Inertial Proximal Alternating Linearized Minimization (iPALM) for Nonconvex and Nonsmooth Problems

Thomas Pock[*] Shoham Sabach[†]

Abstract

In this paper we study nonconvex and nonsmooth optimization problems with semi-algebraic data, where the variables vector is split into several blocks of variables. The problem consists of one smooth function of the entire variables vector and the sum of nonsmooth functions for each block separately. We analyze an inertial version of the Proximal Alternating Linearized Minimization (PALM) algorithm and prove its global convergence to a critical point of the objective function at hand. We illustrate our theoretical findings by presenting numerical experiments on blind image deconvolution, on sparse non-negative matrix factorization and on dictionary learning, which demonstrate the viability and effectiveness of the proposed method.

Key words: Alternating minimization, blind image deconvolution, block coordinate descent, heavy-ball method, Kurdyka-Łojasiewicz property, nonconvex and nonsmooth minimization, sparse non-negative matrix factorization, dictionary learning.

1 Introduction

In the last decades advances in convex optimization have significantly influenced scientific fields such as image processing and machine learning, which are dominated by computational approaches. However, it is also known that the framework of convexity is often too restrictive to provide good models for many practical problems. Several basic problems such as blind image deconvolution are inherently nonconvex and hence there is a vital interest in the development of efficient and simple algorithms for tackling nonconvex optimization problems.

[*]Institute of Computer Graphics and Vision, Graz University of Technology, 8010 Graz, Austria. Digital Safety & Security Department, AIT Austrian Institute of Technology GmbH, 1220 Vienna, Austria. E-mail: `pock@icg.tugraz.at`. Thomas Pock acknowledges support from the Austrian science fund (FWF) under the project EANOI, No. I1148 and the ERC starting grant HOMOVIS, No. 640156.

[†]Department of Industrial Engineering and Management, Technion—Israel Institute of Technology, Haifa 3200003, Israel. E-mail: ssabach@ie.technion.ac.il

A large part of the optimization community has also been devoted to the development of general purpose solvers [23], but in the age of big data, such algorithms often come to their limits since they cannot efficiently exploit the structure of the problem at hand. One notable exception is the general purpose limited quasi Newton method [18] which has been published more than 25 years ago but still remains a competitive method.

A promising approach to tackle nonconvex problems is to consider a very rich the class of problems which share certain structure that allows the development of efficient algorithms. One such class of nonconvex optimization problems is given by the sum of three functions:

$$\min_{\mathbf{x}=(x_1,x_2)} F\left(\mathbf{x}\right) := f_1\left(x_1\right) + f_2\left(x_2\right) + H\left(\mathbf{x}\right), \tag{1.1}$$

where f_1 and f_2 are assumed to be general nonsmooth and nonconvex functions with efficiently computable proximal mappings (see exact definition in the next section) and H is a smooth coupling function which is required to have only partial Lipschitz continuous gradients $\nabla_{x_1} H$ and $\nabla_{x_2} H$ (it should be noted that $\nabla_{\mathbf{x}} H$ might not be a Lipschitz continuous).

Many practical problems frequently used in the machine learning and image processing communities fall into this class of problems. Let us briefly mention two classical examples (another example will be discussed in Section 5).

The first example is Non-negative Matrix Factorization (NMF) [26, 15]. Given a non-negative data matrix $A \in \mathbb{R}_+^{m\times n}$ and an integer $r > 0$, the idea is to approximate the matrix A by a product of again non-negative matrices BC, where $B \in \mathbb{R}_+^{m\times r}$ and $C \in \mathbb{R}_+^{r\times n}$. It should be noted that the dimension r is usually much smaller than $\min\{m,n\}$. Clearly this problem is very difficult to solve and hence several algorithms have been developed (see, for example, [33]). One possibility to solve this problem is by finding a solution for the non-negative least squares model given by

$$\min_{B,C} \tfrac{1}{2}\left\|A - BC\right\|_F^2, \quad \text{s.t. } B \geq 0,\ C \geq 0, \tag{1.2}$$

where the non-negativity constraint is understood pointwise and $\left\|\cdot\right\|_F$ denotes the classical Frobenius norm. The NMF has important applications in image processing (face recognition) and bioinformatics (clustering of gene expressions). Observe that the gradient of the objective function is not Lipschitz continuous but it is partially Lipschitz continuous which enables the application of alternating minimization based methods (see [10]). Additionally, it is popular to impose sparsity constraints on one or both of the unknowns, *e.g.*, $\left\|C\right\|_0 \leq c$, to promote sparsity in the representation. See [10] for the first globally convergent algorithm for solving the sparse NMF problem. As we will see, the complicated sparse NMF can be also simply handled by our proposed algorithm which seems to produce better performances (see Section 5).

The second example we would like to mention is the important but ever challenging problem of blind image deconvolution (BID) [16]. Let $A \in [0,1]^{M\times N}$ be the observed blurred image of size $M \times N$, and let $B \in [0,1]^{M \times N}$ be the unknown sharp image of the same size. Furthermore, let $K \in \Delta_{mn}$ denote a small unknown blur kernel (point spread function) of size $m \times n$, where Δ_{mn} denotes the mn-dimensional standard unit simplex. We further assume that the observed blurred image has been formed by the following linear image formation model:

$$A = B * K + E,$$

where $*$ denotes a two dimensional discrete convolution operation and E denotes a small additive Gaussian noise. A typical variational formulation of the blind deconvolution problem is given by:

$$\min_{U,K} \mathcal{R}\left(U\right) + \frac{1}{2}\left\|B * K - A\right\|_F^2, \quad \text{s.t. } 0 \leq U \leq 1,\ K \in \Delta_{mn}. \tag{1.3}$$

In the above variational model, $\mathcal{R}$ is an image regularization term, typically a function, that imposes sparsity on the image gradient and hence favoring sharp images over blurred images.

We will come back to both examples in Section 5 where we will show how the proposed algorithm can be applied to efficiently solve these problems.

In [10], the authors proposed a proximal alternating linearized minimization method (PALM) that efficiently exploits the structure of problem (1.1). PALM can be understood as a blockwise application of the well-known proximal forward-backward algorithm [17, 11] in the nonconvex setting. In the case that the objective function F satisfy the so-called Kurdyka-Łojasiewicz (KL) property (the exact definition will be given in Section 3), the whole sequence of the algorithm is guaranteed to converge to a critical point of the problem.

In this paper, we propose an inertial version of the PALM algorithm and show convergence of the whole sequence in case the objective function F satisfy the KL property. The inertial term is motivated from the Heavy Ball method of Polyak [29] which in its most simple version applied to minimizing a smooth function f and can be written as the iterative scheme

$$x^{k+1} = x^k - \tau \nabla f\left(x^k\right) + \beta\left(x^k - x^{k-1}\right),$$

where β and τ are suitable parameters that ensure convergence of the algorithm. The heavy ball method differs from the usual gradient method by the additional inertial term $\beta\left(x^k - x^{k-1}\right)$, which adds part of the old direction to the new direction of the algorithm. Therefore for $\beta = 0$, we completely recover the classical algorithm of unconstrained optimization, the Gradient Method. The heavy ball method can be motivated from basically three view points.

First, the heavy ball method can be seen as an explicit finite differences discretization of the heavy ball with friction dynamical system (see [2]):

$$\ddot{x}\left(t\right) + c\dot{x}\left(t\right) + g\left(x\left(t\right)\right) = 0,$$

where $x(t)$ is a time continuous trajectory, $\ddot{x}(t)$ is the acceleration, $c\dot{x}(t)$ for $c > 0$ is the friction (damping), which is proportional to the velocity $\dot{x}(t)$, and $g(x(t))$ is an external gravitational field. In the case that $g = \nabla f$ the trajectory $x(t)$ is running down the "energy landscape" described by the objective function f until a critical point ($\nabla f = 0$) is reached. Due to the presence of the inertial term, it can also overcome spurious critical points of f, *e.g.*, saddle points.

Second, the heavy ball method can be seen as a special case of the so-called multi-step algorithms where each step of the algorithm is given as a linear combination of all previously computed gradients [12], that is, algorithm of the following form

$$x^{k+1} = x^k - \sum_{i=0}^{k} \alpha_i \nabla f\left(x^i\right).$$

Let us note that in the case that the objective function f is quadratic, the parameters α_i can be chosen in a way such that the objective function is minimized at each step. This approach eventually leads to the Conjugate Gradient (CG) method, pointing out a close relationship to inertial based methods.

Third, accelerated gradient methods, as pioneered by Nesterov (see [21] for an overview), are based on a variant of the heavy ball method that use the extrapolated point (based on the inertial force) also for evaluating the gradient in the current step. It turns out that, in the convex setting, these methods improve the worst convergence rate from $\mathcal{O}(1/k)$ to $\mathcal{O}(1/k^2)$, while leaving the computational complexity of each step basically the same.

In [35], the heavy ball method has been analyzed for the first time in the setting of nonconvex problems. It is shown that the heavy ball method is attracted by the connected components of critical points. The proof is based on considering a suitable Lyapunov function that allows to consider the two-step algorithm as a one-step algorithm. As we will see later, our convergence proof is also based on rewriting the algorithm as a one-step method.

In [24], the authors developed an inertial proximal gradient algorithm (iPiano). The algorithm falls into the class of forward-backward splitting algorithms [11], as it performs an explicit forward (steepest descent) step with respect to the smooth (nonconvex) function followed by a (proximal) backward step with respect to the nonsmooth (convex) function. Motivated by the heavy ball algorithm mentioned before, the iPiano algorithm makes use of an inertial force which empirically shows to improve the convergence speed of the algorithm. A related method based on general Bregman proximal-like distance functions has been recently proposed in [14].

Very recently, a randomized proximal linearization method has been proposed in [34]. The method is closely related to our proposed algorithm but convergence is proven only in the case that the function values are strictly decreasing. This is true only if the inertial force is set to be zero or the algorithm is restarted whenever the function values are not decreasing. In this paper, however, we overcome this major drawback and prove convergence of the algorithm without any assumption on the monotonicity of the objective function.

The remainder of the paper is organized as follows. In Section 2 we give an exact definition of the problem and the proposed algorithm. In Section 3 we state few technical results that will be necessary for the convergence analysis, which will be presented in Section 4. In Section 5 we present some numerical results and analyze the practical performance of the algorithm in dependence of its inertial parameters.

2 Problem Formulation and Algorithm

In this paper we follow [10] and consider the broad class of nonconvex and nonsmooth problems of the following form

$$\text{minimize } F\left(\mathbf{x}\right) := f_1\left(x_1\right) + f_2\left(x_2\right) + H\left(\mathbf{x}\right) \text{ over all } \mathbf{x} = \left(x_1, x_2\right) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}, \tag{2.1}$$

where f_1 and f_2 are extended valued (*i.e.*, giving the possibility of imposing constraints separately on the blocks x_1 and x_2) and H is a smooth coupling function (see below for more precise assumptions on the involved functions). We would like to stress from the beginning that *even though* all the discussions and results of this paper derived for two blocks of variables x_1 and x_2, they hold true for any finite number of blocks. This choice was done only for the sake of simplicity of the presentation of the algorithm and the convergence results.

As we discussed in the introduction, the proposed algorithm can be viewed either as a block version of the recent iPiano algorithm [24] or as an inertial based version of the recent PALM algorithm [10]. Before presenting the algorithm it will be convenient to recall the definition of the Moreau proximal mapping [20]. Given a proper and lower semicontinuous function $\sigma : \mathbb{R}^d \to (-\infty, \infty]$, the proximal mapping associated with σ is defined by

$$\operatorname{prox}_t^{\sigma}\left(p\right) := \operatorname{argmin}\left\{\sigma\left(q\right) + \frac{t}{2}\left\|q - p\right\|^2 : \ q \in \mathbb{R}^d\right\}, \quad \left(t > 0\right). \tag{2.2}$$

Following [10], we take the following as our blanket assumption.

Assumption A. (i) $f_1 : \mathbb{R}^{n_1} \to (-\infty, \infty]$ and $f_2 : \mathbb{R}^{n_2} \to (-\infty, \infty]$ are proper and lower semicontinuous functions such that $\inf_{\mathbb{R}^{n_1}} f_1 > -\infty$ and $\inf_{\mathbb{R}^{n_2}} f_2 > -\infty$.

(ii) $H : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$ is differentiable and $\inf_{\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}} F > -\infty$.

(iii) For any fixed x_2 the function $x_1 \to H\left(x_1, x_2\right)$ is $C^{1,1}_{L_1(x_2)}$, namely the partial gradient $\nabla_{x_1} H\left(x_1, x_2\right)$ is globally Lipschitz with moduli $L_1\left(x_2\right)$, that is,

$$\left\|\nabla_{x_1} H\left(u, x_2\right) - \nabla_{x_1} H\left(v, x_2\right)\right\| \leq L_1\left(x_2\right)\left\|u - v\right\|, \quad \forall\, u, v \in \mathbb{R}^{n_1}.$$

Likewise, for any fixed x_1 the function $x_2 \to H\left(x_1, x_2\right)$ is assumed to be $C^{1,1}_{L_2(x_1)}$.

(iv) For $i = 1, 2$ there exists $\lambda_i^-, \lambda_i^+ > 0$ such that

$$\inf \{L_1(x_2) : x_2 \in B_2\} \geq \lambda_1^- \quad \text{and} \quad \inf \{L_2(x_1) : x_1 \in B_1\} \geq \lambda_2^-, \tag{2.3}$$
$$\sup \{L_1(x_2) : x_2 \in B_2\} \leq \lambda_1^+ \quad \text{and} \quad \sup \{L_2(x_1) : x_1 \in B_1\} \leq \lambda_2^+, \tag{2.4}$$

for any compact set $B_i \subseteq \mathbb{R}^{n_i}$, $i = 1, 2$.

(v) ∇H is Lipschitz continuous on bounded subsets of $\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$. In other words, for each bounded subset $B_1 \times B_2$ of $\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ there exists $M > 0$ such that:

$$\begin{aligned}&\|(\nabla_{x_1} H(x_1, x_2) - \nabla_{x_1} H(y_1, y_2), \nabla_{x_2} H(x_1, x_2) - \nabla_{x_2} H(y_1, y_2))\| \\ &\qquad\qquad\qquad\qquad \leq M \|(x_1 - y_1, x_2 - y_2)\|.\end{aligned}$$

We propose now the inertial Proximal Alternating Linearized Minimization (iPALM) algorithm.

iPALM: Inertial Proximal Alternating Linearized Minimization

1. Initialization: start with any $(x_1^0, x_2^0) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$.
2. For each $k = 1, 2, \ldots$ generate a sequence $\left\{\left(x_1^k, x_2^k\right)\right\}_{k \in \mathbb{N}}$ as follows:

   2.1. Take $\alpha_1^k, \beta_1^k \in [0, 1]$ and $\tau_1^k > 0$. Compute

$$y_1^k = x_1^k + \alpha_1^k \left(x_1^k - x_1^{k-1}\right), \tag{2.5}$$
$$z_1^k = x_1^k + \beta_1^k \left(x_1^k - x_1^{k-1}\right), \tag{2.6}$$
$$x_1^{k+1} \in \operatorname{prox}_{\tau_1^k}^{f_1} \left(y_1^k - \frac{1}{\tau_1^k} \nabla_{x_1} H\left(z_1^k, x_2^k\right)\right). \tag{2.7}$$

   2.2. Take $\alpha_2^k, \beta_2^k \in [0, 1]$ and $\tau_2^k > 0$. Compute

$$y_2^k = x_2^k + \alpha_2^k \left(x_2^k - x_2^{k-1}\right), \tag{2.8}$$
$$z_2^k = x_2^k + \beta_2^k \left(x_2^k - x_2^{k-1}\right), \tag{2.9}$$
$$x_2^{k+1} \in \operatorname{prox}_{\tau_2^k}^{f_2} \left(y_2^k - \frac{1}{\tau_2^k} \nabla_{x_2} H\left(x_1^{k+1}, z_2^k\right)\right). \tag{2.10}$$

The parameters τ_1^k and τ_2^k, $k \in \mathbb{N}$, are discussed in Section 4 but for now, we can say that they are proportional to the respective partial Lipschitz moduli of H. The larger the partial Lipschitz moduli the smaller the step-size, and hence the slower the algorithm. As we shall see below, the partial Lipschitz moduli $L_1(x_2)$ and $L_2(x_1)$ are explicitly available for the examples mentioned at the introduction. However, note that if these are unknown, *or still too difficult to compute*, then a backtracking scheme [6] can be incorporated and the convergence results developed below remain true, for simplicity of exposition we omit the details.

In Section 5, we will show that the involved functions of the non-negative matrix factorization model (see (1.2)) and of the blind image deconvulation model (see (1.3)) do satisfy Assumption A. For the general setting we point out the following remarks about Assumption A.

(i) The first item of Assumption A is very general and most of the interesting constraints (via their indicator functions) or regularizing functions fulfill these requirements.

(ii) Items (ii)-(v) of Assumption A are beneficially exploited to build the proposed iPALM algorithm. These requirements do not guarantee that the gradient of H is globally Lipschitz (which is the case in our mentioned applications). The fact that ∇H is not globally Lipschitz reduces the potential of applying the iPiano and PFB methods in concrete applications and therefore highly motivated us to study their block counterparts (PALM in [10] and iPALM in this paper).

(iii) Another advantage of algorithms that exploit block structures inherent in the model at hand is the fact that they achieve better numerical performance (see Section 5) by taking step-sizes which is optimized to each separated block of variables.

(iv) Item (v) of Assumption A holds true, for example, when H is C^2. In this case the inequalities in (2.4) could be obtained if the sequence, which generated by the algorithm, is bounded.

The iPALM algorithm generalizes few known algorithms for different values of the inertial parameters α_i^k and β_i^k, $k \in \mathbb{N}$ and $i = 1, 2$. For example, when $\alpha_i^k = \beta_i^k = 0$, $k \in \mathbb{N}$, we recover the PALM algorithm of [10] which is a block version of the classical Proximal Forward-Backward (PFB) algorithm. When, there is only one block of variables, for instance only $i = 1$, we get the iPiano algorithm [24] which is recovered exactly only when $\beta_1^k = 0$, $k \in \mathbb{N}$. It should be also noted that in [24], the authors additionally assume that the function f_1 is convex (an assumption that is not needed in our case). The iPiano algorithm by itself generalizes two classical and known algorithms, one is the Heavy-Ball method [30] (when $f_1 \equiv 0$) and again the PFB method (when $\alpha_1^k = 0$, $k \in \mathbb{N}$).

3 Mathematical Preliminaries and Proof Methodology

Throughout this paper we are using standard notations and definitions of nonsmooth analysis which can be found in any classical book, see for instance [31, 19]. We recall here few notations and technical results. Let $\sigma : \mathbb{R}^d \to (-\infty, \infty]$ be a proper and lower semicontinuous function. Since we are dealing with nonconvex and nonsmooth functions that can have the value ∞, we use the notion of limiting subdifferential (or simply subdifferential), see [19], which is denoted by $\partial\sigma$. In what follows, we are interested in finding critical points of the objective function F defined in (2.1). Critical points are those points for which the corresponding subdifferential contains the zero vector $\mathbf{0}$. The set of critical points of σ is denoted by $\operatorname{crit}\sigma$, that is,

$$\operatorname{crit}\sigma = \left\{u \in \operatorname{dom}\sigma : \mathbf{0} \in \partial\sigma\left(u\right)\right\}.$$

An important property of the subdifferential is recorded in the following remark (see [31]).

Remark 3.1. Let $\left\{\left(u^k, q^k\right)\right\}_{k\in\mathbb{N}}$ be a sequence in $\operatorname{graph}\left(\partial\sigma\right)$ that converges to $\left(u, q\right)$ as $k \to \infty$. By the definition of $\partial\sigma\left(u\right)$, if $\sigma\left(u^k\right)$ converges to $\sigma\left(u\right)$ as $k \to \infty$, then $\left(u, q\right) \in \operatorname{graph}\left(\partial\sigma\right)$.

The convergence analysis of iPALM is based on the proof methodology which was developed in [5] and more recently extended and simplified in [10]. The main part of the suggested methodology relies on the fact that the objective function of the problem at hand satisfies the Kurdyka-Łojasiewicz (KL) property. Before stating the KL property we will need the definition of the following class of desingularizing functions. For $\eta \in (0, \infty]$ define

$$\Phi_\eta \equiv \left\{\varphi \in C\left[\left[0, \eta\right), \mathbb{R}_+\right] \text{ such that } \begin{cases} \varphi\left(0\right) = 0 & \\ \varphi \in C^1 & \text{on } \left(0, \eta\right) \\ \varphi'\left(s\right) > 0 & \text{for all } s \in \left(0, \eta\right) \end{cases} \right\}. \tag{3.1}$$

The function σ is said to have the Kurdyka-Łojasiewicz (KL) property at $\overline{u} \in \operatorname{dom}\partial\sigma$ if there exist $\eta \in (0, \infty]$, a neighborhood U of $\overline{u}$ and a function $\varphi \in \Phi_\eta$, such that, for all

$$u \in U \cap [\sigma(\overline{u}) < \sigma(u) < \sigma(\overline{u}) + \eta],$$

the following inequality holds

$$\varphi'\left(\sigma\left(u\right) - \sigma\left(\overline{u}\right)\right) \operatorname{dist}\left(0, \partial\sigma\left(u\right)\right) \geq 1, \tag{3.2}$$

where for any subset $S \subset \mathbb{R}^d$ and any point $x \in \mathbb{R}^d$

$$\operatorname{dist}\left(x, S\right) := \inf\left\{\|y - x\| : \ y \in S\right\}.$$

When $S = \emptyset$, we have that $\operatorname{dist}\left(x, S\right) = \infty$ for all x. If σ satisfies property (3.2) at each point of $\operatorname{dom}\partial\sigma$, then σ is called a *KL function*.

The convergence analysis presented in the following section is based on the uniformized KL property which was established in [10, Lemma 6, p. 478].

Lemma 3.1. *Let Ω be a compact set and let $\sigma : \mathbb{R}^d \to (-\infty, \infty]$ be a proper and lower semicontinuous function. Assume that σ is constant on Ω and satisfies the KL property at each point of Ω. Then, there exist $\varepsilon > 0$, $\eta > 0$ and $\varphi \in \Phi_\eta$ such that for all $\overline{u}$ in Ω and all u in the following intersection*

$$\left\{u \in \mathbb{R}^d : \ \operatorname{dist}(u, \Omega) < \varepsilon\right\} \cap \left[\sigma(\overline{u}) < \sigma(u) < \sigma(\overline{u}) + \eta\right], \tag{3.3}$$

one has,

$$\varphi'(\sigma(u) - \sigma(\overline{u})) \operatorname{dist}(0, \partial\sigma(u)) \geq 1. \tag{3.4}$$

We refer the reader to [9] for a depth study of the class of KL functions. For the important relation between semi-algebraic and KL functions see [8]. In [3, 4, 5, 10], the interested reader can find through catalog of functions which are very common in many applications and satisfy the KL property.

Before concluding the mathematical preliminaries part we would like to mention few important properties of the proximal map (defined in (2.2)). The following result can be found in [31].

Proposition 3.1. *Let $\sigma : \mathbb{R}^d \to (-\infty, \infty]$ be a proper and lower semicontinuous function with $\inf_{\mathbb{R}^d} \sigma > -\infty$. Then, for every $t \in (0, \infty)$ the set $\operatorname{prox}_{t\sigma}(u)$ is nonempty and compact.*

It follows immediately from the definition that proxσ is a multi-valued map when σ is nonconvex. The multi-valued projection onto a nonempty and closed set C is recovered when $\sigma = \delta_C$, which is the indicator function of C that defined to be zero on C and ∞ outside.

The main computational effort of iPALM involves a proximal mapping step of a proper and lower semicontinuous but nonconvex function. The following property will be essential in the forthcoming convergence analysis and is a slight modification of [10, Lemma 2, p. 471].

Lemma 3.2 (Proximal inequality)**.** *Let $h : \mathbb{R}^d \to \mathbb{R}$ be a continuously differentiable function with gradient ∇h assumed L_h-Lipschitz continuous and let $\sigma : \mathbb{R}^d \to (-\infty, \infty]$ be a proper and lower semicontinuous function with $\inf_{\mathbb{R}^d} \sigma > -\infty$. Then, for any $v, w \in \operatorname{dom} \sigma$ and any $u^+ \in \mathbb{R}^d$ defined by*

$$u^+ \in \operatorname{prox}_t^\sigma\left(v - \frac{1}{t}\nabla h(w)\right), \quad t > 0, \tag{3.5}$$

we have, for any $u \in \operatorname{dom} \sigma$ and any $s > 0$:

$$g(u^+) \leq g(u) + \frac{L_h + s}{2}\left\|u^+ - u\right\|^2 + \frac{t}{2}\left\|u - v\right\|^2 - \frac{t}{2}\left\|u^+ - v\right\|^2 + \frac{L_h^2}{2s}\left\|u - w\right\|^2, \tag{3.6}$$

where $g := h + \sigma$.

Proof. First, it follows immediately from Proposition 3.1 that u^+ is well-defined. By the definition of the proximal mapping (see (2.2)) we get that

$$u^+ \in \operatorname{argmin}_{\xi \in \mathbb{R}^d} \left\{ \langle \xi - v, \nabla h(w) \rangle + \frac{t}{2} \|\xi - v\|^2 + \sigma(\xi) \right\},$$

and hence in particular, by taking $\xi = u$, we obtain

$$\langle u^+ - v, \nabla h(w) \rangle + \frac{t}{2} \left\|u^+ - v\right\|^2 + \sigma\left(u^+\right) \leq \langle u - v, \nabla h(w) \rangle + \frac{t}{2} \|u - v\|^2 + \sigma(u).$$

Thus

$$\sigma\left(u^+\right) \leq \left\langle u - u^+, \nabla h(w) \right\rangle + \frac{t}{2} \|u - v\|^2 - \frac{t}{2} \left\|u^+ - v\right\|^2 + \sigma(u). \tag{3.7}$$

Invoking first the descent lemma (see [7]) for h, and using (3.7), yields

$$\begin{aligned} h\left(u^+\right) + \sigma\left(u^+\right) &\leq h(u) + \left\langle u^+ - u, \nabla h(u) \right\rangle + \frac{L_h}{2} \left\|u^+ - u\right\|^2 + \left\langle u - u^+, \nabla h(w) \right\rangle \\ &+ \frac{t}{2} \|u - v\|^2 - \frac{t}{2} \left\|u^+ - v\right\|^2 + \sigma(u) \\ &= h(u) + \sigma(u) + \left\langle u^+ - u, \nabla h(u) - \nabla h(w) \right\rangle + \frac{L_h}{2} \left\|u^+ - u\right\|^2 \\ &+ \frac{t}{2} \|u - v\|^2 - \frac{t}{2} \left\|u^+ - v\right\|^2. \end{aligned}$$

Now, using the fact that $\langle p, q \rangle \leq (s/2) \|p\|^2 + (1/2s) \|q\|^2$ for any two vectors $p, q \in \mathbb{R}^d$ and every $s > 0$, yields

$$\begin{aligned} \left\langle u^+ - u, \nabla h(u) - \nabla h(w) \right\rangle &\leq \frac{s}{2} \left\|u^+ - u\right\|^2 + \frac{1}{2s} \|\nabla h(u) - \nabla h(w)\|^2 \\ &\leq \frac{s}{2} \left\|u^+ - u\right\|^2 + \frac{{L_h}^2}{2s} \|u - w\|^2, \end{aligned}$$

where we have used the fact that ∇h is L_h-Lipschitz continuous. Thus, combining the last two inequalities proves that (3.6) holds. $\square$

Remark 3.2. It should be noted that if the nonsmooth function σ is also known to be convex, then we can derive the following tighter upper bound (*cf.* (3.6))

$$g\left(u^+\right) \leq g(u) + \frac{L_h + s - t}{2} \left\|u^+ - u\right\|^2 + \frac{t}{2} \|u - v\|^2 - \frac{t}{2} \left\|u^+ - v\right\|^2 + \frac{L_h^2}{2s} \|u - w\|^2. \tag{3.8}$$

3.1 Convergence Proof Methodology

In this section we briefly summarize (*cf.* Theorem 3.1 below) the methodology recently proposed in [10] which provides the key elements to obtain an abstract convergence result that can be applied to any algorithm and will be applied here to prove convergence of iPALM. Let $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ be a sequence in $\mathbb{R}^d$ which was generated from a starting point $\mathbf{u}^0$ by a generic algorithm $\mathcal{A}$. The set of all limit points of $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ is denoted by $\omega\left(\mathbf{u}^0\right)$, and defined by

$$\left\{\overline{\mathbf{u}}\in\mathbb{R}^d:\ \exists \text{ an increasing sequence of integers } \left\{k_l\right\}_{l\in\mathbb{N}} \text{ such that } \mathbf{u}^{k_l}\to\overline{\mathbf{u}} \text{ as } l\to\infty\right\}.$$

Theorem 3.1. *Let $\Psi:\mathbb{R}^d\to(-\infty,\infty]$ be a proper, lower semicontinuous and semi-algebraic function with $\inf\Psi>-\infty$. Assume that $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ is a bounded sequence generated by a generic algorithm $\mathcal{A}$ from a starting point $\mathbf{u}^0$, for which the following three conditions hold true for any $k\in\mathbb{N}$.*

(C1) *There exists a positive scalar ρ_1 such that*

$$\rho_1\left\|\mathbf{u}^{k+1}-\mathbf{u}^k\right\|^2\le\Psi\left(\mathbf{u}^k\right)-\Psi\left(\mathbf{u}^{k+1}\right),\quad\forall\ k=0,1,\ldots.$$

(C2) *There exists a positive scalar ρ_2 such that for some $\mathbf{w}^k\in\partial\Psi\left(\mathbf{u}^k\right)$ we have*

$$\left\|\mathbf{w}^k\right\|\le\rho_2\left\|\mathbf{u}^k-\mathbf{u}^{k-1}\right\|,\quad\forall\ k=0,1,\ldots.$$

(C3) *Each limit point in the set $\omega\left(\mathbf{u}^0\right)$ is a critical point of Ψ, that is, $\omega\left(\mathbf{u}^0\right)\subset\operatorname{crit}\Psi$.*

Then, the sequence $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ converges to a critical point $\mathbf{u}^*$ of Ψ.

4 Convergence Analysis of iPALM

Our aim in this section is to prove that the sequence $\left\{\left(x_1^k,x_2^k\right)\right\}_{k\in\mathbb{N}}$ which is generated by iPALM converges to a critical point of the objective function F defined in (2.1). To this end we will follow the proof methodology described above in Theorem 3.1. In the case of iPALM, similarly to the iPiano algorithm (see [24]), it is not possible to prove that condition (C1) hold true for the sequence $\left\{\left(x_1^k,x_2^k\right)\right\}_{k\in\mathbb{N}}$ and the function F, namely, this is not a descent algorithm with respect to F. Therefore, we first show that conditions (C1), (C2) and (C3) hold true for an auxiliary sequence and auxiliary function (see details below). Then, based on these properties we will show that the original sequence converges to a critical point of the original function F.

We first introduce the following notations that simplify the coming expositions. For any $k\in\mathbb{N}$, we define

$$\Delta_1^k=\frac{1}{2}\left\|x_1^k-x_1^{k-1}\right\|^2,\quad\Delta_2^k=\frac{1}{2}\left\|x_2^k-x_2^{k-1}\right\|^2\quad\text{and}\quad\Delta^k=\frac{1}{2}\left\|\mathbf{x}^k-\mathbf{x}^{k-1}\right\|^2,\tag{4.1}$$

it is clear, that using these notations, we have that $\Delta^k=\Delta_1^k+\Delta_2^k$ for all $k\in\mathbb{N}$. Using these notations we can easily show few basic relations of the sequences $\left\{x_i^k\right\}_{k\in\mathbb{N}}$, $\left\{y_i^k\right\}_{k\in\mathbb{N}}$, and $\left\{z_i^k\right\}_{k\in\mathbb{N}}$, for $i=1,2$, generated by iPALM.

Proposition 4.1. *Let $\{(x_1^k, x_2^k)\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM. Then, for any $k \in \mathbb{N}$ and $i = 1, 2$, we have*

(i) $\|x_i^k - y_i^k\|^2 = 2\left(\alpha_i^k\right)^2 \Delta_i^k$;

(ii) $\|x_i^k - z_i^k\|^2 = 2\left(\beta_i^k\right)^2 \Delta_i^k$;

(iii) $\|x_i^{k+1} - y_i^k\|^2 \geq 2\left(1 - \alpha_i^k\right)\Delta_i^{k+1} + 2\alpha_i^k\left(\alpha_i^k - 1\right)\Delta_i^k$.

Proof. The first two items follow immediately from the facts that $x_i^k - y_i^k = \alpha_i^k\left(x_i^{k-1} - x_i^k\right)$ and $x_i^k - z_i^k = \beta_i^k\left(x_i^{k-1} - x_i^k\right)$, for $i = 1, 2$ (see steps (2.5), (2.6), (2.8) and (2.9)). The last item follows from the following argument

$$\begin{aligned}
\|x_i^{k+1} - y_i^k\|^2 &= \|x_i^{k+1} - x_i^k - \alpha_i^k\left(x_i^k - x_i^{k-1}\right)\|^2 \\
&= 2\Delta_i^{k+1} - 2\alpha_i^k\left\langle x_i^{k+1} - x_i^k, x_i^k - x_i^{k-1}\right\rangle + 2\left(\alpha_i^k\right)^2\Delta_i^k \\
&\geq 2\left(1 - \alpha_i^k\right)\Delta_i^{k+1} + 2\alpha_i^k\left(\alpha_i^k - 1\right)\Delta_i^k, \qquad (4.2)
\end{aligned}$$

where we have used the fact that

$$2\left\langle x_i^{k+1} - x_i^k, x_i^k - x_i^{k-1}\right\rangle \leq \|x_i^{k+1} - x_i^k\|^2 + \|x_i^k - x_i^{k-1}\|^2 = 2\Delta_i^{k+1} + 2\Delta_i^k,$$

that follows from the Cauchy-Schwartz and Young inequalities. This proves item (iii). □

Now we prove the following property of the sequence $\{(x_1^k, x_2^k)\}_{k\in\mathbb{N}}$ generated by iPALM.

Proposition 4.2. *Suppose that Assumption A holds. Let $\{(x_1^k, x_2^k)\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM, then for all $k \in \mathbb{N}$, we have that*

$$\begin{aligned}
F\left(\mathbf{x}^{k+1}\right) \leq F\left(\mathbf{x}^k\right) &+ \frac{1}{s_1^k}\left(L_1(x_2^k)^2\left(\beta_1^k\right)^2 + s_1^k\tau_1^k\alpha_1^k\right)\Delta_1^k + \frac{1}{s_2^k}\left(L_2(x_1^{k+1})^2\left(\beta_2^k\right)^2 + s_2^k\tau_2^k\alpha_2^k\right)\Delta_2^k \\
&+ \left(L_1(x_2^k) + s_1^k - \tau_1^k\left(1 - \alpha_1^k\right)\right)\Delta_1^{k+1} + \left(L_2(x_1^{k+1}) + s_2^k - \tau_2^k\left(1 - \alpha_2^k\right)\right)\Delta_2^{k+1},
\end{aligned}$$

where $s_1^k > 0$ and $s_2^k > 0$ are arbitrarily chosen, for all $k \in \mathbb{N}$.

Proof. Fix $k \geq 1$. Under our Assumption A(ii), the function $x_1 \to H(x_1, x_2)$ (x_2 is fixed) is differentiable and has a Lipschitz continuous gradient with moduli $L_1(x_2)$. Using the iterative step (2.7), applying Lemma 3.2 for $h(\cdot) := H\left(\cdot, x_2^k\right)$, $\sigma := f_1$ and $t := \tau_1^k$ with the points $u = x_1^k$, $u^+ = x_1^{k+1}$, $v = y_1^k$ and $w = z_1^k$ yields that

$$\begin{aligned} H\left(x_1^{k+1}, x_2^k\right) + f_1\left(x_1^{k+1}\right) &\leq H\left(x_1^k, x_2^k\right) + f_1\left(x_1^k\right) + \frac{L_1(x_2^k) + s_1^k}{2}\left\|x_1^{k+1} - x_1^k\right\|^2 \\ &+ \frac{\tau_1^k}{2}\left\|x_1^k - y_1^k\right\|^2 - \frac{\tau_1^k}{2}\left\|x_1^{k+1} - y_1^k\right\|^2 + \frac{L_1(x_2^k)^2}{2s_1^k}\left\|x_1^k - z_1^k\right\|^2 \\ &\leq H\left(x_1^k, x_2^k\right) + f_1\left(x_1^k\right) + \left(L_1(x_2^k) + s_1^k\right)\Delta_1^{k+1} + \tau_1^k\left(\alpha_1^k\right)^2\Delta_1^k \\ &- \tau_1^k\left(\left(1 - \alpha_1^k\right)\Delta_1^{k+1} + \alpha_1^k\left(\alpha_1^k - 1\right)\Delta_1^k\right) + \frac{L_1(x_2^k)^2\left(\beta_1^k\right)^2}{s_1^k}\Delta_1^k \\ &= H\left(x_1^k, x_2^k\right) + f_1\left(x_1^k\right) + \left(L_1(x_2^k) + s_1^k - \tau_1^k\left(1 - \alpha_1^k\right)\right)\Delta_1^{k+1} \\ &+ \frac{1}{s_1^k}\left(L_1(x_2^k)^2\left(\beta_1^k\right)^2 + s_1^k\tau_1^k\alpha_1^k\right)\Delta_1^k, \end{aligned} \tag{4.3}$$

where the second inequality follows from Proposition 4.1. Repeating all the arguments above on the iterative step (2.10) yields the following

$$\begin{aligned} H\left(x_1^{k+1}, x_2^{k+1}\right) + f_2\left(x_2^{k+1}\right) &\leq H\left(x_1^{k+1}, x_2^k\right) + f_2\left(x_2^k\right) + \left(L_2(x_1^{k+1}) + s_2^k - \tau_2^k\left(1 - \alpha_2^k\right)\right)\Delta_2^{k+1} \\ &+ \frac{1}{s_2^k}\left(L_2(x_1^{k+1})^2\left(\beta_2^k\right)^2 + s_2^k\tau_2^k\alpha_2^k\right)\Delta_2^k. \end{aligned} \tag{4.4}$$

By adding (4.3) and (4.4) we get

$$\begin{aligned} F\left(\mathbf{x}^{k+1}\right) \leq F\left(\mathbf{x}^k\right) &+ \frac{1}{s_1^k}\left(L_1(x_2^k)^2\left(\beta_1^k\right)^2 + s_1^k\tau_1^k\alpha_1^k\right)\Delta_1^k + \frac{1}{s_2^k}\left(L_2(x_1^{k+1})^2\left(\beta_2^k\right)^2 + s_2^k\tau_2^k\alpha_2^k\right)\Delta_2^k \\ &+ \left(L_1(x_2^k) + s_1^k - \tau_1^k\left(1 - \alpha_1^k\right)\right)\Delta_1^{k+1} + \left(L_2(x_1^{k+1}) + s_2^k - \tau_2^k\left(1 - \alpha_2^k\right)\right)\Delta_2^{k+1}. \end{aligned}$$

This proves the desired result. □

Before we proceed and for the sake of simplicity of our developments we would like to chose the parameters s_1^k and s_2^k for all $k \in \mathbb{N}$. The best choice can be derived by minimizing the right-hand side of (3.6) with respect to s. Simple computations yields that the minimizer should be

$$s = L_h\frac{\|u - w\|}{\|u^+ - u\|},$$

where u, u^+, w and L_h are all in terms of Lemma 3.2. In Proposition 4.2 we have used Lemma 3.2 with the following choices $u = x_1^k$, $u^+ = x_1^{k+1}$ and $w = z_1^k$. Thus

$$s_1^k = L_1(x_2^k)\frac{\left\|x_1^k - z_1^k\right\|}{\left\|x_1^{k+1} - x_1^k\right\|} = L_1(x_2^k)\beta_1^k\frac{\left\|x_1^k - x_1^{k-1}\right\|}{\left\|x_1^{k+1} - x_1^k\right\|},$$

where the last equality follows from step (2.6). Thus, from now on, we will use the following parameters:

$$s_1^k = L_1(x_2^k)\beta_1^k \quad \text{and} \quad s_2^k = L_2(x_1^{k+1})\beta_2^k, \quad \forall\, k \in \mathbb{N}. \tag{4.5}$$

An immediate consequence of this choice of parameters which combined with Proposition 4.2 is recorded now.

Corollary 4.1. *Suppose that Assumption A holds. Let $\left\{\left(x_1^k, x_2^k\right)\right\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM, then for all $k \in \mathbb{N}$, we have that*

$$F\left(\mathbf{x}^{k+1}\right) \leq F\left(\mathbf{x}^k\right) + \left(L_1(x_2^k)\beta_1^k + \tau_1^k\alpha_1^k\right)\Delta_1^k + \left(L_2(x_1^{k+1})\beta_2^k + \tau_2^k\alpha_2^k\right)\Delta_2^k$$
$$+ \left(\left(1+\beta_1^k\right)L_1(x_2^k) - \tau_1^k\left(1-\alpha_1^k\right)\right)\Delta_1^{k+1} + \left(\left(1+\beta_2^k\right)L_2(x_1^{k+1}) - \tau_2^k\left(1-\alpha_2^k\right)\right)\Delta_2^{k+1}.$$

Similarly to iPiano, the iPALM algorithm generates a sequence which does not ensure that the function values decrease between two successive elements of the sequence. Thus we can not obtain condition (C1) of Theorem 3.1. Following [24] we construct an auxiliary function which do enjoy the property of function values decreases. Let $\Psi : \mathbb{R}^{n_1\times n_2} \times \mathbb{R}^{n_1\times n_2} \to (-\infty, \infty]$ be the auxiliary function which is defined as follows

$$\Psi_{\delta_1,\delta_2}(\mathbf{u}) := F(u_1) + \frac{\delta_1}{2}\|u_{11} - u_{21}\|^2 + \frac{\delta_2}{2}\|u_{12} - u_{22}\|^2, \tag{4.6}$$

where $\delta_1, \delta_2 > 0$, $u_1 = (u_{11}, u_{12}) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$, $u_2 = (u_{21}, u_{22}) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ and $\mathbf{u} = (u_1, u_2)$.

Let $\left\{\left(x_1^k, x_2^k\right)\right\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM and denote, for all $k \in \mathbb{N}$, $u_1^k = \left(x_1^k, x_2^k\right)$, $u_2^k = \left(x_1^{k-1}, x_2^{k-1}\right)$ and $\mathbf{u}^k = \left(u_1^k, u_2^k\right)$. We will prove now that the sequence $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ and the function Ψ defined above do satisfy conditions (C1), (C2) and (C3) of Theorem 3.1. We begin with proving condition (C1). To this end we will show that there are choices of $\delta_1 > 0$ and $\delta_2 > 0$, such that there exists $\rho_1 > 0$ which satisfies

$$\rho_1\left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\|^2 \leq \Psi\left(\mathbf{u}^k\right) - \Psi\left(\mathbf{u}^{k+1}\right).$$

It is easy to check that using the notations defined in (4.1), we have, for all $k \in \mathbb{N}$, that

$$\Psi\left(\mathbf{u}^k\right) = F\left(\mathbf{x}^k\right) + \frac{\delta_1}{2}\left\|x_1^k - x_1^{k-1}\right\|^2 + \frac{\delta_2}{2}\left\|x_2^k - x_2^{k-1}\right\|^2 = F\left(\mathbf{x}^k\right) + \delta_1\Delta_1^k + \delta_2\Delta_2^k.$$

In order to prove that the sequence $\left\{\Psi\left(\mathbf{u}^k\right)\right\}_{k\in\mathbb{N}}$ decreases we will need the following technical result (we provide the proof in Appendix 7).

Lemma 4.1. *Consider the functions $g : \mathbb{R}_+^5 \to \mathbb{R}$ and $h : \mathbb{R}_+^5 \to \mathbb{R}$ defined as follow*

$$g(\alpha,\beta,\delta,\tau,L) = \tau(1-\alpha) - (1+\beta)L - \delta,$$
$$h(\alpha,\beta,\delta,\tau,L) = \delta - \tau\alpha - L\beta.$$

Let $\varepsilon > 0$ and $\bar{\alpha} > 0$ be two real numbers for which $0 \leq \alpha \leq \bar{\alpha} < 0.5(1-\varepsilon)$. Assume, in addition, that $0 \leq L \leq \lambda$ for some $\lambda > 0$ and $0 \leq \beta \leq \bar{\beta}$ with $\bar{\beta} > 0$. If

$$\delta_* = \frac{\bar{\alpha} + \bar{\beta}}{1 - \varepsilon - 2\bar{\alpha}}\lambda, \tag{4.7}$$

$$\tau_* = \frac{(1+\varepsilon)\delta_* + (1+\beta)L}{1-\alpha}, \tag{4.8}$$

then $g(\alpha,\beta,\delta_*,\tau_*,L) = \varepsilon\delta_*$ and $h(\alpha,\beta,\delta_*,\tau_*,L) \geq \varepsilon\delta_*$.

Based on the mentioned lemma we will set, from now on, the parameters τ_1^k and τ_2^k for all $k \in \mathbb{N}$, as follow

$$\boxed{\tau_1^k = \frac{(1+\varepsilon)\,\delta_1^k + \left(1+\beta_1^k\right) L_1(x_2^k)}{1-\alpha_1^k} \quad \text{and} \quad \tau_2^k = \frac{(1+\varepsilon)\,\delta_2^k + \left(1+\beta_2^k\right) L_2(x_1^{k+1})}{1-\alpha_2^k}. \qquad (4.9)}$$

Remark 4.1. If we additionally know that f_i, $i = 1, 2$, is convex, then a tighter bound can be used in Proposition 3.2 as described in Remark 3.8. Using the tight bound will improve the possible parameter τ_i^k that can be used (*cf.* (4.9)). Indeed, in the convex case, (4.7) and (4.8) are given by

$$\delta_* = \frac{\bar{\alpha} + 2\bar{\beta}}{2\,(1-\varepsilon-\bar{\alpha})}\lambda, \tag{4.10}$$

$$\tau_* = \frac{(1+\varepsilon)\,\delta_* + (1+\beta)\,L}{2-\alpha}, \tag{4.11}$$

Thus, the parameters τ_i^k, $i = 1, 2$, can be taken in the convex case as follows

$$\tau_1^k = \frac{(1+\varepsilon)\,\delta_1^k + \left(1+\beta_1^k\right) L_1(x_2^k)}{2-\alpha_1^k} \quad \text{and} \quad \tau_2^k = \frac{(1+\varepsilon)\,\delta_2^k + \left(1+\beta_2^k\right) L_2(x_1^{k+1})}{2-\alpha_2^k}.$$

This means that in the convex case, we can take smaller τ_i^k, $i = 1, 2$, which means larger step-size in the algorithm. On top of that, in the case that f_i, $i = 1, 2$, is convex, it should be noted that a careful analysis shows that in this case the parameters α_i^k, $i = 1, 2$, can be in the interval $[0, 1)$ and not $[0, 0.5)$ as stated in Lemma 4.1 (see also Assumption B below).

In order to prove condition C1 and according to Lemma 4.1, we will need to restrict the possible values of the parameters α_i^k and β_i^k, $i = 1, 2$, for all $k \in \mathbb{N}$. The following assumption is essential for our analysis.

Assumption B. Let $\varepsilon > 0$ be an arbitrary small number. For all $k \in \mathbb{N}$ and $i = 1, 2$, there exist $0 < \bar{\alpha}_i < (1/2)\,(1-\varepsilon)$ such that $0 \leq \alpha_i^k \leq \bar{\alpha}_i$. In addition, $0 \leq \beta_i^k \leq \bar{\beta}_i$ for some $\bar{\beta}_i > 0$.

Remark 4.2. It should be noted that using Assumption B, we obtain that $\tau_1^k \leq \tau_1^+$ where

$$\tau_1^+ = \frac{(1+\varepsilon)\,\delta_1 + \left(1+\bar{\beta}_1\right)\lambda_1^+}{1-\bar{\alpha}_1},$$

where δ_1 is given in (4.7). Similar arguments show that

$$\tau_2^k \leq \tau_2^+ := \frac{(1+\varepsilon)\,\delta_2 + \left(1+\bar{\beta}_2\right)\lambda_2^+}{1-\bar{\alpha}_2}.$$

Now we will prove a descent property of $\left\{\Psi\left(\mathbf{u}^k\right)\right\}_{k\in\mathbb{N}}$.

Proposition 4.3. *Let $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM which is assumed to be bounded. Suppose that Assumptions A and B hold true. Then, for all $k \in \mathbb{N}$ and $\varepsilon > 0$, we have*

$$\rho_1 \left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\|^2 \leq \Psi\left(\mathbf{u}^k\right) - \Psi\left(\mathbf{u}^{k+1}\right),$$

where $\mathbf{u}^k = \left(\mathbf{x}^k, \mathbf{x}^{k-1}\right)$, $k \in \mathbb{N}$ and $\rho_1 = (\varepsilon/2)\min\{\delta_1, \delta_2\}$ with

$$\delta_1 = \frac{\bar{\alpha}_1 + \bar{\beta}_1}{1 - \varepsilon - 2\bar{\alpha}_1}\lambda_1^+ \quad \textit{and} \quad \delta_2 = \frac{\bar{\alpha}_2 + \bar{\beta}_2}{1 - \varepsilon - 2\bar{\alpha}_2}\lambda_2^+. \tag{4.12}$$

Proof. From the definition of Ψ (see (4.6)) and Corollary 4.1 we obtain that

$$\begin{aligned}
\Psi\left(\mathbf{u}^k\right) - \Psi\left(\mathbf{u}^{k+1}\right) &= F\left(\mathbf{x}^k\right) + \delta_1\Delta_1^k + \delta_2\Delta_2^k - F\left(\mathbf{x}^{k+1}\right) - \delta_1\Delta_1^{k+1} - \delta_2\Delta_2^{k+1} \\
&\geq \left(\tau_1^k\left(1 - \alpha_1^k\right) - \left(1 + \beta_1^k\right)L_1(x_2^k) - \delta_1\right)\Delta_1^{k+1} \\
&\quad + \left(\delta_1 - \tau_1^k\alpha_1^k - L_1(x_2^k)\beta_1^k\right)\Delta_1^k \\
&\quad + \left(\tau_2^k\left(1 - \alpha_2^k\right) - \left(1 + \beta_2^k\right)L_2(x_1^{k+1}) - \delta_2\right)\Delta_2^{k+1} \\
&\quad + \left(\delta_2 - \tau_2^k\alpha_2^k - L_2(x_1^{k+1})\beta_2^k\right)\Delta_2^k \\
&= a_1^k\Delta_1^{k+1} + b_1^k\Delta_1^k + a_2^k\Delta_2^{k+1} + b_2^k\Delta_2^k,
\end{aligned}$$

where

$$\begin{aligned}
a_1^k &:= \tau_1^k\left(1 - \alpha_1^k\right) - \left(1 + \beta_1^k\right)L_1(x_2^k) - \delta_1 \quad \text{and} \quad b_1^k := \delta_1 - \tau_1^k\alpha_1^k - L_1(x_2^k)\beta_1^k, \\
a_2^k &:= \tau_2^k\left(1 - \alpha_2^k\right) - \left(1 + \beta_2^k\right)L_2(x_1^{k+1}) - \delta_2 \quad \text{and} \quad b_2^k := \delta_2 - \tau_2^k\alpha_2^k - L_2(x_1^{k+1})\beta_2^k.
\end{aligned}$$

Let $\varepsilon > 0$ be an arbitrary. Using (4.9) and (4.12) with the notations of Lemma 4.1 we immediately see that $a_1^k = g_1\left(\alpha_1^k, \beta_1^k, \delta_1, \tau_1^k, L_1(x_2^k)\right)$ and $a_2^k = g_2\left(\alpha_2^k, \beta_2^k, \delta_2, \tau_2^k, L_2(x_1^{k+1})\right)$. From Assumptions A and B we get that the requirements of Lemma 4.1 are fulfilled, which means that Lemma 4.1 can be applied. Thus $a_1^k = \varepsilon\delta_1$ and $a_2^k = \varepsilon\delta_2$. Using again the notions of Lemma 4.1, we have that $b_1^k = h_1\left(\alpha_1^k, \beta_1^k, \delta_1, \tau_1^k, L_1(x_2^k)\right)$ and $b_2^k = h_2\left(\alpha_2^k, \beta_2^k, \delta_2, \tau_2^k, L_2(x_1^{k+1})\right)$. Thus we obtain from Lemma 4.1 that $b_1^k \geq \varepsilon\delta_1$ and $b_2^k \geq \varepsilon\delta_2$. Hence, for $\rho_1 = (\varepsilon/2)\min\{\delta_1, \delta_2\}$, we have

$$\begin{aligned}
\Psi\left(\mathbf{u}^k\right) - \Psi\left(\mathbf{u}^{k+1}\right) &\geq a_1^k\Delta_1^{k+1} + b_1^k\Delta_1^k + a_2^k\Delta_2^{k+1} + b_2^k\Delta_2^k \\
&\geq \varepsilon\delta_1\left(\Delta_1^{k+1} + \Delta_1^k\right) + \varepsilon\delta_1\left(\Delta_2^{k+1} + \Delta_2^k\right) \\
&\geq \rho_1\left(\Delta_1^{k+1} + \Delta_1^k\right) + \rho_1\left(\Delta_2^{k+1} + \Delta_2^k\right) \\
&= \rho_1\left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\|^2,
\end{aligned}$$

where the last equality follows from (4.1). This completes the proof. $\square$

Now, we will prove that condition (C2) of Theorem 3.1 holds true for the sequence $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ and the function Ψ.

Proposition 4.4. *Let $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM which is assumed to be bounded. Suppose that Assumptions A and B hold true. Assume that $\mathbf{u}^k = \left(\mathbf{x}^k, \mathbf{x}^{k-1}\right)$, $k \in \mathbb{N}$. Then, there exists a positive scalar ρ_2 such that for some $\mathbf{w}^k \in \partial\Psi\left(\mathbf{u}^k\right)$ we have*

$$\left\|\mathbf{w}^k\right\| \leq \rho_2 \left\|\mathbf{u}^k - \mathbf{u}^{k-1}\right\|.$$

Proof. Let $k \geq 2$. By the definition of Ψ (see (4.6)) we have that

$$\partial\Psi\left(\mathbf{u}^k\right) = \left(\partial_{x_1}F\left(\mathbf{x}^k\right) + \delta_1\left(x_1^k - x_1^{k-1}\right), \partial_{x_2}F\left(\mathbf{x}^k\right) + \delta_2\left(x_2^k - x_2^{k-1}\right), \delta_1\left(x_1^{k-1} - x_1^k\right),\right.$$
$$\left.\delta_2\left(x_2^{k-1} - x_2^k\right)\right).$$

By the definition of F (see (2.1)) and [10, Proposition 1, Page 465] we get that

$$\partial F\left(\mathbf{x}^k\right) = \left(\partial f_1\left(x_1^k\right) + \nabla_{x_1}H\left(x_1^k, x_2^k\right), \partial f_2\left(x_2^k\right) + \nabla_{x_2}H\left(x_1^k, x_2^k\right)\right). \tag{4.13}$$

From the definition of the proximal mapping (see (2.2)) and the iterative step (2.7) we have

$$x_1^k \in \operatorname{argmin}_{x_1\in\mathbb{R}^{n_1}}\left\{\left\langle x_1 - y_1^{k-1}, \nabla_{x_1}H\left(z_1^{k-1}, x_2^{k-1}\right)\right\rangle + \frac{\tau_1^{k-1}}{2}\left\|x_1 - y_1^{k-1}\right\|^2 + f_1\left(x_1\right)\right\}.$$

Writing down the optimality condition yields

$$\nabla_{x_1}H\left(z_1^{k-1}, x_2^{k-1}\right) + \tau_1^{k-1}\left(x_1^k - y_1^{k-1}\right) + \xi_1^k = 0,$$

where $\xi_1^k \in \partial f_1\left(x_1^k\right)$. Hence

$$\nabla_{x_1}H\left(z_1^{k-1}, x_2^{k-1}\right) + \xi_1^k = \tau_1^{k-1}\left(y_1^{k-1} - x_1^k\right) = \tau_1^{k-1}\left(x_1^{k-1} - x_1^k + \alpha_1^{k-1}\left(x_1^{k-1} - x_1^{k-2}\right)\right),$$

where the last equality follows from step (2.5). By defining

$$v_1^k := \nabla_{x_1}H\left(x_1^k, x_2^k\right) - \nabla_{x_1}H\left(z_1^{k-1}, x_2^{k-1}\right) + \tau_1^{k-1}\left(x_1^{k-1} - x_1^k + \alpha_1^{k-1}\left(x_1^{k-1} - x_1^{k-2}\right)\right), \tag{4.14}$$

we obtain from (4.13) that $v_1^k \in \partial_{x_1}F\left(\mathbf{x}^k\right)$. Similarly, from the iterative step (2.10), by defining

$$v_2^k := \nabla_{x_2}H\left(x_1^k, x_2^k\right) - \nabla_{x_2}H\left(x_1^k, z_2^{k-1}\right) + \tau_2^{k-1}\left(x_2^{k-1} - x_2^k + \alpha_2^{k-1}\left(x_2^{k-1} - x_2^{k-2}\right)\right), \tag{4.15}$$

we have that $v_2^k \in \partial_{x_2}F\left(\mathbf{x}^k\right)$.

Thus, for

$$\mathbf{w}^k := \left(v_1^k + \delta_1\left(x_1^k - x_1^{k-1}\right), v_2^k + \delta_2\left(x_2^k - x_2^{k-1}\right)\right), \delta_1\left(x_1^k - x_1^{k-1}, \delta_2\left(x_2^k - x_2^{k-1}\right)\right),$$

we obtain that

$$\left\|\mathbf{w}^k\right\| \leq \left\|v_1^k\right\| + \left\|v_2^k\right\| + 2\delta_1\left\|x_1^k - x_1^{k-1}\right\| + 2\delta_2\left\|x_2^k - x_2^{k-1}\right\|. \tag{4.16}$$

This means that we have to bound from above the norms of v_1^k and v_2^k. Since ∇H is Lipschitz continuous on bounded subsets of $\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ (see Assumption A(v)) and since we assumed that $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ is bounded, there exists $M > 0$ such that

$$\begin{aligned}
\left\|v_1^k\right\| &\leq \tau_1^{k-1}\left\|x_1^{k-1} - x_1^k + \alpha_1^{k-1}\left(x_1^{k-1} - x_1^{k-2}\right)\right\| + \left\|\nabla_{x_1} H\left(x_1^k, x_2^k\right) - \nabla_{x_1} H\left(z_1^{k-1}, x_2^{k-1}\right)\right\| \\
&\leq \tau_1^{k-1}\left\|x_1^{k-1} - x_1^k\right\| + \tau_1^{k-1}\alpha_1^{k-1}\left\|x_1^{k-1} - x_1^{k-2}\right\| + M\left\|\mathbf{x}^k - \left(z_1^{k-1}, x_2^{k-1}\right)\right\| \\
&\leq \tau_1^+\left(\left\|x_1^{k-1} - x_1^k\right\| + \left\|x_1^{k-1} - x_1^{k-2}\right\|\right) + M\left\|\left(x_1^k - x_1^{k-1} - \beta_1^{k-1}\left(x_1^{k-1} - x_1^{k-2}\right), x_2^k - x_2^{k-1}\right)\right\|, \\
&= \tau_1^+\left(\left\|x_1^{k-1} - x_1^k\right\| + \left\|x_1^{k-1} - x_1^{k-2}\right\|\right) + M\left\|\left(\mathbf{x}^k - \mathbf{x}^{k-1}\right) - \beta_1^{k-1}\left(x_1^{k-1} - x_1^{k-2}, \mathbf{0}\right)\right\| \\
&\leq \left(\tau_1^+ + M\right)\left(\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| + \left\|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\right\|\right),
\end{aligned}$$

where the third inequality follows from (2.6), the fact the sequence $\left\{\tau_1^k\right\}_{k\in\mathbb{N}}$ is bounded from above by τ_1^+ (see Remark 4.2) and $\alpha_1^k, \beta_1^k \leq 1$ for all $k \in \mathbb{N}$. On the other hand, from the Lipschitz continuity of $\nabla_{x_2} H\left(x_1, \cdot\right)$ (see Assumption A(iii)), we have that

$$\begin{aligned}
\left\|v_2^k\right\| &\leq \tau_2^{k-1}\left\|x_2^{k-1} - x_2^k + \alpha_2^{k-1}\left(x_2^{k-1} - x_2^{k-2}\right)\right\| + \left\|\nabla_{x_2} H\left(x_1^k, x_2^k\right) - \nabla_{x_2} H\left(x_1^k, z_2^{k-1}\right)\right\| \\
&\leq \tau_2^{k-1}\left\|x_2^{k-1} - x_2^k\right\| + \tau_2^{k-1}\alpha_2^{k-1}\left\|x_2^{k-1} - x_2^{k-2}\right\| + L_1(x_1^k)\left\|x_2^k - z_2^{k-1}\right\| \\
&\leq \tau_2^+\left(\left\|x_2^{k-1} - x_2^k\right\| + \left\|x_2^{k-1} - x_2^{k-2}\right\|\right) + \lambda_2^+\left\|x_2^k - x_2^{k-1} - \beta_2^{k-1}\left(x_2^{k-1} - x_2^{k-2}\right)\right\| \\
&\leq \left(\tau_2^+ + \lambda_2^+\right)\left(\left\|x_2^{k-1} - x_2^k\right\| + \left\|x_2^{k-1} - x_2^{k-2}\right\|\right) \\
&\leq \left(\tau_2^+ + \lambda_2^+\right)\left(\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| + \left\|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\right\|\right),
\end{aligned}$$

where the third and fourth inequalities follow from (2.9), the fact the sequence $\left\{\tau_2^k\right\}_{k\in\mathbb{N}}$ is bounded from above by τ_2^+ (see Remark 4.2) and $\beta_2^k \leq 1$ for all $k \in \mathbb{N}$. Summing up these estimations, we get from (4.16) that

$$\begin{aligned}
\left\|\mathbf{w}^k\right\| &\leq \left\|v_1^k\right\| + \left\|v_2^k\right\| + 2\delta_1\left\|x_1^k - x_1^{k-1}\right\| + 2\delta_2\left\|x_2^k - x_2^{k-1}\right\| \\
&\leq \left\|v_1^k\right\| + \left\|v_2^k\right\| + 2\left(\delta_1 + \delta_2\right)\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| \\
&\leq \left(\tau_1^+ + M + \tau_2^+ + \lambda_2^+\right)\left(\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| + \left\|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\right\|\right) + 2\left(\delta_1 + \delta_2\right)\left\|\mathbf{u}^k - \mathbf{u}^{k-1}\right\| \\
&\leq \left(\sqrt{2}\left(\tau_1^+ + M + \tau_2^+ + \lambda_2^+\right) + 2\left(\delta_1 + \delta_2\right)\right)\left\|\mathbf{u}^k - \mathbf{u}^{k-1}\right\|,
\end{aligned}$$

where the second inequality follows from the fact that $\left\|x_i^k - x_i^{k-1}\right\| \leq \left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\|$ for $i = 1, 2$, the third inequality follows from the fact that $\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| \leq \left\|\mathbf{u}^k - \mathbf{u}^{k-1}\right\|$ and the last inequality follows from the fact that $\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| + \left\|\mathbf{x}^{k-1} - \mathbf{x}^{k-2}\right\| \leq \sqrt{2}\left\|\mathbf{u}^k - \mathbf{u}^{k-1}\right\|$. This completes the proof with $\rho_2 = \sqrt{2}\left(\tau_1^+ + M + \tau_2^+ + \lambda_2^+\right) + 2\left(\delta_1 + \delta_2\right)$. □

So far we have proved that the sequence $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ and the function Ψ (see (4.6)) satisfy conditions (C1) and (C2) of Theorem 3.1. Now, in order to get that $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ converges to a critical point of Ψ, it remains to prove that condition (C3) holds true.

Proposition 4.5. *Let $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM which is assumed to be bounded. Suppose that Assumptions A and B hold true. Assume that $\mathbf{u}^k = \left(\mathbf{x}^k, \mathbf{x}^{k-1}\right)$, $k \in \mathbb{N}$. Then, each limit point in the set $\omega\left(\mathbf{u}^0\right)$ is a critical point of Ψ.*

Proof. Since $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ is assumed to be bounded, the set $\omega\left(\mathbf{u}^0\right)$ is nonempty. Thus there exists $\mathbf{u}^* = \left(x_1^*, x_2^*, \hat{x}_1, \hat{x}_2\right)$ which is a limit point of $\left\{\mathbf{u}^{k_l}\right\}_{l\in\mathbb{N}}$, which is a subsequence of $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$. We will prove that $\mathbf{u}^*$ is a critical point of Ψ (see (4.6)). From condition (C2), for some $\mathbf{w}^k \in \partial\Psi\left(\mathbf{u}^k\right)$, we have that

$$\left\|\mathbf{w}^k\right\| \leq \rho_2 \left\|\mathbf{u}^k - \mathbf{u}^{k-1}\right\|.$$

From Proposition 4.3, it follow that for any $N \in \mathbb{N}$, we have

$$\rho_1 \sum_{k=0}^{N} \left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\|^2 \leq \Psi\left(\mathbf{u}^0\right) - \Psi\left(\mathbf{u}^{N+1}\right). \tag{4.17}$$

Since F is bounded from below (see Assumption A(ii)) and the fact that $\Psi(\cdot) \geq F(\cdot)$ we obtain that Ψ is also bounded from below. Thus, letting $N \to \infty$ in (4.17) yields that

$$\sum_{k=0}^{\infty} \left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\|^2 < \infty, \tag{4.18}$$

which means that

$$\lim_{k\to\infty} \left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\| = 0. \tag{4.19}$$

This fact together with condition (C1) implies that $\left\|\mathbf{w}^k\right\| \to 0$ as $k \to \infty$. Thus, in order to use the closedness property of $\partial\Psi$ (see Remark 3.1) we remain to show that $\left\{\Psi\left(\mathbf{u}^k\right)\right\}_{k\in\mathbb{N}}$ converges to $\Psi\left(\mathbf{u}^*\right)$. Since f_1 and f_2 are lower semicontinuous (see Assumption A(i)), we obtain that

$$\liminf_{k\to\infty} f_1\left(x_1^k\right) \geq f_1\left(x_1^*\right) \quad \text{and} \quad \liminf_{k\to\infty} f_2\left(x_2^k\right) \geq f_2\left(x_2^*\right). \tag{4.20}$$

From the iterative step (2.7), we have, for all integer k, that

$$x_1^{k+1} \in \operatorname{argmin}_{x_1\in\mathbb{R}^{n_1}} \left\{\left\langle x_1 - y_1^k, \nabla_{x_1} H\left(z_1^k, x_2^k\right)\right\rangle + \frac{\tau_1^k}{2}\left\|x_1 - x_1^k\right\|^2 + f_1\left(x_1\right)\right\}.$$

Thus letting $x_1 = x_1^*$ in the above, we get

$$\begin{aligned}\left\langle x_1^{k+1} - y_1^k, \nabla_{x_1} H\left(z_1^k, x_2^k\right)\right\rangle + \frac{\tau_1^k}{2}\left\|x_1^{k+1} - x_1^k\right\|^2 + f_1\left(x_1^{k+1}\right)& \\ \leq \left\langle x_1^* - y_1^k, \nabla_{x_1} H\left(z_1^k, x_2^k\right)\right\rangle + \frac{\tau_1^k}{2}\left\|x_1^* - x_1^k\right\|^2 + f_1\left(x_1^*\right).&\end{aligned}$$

Choosing $k = k_l - 1$ and letting k goes to infinity, we obtain

$$\begin{aligned}\limsup_{l\to\infty} f_1\left(x_1^{k_l}\right) \leq \limsup_{l\to\infty}&\left(\left\langle x_1^* - x_1^{k_l}, \nabla_{x_1} H\left(z_1^{k_l-1}, x_2^{k_l-1}\right)\right\rangle + \frac{\tau_1^{k_l-1}}{2}\left\|x_1^* - x_1^{k_l-1}\right\|^2\right) \\ &+ f_1\left(x_1^*\right),\end{aligned} \tag{4.21}$$

where we have used the facts that both sequences $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ (and therefore $\left\{\mathbf{z}^k\right\}_{k\in\mathbb{N}}$) and $\left\{\tau_1^k\right\}_{k\in\mathbb{N}}$ (see Remark 4.2) are bounded, ∇H continuous and that the distance between two successive iterates tends to zero (see (4.19)). For that very reason we also have that $x_1^{k_l} \to x_1^*$ as $l \to \infty$, hence (4.21) reduces to $\limsup_{l\to\infty} f_1\left(x_1^{k_l}\right) \leq f_1\left(x_1^*\right)$. Thus, in view of (4.20), $f_1\left(x_1^{k_l}\right)$ tends to $f_1\left(x_1^*\right)$ as $k \to \infty$. Arguing similarly with f_2 and x_2^{k+1} we thus finally obtain from (4.19) that

$$\begin{aligned}
\lim_{l\to\infty} \Psi\left(\mathbf{u}^{k_l}\right) &= \lim_{l\to\infty} \left\{ f_1\left(x_1^{k_l}\right) + f_2\left(x_2^{k_l}\right) + H\left(\mathbf{x}^{k_l}\right) + \frac{\delta_1}{2}\left\|x_1^{k_l} - x_1^{k_l-1}\right\|^2 + \frac{\delta_2}{2}\left\|x_2^{k_l} - x_2^{k_l-1}\right\|^2 \right\} \\
&= f_1\left(x_1^*\right) + f_2\left(x_2^*\right) + H\left(x_1^*, x_2^*\right) \\
&= F\left(x_1^*, x_2^*\right) \qquad (4.22) \\
&= \Psi\left(\mathbf{u}^*\right).
\end{aligned}$$

Now, the closedness property of $\partial\Psi$ (see Remark 3.1) implies that $\mathbf{0} \in \partial\Psi\left(\mathbf{u}^*\right)$, which proves that $\mathbf{u}^*$ is a critical point of Ψ. This proves condition (C3). □

Now using the convergence proof methodology of [10] which is summarized in Theorem 3.1 we can obtain the following result.

Corollary 4.2. *Let $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM which is assumed to be bounded. Suppose that Assumptions A and B hold true. Assume that $\mathbf{u}^k = \left(\mathbf{x}^k, \mathbf{x}^{k-1}\right)$, $k \in \mathbb{N}$. If F is a KL function, then the sequence $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ converges to a critical point $\mathbf{u}^*$ of Ψ.*

Proof. The proof follows immediately from Theorem 3.1 since Proposition 4.3 proves that condition (C1) holds true, Proposition 4.4 proves that condition (C2) holds true, and condition (C3) was proved in Proposition 4.5. It is also clear that if F is a KL function then obviously Ψ is a KL function since we just add two quadratic functions. □

To conclude the convergence theory of iPALM we have to show that the sequence $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ which is generated by iPALM converges to a critical point of F (see (2.1)).

Theorem 4.1 (Convergence of iPALM)**.** *Let $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ be a sequence generated by iPALM which is assumed to be bounded. Suppose that Assumptions A and B hold true. If F is a KL function, then the sequence $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ converges to a critical point $\mathbf{x}^*$ of F.*

Proof. From Corollary 4.2 we have that the sequence $\left\{\mathbf{u}^k\right\}_{k\in\mathbb{N}}$ converges to a critical point $\mathbf{u}^* = \left(u_{11}^*, u_{12}^*, u_{21}^*, u_{22}^*\right)$ of Ψ. Therefore, obviously also the sequence $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$ converges. Let $\mathbf{x}^*$ be the limit point of $\left\{\mathbf{x}^k\right\}_{k\in\mathbb{N}}$. Thence $u_1^* = \left(u_{11}^*, u_{12}^*\right) = \mathbf{x}^*$ and $u_2^* = \left(u_{21}^*, u_{22}^*\right) = \mathbf{x}^*$ (see the discussion on Page 14). We will prove that $\mathbf{x}^*$ is a critical point of F (see (2.1)), that is, we have to show that $0 \in \partial F\left(\mathbf{x}^*\right)$. Since $\mathbf{u}^*$ is a critical point of Ψ, it means that $0 \in \partial\Psi\left(\mathbf{u}^*\right)$. Thus

$$0 \in \left(\partial_{x_1} F\left(u_1^*\right) + \delta_1\left(u_{11}^* - u_{12}^*\right), \partial_{x_2} F\left(u_2^*\right) + \delta_2\left(u_{21}^* - u_{22}^*\right), \delta_1\left(u_{11}^* - u_{12}^*\right), \delta_2\left(u_{21}^* - u_{22}^*\right)\right),$$

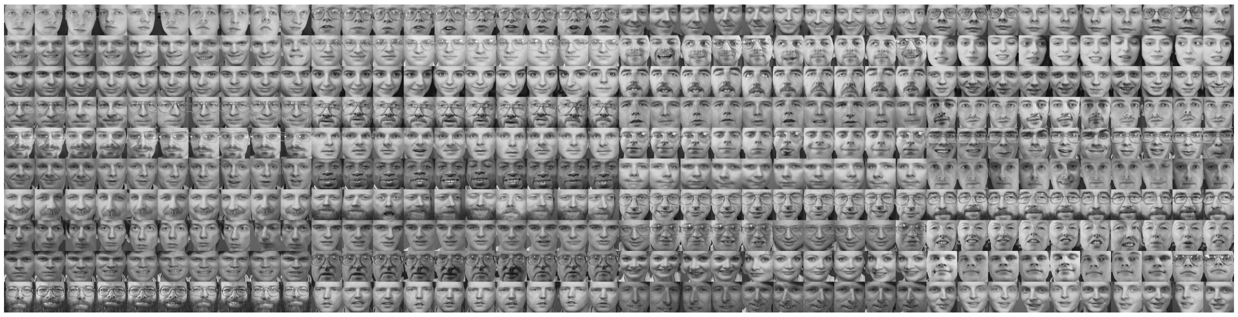


Figure 1: ORL database which includs 400 faces which we used in our NMF example.

which means that

$$0 \in \left(\partial_{x_1} F \left(u_1^* \right), \partial_{x_2} F \left(u_2^* \right) \right) = \partial F \left(\mathbf{x}^* \right).$$

This proves that $\mathbf{x}^*$ is a critical point of F. $\square$

5 Numerical Results

In this section we consider several important applications in image processing and machine learning to illustrate the numerical performance of the proposed iPALM method. All algorithms have been implemented in Matlab R2013a and executed on a server with Xeon(R) E5-2680 v2 @ 2.80GHz CPUs and running Linux.

5.1 Non-Negative Matrix Factorization

In our first example we consider the problem of using the Non-negative Matrix Factorization (NMF) to decompose a set of facial images into a number of sparse basis faces, such that each face of the database can be approximated by using a small number of those parts. We use the ORL database [32] that consists of 400 normalized facial images. In order to enforce sparsity in the basis faces, we additionally consider a ℓ_0 sparsity constraint (see, for example, [27]). The sparse NMF problem to be solved is given by

$$\min_{B,C} \left\{ \frac{1}{2} \left\| A - BC \right\|^2 : \, B, C \geq 0, \; \left\| b_i \right\|_0 \leq s, \, i = 1, 2, \ldots, r \right\}, \tag{5.1}$$

where $A \in \mathbb{R}^{m \times n}$ is the data matrix, organized in a way that each column of the matrix A corresponds to one face of size $m = 64 \times 64$ pixels. In total, the matrix holds $n = 400$ faces (see Figure 1 for a visualization of the data matrix A). The matrix $B \in \mathbb{R}^{m \times r}$ holds the r basis vectors $b_i \in \mathbb{R}^{m \times 1}$, where r corresponds to the number of sparse basis faces. The sparsity constraint applied to each basis face requires that the number of non-zero elements in each column vector b_i, $i = 1, 2, \ldots, r$ should have less or equal to s non-zero elements. Finally, the matrix $C \in \mathbb{R}^{r \times n}$ corresponds to the coefficient vectors.

The application of the proposed iPALM algorithm to this problem is straight-forward. The first block of variables corresponds to the matrix B and the second block of variables corresponds to the matrix C. Hence, the smooth coupling function of both blocks is given by

$$H(B,C) = \frac{1}{2}\|A - BC\|^2 .$$

The block-gradients and respective block-Lipschitz constants are easily computed via

$$\begin{aligned} \nabla_B H(B,C) &= (BC - A)\,C^T, \quad L_1(C) = \left\|CC^T\right\|_2 , \\ \nabla_C H(B,C) &= B^T (BC - A), \quad L_2(B) = \left\|B^T B\right\|_2 . \end{aligned}$$

The nonsmooth function for the first block, $f_1(B)$, is given by the non-negativity constraint $B \geq 0$ and the ℓ_0 sparsity constraint applied to each column of the matrix B, that is,

$$f_1(B) = \begin{cases} 0, & B \geq 0,\ \|b_i\|_0 \leq s,\ i = 1,2,\ldots,r, \\ \infty, & \text{else.} \end{cases}$$

Although this function is an indicator function of a nonconvex set, it is shown in [10], that its proximal mapping can be computed very efficiently (in fact in linear time) via

$$B = \operatorname{prox}_{f_1}\left(\hat{B}\right) \Leftrightarrow b_i = T_s\left(\hat{b_i^+}\right), \quad i = 1,2,\ldots,r,$$

where $\hat{b_i^+} = \max\{\hat{b}_i, 0\}$ denotes an elementwise truncation at zero and the operator $T_s\left(\hat{b_i^+}\right)$ corresponds to first sorting the values of $\hat{b_i^+}$, keeping the s largest values and setting the remaining $m - s$ values to zero.

The nonsmooth function corresponding to the second block is simply the indicator function of the non-negativity constraint of C, that is,

$$f_2(C) = \begin{cases} 0, & C \geq 0, \\ \infty, & \text{else,} \end{cases}$$

and its proximal mapping is trivially given by

$$C = \operatorname{prox}_{f_2}\left(\hat{C}\right) = \hat{C^+},$$

which is again an elementwise truncation at zero.

In our numerical example we set $r = 25$, that is we seek for 25 sparse basis images. Figure 2 shows the results of the basis faces when running the iPALM algorithm for different sparsity settings. One can see that for smaller values of s, the algorithm leads to more compact representations. This might improve the generalization capabilities of the representation.

In order to investigate the properties of iPALM on the inertial parameters, we run the algorithm for a specific sparsity setting ($s = 33\%$) using different constant settings of α_i and β_i for $i = 1, 2$. From our convergence theory (see Proposition 4.3) it follows that the parameters δ_i, $i = 1, 2$, have to be chosen as constants. However, in practice we shall use a varying parameter δ_i^k, $i = 1, 2$, and assume that the parameters will become constant after a certain number of iterations. Observe, that the nonsmooth function of the first block is nonconvex (ℓ_0 constraint), while the nonsmooth function of the second block is convex (non-negativity constraint) which will affect the rules to compute the parameters (see Remark 4.1).

We compute the parameters τ_i^k, $i = 1, 2$, by invoking (4.7) and (4.8), where we practically choose $\varepsilon = 0$. Hence, for the first, completely nonconvex block, we have

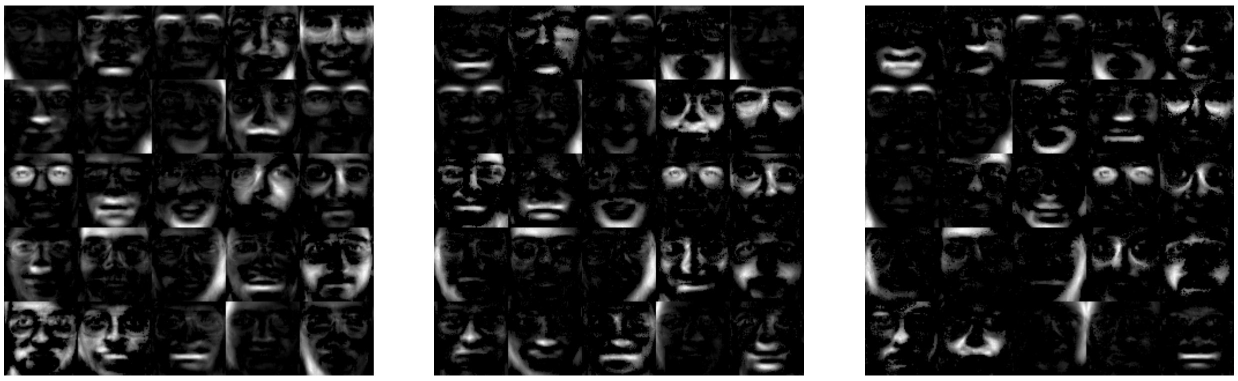
$$\delta_1^k = \frac{\alpha_1^k + \beta_1^k}{1 - 2\alpha_1^k} L_1(x_2^k), \quad \tau_1^k = \frac{\delta_1^k + \left(1 + \beta_1^k\right) L_1(x_2^k)}{1 - \alpha_1^k} \quad \Rightarrow \quad \tau_1^k = \frac{1 + 2\beta_1^k}{1 - 2\alpha_1^k} L_1(x_2^k),$$

from which it directly follows that $\alpha \in [0, 0.5)$.

For the second block, where the nonsmooth function is convex we follow Remark 4.1 and invoke (4.10) and (4.11) to obtain

$$\delta_2^k = \frac{\alpha_2^k + 2\beta_2^k}{2\left(1 - 2\alpha_2^k\right)} L_2(x_1^{k+1}), \quad \tau_2^k = \frac{\delta_2^k + \left(1 + \beta_2^k\right) L_2(x_1^{k+1})}{2 - \alpha_2^k} \quad \Rightarrow \quad \tau_2^k = \frac{1 + 2\beta_2^k}{2\left(1 - \alpha_2^k\right)} L_2(x_1^{k+1}).$$

Comparing this parameter to the parameter of the first block we see that now $\alpha \in [0, 1)$ and the value of τ is smaller by a factor of 2. Hence, convexity in the nonsmooth function allows for twice larger steps.



(a) $s = 50\%$ (b) $s = 33\%$ (c) $s = 25\%$

Figure 2: 25 basis faces using different sparsity settings. A sparsity of $s = 25\%$ means that each basis face contains only 25% non-zero pixels. Clearly, stronger sparsity leads to a more compact representation.

In Tables 1 and 2, we report the performance of the iPALM algorithm for different settings of the inertial parameters α_i^k and β_i^k, $i = 1, 2$ and $k \in \mathbb{N}$. Since we are solving a

	$K = 100$	$K = 500$	$K = 1000$	$K = 5000$	time (s)
$\alpha_{1,2} = \beta_{1,2} = 0.0$	12968.17	7297.70	5640.11	4088.22	196.63
$\alpha_{1,2} = \beta_{1,2} = 0.2$	12096.91	8453.29	6810.63	4482.00	190.47
$\alpha_{1,2} = \beta_{1,2} = 0.4$	12342.81	11496.57	9277.11	5617.02	189.27
$\alpha_1 = \beta_1 = 0.2, \alpha_2 = \beta_2 = 0.4$	12111.55	8488.35	6822.95	4465.59	201.05
$\alpha_1 = \beta_1 = 0.4, \alpha_2 = \beta_2 = 0.8$	12358.00	11576.72	9350.37	5593.84	200.11
$\alpha^k_{1,2} = \beta^k_{1,2} = (k-1)/(k+2)$	5768.63	3877.41	3870.98	3870.81	186.62

Table 1: Values of the objective function of the sparse NMF problem after K iterations, using different settings for the inertial parameters α_i and β_i, $i = 1, 2$ and using computation of the exact Lipschitz constant.

	$K = 100$	$K = 500$	$K = 1000$	$K = 5000$	time (s)
$\alpha_{1,2} = \beta_{1,2} = 0.0$	8926.23	5037.89	4356.65	4005.53	347.17
$\alpha_{1,2} = \beta_{1,2} = 0.2$	8192.71	4776.64	4181.40	4000.41	349.42
$\alpha_{1,2} = \beta_{1,2} = 0.4$	8667.62	4696.64	4249.57	4060.95	351.78
$\alpha_1 = \beta_1 = 0.2, \alpha_2 = \beta_2 = 0.4$	8078.14	4860.74	4274.46	3951.28	353.53
$\alpha_1 = \beta_1 = 0.4, \alpha_2 = \beta_2 = 0.8$	8269.27	4733.76	4243.29	4066.63	357.35
$\alpha^k_{1,2} = \beta^k_{1,2} = (k-1)/(k+2)$	5071.71	3902.91	3896.40	3869.13	347.90
iPiano ($\beta = 0.4$)	14564.65	12200.78	11910.65	7116.22	258.42

Table 2: Values of the objective function of the sparse NMF problem after K iterations, using different settings for the inertial parameters α_i and β_i, $i = 1, 2$ and using backtracking to estimate the Lipschitz constant.

nonconvex problem, we report the values of the objective function after a certain number of iterations ($100, 500, 1000$ and 5000). As already mentioned, setting $\alpha_i = \beta_i = 0$, $i = 1, 2$, reverts the proposed iPALM algorithm to the PALM algorithm [10]. In order to estimate the (local) Lipschitz constants $L_1(x_2^k)$ and $L_2(x_1^{k+1})$ we computed the exact values of the Lipschitz constants by computing the largest eigenvalues of $C^k(C^k)^T$ and $(B^k)^T B^k$, respectively. The results are given in table 1. Furthermore, we also implemented a standard backtracking procedure, see for example [6, 24], which makes use of the descent lemma in order to estimate the value of the (local) Lipschitz constant. In terms of iterations of iPALM, the backtracking procedure generally leads to a better overall performance but each iteration also takes more time compared to the exact computation of the Lipschitz constants. The results based on backtracking are shown in table 2.

We tried the following settings for the inertial parameters.

- **Equal:** Here we used the same inertial parameters for both the first and the second block. In case all parameters are set to be zero, we recover PALM. We observe that the use of the inertial parameters can speed up the convergence but for too large inertial parameters we also observe not as good results as in the case that we use PALM, *i.e.*, no inertial is used. Since the problem is highly nonconvex the final value of the objective function can be missleading since it could correspond to a bad stationary point.

- **Double:** Since the nonsmooth function of the second block is convex, we can take twice larger inertial parameters. We observe an additional speedup by taking twice larger inertial parameters. Again, same phenomena occur here, too large inertial parameters yields inferior performances of the algorithm.

- **Dynamic:** We also report the performance of the algorithm in case we choose dynamic inertial parameters similar to accelerated methods in smooth convex optimization [21]. We use $\alpha_i^k = \beta_i^k = (k-1)\,/\,(k+2)$, $i = 1, 2$, and we set the parameters $\tau_1^k = L_1(x_2^k)$ and $\tau_2^k = L_2(x_1^{k+1})$. One can see that this setting outperforms the other settings by a large margin. Although our current convergence analysis does not support this setting, it shows the great potential of using inertial algorithms when tackling nonconvex optimization problems. The investigation of the convergence properties in this setting will be subject to future research.

Finally, we also compare the proposed algorithm to the iPiano algorithm [24], which is similar to the proposed iPALM algorithm but does not make use of the block structure. Note that in theory, iPiano is not applicable since the gradient of the overall problem is not Lipschitz continuous. However, in practice it turns out that using a backtracking procedure to determine the Lipschitz constant of the gradient is working and hence we show comparisons. In the iPiano algorithm, the step-size parameter τ (α in terms of [24]) was set as

$$\tau \leq \frac{1-2\beta}{L},$$

from which it follows that the inertial parameter β can be chosen in the interval $[0, 0.5)$. We used $\beta = 0.4$ since it give the best results in our tests. Observe that the performance of iPALM is much better compared to the performance of iPiano. In terms of CPU time, one iteration of iPiano is clearly slower than one iteration of iPALM using the exact computation of the Lipschitz constant, because the backtracking procedure is computationally more demanding than the computation of the largest eigenvalues of $C^k(C^k)^T$ and $(B^k)^T B^k$. Comparing the iPiano algorithm to the version of iPALM which uses backtracking, one iteration of iPiano is faster since iPALM needs to backtrack the Lipschitz constants for both of the two blocks.

5.2 Blind Image Deconvolution

In our second example, we consider the well-studied (yet challenging) problem of blind image deconvolution (BID). Given a blurry and possibly noisy image $f \in \mathbb{R}^M$ of $M =$

$m_1 \times m_2$ pixels, the task is to recover both a sharp image u of the same size and the unknown point spread function $b \in \mathbb{R}^N$, which is a small $2D$ blur kernel of size $N = n_1 \times n_2$ pixels. We shall assume that the blur kernel is normalized, that is $b \in \Delta^N$, where Δ^N denotes the standard unit simplex defined by

$$\Delta^N = \left\{ b \in \mathbb{R}^N : \, b_i \geq 0, \; i = 1, 2, \ldots, N, \; \sum_{i=1}^{N} b_i = 1 \right\}. \tag{5.2}$$

Furthermore, we shall assume that the pixel intensities of the unknown sharp image u are normalized to the interval $[0,1]$, that is $u \in U^M$, where

$$U^M = \left\{ u \in \mathbb{R}^M : \, u_i \in [0,1] \, , \; i = 1, 2, \ldots, M \right\}. \tag{5.3}$$

We consider here a classical blind image deconvolution model (see, for example, [28]) defined by

$$\min_{u,b} \left\{ \sum_{p=1}^{8} \phi \left(\nabla_p u \right) + \frac{\lambda}{2} \left\| u *_{m_1,m_2} b - f \right\|^2 : \, u \in U^M, \; b \in \Delta^N \right\}. \tag{5.4}$$

The first term is a regularization term which favors sharp images and the second term is a data fitting term that ensures that the recovered solution approximates the given blurry image. The parameter $\lambda > 0$ is used to balance between regularization and data fitting. The linear operators ∇_p are finite differences approximation to directional image gradients, which in implicit notation are given by

$$\begin{aligned}
\left(\nabla_1 u\right)_{i,j} &= u_{i+1,j} - u_{i,j}, & \left(\nabla_2 u\right)_{i,j} &= u_{i,j+1} - u_{i,j}, \\
\left(\nabla_3 u\right)_{i,j} &= \frac{u_{i+1,j+1} - u_{i,j}}{\sqrt{2}}, & \left(\nabla_4 u\right)_{i,j} &= \frac{u_{i+1,j-1} - u_{i,j}}{\sqrt{2}} \\
\left(\nabla_5 u\right)_{i,j} &= \frac{u_{i+2,j+1} - u_{i,j}}{\sqrt{5}}, & \left(\nabla_6 u\right)_{i,j} &= \frac{u_{i+2,j-1} - u_{i,j}}{\sqrt{5}}, \\
\left(\nabla_7 u\right)_{i,j} &= \frac{u_{i+1,j+2} - u_{i,j}}{\sqrt{5}}, & \left(\nabla_8 u\right)_{i,j} &= \frac{u_{i-1,j+2} - u_{i,j}}{\sqrt{5}},
\end{aligned}$$

for $1 \leq i \leq m_1$ and $1 \leq j \leq m_2$. We assume natural boundary conditions, that is $\left(\nabla_p\right)_{i,j} = 0$, whenever the operator references a pixel location that lies outside the domain. The operation $u *_{m_1,m_2} b$ denotes the usual 2D modulo - m_1, m_2 discrete circular convolution operation defined by (and interpreting the image u and the blur kernel b as 2D arrays)

$$\left(u *_{m_1,m_2} b\right)_{i,j} = \sum_{k=0}^{n_1} \sum_{l=0}^{n_2} b_{k,l} \, u_{(i-k)_{\bmod m_1},(j-l)_{\bmod m_2}}, \quad 1 \leq i \leq m_1, \; 1 \leq j \leq m_2. \tag{5.5}$$

For ease the notation, we will rewrite the 2D discrete convolutions as the matrix vector products of the form

$$v = u *_{m_1,m_2} b \Leftrightarrow v = K\left(b\right) u \Leftrightarrow v = K\left(u\right) b, \tag{5.6}$$

where $K(b) \in \mathbb{R}^{M\times M}$ is a sparse matrix, where each row holds the values of the blur kernel b, and $K(u) \in \mathbb{R}^{M\times N}$ is a dense matrix, where each column is given by a circularly shifted version of the image u. Finally, the function $\phi(\cdot)$ is a differentiable robust error function, that promotes sparsity in its argument. For a vector $x \in \mathbb{R}^M$, the function ϕ is defined as

$$\phi(x) = \sum_{i=1}^{M} \log\left(1 + \theta x_i^2\right), \tag{5.7}$$

where $\theta > 0$ is a parameter. Here, since the argument of the function are image gradients, the function promotes sparsity in the edges of the image. Hence, we can expect that sharp images result in smaller values of the objective function than blurry images. For images that can be well described by piecewise constant functions (see, for example, the books image in Figure 3), such sparsity promoting function might be well suited to favor sharp images, but we would like to stress that this function could be a bad choice for textured images, since a sharp image usually has much stronger edges than the blurry image. This often leads to the problem that the trivial solution (b being the identity kernel and $u = f$) has a lower energy compared to the true solution.

In order to apply the iPALM algorithm, we identify the following functions

$$H(u, b) = \sum_{p=1}^{8} \phi(\nabla_p u) + \frac{\lambda}{2}\left\|u *_{m_1,m_2} b - f\right\|^2, \tag{5.8}$$

which is smooth with block Lipschitz continuous gradients given by

$$\begin{aligned}
\nabla_u H(u, b) &= 2\theta \sum_{p=1}^{8} \nabla_p^T \mathrm{vec}\left(\frac{(\nabla_p u)_{i,j}}{1 + \theta (\nabla_p u)_{i,j}^2}\right)_{i,j=1}^{m_1,m_2} + \lambda K^T(b)\left(K(b)u - f\right),\\
\nabla_b H(u, b) &= \lambda K^T(u)\left(K(u)b - f\right),
\end{aligned}$$

where the operation $\mathrm{vec}(\cdot)$ denotes the formation of a vector from the values passed to its argument. The nonsmooth function of the first block is given by

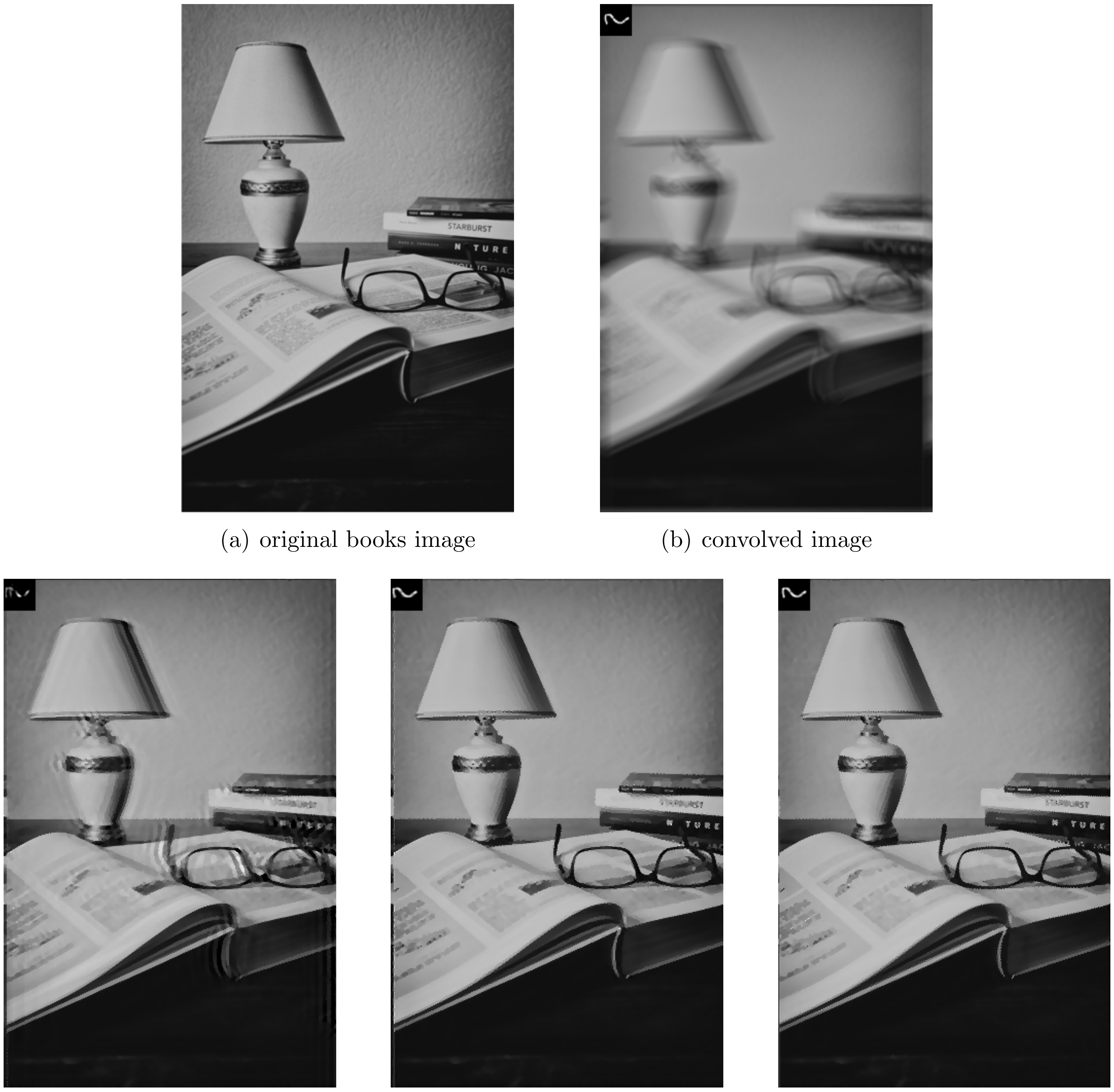
$$f_1(u) = \begin{cases} 0, & u \in U^M, \\ \infty, & \text{else,} \end{cases} \tag{5.9}$$

and the proximal map with respect to f_1 is computed as

$$u = \mathrm{prox}_{f_1}(\hat{u}) \Leftrightarrow u_{i,j} = \max\{0, \min(1, \hat{u_{i,j}})\}. \tag{5.10}$$

The nonsmooth function of the second block is given by the indicator function of the unit simplex constraint, that is,

$$f_2(b) = \begin{cases} 0, & b \in \Delta^N, \\ \infty, & \text{else.} \end{cases} \tag{5.11}$$


(a) original books image

(b) convolved image

(c) $\alpha_{1,2} = \beta_{1,2} = 0$

(d) $\alpha_{1,2} = \beta_{1,2} = 0.4$

(e) $\alpha_{1,2} = \beta_{1,2} = \frac{k-1}{k+2}$

Figure 3: Results of blind deconvolution using $K = 5000$ iterations and different settings of the inertial parameters. The result without inertial terms ($i.e.$, $\alpha_i = \beta_i = 0$ for $i = 1, 2$) is significantly worse compared to the result using inertial terms. The best results are obtained using the dynamic choice of the inertial parameters.

In order to compute the proximal map with respect to f_2, we use the algorithm proposed in [13] which computes the projection onto the unit simplex in $O(N \log N)$ time.

We applied the BID problem to the books image of size $m_1 \times m_2 = 495 \times 323$ pixels

	$K = 100$	$K = 500$	$K = 1000$	$K = 5000$	time (s)
$\alpha_{1,2} = \beta_{1,2} = 0.0$	2969668.92	1177462.72	1031575.57	847268.70	1882.63
$\alpha_{1,2} = \beta_{1,2} = 0.4$	5335748.90	1402080.44	1160510.16	719295.30	1895.61
$\alpha_{1,2} = \beta_{1,2} = 0.8$	5950073.38	1921105.31	1447739.06	780109.56	1888.25
$\alpha_{1,2}^k = \beta_{1,2}^k = (k-1)/(k+2)$	2014059.03	978234.23	683694.72	678090.51	1867.19

Table 3: Values of the objective function of the BID problem, after K iterations and using different settings for the inertial parameters α_i and β_i for $i = 1, 2$.

(see Figure 3). We set $\lambda = 10^6$, $\theta = 10^4$ and generated the blurry image by convolving it with a s-shaped blur kernel of size $n_1 \times n_2 = 31 \times 31$ pixels. Since the nonsmooth functions of both blocks are convex, we can set the parameters as (compare to the first example)

$$\tau_1^k = \frac{1 + 2\beta_1^k}{2\left(1 - \alpha_1^k\right)} L_1(x_2^k) \quad \text{and} \quad \tau_2^k = \frac{1 + 2\beta_2^k}{2\left(1 - \alpha_2^k\right)} L_2(x_1^{k+1}). \tag{5.12}$$

To determine the values of the (local) Lipschitz constants, we used again a backtracking scheme [6, 24]. In order to avoid the trivial solution (that is, $u = f$ and b being the identity kernel) we took smaller descent steps in the blur kernel which was realized by multiplying τ_2^k, $k \in \mathbb{N}$, by a factor of $c = 5$. Note that this form of "preconditioning" does not violate any step-size restrictions as we can always take larger values for τ than the value computed in (5.12).

Table 3 shows an evaluation of the iPALM algorithm using different settings of the inertial parameters α_i and β_i for $i = 1, 2$. First, we observe that the use of inertial parameters lead to higher values of the objective function after a smaller number of iterations. However, for a larger number of iterations the use of inertial forces leads to significantly lower values. Again, the use of dynamic inertial parameters together with the parameter $\tau_1^k = L_1(x_2^k)$ and $\tau_2^k = L_2(x_1^{k+1})$ leads to the best overall performance. In Figure 3 we show the results of the blind deconvolution problem. One can see that the quality of the recovered image as well as the recovered blur kernel is much better using inertial forces. Note that the recovered blur kernel is very close to the true blur kernel but the recovered image appears slightly more piecewise constant than the original image.

5.3 Convolutional LASSO

In our third experiment we address the problem of sparse approximation of an image using dictionary learning. Here, we consider the convolutional LASSO model [36], which is an interesting variant of the well-known patch-based LASSO model [25, 1] for sparse approximations. The convolutional model inherently models the transnational invariance of images, which can be considered as an advantage over the usual patch-based model which treats every patch independently.

	$K = 100$	$K = 200$	$K = 500$	$K = 1000$	time (s)
$\alpha_{1,2} = \beta_{1,2} = 0.0$	336.13	328.21	322.91	321.12	3274.97
$\alpha_{1,2} = \beta_{1,2} = 0.4$	329.20	324.62	321.51	319.85	3185.04
$\alpha_{1,2} = \beta_{1,2} = 0.8$	325.19	321.38	319.79	319.54	3137.09
$\alpha_{1,2}^k = \beta_{1,2}^k = (k-1)/(k+2)$	323.23	319.88	318.64	318.44	3325.37

Table 4: Values of the objective function for the convolutional LASSO model using different settings of the inertial parameters.

The idea of the convolutional LASSO model is to learn a set of small convolution filters $d_j \in \mathbb{R}^{l\times l}$, $j = 1, 2, \ldots, p$, such that a given image $f \in \mathbb{R}^{m\times n}$ can be written as $f \approx \sum_{j=1}^{p} d_j *_{m,n} v_j$, where $*_{m,n}$ denotes again the $2D$ modulo m, n discrete circular convolution operation and $v_j \in \mathbb{R}^{m\times n}$ are the corresponding coefficient images which are assumed to be sparse. In order to make the convolution filters capture the high-frequency information in the image, we fix the first filter d_1 to be a Gaussian (low pass) filter g with standard deviation σ_l and we set the corresponding coefficient image v_j equal to the initial image f. Furthermore, we assume that the remaining filters d_j, $j = 2, 3, \ldots, p$ have zero mean as well as a ℓ_2-norm less or equal to 1. In order to impose a sparsity prior on the coefficient images v_j we make use of the ℓ_1-norm. The corresponding objective function is hence given by

$$\min_{(d_j)_{j=1}^p, (v_j)_{j=1}^p} \sum_{j=1}^{p} \lambda \left\| v_j \right\|_1 + \frac{1}{2} \left\| \sum_{j=1}^{p} d_j *_{m,n} v_j - f \right\|_2^2, \tag{5.13}$$

$$\text{s.t. } d_1 = g,\ v_1 = f\ \sum_{a,b=1}^{l} (d_j)_{a,b} = 0,\ \|d_j\|_2 \le 1,\ j = 2, 3, \ldots, p,$$

where the parameter $\lambda > 0$ is used to control the degree of sparsity. It is easy to see that the convolutional LASSO model nicely fits to the class of problems that can be solved using the proposed iPALM algorithm. We leave the details to the interested reader. In order to compute the (local) Lipschitz constants we again made use of a backtracking procedure and the parameters τ_i^k, $i = 1, 2$ were computed using (5.12).

We applied the convolutional LASSO problem to the Barbara image of size 512×512 pixels, which is shown in Figure 4. The Barbara image contains a lot of stripe-like texture and hence we expect that the learned convolution filters will contain these characteristic structures. In our experiment, we learned a dictionary made of 81 filter kernels, each of size 9×9 pixels. The regularization parameter λ was set to $\lambda = 0.2$. From Figure 4 it can be seen that the learned convolution filters indeed contain stripe-like structures of different orientations but also other filters that are necessary to represent the other structures in the image. Table 4 summarizes the performance of the iPALM algorithm for different settings of the inertial parameters. From the results, one can see that larger settings of the inertial parameters lead to a consistent improvement of the convergence speed. Again, using a dynamic choice of the inertial parameters clearly outperforms the other settings.

For illustration purposes we finally applied the learned dictionary to denoise a noisy variant of the same Barbara image. The noisy image has been generated by adding zero-mean Gaussian noise with standard deviation $\sigma = 0.1$ to the original image. For denoising we use the previously learned dictionary d and minimizing the convolutional LASSO problem only with respect to the coefficient images v. Note that this is a convex problem and hence,it can be efficiently minimized using for example the FISTA algorithm [6]. The denoised image is again shown in Figure 4. Observe that the stripe-like texture is very well preserved in the denoised image.

6 Conclusion

In this paper we proposed iPALM which an inertial variant of the Proximal Alternating Linearized Minimization (PALM) method proposed in [10] for solving a broad class of nonconvex and nonsmoooth optimization problems consisting of block-separable nonsmooth, nonconvex functions with easy to compute proximal mappings and a smooth coupling function with block-Lipschitz continuous gradients. We studied the convergence properties of the algorithm and provide bounds on the inertial and step-size parameters that ensure convergence of the algorithm to a critical point of the problem at hand. In particular, we showed that in case the objective function satisfies the Kurdyka-Łojasiewicz (KL) property, we can obtain finite length property of the generated sequence of iterates. In several numerical experiments we show the advantages of the proposed algorithm to minimize a number of well-studied problems and image processing and machine learning. In our experiments we found that choosing the inertial and step-size parameters dynamically, as pioneered by Nesterov [22], leads to a significant performance boost, both in terms of convergence speed and convergence to a "better" critical point of the problem. Our current convergence theory does not support this choice of parameters but developing a more general convergence theory will be interesting and a subject for future research.

7 Appendix A: Proof of Lemma 4.1

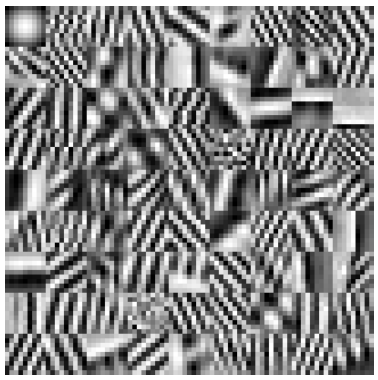
We first recall the result that should be proved.

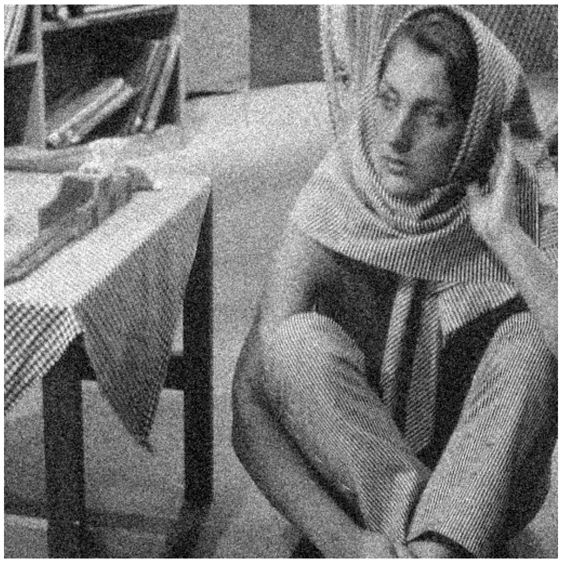
Lemma 7.1. *Consider the functions $g : \mathbb{R}_+^5 \to \mathbb{R}$ and $h : \mathbb{R}_+^5 \to \mathbb{R}$ defined as follow*

$$\begin{aligned} g(\alpha, \beta, \delta, \tau, L) &= \tau(1 - \alpha) - (1 + \beta) L - \delta, \\ h(\alpha, \beta, \delta, \tau, L) &= \delta - \tau\alpha - L\beta. \end{aligned}$$

Let $\varepsilon > 0$ and $\bar{\alpha} > 0$ be two real numbers for which $0 \leq \alpha \leq \bar{\alpha} < 0.5(1 - \varepsilon)$. Assume, in

(a) Original Barbara image


(b) Learned 9×9 dictionary


(c) Noisy Barbara image ($\sigma = 0.1$)

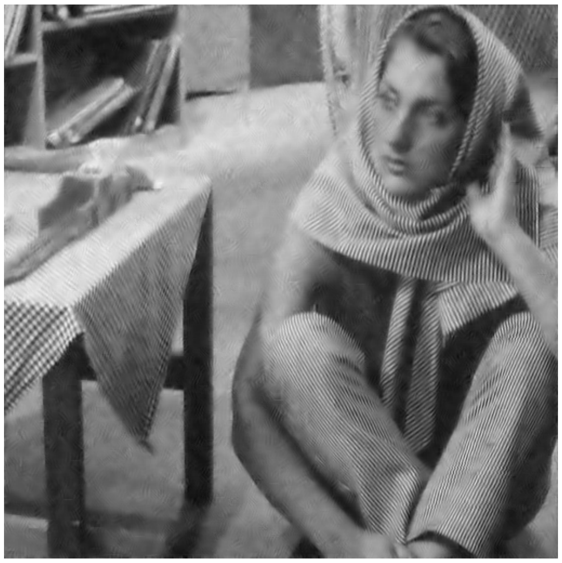

(d) Denoised Barbara image (PSNR=28.33)

Figure 4: Results of dictionary learning using the convolutional LASSO model. Observe that the learned dictionary very well captures the stripe-like texture structures of the Barbara image.

addition, that $0 \leq L \leq \lambda$ *for some* $\lambda > 0$ *and* $0 \leq \beta \leq \bar{\beta}$ *with* $\bar{\beta} > 0$*. If*

$$\delta_* = \frac{\bar{\alpha} + \bar{\beta}}{1 - \varepsilon - 2\bar{\alpha}} \lambda, \tag{7.1}$$

$$\tau_* = \frac{(1+\varepsilon)\,\delta_* + (1+\beta)\,L}{1-\alpha}, \tag{7.2}$$

then $g(\alpha,\beta,\delta_*,\tau_*,L)=\varepsilon\delta_*$ *and* $h(\alpha,\beta,\delta_*,\tau_*,L)\geq\varepsilon\delta_*$.

Proof. From the definition of g we immediately obtain that

$$g(\alpha,\beta,\delta_*,\tau_*,L)=\tau_*(1-\alpha)-(1+\beta)L-\delta_*=\varepsilon\delta_*,$$

this proves the first desired result. We next simplify $h(\alpha,\beta,\delta_*,\tau_*,L)-\varepsilon\delta_*$ as follows

$$\begin{aligned} h(\alpha,\beta,\delta_*,\tau_*,L)-\varepsilon\delta_* &= (1-\varepsilon)\delta_*-\tau_*\alpha-L\beta \\ &= (1-\varepsilon)\delta_*-\frac{(1+\varepsilon)\delta_*+L(1+\beta)}{1-\alpha}\alpha-L\beta \\ &= (1-\varepsilon)\delta_*-\frac{(1+\varepsilon)\delta_*\alpha+L(1+\beta)\alpha+L\beta(1-\alpha)}{1-\alpha} \\ &= (1-\varepsilon)\delta_*-\frac{(1+\varepsilon)\delta_*\alpha+L(\alpha+\beta)}{1-\alpha}. \end{aligned}$$

Thus, we only remain to show that

$$(1-\varepsilon)\delta_*-\frac{(1+\varepsilon)\delta_*\alpha+L(\alpha+\beta)}{1-\alpha}\geq 0.$$

Indeed, simple manipulations yields the following equivalent inequality

$$(1-\varepsilon-2\alpha)\delta_*\geq L(\alpha+\beta). \tag{7.3}$$

Using now (7.1) and the facts that $\alpha\leq\bar{\alpha}$, $\beta\leq\bar{\beta}$ and $0<L\leq\lambda$ we obtain that (7.3) holds true. This completes the proof of the lemma. □

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, 2006.

[2] F. Alvarez and H. Attouch. An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping. *Set-Valued Analysis*, 9(1-2):3–11, 2001.

[3] H. Attouch and J. Bolte. On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Math. Program.*, 116(1-2, Ser. B):5–16, 2009.

[4] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: an approach based on the Kurdyka-L ojasiewicz inequality. *Math. Oper. Res.*, 35(2):438–457, 2010.

[5] H. Attouch, J. Bolte, and B. F. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Math. Program.*, 137(1-2, Ser. A):91–129, 2013.

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.

[7] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation.* Prentice-Hall International Editions, Englewood Cliffs, NJ, 1989.

[8] J. Bolte, A. Daniilidis, and A. Lewis. The Lojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. *SIAM J. Optim.*, 17(4):1205–1223, 2006.

[9] J. Bolte, A. Daniilidis, O. Ley, and L. Mazet. Characterizations of łojasiewicz inequalities: subgradient flows, talweg, convexity. *Trans. Amer. Math. Soc.*, 362(6):3319–3363, 2010.

[10] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Math. Program. Series A*, 146:459–494, 2014.

[11] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Model. Simul.*, 4(4):1168–1200, 2005.

[12] Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: a novel approach. *Math. Program.*, 145(1-2, Ser. A):451–482, 2014.

[13] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T.Chandra. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 272–279, 2008.

[14] R. .I.BoŢ and E. R. Csetnek. An inertial tseng's type proximal algorithm for nonsmooth and nonconvex optimization problems. *Journal of Optimization Theory and Applications*, pages 1–17, 2015.

[15] D. D. Lee and H. S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401:788–791, 1999.

[16] A. Levin, Y. Weiss, F. Durand, and W.T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Computer Vision and Patter Recognition (CVPR)*, 2009.

[17] P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Applied Mathematics*, 16(6):964–979, 1979.

[18] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

[19] B. S. Mordukhovich. *Variational analysis and generalized differentiation. I*, volume 330 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 2006. Basic theory.

[20] J. J. Moreau. Proximitéet dualité dans un espace hilbertien. *Bull. Soc. Math. France*, 93:273–299, 1965.

[21] Y. Nesterov. *Introductory Lectures on Convex Optimization*, volume 87 of *Applied Optimization*. Kluwer Academic Publishers, Boston, MA, 2004.

[22] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Dokl. Akad. Nauk SSSR*, 269(3):543–547, 1983.

[23] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[24] P. Ochs, Y. Chen, T. Brox, and T.Pock. iPiano: inertial proximal algorithm for nonconvex optimization. *SIAM J. Imaging Sci.*, 7(2):1388–1419, 2014.

[25] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311 – 3325, 1997.

[26] P. Paatero and U. Tapper. Positive matrix factorization: A nonnegative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

[27] R. Peharz and F. Pernkopf. Sparse nonnegative matrix factorization with 0-constraints. *Neurocomputing*, 80(0):38 – 46, 2012.

[28] D. Perrone, R. Diethelm, and P. Favaro. Blind deconvolution via lower-bounded logarithmic image priors. In *International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 2015.

[29] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[30] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *{U.S.S.R.} Comput. Math. and Math. Phys.*, 4(5):1–17, 1964.

[31] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*, volume 317 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1998.

[32] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *WACV*, pages 138–142. IEEE, 1994.

[33] S. Sra and I. S. Dhillon. Generalized nonnegative matrix approximations with Bregman divergences. In Y. Weiss, B. Schölkopf, and J.C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 283–290. MIT Press, 2006.

[34] Y. Xu and W. Yin. A globally convergent algorithm for nonconvex optimization based on block coordinate update. Technical report, Arxiv preprint, 2014.

[35] S.K. Zavriev and F.V. Kostyuk. Heavy-ball method in nonconvex optimization problems. *Computational Mathematics and Modeling*, 4(4):336–341, 1993.

[36] M. Zeiler, D. Krishnan, G. Taylor, and R. Fergus. Deconvolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2528–2535, 2010.