/2]$ then there exists $\omega_1 > 0$ and $\omega_2 \in [0, 1)$ such that $\left\|z^{(k)} - \bar{z}\right\| \leq \omega_1\omega_2^k$.

(iii) If $\omega \in (1/2, 1)$ then there exists $\omega_1 > 0$ such that $\left\|z^{(k)} - \bar{z}\right\| \leq \omega_1 k^{-(1-\omega)/(2\omega-1)}$.

4.2. Global Convergence of IBP and IBPG

We need the use of the following auxiliary function

$$\Psi(\acute{y}, \breve{y}) := F(\acute{y}) + \rho D(\acute{y}, \breve{y}),$$

where $\rho > 0$ and $D(\acute{y}, \breve{y}) = \sum_{i=1}^{s} D_i(\acute{y}_i, \breve{y}_i)$. Recall that $(\tilde{x}_{\text{prev}}^{(k)})_i = \bar{x}_i^{(k,d_i^k-1)}$. Let us consider the sequence $\{Y^{(k)}\}$ with $Y^{(k)} = \left(\acute{y}^{(k)}, \breve{y}^{(k)}\right) = \left(\tilde{x}^{(k)}, \tilde{x}_{\text{prev}}^{(k)}\right)$. We then have

$$\Psi(Y^{(k)}) = F(\tilde{x}^{(k)}) + \rho D(\tilde{x}^{(k)}, \tilde{x}_{\text{prev}}^{(k)}), \qquad (18)$$

$$\|Y^{(k)} - Y^{(k+1)}\|^2 = \|\tilde{x}^{(k)} - \tilde{x}^{(k+1)}\|^2 + \|\tilde{x}_{\text{prev}}^{(k)} - \tilde{x}_{\text{prev}}^{(k+1)}\|^2.$$

We define

$$\begin{aligned}\varphi_k^2 &:= \sum_{i=1}^{s} \sum_{m=0}^{d_i^{(k+1)}} \|\bar{x}_i^{(k+1,m)} - \bar{x}_i^{(k+1,m-1)}\|^2 \\ &= \sum_{i=1}^{s} \sum_{m=1}^{d_i^{(k+1)}} \|\bar{x}_i^{(k+1,m)} - \bar{x}_i^{(k+1,m-1)}\|^2 \\ &\quad + \|\tilde{x}^{(k)} - \tilde{x}_{\text{prev}}^{(k)}\|^2.\end{aligned}$$

We make the following additional assumption.

Assumption 4 *The sequences $\left\{\tilde{x}^{(k)}\right\}_{k\in\mathbb{N}}$ generated by Algorithm 2 are bounded.*

In Proposition 2 we will prove that $\Psi(Y^{(k)})$ is non-increasing; thus, $\Psi(Y^{(k)})$ is upper bounded by $\Psi(Y^{(-1)})$. Moreover, note that $D(\tilde{x}^{(k)}, \tilde{x}_{\text{prev}}^{(k)}) \geq 0$. Hence, from (18) this implies that $F\left(\tilde{x}^{(k)}\right)$ is also upper bounded by $\Psi\left(Y^{(-1)}\right)$. Therefore, we can say that Assumption 4 is satisfied when F has bounded level sets. Denote $\sigma = \min\{\sigma_1, \ldots, \sigma_s\}$ and $L_H = \max\{L_{H_1}, \ldots, L_{H_s}\}$.

The following proposition gives an upper bound for the subgradients and a sufficient decrease property for $\left\{\Psi\left(Y^{(k)}\right)\right\}$.

Proposition 2 *Suppose Assumption 1, 2, 3 and 4 hold.*

(i) Suppose f is continuously differentiable and ∇f is Lipschitz continuous on bounded subsets of $\mathbb{E}$ (this is a standard assumption, see (Xu & Yin, 2013, Lemma 2.6), (Bolte et al., 2014, Assumption 2 iv)). We have $\|\hat{q}^{(k+1)}\| = O(\varphi_k)$ for some $\hat{q}^{(k)} \in \partial\Psi\left(Y^{(k)}\right)$.

(ii) Together with the condition in Proposition 2 (i), assume that there exists a constant W_2 such that $\forall\, k \in \mathbb{N}$, $m = 1, \ldots, d_i^k$ and $i = 1, \ldots, s$, we have $\theta_i^{(k,m)} \leq W_2$ for IBP, $\lambda_i^{(k,m)} \leq W_2$ for IBPG and $\delta > (L_H W_2)/(\sigma W_1)$. Let $\rho = \frac{\delta W_1}{L_H} + \frac{W_2}{\sigma}$ in (18) and let $\rho_2 = \frac{\delta\sigma W_1}{2L_H} - \frac{W_2}{2}$. Then

$$\Psi(Y^{(k)}) - \Psi(Y^{(k+1)}) \geq \rho_2 \varphi_k^2.$$

We are now ready to state our global convergence result.

Theorem 4 *Assume F is a KL-function and the conditions of Proposition 2 are satisfied. Then the whole sequence $\{\tilde{x}^{(k)}\}$ generated by IBP or IBPG converges to a critical point type II of F.*

We note that $\left\|Y^{(k)} - Y^*\right\| \geq \left\|\tilde{x}^{(k)} - x^*\right\|$, hence the convergence rate of the sequence $\{\tilde{x}^{(k)}\}$ is at least the same order as the rate of $\{Y^{(k)}\}$. If Ψ is a KL function with $\xi(a) = Ca^{1-\omega}$, then we can apply Theorem 3 to derive the convergence rate of $\{Y^{(k)}\}$.

Remark 4 *Note that we need the additional condition $\delta > \frac{L_H W_2}{\sigma W_1}$ in order to obtain the global convergence in Theorem 4. Therefore, it makes sense to show that there exists δ such that Condition (14) for IBPG (or Condition (13) for IBP) is also satisfied. See the supplementary material for the proof.*

The parameters of IBP for NMF in Section 3.3 satisfy the conditions for global convergence.

5. Numerical Results for NMF

In this section, we compare our IBP, IBPG and IBPG-A (see Section 3.3) with the following NMF algorithms:

+ A-HALS: the accelerated hierarchical alternating least squares algorithm (Gillis & Glineur, 2012). A-HALS outperforms standard projected gradient, the popular multiplicative updates and alternating non-negative least squares (Kim et al., 2014; Gillis, 2014).

+ E-A-HALS: the acceleration version of A-HALS proposed by Ang & Gillis (2019). This algorithm was experimentally shown to outperform A-HALS. This is, as far as we know, one of the most efficient NMF algorithms. Note that E-A-HALS is a heuristic with no convergence guarantees.

+ APGC: the accelerated proximal gradient coordinate descent method proposed by Xu & Yin (2013) which corresponds exactly to IBPG with $\tilde{\gamma} = \breve{\alpha} = 0.9999$.

+ iPALM: the inertial proximal alternating linearized minimization method proposed by Pock & Sabach (2016).

We define the relative errors $\text{relerror}_k = \frac{\|X - \tilde{U}^{(k)}\tilde{V}^{(k)}\|_F}{\|X\|_F}$. We let $e_{\min} = 0$ for the experiments with low-rank synthetic data sets, and in the other experiments $e_{\min}$ is the lowest relative error obtained by any algorithms with any initializations. We define $E(k) = \text{relerror}_k - e_{\min}$. These are the same settings as in (Gillis & Glineur, 2012). All tests are preformed using Matlab R2015a on a laptop Intel CORE i7-8550U CPU @1.8GHz 16GB RAM. The code is available at https://github.com/LeThiKhanhHien/IBPG

Experiments with synthetic data sets. Two low-rank matrices of size 200×200 and 200×500 are generated by letting $X = UV$, where U and V are generated by commands $rand(\mathbf{m}, \mathbf{r})$ and $rand(\mathbf{r}, \mathbf{n})$ with $\mathbf{r} = 20$. For

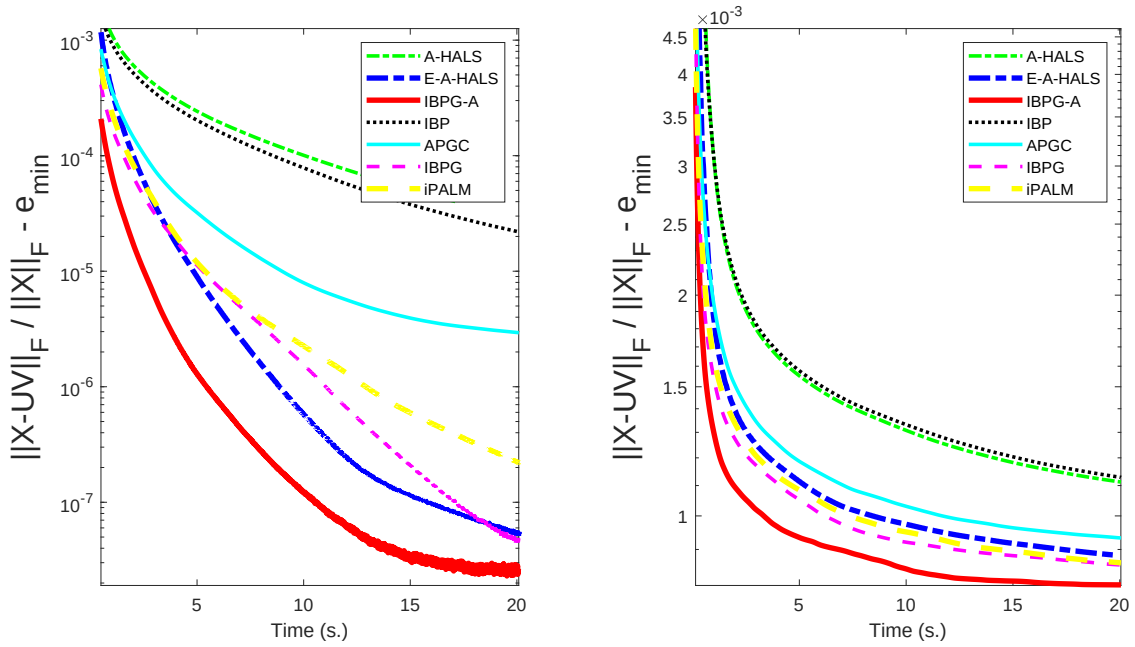

Figure 1. Average value of $E(k)$ with respect to time on 2 random low-rank matrices: 200×200 (the left) and 200×500 (the right).

each X, we run all algorithms with the same 50 random initializations $U_0 = rand(\mathbf{m}, \mathbf{r})$, $V_0 = rand(\mathbf{r}, \mathbf{n})$, and for each initialization we run each algorithm for 20 seconds. Figure 1 illustrates the evolution of the average of $E(k)$ over 50 initializations with respect to time.

To compare the accuracy of the solutions, we generate 80 random $\mathbf{m} \times \mathbf{n}$ matrices, $\mathbf{m}$ and $\mathbf{n}$ are random integer numbers in the interval [200,500]. For each X we run the algorithms for 20 seconds with 1 random initialization. Table 2 reports the average and standard deviation (std) of the errors. It also provides a ranking between the different algorithms: the ith entry of the ranking vector indicates how many times the corresponding algorithm obtained the ith best solution.

We observe that (i) in terms of convergence speed and the final errors obtained, IBPG-A outperforms the other algorithms, and (ii) APGC and iPALM converge slower than IBPG and APGC produces worse solutions. This illustrates the fact that using two extrapolated points may lead to a faster convergence.

Experiments with real data sets. In these experiments, we will only keep the best performing algorithms, namely IBPG-A and E-A-HALS, along with APGC for our observation purpose. For each data set, we generate 35 random initializations and for each initialization we run each algorithm for 200 seconds. We test the algorithms on two widely used hyperspectral images, namely the Urban and San Diego data sets; see (Gillis et al., 2015). We let $\mathbf{r} = 10$.

Figure 2 reports the evolution of the average value of $E(k)$,

Table 2. Average, standard deviation and ranking of the value of $E(k)$ at the last iteration among the different runs on the low-rank synthetic data sets. The best performance is highlighed in bold.

Algorithm	mean $\pm$ std	ranking
A-HALS	$1.227\,10^{-3} \pm 7.365\,10^{-4}$	(1, 0, 3, 4, 7, 24, 41)
E-A-HALS	$8.501\,10^{-4} \pm 6.882\,10^{-4}$	(16, 10, 12, 13, 17, 3, 9)
IBPG-A	$\mathbf{5.036\,10^{-4}} \pm 5.522\,10^{-4}$	(**39**, 10, 14, 10, 3, 2, 2)
IPG	$1.209\,10^{-3} \pm 7.386\,10^{-4}$	(0, 3, 5, 7, 15, 39, 11)
APGC	$8.726\,10^{-4} \pm 6.561\,10^{-4}$	(3, 10, 14, 22, 18, 3, 10)
IBPG	$6.621\,10^{-4} \pm 6.371\,10^{-4}$	(17, 17, 15, 11, 14, 2, 4)
iPALM	$6.759\,10^{-4} \pm 6.302\,10^{-4}$	(17, 22, 13, 12, 6, 7, 3)

Table 3. Average error, standard deviation and ranking among the different runs for urban and SanDiego data sets.

Algorithm	mean $\pm$ std	ranking
E-A-HALS	$0.018823 \pm 6.739\,10^{-4}$	(17, 28, 25)
IBPG-A	$\mathbf{0.018316} \pm 9.745\,10^{-4}$	(**53**, 15, 2)
APGC	$0.018728 \pm 7.779\,10^{-4}$	(0, 27, 43)

and Table 3 reports the average error, standard deviation and ranking of the final value of $E(k)$ among the 70 runs (2 data sets with 35 initializations for each data set).

We see that IBPG-A outperforms E-A-HALS and APGC both in terms of convergence speed and accuracy.

6. Conclusion

We have analysed inertial versions of proximal BCD and proximal gradient BCD methods for solving non-convex non-smooth composite optimization problems. Our methods do not require restarting steps, and allow the use of

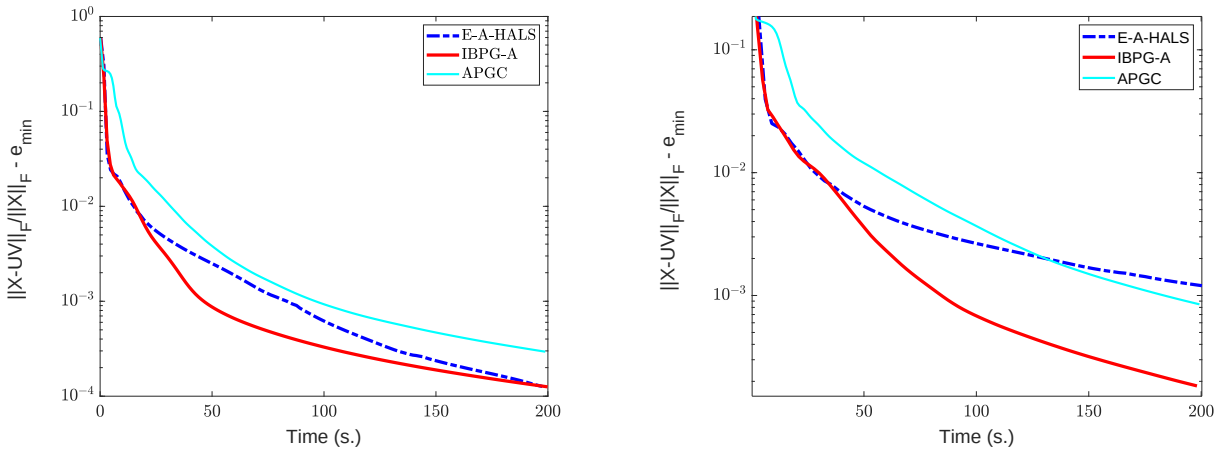

Figure 2. Average value of $E(k)$ with respect to time on 2 hyperspectral images: urban (the left) and SanDiego (the right).

randomized strategies and of two extrapolation points. We first proved sub-sequential convergence of the generated sequence to a critical point of F (Theorem 1) and then, under some additional assumptions, convergence of the whole sequence (Theorem 4). We showed that the proposed methods compared favourably with state-of-the-art algorithms for NMF. Additional experiments on NMF and NCPD are given in the supplementary material. Exploring other Bregman divergences for IBP and IBPG to solve NMF and NCPD may lead to other efficient algorithms for NMF and NCPD. This is one of our future research directions.

References

Aharon, M., Elad, M., Bruckstein, A., et al. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311, 2006.

Ang, A. M. S. and Gillis, N. Accelerating nonnegative matrix factorization algorithms using extrapolation. *Neural Computation*, 31(2):417–439, 2019.

Attouch, H., Bolte, J., Redont, P., and Soubeyran, A. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010. doi: 10.1287/moor.1100.0449. URL https://doi.org/10.1287/moor.1100.0449.

Attouch, H., Bolte, J., and Svaiter, B. F. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. *Mathematical Programming*, 137(1):91–129, Feb 2013.

Auslender, A. Asymptotic properties of the Fenchel dual functional and applications to decomposition problems. *Journal of Optimization Theory and Applications*, 73(3): 427–449, Jun 1992. ISSN 1573-2878. doi: 10.1007/BF00940050. URL https://doi.org/10.1007/BF00940050.

Bauschke, H. H., Bolte, J., and Teboulle, M. A descent lemma beyond Lipschitz gradient continuity: First-order methods revisited and applications. *Mathematics of Operations Research*, 42(2):330–348, 2017. doi: 10.1287/moor.2016.0817. URL https://doi.org/10.1287/moor.2016.0817.

Blumensath, T. and Davies, M. E. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265 – 274, 2009. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2009.04.002. URL http://www.sciencedirect.com/science/article/pii/S1063520309000384.

Bochnak, J., Coste, M., and Roy, M.-F. *Real Algebraic Geometry*. Springer, 1998.

Bolte, J., Sabach, S., and Teboulle, M. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1):459–494, Aug 2014.

Bolte, J., Sabach, S., Teboulle, M., and Vaisbourd, Y. First order methods beyond convexity and lipschitz gradient continuity with applications to quadratic inverse problems. *SIAM Journal on Optimization*, 28(3):2131–2151, 2018. doi: 10.1137/17M1138558. URL https://doi.org/10.1137/17M1138558.

Boţ, R. I. and Csetnek, E. R. An inertial Tseng's type proximal algorithm for nonsmooth and nonconvex optimization problems. *Journal of Optimization Theory and*

Applications, 171(2):600–616, Nov 2016. ISSN 1573-2878. doi: 10.1007/s10957-015-0730-z.

Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

Eckstein, J. Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming. *Mathematics of Operations Research*, 18 (1):202–226, 1993. doi: 10.1287/moor.18.1.202. URL https://doi.org/10.1287/moor.18.1.202.

Fercoq, O. and Richtarik, P. Accelerated, parallel, and proximal coordinate descent. *SIAM J. Optim.*, 25(4): 1997–2023, 2015.

Fu, X., Huang, K., Sidiropoulos, N. D., and Ma, W.-K. Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications. *IEEE Signal Process. Mag.*, 36(2):59–80, 2019.

Gillis, N. The why and how of nonnegative matrix factorization. *Regularization, Optimization, Kernels, and Support Vector Machines*, 12(257):257–291, 2014.

Gillis, N. and Glineur, F. Accelerated multiplicative updates and hierarchical als algorithms for nonnegative matrix factorization. *Neural Computation*, 24(4):1085–1105, 2012.

Gillis, N., Kuang, D., and Park, H. Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization. *IEEE Transactions on Geoscience and Remote Sensing*, 53(4):2066–2078, 2015.

Grippo, L. and Sciandrone, M. On the convergence of the block nonlinear gauss–seidel method under convex constraints. *Operations Research Letters*, 26(3):127 – 136, 2000. ISSN 0167-6377. doi: https://doi.org/10.1016/S0167-6377(99)00074-7. URL http://www.sciencedirect.com/science/article/pii/S0167637799000747.

Hildreth, C. A quadratic programming procedure. *Naval Research Logistics Quarterly*, 4(1):79–85, 1957. doi: 10.1002/nav.3800040113. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800040113.

Kim, J., He, Y., and Park, H. Algorithms for nonnegative matrix and tensor factorizations: A unified view based on block coordinate descent framework. *Journal of Global Optimization*, 58(2):285–319, 2014.

Natarajan, B. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995. doi: 10.1137/S0097539792240406. URL https://doi.org/10.1137/S0097539792240406.

Nesterov, Y. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2), 1983.

Nesterov, Y. On an approach to the construction of optimal methods of minimization of smooth convex functions. *Ekonom. i. Mat. Metody*, 24:509–517, 1998.

Nesterov, Y. *Introductory lectures on convex optimization: A basic course*. Kluwer Academic Publ., 2004.

Nesterov, Y. Smooth minimization of non-smooth functions. *Math. Prog.*, 103(1):127–152, 2005.

Nesterov, Y. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012. doi: 10.1137/100802001. URL https://doi.org/10.1137/100802001.

Ochs, P. Unifying abstract inexact convergence theorems and block coordinate variable metric ipiano. *SIAM Journal on Optimization*, 29(1):541–570, 2019. doi: 10.1137/17M1124085. URL https://doi.org/10.1137/17M1124085.

Ochs, P., Chen, Y., Brox, T., and Pock, T. iPiano: Inertial proximal algorithm for nonconvex optimization. *SIAM Journal on Imaging Sciences*, 7(2):1388–1419, 2014. doi: 10.1137/130942954. URL https://doi.org/10.1137/130942954.

Pock, T. and Sabach, S. Inertial proximal alternating linearized minimization (iPALM) for nonconvex and nonsmooth problems. *SIAM Journal on Imaging Sciences*, 9 (4):1756–1787, 2016. doi: 10.1137/16M1064064. URL https://doi.org/10.1137/16M1064064.

Polyak, B. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1 – 17, 1964. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(64)90137-5. URL http://www.sciencedirect.com/science/article/pii/0041555364901375.

Razaviyayn, M., Hong, M., and Luo, Z. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23(2):1126–1153, 2013. doi: 10.1137/120891009.

Reem, D., Reich, S., and Pierro, A. D. Re-examination of Bregman functions and new properties of their divergences. *Optimization*, 68(1):279–348, 2019. doi: 10.1080/02331934.2018.1543295. URL https://doi.org/10.1080/02331934.2018.1543295.

Teboulle, M. Convergence of proximal-like algorithms. *SIAM J. Optim.*, 7(4):1069–1083, 1997.

Teboulle, M. A simplified view of first order methods for optimization. *Mathematical Programming*, 170(1): 67–96, Jul 2018. ISSN 1436-4646. doi: 10.1007/s10107-018-1284-2. URL https://doi.org/10.1007/s10107-018-1284-2.

Tseng, P. and Yun, S. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1):387–423, Mar 2009.

Xu, Y. and Yin, W. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013. doi: 10.1137/120887795. URL https://doi.org/10.1137/120887795.

Xu, Y. and Yin, W. A fast patch-dictionary method for whole image recovery. *Inverse Problems & Imaging*, 10: 563, 2016. ISSN 1930-8337. doi: 10.3934/ipi.2016012.

Xu, Y. and Yin, W. A globally convergent algorithm for nonconvex optimization based on block coordinate update. *Journal of Scientific Computing*, 72(2):700–734, Aug 2017.

Zavriev, S. and Kostyuk, F. Heavy-ball method in nonconvex optimization problems. *Computational Mathematics and Modeling*, 1993.