Direct Optimization of the dictionary learning problem

Alain Rakotomamonjy

Direct Optimization of the Dictionary Learning Problem

Alain Rakotomamonjy*
LITIS, EA 4108 - Université/INSA de Rouen
Avenue de l'Université - 76801 Saint-Etienne du Rouvray Cedex
firstname.lastname@insa-rouen.fr

***Abstract*—A novel way of solving the dictionary learning problem is proposed in this paper. It is based on a so-called direct optimization as it avoids the usual technique which consists in alternatively optimizing the coefficients of a sparse decomposition and in optimizing dictionary atoms. The algorithm we advocate simply performs a joint proximal gradient descent step over the dictionary atoms and the coefficient matrix. After having derived the algorithm, we also provided in-depth discussions on how the stepsizes of the proximal gradient descent have been chosen. In addition, we uncover the connection between our direct approach and the alternating optimization method for dictionary learning. We have shown that it can be applied to a broader class of non-convex optimization problems than the dictionary learning one. As such, we have denoted the algorithm as a *one-step block-coordinate proximal gradient descent*. The main advantage of our novel algorithm is that, as suggested by our simulation study, it is more efficient than alternating optimization algorithms.**

***Index Terms*—dictionary learning, non-convex proximal, one-step block-coordinate descent .**

I. Introduction

In the recent years, a lot of works have been devoted to the problem of sparse representation of signals and images. Several research communities [17], [26], [12] have focused on this problem in order to develop some new tools for analyzing signals or images, to select features for discrimination tasks or to study the theoretical properties of sparse representation. This large success of "sparsity" is essentially due to the fact that many real-world signals or natural images can be represented as a linear combination of few representative elements, denoted as atoms or dictionary elements.

One of the key problems related to sparse representation is the choice of the dictionary on which the signal of interest is decomposed. One simple approach is to consider an off-the-shelf dictionary such as wavelet basis, wavelet packet basis, Gabor atoms or Discrete Cosine Basis. A recent trend which has been shown to achieve state-of-the art results on many low-level signal and image processing tasks is to learn the dictionary from the data [11], [15], [18], [14], [10]. Several algorithms for dictionary learning have been proposed and most of them are based on an alternating optimization scheme which involves a signal sparse coding step and a dictionary optimization step [19], [15], [11], [17], [13], [25]. Recently, approaches based on alternating optimization, which also allows sparse code updates during the dictionary optimization step, have been proposed [1], [9].

This paper deviates from this mainstream approach and proposes a novel way of solving the dictionary problem by means of what we call a "direct" optimization. Indeed, we propose an algorithm which does not perform alternating optimization but does at each iteration, jointly optimize both the dictionary elements and the sparse coding of each signal. This idea of joint optimization has already been explored in [1] and [9] and the approach we present here goes farther as we do optimize all the elements of the dictionary learning problem at once, without restriction of coefficient supports.

The algorithm for this "direct" optimization is based on a proximal gradient descent and it has been derived by casting the dictionary learning problem into a recent non-convex proximal splitting framework introduced by Sra [22]. We show that, with slight additional constraints, the dictionary learning problem satisfies all conditions necessary to ensure convergence of the algorithm. A careful analysis of the stepsize involved in the proximal gradient descent is also provided. Interestingly, we also point out how our novel algorithm can be related to alternating optimization approach like the one proposed by Yaghoobi et al. [25]. In addition, we have cast the algorithm in a broader framework and have shown that it can actually solve a larger class of non-convex optimization problems which involve a smooth function and a sum of separable convex and possibly non-smooth functions. Because of its specific nature, we have denoted the algorithm as a *one-step block-coordinate proximal gradient descent*. The main practical result achieved by our approach is that the dictionary learning problem can be solved significantly faster compared to other existing algorithms such as the one of Yaghoobi et al. [25] which is already faster than methods like K-SVD [1].

The paper is organized as follows. Section II introduces the dictionary learning framework we are dealing with and reviews some related works. The non-convex proximal splitting framework of Sra [22] is described in Section III as well as its application to dictionary learning. Most parts of this section also discuss some important algorithmic issues including step-size selection. Section IV uncovers the relationship between direct and alternating optimization approaches and explains how the algorithm we propose for dictionary learning applies to a larger class of problems. Experimental studies described in Section V clearly establish the practical benefit of using direct optimization of the dictionary learning problem. We indeed show that such an approach is significantly faster than alternating optimization without compromising quality of the

learned dictionary. The code used in this paper is available on the author's website.

II. Dictionary learning

In this section, we formally state the dictionary learning problem we are interested in and present the current state-of-the-art methods for solving this problem.

A. Simultaneous Sparse approximation and dictionary learning

The problem of simultaneous sparse approximation (SSA) [23], [20] consists in looking for a sparse decomposition, over a fixed dictionary, of a set of signals under the constraint that, to some extents, they all share the same sparsity profile. Formally, this translates as follows. Suppose we have a set of L signals $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_L]$, with $\mathbf{X} \in \mathbb{R}^{N\times L}$, a dictionary $\mathbf{D} \in \mathbb{R}^{N\times M}$, then the SSA problem solves :

$$\min_{\mathbf{A}\in\mathbb{R}^{M\times L}} \frac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda\Omega(\mathbf{A}) \tag{1}$$

where $\Omega(\mathbf{A})$ is a sparsity-inducing regularizer on $\mathbf{A}$ and λ a trade-off parameter that balances the data fitting term (the square loss) and the regularization term. Typically, for inducing a shared sparsity profile on the signal approximation, which means that all $\mathbf{x}_i$ should preferably be approximated by the same set of dictionary elements, one can consider a mixed-norm over the rows of $\mathbf{A}$ of the form :

$$\Omega(\mathbf{A}) = \left(\sum_{i=1}^{M} \|\mathbf{A}_{i,\cdot}\|_q^p\right)^{1/p} \tag{2}$$

where typically $p = 1$ and $q \in \{2, \infty\}$ [8], [23]. Note that for all matrices, we will respectively denote as $\mathbf{A}_{i,\cdot}$ and $\mathbf{A}_{\cdot,j}$ their i-th row and the j-th column. When the sparsity profile is known to be different for each single signal to be approximated, one instead may consider $p = 1$ and $q = 1$ [5] which allows $\Omega(\mathbf{A})$ to decouple and thus untie coefficients of $\mathbf{A}$.

Dictionary learning problems go beyond simultaneous sparse approximation by jointly optimizing over the coefficient matrix $\mathbf{A}$ and the dictionary $\mathbf{D}$ leading then to the optimization problem

$$\min_{\mathbf{A},\mathbf{D}\in\mathcal{D}} \frac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda\Omega(\mathbf{A}) \tag{3}$$

where $\mathcal{D}$ is a set that imposes some constraints on the dictionary elements. This set is usually chosen according to prior knowledge on the problem or chosen so as to help in resolving the scale invariance of the problem. The most frequent constraints imposed to $\mathbf{D}$, and already used in other works, are the one that induces each column of $\mathbf{D}$ to have unit ℓ_2 norm or the one that enforces $\mathbf{D}$ to have an unit Frobenius norm [11], [15], [25]. In a more general form, we can consider the following problem for dictionary learning

$$\min_{\mathbf{A},\mathbf{D}} \Phi(\mathbf{A},\mathbf{D}) = \frac{1}{2}\|\mathbf{X}-\mathbf{D}\mathbf{A}\|_F^2+\lambda_A\Omega_A(\mathbf{A})+\lambda_D\Omega_D(\mathbf{D}) \tag{4}$$

Algorithm 1 : Alternating optimization for dictionary learning

1: set k=1, initialize $\mathbf{A}^1$, $\mathbf{D}^1$
2: **repeat**
3: $\quad\mathbf{D}^{k+1} = \min_{\mathbf{D}} \Phi(\mathbf{A}^k, \mathbf{D})$
4: $\quad\mathbf{A}^{k+1} = \min_{\mathbf{A}} \Phi(\mathbf{A}, \mathbf{D}^{k+1})$
5: $\quad k \leftarrow k + 1$
6: **until** stopping criterion is met

where $\Omega_A(\cdot)$ and $\Omega_D(\cdot)$ are general regularizers on $\mathbf{A}$ and $\mathbf{D}$ that can involve mixed terms (*e.g* non-negativity and sparseness) or indicator functions that are related to projection on convex sets defining some constraints. However, for a sake of simplicity, we will consider for now on, the constraints on $\mathbf{D}$ as the projection on the unit-norm ball, denoted as $\Pi(\mathbf{D})$, which imposes each dictionary atom norm to be bounded by 1, and the regularizer on $\mathbf{A}$ as $\lambda_A\|\mathbf{A}\|_1 = \lambda_A \sum\sum |a_{i,j}|$.

B. Related works

Several methods for solving the dictionary learning problem given in Equation (4) have been investigated in the last decade. These algorithms usually solve problem (4) by means of an alternating optimization procedure which is summarized in Algorithm 1. Basically, this algorithm consists in alternatively learning the sparse approximation when the dictionary is considered fixed and then in updating the dictionary with fixed matrix approximation $\mathbf{A}$. For instance, Engan et al. [11] consider the problem with no constraints on $\mathbf{A}$ and impose unit-norm dictionary elements. Accordingly, the solution at each step can be straightforwardly obtained by solving the related least-square problem over $\mathbf{A}$ and $\mathbf{D}$. For the $\mathbf{D}$ update, this gives

$$\mathbf{D}^{k+1} = \mathbf{X}(\mathbf{A}^k)^\top\left(\mathbf{A}^k(\mathbf{A}^k)^\top\right)^{-1} \tag{5}$$

and unit-norm atoms are obtained by normalizing each column of $\mathbf{D}^{k+1}$. This results in an algorithm known as the method of optimal directions (MOD) algorithm. In a similar way, Kreutz-Delgado et al. have proposed an alternating optimization algorithm that solves a Bayesian interpretation model of the dictionary learning problem [15].

The recent work of Yaghoobi et al. investigates algorithms for solving problem (4) when several types of constraints and regularizations on $\mathbf{D}$ as well as sparsity-inducing regularizers on $\mathbf{A}$ are in play. For instance, they consider both bounded Frobenius norm or bounded ℓ_2 norm constraints on the dictionary atoms jointly to sparsity-inducing regularizers of the same form of the one given in Equation (2) for the coefficient matrix. The main idea of their optimization algorithm is based on the use of a surrogate function that majorizes the data fidelity term $\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2$ in problem (4). For instance, when updating the dictionary $\mathbf{D}$, they replace the problem of minimizing $\Phi(\mathbf{A}^k, \mathbf{D})$ with respects to $\mathbf{D}$ by the following surrogate problem :

$$\begin{aligned}\min_{\mathbf{D}} \quad & \tfrac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}^k\|_F^2 + \Pi(\mathbf{D}) + C_D\|\mathbf{D} - \mathbf{D}^{k-1}\|_F^2 \\ & - \|\mathbf{D}\mathbf{A}^k - \mathbf{D}^{k-1}\mathbf{A}^k\|_F^2\end{aligned}$$

where $C_D > \|(\mathbf{A}^k)^\top \mathbf{A}^k\|_2$ is a constant that ensures that the majorization holds. Yaghoobi et al. solve this problem through some Landweber iterations which first consist in updating $\mathbf{D}$ according to a gradient descent step and then in projecting this novel update on the convex set imposed by the constraints. This surrogate approach helps them in dealing with several types of constraints over the dictionary.

Still in this framework of alternating optimization on $\mathbf{A}$ and $\mathbf{D}$, several interesting research outcomes have also recently been introduced by the machine learning community [16], [17]. For instance, Jenatton et al. have investigated methods that are able to exploit relationship between the dictionary elements through the definition of appropriate regularizers $\Omega_A(\mathbf{A})$ [13]. They proposed algorithms that are able to learn some structures on the dictionary elements [14].

The K-SVD of Aharon et al. [1] presents a different perspective on the dictionary learning problem. Indeed, their dictionary update step also involves some updates of the approximation coefficients. This update is performed through a SVD decomposition on a representation error matrix of the set of signal $\mathbf{X}$. The resulting decomposition redefines a novel dictionary element and the approximation coefficient related to an atom. This SVD decomposition is performed M times (M being the dictionary size) for each dictionary update in the alternating optimization algorithm (1). A recent work of Dai et al. [9] extends the K-SVD algorithm by allowing, in the dictionary update step, the simultaneous optimization of several dictionary elements and the related approximation coefficients. Owing to this possibility of optimizing several atoms at a time, the running efficiency of their algorithm is better than the one of K-SVD although still worse than the one of MOD. It is also interesting to mention that two works have lately investigated the problem of online dictionary learning [21], [17]. At the contrary of the batch approaches presented above, in the online framework, it is supposed that signal examples $\{\mathbf{x}_i\}$ are available on the fly. Skretting et al. [21] proposes an extension of the method of optimal direction based on recursive least-squares for dealing with this online framework while Mairal et al. [17] introduce a stochastic approximation algorithm that is able to take into account constraints on $\mathbf{A}$ and $\mathbf{D}$.

While most of the above-described works solve the dictionary learning problem by means of alternating optimization or block-coordinatewise descent, we take another route by investigating an optimization method which addresses directly the non-convex problem (4). The algorithm we advocate is based on a proximal gradient descent that, at each iteration, updates both the dictionary $\mathbf{D}$ and the coefficient matrix $\mathbf{A}$. While several studies concern the use of proximal methods in the convex case (see for instance [6], [7] and references therein), few works have addressed their applications in the non-convex case. In particular, we can mention the seminal works of Attouch et al. [2] and Sra [22] which both propose a proximal framework for solving non-convex problems. In this work, we will consider the framework introduced in [22] since it is especially tailored for non-convex problems with composite objective functions, one of which is smooth. However, its application to dictionary learning poses some issues regarding the choice of the stepsize. We propose several ways of selecting this stepsize based on local estimation of the objective function curvature. A backtracking of the stepsize has also been introduced and convergence of the sequence of objective values of (4) is discussed. From a practical point of view, we show that this direct optimization approach is far more efficient than alternating optimization methods since it avoids unnecessary and costly updates at the early stage of the learning process.

III. Non-convex proximal splitting for dictionary learning

In this section, we first introduce the optimization framework we will consider. We then discuss how it can be applied to the dictionary learning problem (4). Technical aspects related to the choice of the stepsizes for the proximal gradient descent are also examined.

A. Non-convex proximal splitting algorithm

One of the main reasons for solving the dictionary learning problem (4) using an alternating optimization algorithm is that the global problem is not convex with respects to the optimization variables whereas each alternating problem is. Hence, it seems more beneficial to take into account this property. We show in the sequel that direct optimization is equally simple to implement and leads to a faster algorithm.

Our direct optimization algorithm is based on the framework introduced by Sra [22] which aim is to solve problems of the form

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) + g(\mathbf{x}) \tag{6}$$

where $f(\cdot)$ is some non-convex function of $\mathbf{x}$ which has Lipschitz gradient and $g(\mathbf{x})$ a lower semi-continuous, possibly non-smooth, convex function of the form $\psi(\mathbf{x}) + \delta(\mathbf{x}|\mathcal{X})$, where $\delta(\mathbf{x}|\mathcal{X})$ is the indicator function of a compact subset of $\mathbb{R}^d$ denoted $\mathcal{X}$:

$$\delta(\mathbf{x}|\mathcal{X}) = \begin{cases} 0 & \text{if } \mathbf{x} \in \mathcal{X} \\ +\infty & \text{otherwise} \end{cases}$$

The fact that the optimization variable $\mathbf{x}$ needs to lie in a compact set is of primary importance for the convergence analysis [22]. For the non-smooth function g, we define as its proximity operator $\text{prox}_{\eta g}$, parametrized by $\eta > 0$, the mapping [7]

$$\text{prox}_{\eta g}(\mathbf{y}) : \mathbf{y} \mapsto \arg\min_{\mathbf{x}} \frac{1}{2\eta}\|\mathbf{x}-\mathbf{y}\|^2 + g(\mathbf{x})$$

The algorithm proposed by Sra for solving problem given in Equation (6) is based on the following iterative scheme :

$$\mathbf{x}^{k+1} = \text{prox}_{\eta_k g}(\mathbf{x}^k - \eta_k \nabla f(\mathbf{x}^k) + \eta_k \zeta(\mathbf{x}_k)) \tag{7}$$

where $\zeta(\mathbf{x}_k)$ is some perturbations that may occur on the gradient computations and the stepsize η_k has to satisfy the conditions :

$$0 < c \leq \eta_k \leq \min(1, 2/L - c) \quad \text{with, } 0 < c < \frac{1}{L} \tag{8}$$

L being the Lipschitz constant of the function ∇f. Interestingly, the following theorem proves convergence of the simple iterative scheme (7) under some mild conditions.

Theorem 1: ([22]) Given a function f gradient Lipschitz on some compact subset $\mathcal{X}$ of $\mathbb{R}^d$, and f bounded from below, given g lower semi-continuous and convex on $\mathcal{X}$. Supposing that the following condition on the perturbations holds, given a fixed $\bar{\eta}$

$$\forall \eta \leq \bar{\eta}, \quad \eta \|\zeta(\mathbf{x})\| \leq \bar{\epsilon}, \quad \text{for some } \bar{\epsilon} \geq 0, \forall \mathbf{x} \in \mathcal{X}, \tag{9}$$

then, the sequence $\{\mathbf{x}^k\}$ generated by iterates (7) admits a limit point $\mathbf{x}^\star$ and there exists a constant $K > 0$ such that

$$\|\mathbf{x}^\star - \text{prox}_g(\mathbf{x}^\star - \nabla f(\mathbf{x}^\star))\| \leq K\epsilon(\mathbf{x}^\star) \tag{10}$$

with $\epsilon(\mathbf{x}^\star)$ being the level of perturbation at $\mathbf{x}^\star$. ■

Note that the condition on the perturbation given in Equation (9) can be understood as an uniform bound on the perturbation's norm at any point $\mathbf{x}$. It tells us that in some ways the perturbation should not be too large. Instead, Equation 10 can be interpreted as a measure of quality of the limit point with respects to a stationary point. Indeed, from simple algebras [6, Prop 3.1], a stationary point of the problem given in Equation 6 can be proven to satisfy the fixed-point condition :

$$\mathbf{x}^* = \text{prox}_{\eta g}(\mathbf{x}^* - \eta \nabla f(\mathbf{x}^*)), \quad \text{for } \eta > 0$$

and left-hand side of Equation 10 is obtained by choosing $\eta = 1$ in the above equation. Hence, the theorem actually gives us the guarantee that a sequence $\{\mathbf{x}_k\}_k$ generated by the numerical scheme (7) with some perturbations on the gradient computations still converges towards a limit point which approximately satisfies the stationary condition defined by Equation (10). It is worth mentioning that if the number of perturbed gradient computations is finite or the perturbation norms converge towards 0, then the sequence's limit point is an exact stationary point.

As shown in the sequel, in practice for the dictionary learning problem, we will be able to compute the exact gradient of the problem. However, the perturbations will be in play essentially when bridging the gap between direct and alternating optimization of the dictionary learning problem.

B. Application to dictionary learning

In order to apply the above-described framework to a dictionary learning problem, Problem (4) must consider some specific constraints, in particular, the fact that the variables to be optimized must leave in some compact sets. Hence, we consider Problem (4) with the following regularizers

$$\Omega_A(\mathbf{A}) = \lambda_{\mathbf{A}} \|\mathbf{A}\|_1 + \delta(\mathbf{A}|\mathcal{A})$$

and

$$\Omega_D(\mathbf{D}) = \Pi(\mathbf{D}) + \delta(\mathbf{D}|\mathcal{D})$$

where $\delta(\mathbf{A}|\mathcal{A})$ and $\delta(\mathbf{D}|\mathcal{D})$ are respectively the indicator functions for the set $\mathcal{A}$ and $\mathcal{D}$ in which $\mathbf{A} \in \mathbb{R}^{M \times L}$ and $\mathbf{D} \in \mathbb{R}^{N \times M}$ live. Note that for the dictionary element learning, $\mathcal{D}$ can be chosen to be $\mathbb{R}^{N \times M}$ since $\Pi(\mathbf{D})$ imposes the dictionary atoms to be in the unit-norm ball which is a compact set. For the coefficient matrix elements, we choose the set $\mathcal{A}$ to be of the form

$$\mathcal{A} = \{a_{i,j} \in \mathbb{R} : |a_{i,j}| \leq B_a, \forall i, j\}$$

which is naturally compact. In practice, we set B_a to be very large so that this indicator $\delta(\mathbf{A}|\mathcal{A})$ has no influence on the problem solution.

Now that our dictionary learning problem fits into the non-convex proximal splitting framework of Sra, we consider as the optimization variables all elements of $\mathbf{D}$ and $\mathbf{A}$ and define

$$f(\mathbf{D}, \mathbf{A}) = \frac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2$$

and $g(\mathbf{D}, \mathbf{A}) = \Pi(\mathbf{D}) + \lambda_A \|\mathbf{A}\|_1 + \delta(\mathbf{A}|\mathcal{A})$. According to these definitions, it is easy to show that the gradient of f with respects to $\mathbf{D}$ and $\mathbf{A}$ writes :

$$\nabla_{\mathbf{D}} f = -(\mathbf{X} - \mathbf{D}\mathbf{A})\mathbf{A}^\top \quad \text{and} \quad \nabla_{\mathbf{A}} f = -\mathbf{D}^\top(\mathbf{X} - \mathbf{D}\mathbf{A}) \tag{11}$$

and the full gradient of f is (with some slight abuse of notations) :

$$\nabla_{\mathbf{D},\mathbf{A}} f = \begin{pmatrix} \nabla_{\mathbf{D}} f \\ \nabla_{\mathbf{A}} f \end{pmatrix}$$

The non-convex proximal splitting iterations given in Equation (7) become :

$$\begin{pmatrix} \mathbf{D}^{k+1} \\ \mathbf{A}^{k+1} \end{pmatrix} = \text{prox}_{\eta_k g}\left(\begin{pmatrix} \mathbf{D}^k \\ \mathbf{A}^k \end{pmatrix} - \eta_k \nabla_{\mathbf{D},\mathbf{A}} f\right)$$

Now, since $g(\mathbf{D}, \mathbf{A})$ is separable in its parameters, these iterations can be splitted in two parts. The iteration on $\mathbf{D}$ is

$$\mathbf{D}^{k+1} = \mathcal{P}_\Pi(\mathbf{D}^k + \eta_k(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)(\mathbf{A}^{k)})^\top) \tag{12}$$

where $\mathcal{P}_\Pi$ defines the following projection on the convex set defined by Π

$$\mathcal{P}_\Pi(\mathbf{D}) = \{\mathbf{d}_i\}_{i=1}^M = \begin{cases} \mathbf{d}_i & \text{if } \|\mathbf{d}_i\|_2 < 1 \\ \frac{\mathbf{d}_i}{\|\mathbf{d}_i\|_2} & \text{otherwise} \end{cases} \tag{13}$$

and the iteration on $\mathbf{A}$ results to be

$$\mathbf{A}^{k+1} = \text{prox}_{\eta_k \lambda_A \|\mathbf{A}\|_1 + \delta(\mathbf{A}|\mathcal{A})}(\mathbf{A}^k + \eta_k(\mathbf{D}^k)^\top(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)) \tag{14}$$

which boils down to

$$\mathbf{A}^{k+1} = \mathcal{S}^{B_a}_{\eta_k \lambda_{\mathbf{A}}}\left(\mathbf{A}^k + \eta_k(\mathbf{D}^k)^\top(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)\right) \tag{15}$$

with $\mathcal{S}^{\mathcal{B}}_\lambda$ being the clipped soft-thresholding operator

$$\mathcal{S}^B_\lambda(\mathbf{A}) = \begin{cases} B & \text{if } (a_{i,j} - \lambda) \geq B \\ -B & \text{if } (a_{i,j} + \lambda) \leq -B \\ a_{i,j} - \lambda \text{sign}(a_{i,j}) & \text{if } \lambda < |a_{i,j}| \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

with $B > \lambda$.

These iterations on $\mathbf{D}$ and $\mathbf{A}$ lead to Algorithm 2. As stated by the above theorem, this algorithm converges as soon as f is gradient Lipschitz on a compact subset of $\mathbb{R}^{N \times M + M \times L}$. Note that since we are able to compute the exact gradient of f, no perturbations occur. Hence, the proximal iterates converge to an exact stationary point of the dictionary learning problem.

Algorithm 2 : Direct optimization for dictionary learning

set k=1, initialize $\mathbf{A}^1$, $\mathbf{D}^1$
choose η_k that satisfies Equation (8)
repeat
  $\mathbf{D}^{k+\frac{1}{2}} = \mathbf{D}^k + \eta_k(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)(\mathbf{A}^k)^\top$
  $\mathbf{D}^{k+1} = \mathcal{P}_\Pi(\mathbf{D}^{k+\frac{1}{2}})$
  $\mathbf{A}^{k+\frac{1}{2}} = \mathbf{A}^k + \eta_k(\mathbf{D}^k)^\top(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)$
  $\mathbf{A}^{k+1} = \mathcal{S}^{B_a}_{\eta_k\lambda_\mathbf{A}}\left(\mathbf{A}^{k+\frac{1}{2}}\right)$
  $k \leftarrow k+1$
until stopping criterion is met

The remainder of the section now discusses on the gradient Lipschitz property of f and how this constant L can be estimated.

C. The Lipschitz constant

The Lipschitz constant of the function ∇f plays an important role in the algorithm since its existence guarantees convergence of Algorithm 2 and its knowledge helps in choosing a proper stepsize for the proximal gradient descent (see conditions (8)). Its existence is proven from the following proposition.

Proposition 1: For the twice differentiable order 4 loss $f(\cdot,\cdot)$ whose entries belong to a compact subset $\mathcal{X}$ of $\mathbb{R}^{N\times M+M\times L}$, there always exists a constant L so that the Hessian of f satisfies the condition

$$\|\nabla^2 f\|_2 \leq L$$

and thus f is Lipschitz gradient of $\mathcal{X}$.

Proof: After some algebras given in the appendix, one can show that each entry of the Hessian of $f(\cdot,\cdot)$ is a continuous function of the entries of $\mathbf{A}$ and $\mathbf{D}$. Since these entries live in a compact subset of $\mathbb{R}$, the Frobenius norm of the Hessian is thus bounded since it is a continuous function of these entries. Thus, there exists L so that

$$\|\nabla^2 f\|_2 \ \leq \|\nabla^2 f\|_F \leq L$$

Hence, the largest eigenvalue of the Hessian is bounded for any $\mathbf{D}$ and $\mathbf{A}$ and thus the function f is gradient Lipschitz [4, p. 48 and p. 54 1.2.5]. ■

The above proposition does not give us any hint on the value of L. Hence, we have left few options for retrieving L, since computing the largest eigenvalue is rapidly intractable as the Hessian is a matrix of $M^2(L+N)^2$ entries.

D. Local estimation of the Lipschitz constant

There exists several situations where even if the Lipschitz constant of the gradient is known, it may be better to consider local estimation of that constant.

Indeed, this Lipschitz constant usually defines the stepsize of a (proximal) gradient descent algorithm and that constant is related to the maximal curvature of the objective function. Hence, if the function, at some point, is rather steep, the global constant L will be large imposing the use of small stepsize updates, even in regions where the function is flat. In such a situation, it is easily understandable that adapting the stepsize to the local function curvature may help in achieving faster convergence.

In this paragraph, we will devise on two ways for estimating the local Lipschitz constant (and thus a proper stepsize). In addition, because, the resulting stepsize may not satisfy conditions (8), we next introduce and discuss a modified non-convex proximal algorithm, with backtracking of the stepsize.

One simple local estimate of the Lispchitz constant can be straightforwardly derived from its definition. At iteration k, given $\mathbf{A}^k$, $\mathbf{A}^{k-1}$, $\mathbf{D}^k$, $\mathbf{D}^{k-1}$, we define the constant L_k as

$$L_k = \frac{\|\nabla f(\mathbf{A}^k,\mathbf{D}^k) - \nabla f(\mathbf{A}^{k-1},\mathbf{D}^{k-1})\|_F}{\sqrt{\|\mathbf{A}^k - \mathbf{A}^{k-1}\|_F^2 + \|\mathbf{D}^k - \mathbf{D}^{k-1}\|_F^2}} \tag{17}$$

Another estimate of the Lipschitz constant can be obtained from the eigenvalues of an approximation of the local Hessian. As we have already stated, computing the largest eigenvalue of the Hessian is intractable, however, it is possible to estimate it, in a rather cheap way.

Indeed, ignoring the off-diagonal block terms $\mathbb{B}$ and $\mathbb{B}^\top$ in the Hessian's definition (see Equation 28) leaves us with a nice block-diagonal matrix which components are either $\mathbf{A}\mathbf{A}^\top$ or $\mathbf{D}^\top\mathbf{D}$. Consequently, we can at iteration k estimate the Lipschitz constant as

$$\tilde{L}_k = \max\left(\|\mathbf{A}^k(\mathbf{A}^k)^\top\|_2, \|(\mathbf{D}^k)^\top\mathbf{D}^k\|_2\right) \tag{18}$$

which results in the largest eigenvalue between those of $\mathbf{A}\mathbf{A}^\top$ and $\mathbf{D}^\top\mathbf{D}$.

Another simple estimate of the Lipschitz constant at an iteration k is to use the estimation at previous iteration. This has the advantage of having an estimation at zero cost and it may be accurate enough if the curvature of the function f is almost constant along some iterations.

E. Non-convex proximal splitting and backtracking

As we have stated, the above-given Lipschitz constants are estimations that may be rough and thus they may not be compliant to the stepsize conditions of the algorithm given in Equation (8).

Consequently, we propose at each iteration to check whether the estimation satisfies a condition, related to the descent lemma [4] and update the current estimation of L if necessary. That lemma states that :

Proposition 2: For a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ with gradient Lipschitz constant L and for any $L' \geq L$, the following equation holds

$$h(\mathbf{x}) \leq h(\mathbf{y}) + \langle \mathbf{x}-\mathbf{y}, \nabla h(\mathbf{y})\rangle + \frac{L'}{2}\|\mathbf{x}-\mathbf{y}\|^2 \quad \forall \mathbf{x},\mathbf{y} \in \mathbb{R}^n$$

For our dictionary learning problem and the function $f(\cdot,\cdot)$, this descent condition at iteration k, can be translated as

$$\begin{aligned} f(\mathbf{D},\mathbf{A}) \leq \quad & f(\mathbf{D}^k,\mathbf{A}^k) + \text{trace}\left((\mathbf{A}-\mathbf{A}^k)^\top\nabla_\mathbf{A} f\right) \\ & + \text{trace}\left((\mathbf{D}-\mathbf{D}^k)^\top\nabla_\mathbf{D} f\right) \\ & + \tfrac{L'}{2}\|\mathbf{A}-\mathbf{A}^k\|_F^2 + \tfrac{L'}{2}\|\mathbf{D}-\mathbf{D}^k\|_F^2 \end{aligned} \tag{19}$$

Again this inequality has been obtained owing to the separability of the trace operator and the Frobenius norm with respect to

$\mathbf{D}$ and $\mathbf{A}$. For backtracking the value of the Lipschitz constant L, and consequently for choosing a correct stepsize, we define a quadratic approximation of $f + g$ at $\mathbf{D}^k$ and $\mathbf{A}^k$ as

$$\begin{aligned} Q_\eta(\mathbf{D},\mathbf{A}) = \quad & f(\mathbf{D}^k,\mathbf{A}^k) + g(\mathbf{D},\mathbf{A}) \\ & + \text{trace}\left((\mathbf{A}-\mathbf{A}^k)^\top \nabla_{\mathbf{A}} f\right) \\ & + \text{trace}\left((\mathbf{D}-\mathbf{D}^k)^\top \nabla_{\mathbf{D}} f\right) \\ & + \tfrac{1}{2\eta}\|\mathbf{A}-\mathbf{A}^k\|_F^2 + \tfrac{1}{2\eta}\|\mathbf{D}-\mathbf{D}^k\|_F^2 \end{aligned}$$

which is the sum of $g(\mathbf{D},\mathbf{A})$ and the right-hand side of Equation (19) with $L' = \frac{1}{\eta}$. Our choice of the stepsize η is then the largest one for which the following condition holds :

$$f(\mathbf{D}^{k+1},\mathbf{A}^{k+1}) + g(\mathbf{D}^{k+1},\mathbf{A}^{k+1}) \leq Q_\eta(\mathbf{D}^{k+1},\mathbf{A}^{k+1}) \quad (20)$$

It should be noted that this backtracking strategy involves a quadratic approximation of the function to be minimized and this approximation needs an exact first-order approximation. Hence, this strategy is possible only if the gradient can be computed exactly, which is the case for our dictionary learning problem.

From simple algebras and owing to the separability of $g(\mathbf{D},\mathbf{A})$, we can show that $\mathbf{D}^{k+1}$ and $\mathbf{A}^{k+1}$ as defined in Equations (12) and (14) are actually the minimizers of $Q_{\eta_k}(\mathbf{D},\mathbf{A})$. This means that the qualification condition of a stepsize η_k is based on whether the objective value at the novel iterates $\mathbf{D}^{k+1}$ and $\mathbf{A}^{k+1}$ of our original optimization problem is smaller than the minimum objective value of the quadratic approximation $Q_{\eta_k}(\mathbf{D},\mathbf{A})$. This kind of backtracking strategy is exactly the one used by Beck et al. [3] for minimizing the sum of two convex functions. The resulting numerical scheme with backtracking step is given in Algorithm 3 and the following proposition formalizes its convergence properties.

Proposition 3: Suppose that at each iteration k, the stepsize η_k is initialized at $\frac{1}{L_k}$ and then backtracked until Equation (20) holds. Define $\mathbf{D}^{k+1}$ and $\mathbf{A}^{k+1}$ as in Equations (12) and (14) then the sequence $\{f(\mathbf{D}^k,\mathbf{A}^k)+g(\mathbf{D}^k,\mathbf{A}^k)\}_k$ is strictly decreasing and it converges towards a limit point.

Proof: At first, we check that at each iteration the stepsize η_k is lower bounded by a constant. This can be easily proven since according to our algorithm, at each iteration the chosen $\eta_k = \frac{1}{\beta^{h_i}\bar{L}_k}$ is so that $\frac{1}{\beta L} \leq \eta_k$, ($L$ being the Lipschitz constant of f) and thus η_k is lower bounded. For showing this, we first note that condition (20) is satisfied for any $\eta_k \leq \frac{1}{L}$, L being the gradient Lipschitz constant of f, owing to the descent lemma. Suppose now that the selected stepsize $\eta_k = \frac{1}{\beta^{h_i}\bar{L}}$ is so that $\eta_k \leq \frac{1}{\beta L}$, then since $\frac{1}{\beta L} \leq \frac{1}{L}$ it is always possible to choose a smaller h_i' so that

$$\frac{1}{\beta L} \leq \frac{1}{\beta^{h_i'}\bar{L}} \leq \frac{1}{L}$$

with $\frac{1}{\beta^{h_i'}\bar{L}}$ satisfying Equation (20). Defining the stepsize as this value $\frac{1}{\beta^{h_i'}\bar{L}}$, we can conclude that the stepsize is lower bounded since the Lipschitz constant L is finite. Hence, the following inequalities hold for the stepsize η_k at any iteration :

$$\frac{1}{\beta L} \leq \eta_k \leq \frac{1}{\bar{L}}$$

Algorithm 3 : Direct optimization with backtracking for dictionary learning

set k=1, initialize $\mathbf{A}^1$, $\mathbf{D}^1$
choose $\beta > 1$
repeat
  $\bar{L}_k$ = estimation of L_k using either Equation (17) or (18) or a $\bar{L}_{k-k'}$, $k' > 0$
  $h_i = 0$
  repeat
    $\eta_k = \frac{1}{\beta^{h_i}\bar{L}_k}$
    $\mathbf{D}^{k+\frac{1}{2}} = \mathbf{D}^k + \eta_k(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)(\mathbf{A}^k)^\top$
    $\mathbf{D}^{k+1} = \mathcal{P}_\Pi(\mathbf{D}^{k+\frac{1}{2}})$
    $\mathbf{A}^{k+\frac{1}{2}} = \mathbf{A}^k + \eta_k(\mathbf{D}^k)^\top(\mathbf{X} - \mathbf{D}^k\mathbf{A}^k)$
    $\mathbf{A}^{k+1} = \mathcal{S}^{B_a}_{\eta_k\lambda_{\mathbf{A}}}\left(\mathbf{A}^{k+\frac{1}{2}}\right)$
    $h_i \leftarrow h_i + 1$
  until $(\mathbf{A}^{k+1},\mathbf{D}^{k+1})$ satisfies Equation (20) for η_k
  $k \leftarrow k + 1$
until stopping criterion is met

Now, we show that the sequence of the objective value $f(\mathbf{D}^k,\mathbf{A}^k) + g(\mathbf{D}^k,\mathbf{A}^k)$ is strictly decreasing. This property simply comes from Equation (20). Indeed, for a couple $\{\mathbf{D}^{k+1},\mathbf{A}^{k+1}\}$ and η_k that satisfy Equation 20, we have

$$Q_{\eta_k}(\mathbf{D}^{k+1},\mathbf{A}^{k+1}) < Q_{\eta_k}(\mathbf{D}^k,\mathbf{A}^k) = f(\mathbf{D}^k,\mathbf{A}^k)+g(\mathbf{D}^k,\mathbf{A}^k)$$

as $\mathbf{D}^{k+1}$ and $\mathbf{A}^{k+1}$ minimize $Q_{\eta_k}(\mathbf{D},\mathbf{A})$ which is a strictly convex function, η_k being non-zero and finite. Thus, we naturally have

$$f(\mathbf{D}^{k+1},\mathbf{A}^{k+1})+g(\mathbf{D}^{k+1},\mathbf{A}^{k+1}) < f(\mathbf{D}^k,\mathbf{A}^k)+g(\mathbf{D}^k,\mathbf{A}^k)$$

As the sequence $\{f(\mathbf{D}^k,\mathbf{A}^k)+g(\mathbf{D}^k,\mathbf{A}^k)\}$ is lower bounded and strictly decreasing, it is thus convergent. ■

Remark that by allowing backtracking of the stepsize, we have lost some properties of the original Sra's algorithm since the convergence of the algorithm is now pertaining to the objective value instead of the minimizers $\mathbf{D}^k$ and $\mathbf{A}^k$.

F. Coordinatewise stepsize adaptation

The stepsize η_k we choose in Algorithm 3 is the same for the proximal gradient descent step in $\mathbf{D}$ and $\mathbf{A}$. However, it is unlikely that this stepsize is adapted for both coordinate descent. Hence, in order to further speed-up convergence, since this stepsize is conservative, it may be useful to consider a proper stepsize for each coordinate descent and thus we should allow larger stepsize for the smoother coordinate.

To this end, in Algorithm 3 we propose to consider $\bar{L}_{k,\mathbf{D}} = \|\mathbf{A}^k(\mathbf{A}^k)^\top\|_2$ and $\bar{L}_{k,\mathbf{A}} = \|(\mathbf{D}^k)^\top\mathbf{D}^k\|_2$ instead of $\bar{L}_k$ and then define the stepsizes accordingly as

$$\eta_{k,\mathbf{D}} = \frac{1}{\beta^{h_i}\bar{L}_{k,\mathbf{D}}} \quad \text{and} \quad \eta_{k,\mathbf{A}} = \frac{1}{\beta^{h_i}\bar{L}_{k,\mathbf{A}}} \qquad (21)$$

which are respectively used in the dictionary and coefficient matrix update. Note that as the stepsizes are coordinate-dependent, the backtracking condition should now involve

Algorithm 4 : Alternating optimization for dictionary learning

set $k = 1$, initialize $\mathbf{A}^1$, $\mathbf{D}^1$
repeat
  compute $L_{k,\mathbf{D}}$
  choose $\eta_k \leq \frac{1}{L_{k,\mathbf{D}}}$
  $\mathbf{D}_1 = \mathbf{D}^k$
  set $n = 1$
  repeat
    $\mathbf{D}_{n+1} = \mathcal{P}_\Pi\Big(\mathbf{D}_n + \eta_k(\mathbf{X} - \mathbf{D}_n\mathbf{A}^k)(\mathbf{A}^k)^\top\Big)$
    $n \leftarrow n + 1$
  until stopping criterion is met
  $\mathbf{D}^{k+1} = \mathbf{D}_n$
  compute $L_{k,\mathbf{A}}$
  choose $\eta_k \leq \frac{1}{L_{k,\mathbf{A}}}$
  $\mathbf{A}_1 = \mathbf{A}^k$
  $n = 1$
  repeat
    $\mathbf{A}_{n+1} = \mathcal{S}^\infty_{\eta_k\lambda_A}(\mathbf{A}_n + \eta_k(\mathbf{D}^{k+1})^\top(\mathbf{X} - \mathbf{D}^{k+1}\mathbf{A}_n))$
    $n \leftarrow n + 1$
  until stopping criterion is met
  $\mathbf{A}^{k+1} = \mathbf{A}_n$
  $k \leftarrow k + 1$
until stopping criterion is met

$\eta_{k,\mathbf{D}}$ and $\eta_{k,\mathbf{A}}$. In this case, the quadratic approximation of $f + g$ becomes

$$\begin{aligned} Q_{\eta_\mathbf{D},\eta_\mathbf{A}}(\mathbf{D},\mathbf{A}) = \quad & f(\mathbf{D}^k,\mathbf{A}^k) + g(\mathbf{D},\mathbf{A}) \\ & + \operatorname{trace}\big((\mathbf{A}-\mathbf{A}^k)^\top\nabla_\mathbf{A} f\big) \\ & + \operatorname{trace}\big((\mathbf{D}-\mathbf{D}^k)^\top\nabla_\mathbf{D} f\big) \\ & + \tfrac{1}{2\eta_\mathbf{A}}\|\mathbf{A}-\mathbf{A}^k\|_F^2 + \tfrac{1}{2\eta_\mathbf{D}}\|\mathbf{D}-\mathbf{D}^k\|_F^2 \end{aligned}$$

and the backtracking condition is the same as in Equation (20) with $Q_{\eta_\mathbf{D},\eta_\mathbf{A}}$ replacing Q_η. Note that the above proof of convergence can be straightforwardly extended to handle this coordinatewise stepsize situation.

IV. Discussions

A. Relations with alternating optimization

Now that we have presented the direct optimization approach for dictionary learning, we discuss its relation with the approach of Yaghoobi et al. [25] that is based on alternating minimization over $\mathbf{D}$ and $\mathbf{A}$ as depicted in Algorithm 4. We will make this relation clear by stating that the alternating minimization algorithm is actually a direct optimization approach for which gradient computations are corrupted.

From Algorithm 4, we can see that the main difference between alternating and direct optimization is that the former considers several proximal updates for $\mathbf{D}$ and $\mathbf{A}$ at each main iteration while direct optimization only uses a single update. It is intuitive that a full optimization over $\mathbf{A}$ and $\mathbf{D}$ may be useless at the early stage of the dictionary learning problem. This point suggests that direct optimization should be far more efficient than alternating optimization.

Let us now exhibit a more formal relation between these two approaches. In the sequel, we focus only on the updates of $\mathbf{D}$. In alternating optimization, several updates of $\mathbf{D}$ are performed during each main iteration. These updates can be related to those of the direct approach in which the gradients on $\mathbf{A}$ are perturbed so that $\mathbf{A}^k$ remains a fixed point. More formally, consider the direct approach updates

$$\begin{pmatrix}\mathbf{D}^{k+1}\\ \mathbf{A}^{k+1}\end{pmatrix} = \operatorname{prox}_{\eta_k g}\left(\begin{pmatrix}\mathbf{D}^k\\ \mathbf{A}^k\end{pmatrix} - \eta_k\begin{pmatrix}\nabla_\mathbf{D} f\\ \nabla_\mathbf{A} f + \zeta_\mathbf{A}\end{pmatrix}\right) \tag{22}$$

where $\zeta_\mathbf{A}$ is the perturbation. If this matrix perturbation $\zeta_\mathbf{A}$ is so that the iterates lead to

$$\mathbf{A}^k = \operatorname{prox}_{\lambda_A\eta_k\Omega_A}(\mathbf{A}^k - \eta_k\nabla_\mathbf{A} f - \eta_k\zeta_\mathbf{A}) \tag{23}$$

then these direct optimization updates (22) is exactly the $\mathbf{D}$ updates of the alternating approach. In order to satisfy this fixed-point equation (23), we can show, from the optimality condition of a proximity operator, that the perturbation is so that there exists a vector $g_\mathbf{A} \in \partial\Omega_\mathbf{A}$ that satisfies the equation

$$-\zeta_A = \nabla_\mathbf{A} f(\mathbf{D}^k,\mathbf{A}^k) + \lambda_\mathbf{A} g_\mathbf{A}$$

with $\partial\Omega_\mathbf{A}$ being the subdifferential of Ω_A.

By following the same reasoning for the updates of $\mathbf{A}$, we can conclude that alternating optimization of the dictionary learning problem is equivalent to direct optimization with perturbed gradients.

Another insight into the relation between alternating optimization and direct optimization can be given. Indeed, one can notice that if only a single update of $\mathbf{D}_n$ or $\mathbf{A}_n$ is allowed for each alternating optimization step in Algorithm 4 then algorithms 3 and 4 becomes very similar. Hence, our direct optimization approach is strongly related to a special case of an alternating optimization strategy in which early stopping is allowed. The point that differs in the two algorithms occurs in the gradient step of $\mathbf{A}$ which is computed at $\mathbf{D}^k$ for our approach and at $\mathbf{D}^{k+1}$ for the alternating optimization approach. While one may argue that using $\mathbf{D}^{k+1}$ should accelerate convergence, the update based on $\mathbf{D}^k$ is derived from the exact gradient of the non-convex cost function $f(\cdot,\cdot)$ at $\{\mathbf{D}_k,\mathbf{A}_k\}$. This strong connection with alternating optimization also suggests that the stepsizes based on $L_{k,\mathbf{D}}$ and $L_{k,\mathbf{A}}$ as given in Equation (21) are accurate enough so that few backtracking steps would be needed in Algorithm 3. We base this hypothesis on the fact that the alternating optimization uses similar stepsizes and achieves monotone descent.

According to this insight, we have also experimented with a direct optimization with no backtracking of the stepsizes. We have shown that such approach is able to achieve the same recovery rate as with backtracking while being more efficient. Proof of convergence of such approach has not been derived yet and is left for future works but we believe that this nice behavior is due to its resemblance with the alternating optimization approach with a single update. Indeed, if $\mathbf{D}^k$ and $\mathbf{D}^{k+1}$ are close enough, then the gradient and the stepsize in $\mathbf{A}$ of our direct approach is similar to the one of alternating approach and such a step induces descent of the objective value.

B. Generalization of the numerical scheme

The algorithm we propose for solving dictionary learning problem can be extended to a more general setting. Suppose that we want to solve the following optimization problem

$$\min_{\{\mathbf{x}_i \in \mathbb{R}^{d_i}\}} \underbrace{f(\mathbf{x}_1, \cdots, \mathbf{x}_m) + \sum_{i=1}^{m} \Omega_i(\mathbf{x}_i)}_{J(\mathbf{x}_1, \cdots, \mathbf{x}_m)} \tag{24}$$

where f is a smooth non-convex function from $\mathbb{R}^{\sum_i d_i} \mapsto \mathbb{R}$ with gradient Lipschitz of constant L and so that $\inf f > -\infty$ and $\Omega_i(\mathbf{x}_i)$ a lower semi-continuous convex function that acts as a regularizer on the component $\mathbf{x}_i$. $\Omega_i(\mathbf{x}_i)$ is supposed to be of the form $\Omega_i(\mathbf{x}_i) = \Omega_i'(\mathbf{x}_i) + \delta(\mathbf{x}_i | \mathcal{X}_i)$ where $\mathcal{X}_i$ is a compact subset of $\mathbb{R}^{d_i}$. When $J(\mathbf{x}_1, \cdots, \mathbf{x}_m)$ has a special structure (for instance, J is convex in each of its coordinate as in our dictionary learning problem), block-coordinate descent (BCD) algorithms can be considered especially when non-smooth regularizers are in play [24]. For such BCD algorithms, one usually fully optimizes the problem with respects to one coordinate $\mathbf{x}_i$ while keeping the others fixed and then follows the same step for other coordinates.

For our direct approach, we can consider the following iterates for any block-coordinate i

$$\mathbf{x}_i^{k+1} = \text{prox}_{\nu_k \Omega_i}(\mathbf{x}_i^k - \nu_k \nabla_{\mathbf{x}_i} f(\mathbf{x}_1^k, \cdots, \mathbf{x}_i^k, \cdots, \mathbf{x}_m^k)) \tag{25}$$

which update each component $\mathbf{x}_i$, $i \in 1, \cdots, m$ at iteration k with a proximal gradient step. The stepsize ν_k should satisfy conditions given in Equation (8). For simplicity, we have supposed here that the gradients can be computed exactly. This approach that we denote as a *one-step block-coordinate proximal gradient descent* provides a solution of problem (24) in the sense that the iterates defined by Equation (25) converge toward an exact stationary point of (24). This last proposition can be easily proved by following the same lines as for the dictionary learning problem and by invoking Theorem 1.

This algorithm differs from those proposed in the literature [24] by the fact that only a single proximal step is performed along one component instead of a full optimization. While we have not applied this algorithm to other problems than dictionary learning problem, we conjecture that the *one-step block-coordinate proximal gradient descent* is significantly more efficient than classical block coordinate descent because it avoids unnecessary updates while being far from the optimal solution.

V. Simulation study

In this section, we describe the study we have carried out for showing the benefits of the algorithm we propose for dictionary learning. Experiments on simulated data and on an audio dataset are reported. They have been run on a Linux machine with 24Gb of memory and a 8-core Intel Xeon E5530 processor clocked at 2.4GHz. All the codes have been written in Matlab.

A. Experimental set-up

Evaluating performance of dictionary learning algorithms is rather a difficult task unless the true dictionary is known. This is why for this experiment we focus on synthetic signals built from a known dictionary.

We have built these signals as follows. Let us define $\mathbf{D}^\star$ the true dictionary composed of M elements. Signal $\{\mathbf{x}_i\}_{i=1}^L$ are obtained as sparse linear combinations of dictionary elements according to the model

$$\mathbf{X} = \mathbf{D}^\star \mathbf{A}^\star + \mathbf{E}$$

where $\mathbf{E}$ and $\mathbf{A}^\star$ are respectively a noise matrix and a sparse coefficient matrix. Components of $\mathbf{E}$ have being drawn *i.i.d* from a Gaussian distribution which variance is adjusted so that each signal has a pre-defined signal-to-noise ratio (here 30). For each signal, the T atoms which get non-zero weights are randomly chosen and their absolute weight values are drawn according to a uniform distribution in $[0.2, 1]$. The entries of the dictionary $\mathbf{D}^\star$ is obtained from *i.i.d* random sampling of a Gaussian distribution followed by a unit-norm normalization of each column.

We have solved the dictionary learning problem (4) with the matrix $\mathbf{X}$ as input, by means of our direct optimization approach, the alternating method (MM) of Yaghoobi et al. [25] and the MOD approach [11]. For constraints and regularizations, we have used an ℓ_1 norm for the coefficient matrix. For the dictionary elements, we used the unit-norm ball constraint for our approach and MM and the unit-norm sphere constraint for MOD. We have evaluated the quality of the learned dictionary atoms compared to the true ones and the time needed for the two algorithms for solving problem (4). For all algorithms, the stopping criterion is based on the relative variation of the objective value which should be lower than 10^{-5}. For the inner problems of the alternating approach and MOD, the same stopping criterion has been used with a tolerance of 10^{-6}. The maximal number of iterations for the direct approach has been set to 30000 while the ones of alternating optimization and MOD are fixed at 10000. Performance results have been averaged over 10 trials and all algorithms have been initialized from the same matrices $\mathbf{A}^1$ and $\mathbf{D}^1$. Unless specified, the gradient Lipschitz constant has been estimated according to Equation (21). We have reported on three variants of our approach depending on whether backtracking has been performed and on how often the gradient Lipschitz constant has been estimated. We have denoted them as

- **Back** : the approach where backtracking is performed and stepsizes have been computed coordinatewise every 2 iterations.
- **Back FewEig** : the approach where backtracking is performed and the stepsizes have been computed still coordinatewise but every 10 iterations.
- **NoBack** : the approach where backtracking is not performed and stepsizes have been computed coordinatewise again every 2 iterations.

These three variants have been denoted in the figure as **Dir B**, **Dir FB** and **Dir NoB**. We have also tried an approach which

performs backtracking and use a global stepsize defined as $\min(\eta_{k,\mathbf{D}}, \eta_{k,\mathbf{A}})$ obtained from the estimation of the Lipschitz constant through Equation (18). Results of this variant has not been reported since it was a bit less efficient than the alternating approach of Yaghoobi et al. [25] which also adapts the stepsize to each alternating step.

In these experiments, we have used as a measure of performance the recovery rate of the true dictionary elements. This rate has been computed as the number of atoms in $\mathbf{D}^\star$ that also appear in the estimated dictionary. We say that an atom $\mathbf{D}^\star_{\cdot,j}$ of the true dictionary is present in an estimated dictionary $\hat{\mathbf{D}}$ if

$$\min(1 - |\hat{\mathbf{D}}^\top \mathbf{D}^\star_{\cdot,j}|) < 0.01$$

where the $\min$ applies to coordinates of the resulting vector.

B. Comparing computational efficiency and dictionary recovery

Computational efficiency of the direct approaches, the alternating method and MOD have been reported in Figure 1 and 2 for two different settings of the dictionary learning problem and for increasing number T of dictionary atoms involved in the process of each signal.

Results for the first setting involving 1300 signals in dimension 50 are reported in in Figure 1. We notice that performances on recovering the exact dictionary are rather similar for all approaches, although a very slight advantage goes to the MOD approach when few atoms compose the signal and direct methods are slightly better in the phase transition. Regarding objective values after convergence of all algorithms, we can note that they all converge towards a local minimal of the learning problem with very similar objective values. Note that **NoBack** performs as well as the other methods suggesting that the estimation of the gradient Lipschitz constant using Equation (21) is rather accurate and robust. This can be justified by the resemblance of the **NoBack** approach to an alternating optimization method using early stopping, as we have explained above.

The most important benefit of the direct approaches are related to the gain in computational efficiency compared to the alternating method and MOD as reported in right panel of Figure 1. For the genuine approach **Back** the gain with respect to alternating method varies between from 1.5 to 2.5 in regions where recovery rate is about 1. As the recovery rate decreases, the running time of **Back** becomes slightly worse than the one of MM. Compared to MOD, **Back** is about 2 times faster. Computing the stepsize less frequently as in **Back FewEig** leads to slightly better computational efficiency than **Back**, especially in regions with few atoms composing the signals. When backtracking is not performed (**NoBack**), the direct approach is always more efficient than MOD and MM and the gain in running time is always more important than those of other direct approaches (ranging from 3 to 8).

Interestingly, according to our results, MOD is more efficient than MM when T is smaller than 8. This result is in accordance with the one reported by Yaghoobi et al. [25]. However, as the number T of atoms in the signal increases, MOD becomes far less efficient than the majorization method.

Figure 2 presents the same results for a larger-scale problem (with 5000 signals and 200 dictionary elements to learn). Recovery rates and objective values follow the same trends as for the smaller-scale settings. Gain in computational efficiency are still achieved by our direct approach in this setting. This gain is about a factor of 2 in regions where T is small. Again, when no algorithms are able to retrieve the correct dictionary atoms, as the number of atoms composing the signals are too large, our direct approaches, included **NoBack**, have a slightly worse running time than MM. Interestingly, **NoBack** is always more efficient than the other two variants. This is natural since backtracking can be expensive.

These two experiments suggest us that our direct approaches are efficient and robust and they can provided similar recovery rate of the dictionary atoms to those of the majorization method or MOD while being more efficient, at least when atoms can be recovered. In term of efficiency, **NoBack** is the best performing algorithm while **Back** and **Back FewEig** are just slightly worse when some true dictionary elements can be retrieved. However, these two latter algorithms have the advantage of being provided with some proofs of convergence.

C. Influence of the sparsity-inducing hyperparameter λ_A

It is well known that algorithms solving sparse approximation problems using ℓ_1 norm sparsity-inducing regularizer get less efficient as the related hyperparameter decreases leading thus to solutions that are not very sparse [12]. In this experiment, we investigate the gain in efficiency brought by our direct optimization approach as the hyperparameter λ_A related to $\Omega_{\mathbf{A}}(\mathbf{A}) = \sum_{i,j} |a_{i,j}|$ varies. Note that we have increased the maximal number of iterations of MM and MOD to 30000. Figure 3 depicts the recovery rates, the objective values and the computational efficiency of all algorithms with respects to parameter λ_A. We can note that recovery rate is better when solutions get denser and at some points (for $\lambda_{\mathbf{A}} = 0.02$) the majorization method sometimes fails to recover the correct atoms while MOD never achieves in getting these dictionary elements. As depicted by the objective value plots, For MOD, this is essentially either to lack of convergence after the 30000 iterations or because the iterates are trapped in bad local minima. For MM, as its objective values are rather similar to those of our direct approaches, we think that these failure are essentially due to lack of convergence after the maximal number of iterations. For course, we could have made MM properly converge by increasing this maximal number of iterations, but we think that it is an interesting point to highlight this slow convergence behaviour of MM. Gain in computational efficiency brought by our direct optimization with backtracking approach is about 1.5 in regions where competing algorithms have reached proper convergence. We can again highlight the practical benefit of not backtracking the stepsize as the gain raises to about 3.

D. Comparing the Lipschitz constant estimations

Up to now, all the experiments have been carried by estimating the stepsizes in Algorithm 3 through the spectral norm given by Equation (21). We have made this choice

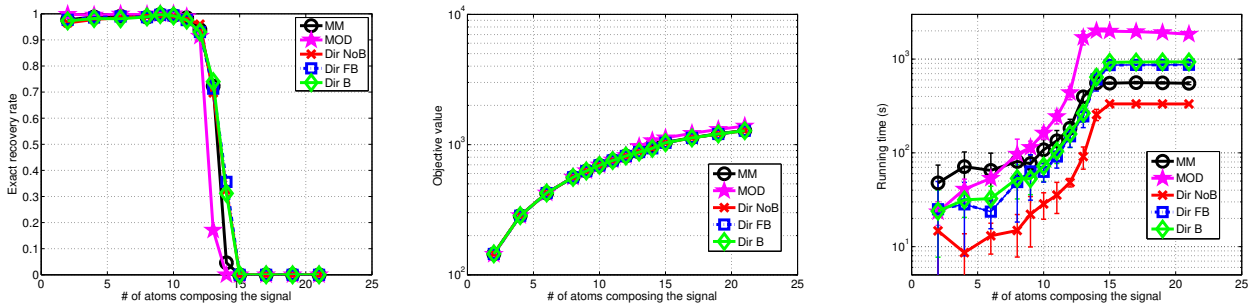

Fig. 1. (left) Recovery rate of different parametrizations of our direct approach and a baseline algorithm which is the alternating method of Yaghoobi et al. (middle) Averaged objective values at convergence. (right) computational running time of our direct optimization approach compared to the baseline algorithms. All performances and gains are evaluated with respects to the number of non-zero elements composing the signals. Other settings of the dictionary learning problem are $N = 50$, $L = 1300$, $M = 100$, $\lambda_{\mathbf{A}} = 0.1$ and $\Omega_{\mathbf{A}}(\mathbf{A}) = \sum_{i,j} |a_{i,j}|$

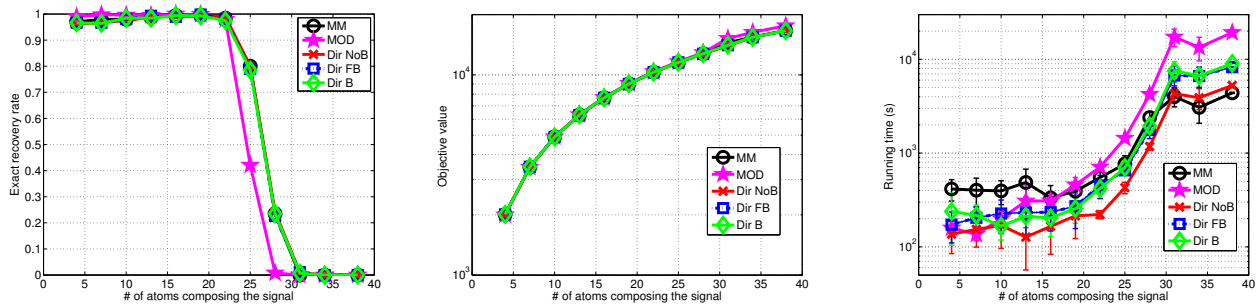

Fig. 2. (left) Recovery rate of different parametrizations of our direct approach and a baseline algorithm which is the alternating method of Yaghoobi et al. (middle) Averaged objective values at convergence. (right) computational running time of our direct optimization approach compared to the baseline algorithms. All performances and gains are evaluated with respects to the number of non-zero elements composing the signals. Other settings of the dictionary learning problem are $N = 100$, $L = 5000$, $M = 200$, $\lambda_{\mathbf{A}} = 0.2$ and $\Omega_{\mathbf{A}}(\mathbf{A}) = \sum_{i,j} |a_{i,j}|$

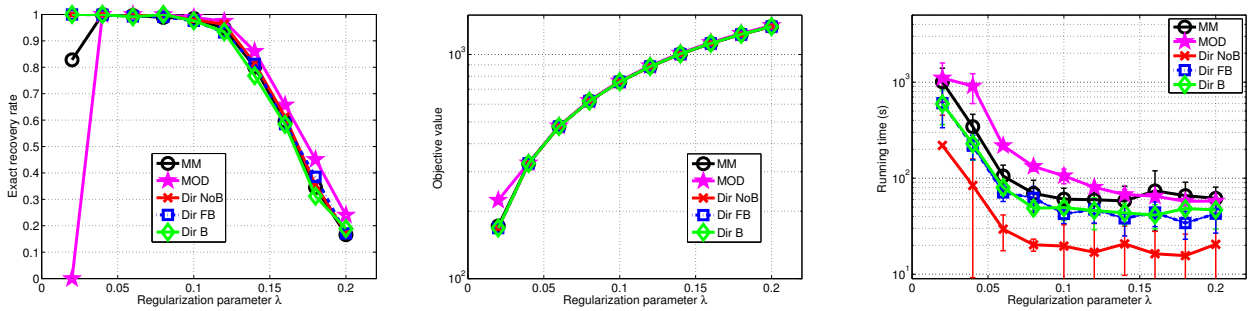

Fig. 3. (left) Comparing recovery rate of different parametrizations of our direct approach and a baseline algorithm which is the alternating method of Yaghoobi et al. (right) Gain in computational time of our direct optimization approach compared to the baseline. All performances and gains are evaluated with respects to the choice of the hyperparameter $\lambda_{\mathbf{A}}$. Recovery Rate and Gain in computational time for varying values of hyperparameters $\lambda_{\mathbf{A}}$ and for 11 non-zero elements composing each signal. The other settings are the same of those used in Figure 1.

because these estimations empirically appeared to be more robust than the estimations obtained using Equation (17). Figure 4 gives an example, for $T = 2$, $N = 50$, $M = 100$ and $L = 1300$ of how the coordinate-wise stepsizes evolve across iterations in both cases. For producing this figure, we have used the same experimental set-up as above and both algorithms have the same stopping criterion. We clearly note that the stepsizes computed from the gradient definition clearly tends to oscillate. At some point, due to too small stepsizes, the algorithm meets its stopping criterion and finishes before a proper convergence. For instance, for this figure, the learned dictionary has not recovered any atom from the true dictionary. Detailed performances have not been reported because the gradient-based Lipschitz constant estimation performs very poorly with, for instance, a recovery rate of less than 0.55 for $T = 2$, which drops at 0.4 for $T = 6$ and 0 for $T = 10$.

E. Application on Audio data

We have compared the efficiency of our direct approaches and the alternating optimization approach of Yaghoobi et al. [25] on a real signal processing application. The audio dataset we use is the one considered by Yaghoobi et al. [25]. This dataset is composed of an audio sample recorded from a BBC radio session which plays classical music. From that audio sample, 8192 pieces of signal have been extracted, each being

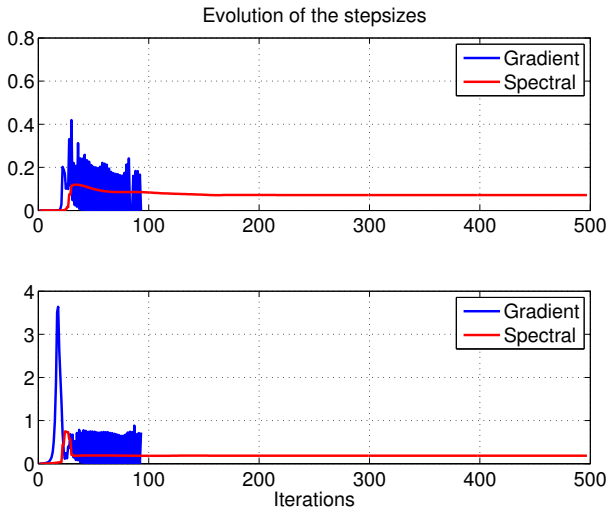

Fig. 4. Evolution of stepsizes along the iterations. (top) stepsize on the dictionary update. (bottom) stepsize on the weight update. "Gradient" denotes the stepsize as computed by coordinatewise version of Equation (17) while "Spectral" denotes the one obtained through Equation (21).

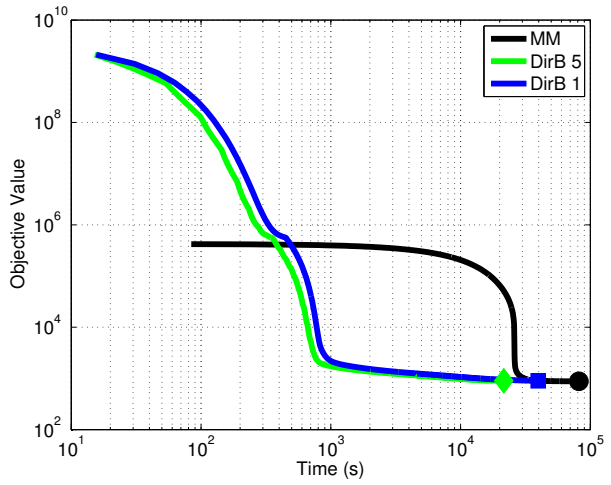

Fig. 5. Evolution of the objective value along time. (MM) majorization method. (dir B) direct optimization with backtracking and evaluation of the stepsize at each iteration (dirB 1) and every five iterations (dirB 5).

composed of 1024 time samples. Details about the dataset can be found in [25].

Our objective is to learn 2048 dictionary atoms from this set of signals and we have run the MM algorithm for 1000 iterations while allowing only 20 iterations for each inner problems. Two variants of the direct optimization approach have been compared : they respectively evaluate the stepsize coordinatewise using Equations (21) at every single and every five iterations. These variants have been run for 2000 and 3000 iterations. These maximal number of iterations have been chosen in order to allow these approaches to reach similar objective values than the MM approach. Note that all algorithms have been initialized with the same Gaussian random dictionary and $\lambda_{\mathbf{A}}$ has been set to 0.1. Figure 5 depicts the evolution the objective values of all algorithms with respect to the running time. We can see that our direct optimization

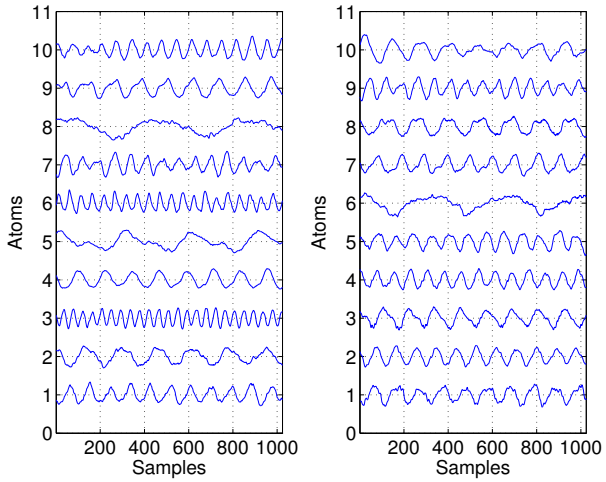

Fig. 6. Examples of learned atoms from the audio data. (left) alternating optimization. (right) direct optimization with backtracking and evaluation of the stepsizes at every iteration. The presented atoms are among those that are the most frequently used by the sparse coding step.

approaches achieve faster convergence. Interestingly, these methods yield to a rapid decrease of the objective value and then reach a plateau. This means that the methods we advocate is able to provide "reasonable" dictionary atoms rapidly. This is an interesting feature if one want to solve a dictionary learning problem on a budget of time. Figure 6 shows the learned dictionary atoms for both the MM method and one of our direct optimization method. We note that both algorithms were able to retrieve dictionary atoms that oscillate at some specific frequencies.

VI. Conclusion

Departing from the mainstream approach for solving dictionary learning problems, we have proposed in this paper a more direct approach based on the so-called *one-step block coordinate proximal gradient descent*. This algorithm is tailored for the resolution of an optimization problem involving a smooth non-convex function and a sum of separable and convex functions. The dictionary learning problem we address fits into this framework and we show that this novel algorithm allows direct resolution of the non-convex optimization problem. The choice of the stepsize of the proximal gradient descent is an issue that we have addressed by proposing a method which first roughly estimates the stepsize and then adapts it by backtracking. Simulation studies show that our novel optimization method is more efficient than usual alternating optimization methods.

VII. Appendix

A. Computing the components of the Hessian

For proving that the gradient of the cost function is Lipschitz, we need a closed-form expression of its components . In the sequel, we derive that expression as well as those of the Hessian since f is twice differentiable. We have :

$$f(\mathbf{D},\mathbf{A}) = \frac{1}{2}\|\mathbf{X}-\mathbf{D}\mathbf{A}\|_F^2 = \frac{1}{2}\sum_{i,j}\left(x_{j,i} - \sum_k d_{j,k}a_{k,i}\right)^2$$

Hence, we can derive :

$$\frac{\partial f}{\partial d_{m,n}} = -\sum_i a_{n,i}x_{m,i} + \sum_{k,i} d_{m,k}a_{k,i}a_{n,i}$$

which gives for the second-order partial derivatives

$$\frac{\partial^2 f}{\partial d_{m,n}\partial d_{s,t}} = \begin{cases} 0 & \text{if } m \neq s \\ \sum_i a_{t,i}a_{n,i} & \text{otherwise} \end{cases} \quad (26)$$

and

$$\frac{\partial^2 f}{\partial d_{m,n}\partial a_{s,t}} = \begin{cases} d_{m,s}a_{n,t} & \text{if } n \neq s \\ -x_{m,t} + d_{m,s}a_{n,t} + \sum_k d_{m,k}a_{k,t} & \text{otherwise} \end{cases}$$

In a similar way, we have

$$\frac{\partial f}{\partial a_{m,n}} = -\sum_j d_{j,m}x_{j,n} + \sum_{k,j} d_{j,k}a_{k,n}d_{j,m}$$

$$\frac{\partial^2 f}{\partial a_{m,n}\partial a_{s,t}} = \begin{cases} 0 & \text{if } t \neq n \\ \sum_j d_{j,s}d_{j,m} & \text{otherwise} \end{cases} \quad (27)$$

and

$$\frac{\partial^2 f}{\partial a_{m,n}\partial d_{s,t}} = \begin{cases} d_{s,m}a_{t,n} & \text{if } t \neq m \\ -x_{s,n} + d_{s,m}a_{t,n} + \sum_k d_{s,k}a_{k,n} & \text{otherwise} \end{cases}$$

According to these equations, we can note that the Hessian of f has a specific structure. Indeed, if we suppose that the variables are sorted as $d_{1,1}, \cdots, d_{1,M}, d_{2,1}, \cdots, d_{N,M}$ and then $a_{1,1}, \cdots, a_{M,1}, a_{1,2}, \cdots, a_{M,L}$, the Hessian is of the form

$$\begin{pmatrix} \mathbb{A} & \mathbb{B} \\ \mathbb{B}^\top & \mathbb{D} \end{pmatrix} \quad (28)$$

where $\mathbb{A}$ and $\mathbb{D}$ are block-diagonal matrices where, according to Equation (26), for $\mathbb{A}$ each block is $\mathbf{A}\mathbf{A}^\top$ and for $\mathbb{D}$ is $\mathbf{D}^\top\mathbf{D}$ as given by Equation (27).

REFERENCES

[1] M. Aharon, E. Elad, and A. Bruckstein, "K-svd : and algorithm for designing overcomplete dictionaries for sparse representations." *IEEE Trans. on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[2] 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 gaussseidel methods," *Mathematical Programming*, vol. 137, no. 1-2, pp. 91–129, 2013.

[3] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, pp. 183–202, 2009.

[4] D. Bertsekas, *Nonlinear programming*. Athena scientific, 1999.

[5] J. Chen and X. Huo, "Sparse representations for multiple measurements vectors (mmv) in an overcomplete dictionary," in *Proc IEEE Int. Conf Acoustics, Speech Signal Processing*, vol. 4, 2005, pp. 257–260.

[6] P. Combettes and V. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling and Simulation*, vol. 4, pp. 1168–1200, 2005.

[7] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. H. Bauschke, R. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, Eds. Springer-Verlag, 2010.

[8] S. Cotter, B. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, 2005.

[9] W. Dai, T. Xu, and W. Wang, "Simultaneous codeword optimization (simco) for dictionary update and learning," *IEEE Trans. on Signal Processing*, vol. 60, no. 12, pp. 6340–6353, 2012.

[10] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Processing*, vol. 54, pp. 3736–3745, 2006.

[11] K. Engan, S. Aase, and J. Hakon-Husoy, "Method of optimal directions for frame designs," in *Proc of the International Conference on Acoustics, Speech and Signal Processing*, 1999.

[12] M. Figueiredo, R. Nowak, and S. Wright, "Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing*, vol. 1, no. 4, pp. 586–598, 2007.

[13] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach, "Proximal methods for sparse hierarchical dictionary learning," in *Proceedings of International Conference on Machine Learning*, 2010.

[14] R. Jenatton, G. Obozinski, and F. Bach, "Structured sparse principal component analysis," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010.

[15] K. Kreutz-Delgado, J. Murray, B. Rao, K. Engan, T. Lee, and T.Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Computation*, vol. 15, pp. 349–396, 2003.

[16] H. Lee, A. Battle, R. Raina, and A. Ng, "Efficient sparse coding algorithms," in *Advances in Neural Information Processing Systems*, 2007.

[17] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[18] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," in *Proceedings of Neural Information Processing Systems*, 2008.

[19] B. Olshausen and D. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[20] A. Rakotomamonjy, "Surveying and comparing simultaneous sparse approximation (or group lasso) algorithms," *Signal Processing*, vol. 91, no. 7, pp. 1505–1526, 2011.

[21] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2121–2130, 2010.

[22] S. Sra, "nonconvex proximal splitting : batch and incremental algorithms," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[23] J. Tropp, "Algorithms for simultaneous sparse approximation. part II: Convex relaxation," *Signal Processing*, vol. 86, pp. 589–602, 2006.

[24] P. Tseng, "Convergence of block coordinate descent method for nondifferentiable minimization," *Journal of Optimization Theory and Application*, vol. 109, pp. 475–494, 2001.

[25] M. Yaghoobi, T. Blumensath, and M. Davies, "Dictionary learning for sparse approximations with the majorization method," *IEEE Transaction on Signal Processing*, vol. 57, no. 6, pp. 2178–2191, 2009.

[26] H. Zou, "The adaptive lasso and its oracle properties," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.