Commun. Math. Stat. (2015) 3:37–55
DOI 10.1007/s40304-015-0050-5

Alternating Direction Method of Multipliers for Solving Dictionary Learning Models

Yusheng Li[1] · **Xinchang Xie**[2] · **Zhouwang Yang**[1]

Received: 12 November 2014 / Revised: 12 February 2015 / Accepted: 26 February 2015 /
Published online: 4 April 2015

Abstract In recent years, there has been a growing usage of sparse representations in signal processing. This paper revisits the K-SVD, an algorithm for designing overcomplete dictionaries for sparse and redundant representations. We present a new approach to solve dictionary learning models by combining the alternating direction method of multipliers and the orthogonal matching pursuit. The experimental results show that our approach can reliably obtain better learned dictionary elements and outperform other algorithms.

Keywords Dictionary learning · K-SVD · Alternating direction method of multipliers · Orthogonal matching pursuit

Mathematics Subject Classification 49N45 · 65K05 · 68U10

1 Introduction

In recent years, a great volume of work has been devoted to the problem of sparse representation of signals. Using an overcomplete dictionary $\mathbf{D} \in \mathbb{R}^{m\times n}$ $(m \ll n)$ that contains n prototype signal-atoms for columns $\{\mathbf{d}_j\}_{j=1}^n$, a signal $\mathbf{y} \in \mathbb{R}^m$ can

✉ Yusheng Li
lysh@mail.ustc.edu.cn

Xinchang Xie
xcxie@mail.ustc.edu.cn

Zhouwang Yang
yangzw@ustc.edu.cn

1 University of Science and Technology of China, Hefei, China

2 Duke University, Durham, USA

be represented by sparse linear combinations of these atoms. The representation of $\mathbf{y}$ may either be exact $\mathbf{y} = \mathbf{D}\mathbf{x}$ or approximate (i.e., satisfying $\|\mathbf{y} - \mathbf{D}\mathbf{x}\|_p \leq \varepsilon$). The vector $\mathbf{x} \in \mathbb{R}^n$ contains the representation coefficients of the signal $\mathbf{y}$. In approximation methods, the typical norms used for measuring the deviation are the ℓ_p-norms for $p = 1, 2$ and ∞. In this paper, we shall concentrate on the case of $p = 2$.

If $m < n$ and $\mathbf{D}$ is a full-rank matrix, an infinite number of solutions are available for the representation problem, and hence, constraints on the solution must be set for a unique solution. The solution with the fewest numbers of nonzero coefficients is certainly an appealing representation. This sparsest representation is the solution of either

$$(P_0) \quad \min_{\mathbf{x}} \|\mathbf{x}\|_0 \ \text{ subject to } \ \mathbf{y} = \mathbf{D}\mathbf{x} \tag{1.1}$$

or

$$(P_0^{\varepsilon}) \quad \min_{\mathbf{x}} \|\mathbf{x}\|_0 \ \text{ subject to } \ \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2 \leq \varepsilon, \tag{1.2}$$

where $\|\cdot\|_0$ is the ℓ_0 norm, counting the nonzero entries of a vector.

Applications that can benefit from the sparsity and overcompleteness properties (together or separately) include compression, regularization in inverse problems, feature extraction, and more. We will give some application examples in the numerical part.

An overcomplete dictionary $\mathbf{D}$, which leads to sparse representations, can either be chosen as a prespecified set of functions or designed by adapting its content to fit a given set of signal examples. Choosing a prespecified transform matrix is appealing because of its simplicity. In this paper, we consider a different route for designing dictionaries based on learning. Our goal is to find the dictionary that yields sparse representations for the training signals. We apply the ADMM [1] method to the training model, which is different from the K-SVD [2].

The paper is organized as follows. Section 2 introduces the dictionary learning knowledge, and reviews works related to it. The ADMM [1] algorithm we promote for solving the problem is presented and discussed in Sect. 3. Numerical experiments on image inpainting problems are presented in Sect. 4. Section 5 shows our conclusion.

2 Prior Art

2.1 Sparse Coding

Sparse coding is the process of computing the representation coefficients $\mathbf{x}$ based on the given signal $\mathbf{y}$ and the dictionary $\mathbf{D}$. This process, commonly referred to as “atom decomposition,” requires solving (1.1) or (1.2). This is typically done by a “pursuit algorithm” that finds an approximate solution. In this section, we briefly show several of these algorithms. A more detailed description of these methods can be found in [2].

Exact determination of sparsest representations proves to be an NP-hard problem [3]; thus, approximate solutions are considered instead. In the past decade, several efficient pursuit algorithms have been proposed. The simplest ones are the matching pursuit (MP) [4], and the orthogonal matching pursuit (OMP) algorithms [5–10]. See [11] for a review of a paper from December 2008. There are also some other greedy solvers improved based on OMP, such as stagewise OMP (StOMP) [12], Robust OMP (ROMP) [13,14], compressive sampling matching pursuit (CoSaMP) [15], Batch OMP (BOMP) [16], etc. In this paper, we apply BOMP to solve the sparse coding stage.

A second well-known pursuit approach is the basis pursuit (BP) [17]. It suggests a convexification of the problems posed in (1.1) and (1.2) by replacing the ℓ_0-norm with an ℓ_1-norm. A series of methods were developed in this mode, like FISTA [18], SPGL1 [19], TFOCS [20], and so on.

Extensive study of these algorithms in recent years has established that, if the sought solution is sparse enough, these techniques recover it well in the exact case [8,21–23]. Further work has considered the approximated versions and has shown stability in recovery of [24] and [25]. The properties of the dictionary set the limits on the sparsity of the coefficient vector, consequently leading to its successful evaluation.

2.2 Design of Dictionaries

We now come to the main topic of the paper, the training of dictionaries based on a set of examples. Given such a set, $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{L}$, we assume that there is a dictionary $\mathbf{D}$ that gives rise to the given signal examples via sparse combinations, i.e., we assume that $\mathbf{D}$ exists, so that solving (P_0) for each example $\mathbf{y}_i$ gives a sparse representation $\mathbf{x}_i$. It is in this setting where we ask what the proper dictionary $\mathbf{D}$ is.

The problem can be formulated as

$$\min_{\mathbf{D},\mathbf{X}}\{\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2\} \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \leq \tau_0,\, i = 1, \ldots, L, \tag{2.1}$$

where τ_0 is a bound on the sparsity of the representation, $\mathbf{x}_i$ denotes the ith column of the coefficient matrix $\mathbf{X}$, and $\|\cdot\|_F$ is the Frobenius norm of a matrix.

A dual objective could alternatively be met by considering

$$\min_{\mathbf{D},\mathbf{X}} \sum_{i=1}^{L} \|\mathbf{x}_i\|_0 \quad \text{subject to} \quad \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 \leq \varepsilon \tag{2.2}$$

for a fixed value ε. In this paper, we mainly discuss the first problem (2.1).

We now briefly review important work that has been completed in the bases learning domain. The usual method for solving (2.1) is alternating minimization: first, $\mathbf{D}$ is fixed and $\mathbf{X}$ is approximated by some of the algorithms reviewed in the previous section. This stage is called sparse coding. In the next step, $\mathbf{X}$ is fixed and $\mathbf{D}$ is updated. This alternating minimization is repeated until converging to a local minimum (it is also possible that the algorithm can get stuck to a saddle point of the problem). The MOD [26] algorithm follows this approach. The dictionary update step in MOD is calculated as $\mathbf{D}^{(k+1)} = \mathbf{Y}\mathbf{X}^{(k)+}$, where the superscript denotes iteration number and $\mathbf{X}^{+}$ is pseudoinverse of $\mathbf{X}$. Calculating the pseudoinverse for each iteration of the algorithm makes MOD slow. The K-SVD algorithm, which is reviewed next, overcomes this problem partially.

The K-SVD algorithm [2] is a generalization of the K-means clustering approach to basis learning with sparseness constraints when number of clusters (that corresponds to the level of sparseness τ_0) is greater than 1. Therefore, the basis learning philosophy of the K-SVD algorithm reflects the knowledge that clustering yields the most efficient signal representation, i.e., representing the signal by one coefficient only. Its development was inspired in part by the computational inefficiency that is inherent in the MOD basis learning algorithms.

The K-SVD algorithm involves two basic steps, which together constitute the algorithm iteration: (i) the signals in $\mathbf{Y}$ are sparse-coded given the current dictionary estimate, producing the sparse representations matrix $\mathbf{X}$, and (ii) the dictionary atoms are updated according to the current sparse representations; see Algorithm 1. The sparse coding part is commonly implemented by using OMP. The dictionary update is performed one atom at a time, optimizing the target function for each atom individually, while keeping the rest fixed.

The framework of K-SVD [2]:

Algorithm 1 K-SVD

```
Require: Signal set $\mathbf{Y}$, initial dictionary $\mathbf{D} = \mathbf{D}^{(0)}$, target sparsity $\tau_0$, number of iterations $K$.
Ensure: Dictionary $\mathbf{D}$ and sparse matrix $\mathbf{X}$ such that $\mathbf{Y} \approx \mathbf{DX}$.
  for $k = 1 \dots K$ do
    Solve $\mathbf{x}_i := \arg\min_{\mathbf{x}} \|\mathbf{y}_i - \mathbf{Dx}\|_2^2$ subject to $\|\mathbf{x}\|_0 \le \tau_0,\ \forall i \in \{1, \dots, L\}$
    for $j = 1 \dots n$ do
      $\mathbf{E}_j := \mathbf{Y} - \sum_{l \neq j} \mathbf{d}_l \mathbf{x}_l$ ($\mathbf{x}_l$ is the $l$th row of $\mathbf{X}$)
      $(\mathbf{d}^*, \mathbf{g}^*) := \arg\min_{\mathbf{d},\mathbf{g}} \|\mathbf{E}_j - \mathbf{dg}^T\|_F^2$ subject to $\|\mathbf{d}\|_2 = 1$
      $\mathbf{d}_j := \mathbf{d}^*$
      $\mathbf{x}_j := \mathbf{g}^*$
    end for
  end for
```

The problem in the dictionary update stage can be solved directly via a singular value decomposition (SVD), which makes the algorithm slow when the sample size is large. K-SVD performs well in different kinds of applications including image compression, image denoising, image inpainting, and so on [2,27–30].

3 ADMM for Dictionary Learning

3.1 Alternating Direction Method of Multipliers

In this section, we give an overview of ADMM [1]. The alternating direction method of multipliers (ADMM) is a particular optimization technique that is well suited for constrained minimization problem in the following form:

$$\min_{\mathbf{x},\mathbf{y}} f(\mathbf{x}) + g(\mathbf{y}) \ \text{ subject to } \ \mathbf{Ax} + \mathbf{By} = \mathbf{c}, \tag{3.1}$$

where $f, g : \mathbb{R}^n \to \mathbb{R}$ are closed, convex, proper functions, and $\mathbf{A}$ and $\mathbf{B}$ are linear transformations (e.g., matrices).

To solve the problem, one considers the augmented Lagrangian function and seeks its stationary points.

$$L_\beta(\mathbf{x}, \mathbf{y}, \boldsymbol{\lambda}) := f(\mathbf{x}) + g(\mathbf{y}) + \langle \boldsymbol{\lambda}, \mathbf{Ax} + \mathbf{By} - \mathbf{c} \rangle + \frac{\beta}{2} \|\mathbf{Ax} + \mathbf{By} - \mathbf{c}\|_2^2 \tag{3.2}$$

ADMM consists of the iterations

$$\begin{cases} \mathbf{x}^{(k+1)} := \arg\min\limits_{\mathbf{x}} L_\beta(\mathbf{x}, \mathbf{y}^{(k)}, \boldsymbol{\lambda}^{(k)}) \\ \mathbf{y}^{(k+1)} := \arg\min\limits_{\mathbf{y}} L_\beta(\mathbf{x}^{(k+1)}, \mathbf{y}, \boldsymbol{\lambda}^{(k)}) \\ \boldsymbol{\lambda}^{(k+1)} := \boldsymbol{\lambda}^{(k)} + \beta(\mathbf{Ax}^{(k+1)} + \mathbf{By}^{(k+1)} - \mathbf{c}) \end{cases} \tag{3.3}$$

where $\beta > 0$.

The ADMM algorithm was originally introduced in early 1970s, and has since been studied extensively. Recently, it has become widely popular in modern big data related problems arising in machine learning, computer vision, signal processing, networking, and so on.

3.2 Application of ADMM to Dictionary Learning

ADMM has been widely used in different kinds of optimization problems. Additional information regarding ADMM's applications can be found in [1], wherein the author explores the use of ADMM in nonconvex problems and gives some examples. However, there is no theoretical evidence to confirm that ADMM is able to converge to an optimal point.

Rakotomamonjy [31] tried to apply ADMM to the convex model of dictionary learning and obtained satisfactory results. However, for most applications, the nonconvex model always performs better. That is also the reason why we choose to solve the nonconvex model. Nevertheless, it has been observed by many researchers that the ADMM works very well for various applications involving nonconvex objectives, such as the nonnegative matrix factorization, phase retrieval, distributed matrix factorization, etc.; see [32–36] and the references therein. Also in practice, ADMM of ten exhibits faster convergence than traditional primal-dual type algorithms such as the dual ascent algorithm or the method of multipliers. It is also particularly suitable for parallel implementation. In this section, we introduce the ADMM algorithm for nonconvex dictionary training problem. The nonconvex dictionary learning model we primarily focus on is

$$\min_{\mathbf{D},\mathbf{X}} \{\|\mathbf{Y} - \mathbf{DX}\|_F^2\} \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \le \tau_0, i = 1, \dots, L \tag{3.4}$$

with $\mathbf{Y} \in \mathbb{R}^{m \times L}$, $\mathbf{D} \in \mathbb{R}^{m \times n}$ and the coefficient $\mathbf{X} \in \mathbb{R}^{n \times L}$.

Before applying ADMM to this model, we let $\mathbf{Z} = \mathbf{DX}$, and the formulation becomes

$$\min_{\mathbf{D},\mathbf{X},\mathbf{Z}} \left\{\|\mathbf{Y}-\mathbf{Z}\|_F^2\right\} \text{ s.t } \mathbf{Z}=\mathbf{D}\mathbf{X},\ \|\mathbf{x}_i\|_0 \le \tau_0, i=1,\dots,L. \tag{3.5}$$

As proposed before, the augmented Lagrangian function can be written as

$$L_\beta(\mathbf{D},\mathbf{X},\mathbf{Z},\mathbf{\Lambda}) := \|\mathbf{Y}-\mathbf{Z}\|_F^2 + \sum_{i=1}^{L}\langle \mathbf{\Lambda}_i, (\mathbf{Z}-\mathbf{D}\mathbf{X})_i\rangle + \frac{\beta}{2}\|\mathbf{Z}-\mathbf{D}\mathbf{X}\|_F^2, \tag{3.6}$$

where $\mathbf{\Lambda}$ is the Lagrange multiplier matrix, and $\mathbf{\Lambda}_i$ is the ith column of $\mathbf{\Lambda}$.

Applying ADMM to the model, the $\mathbf{X}$ subproblem is

$$\min_{\mathbf{X}} \sum_{i=1}^{L}\langle \mathbf{\Lambda}_i, (\mathbf{Z}-\mathbf{D}\mathbf{X})_i\rangle + \frac{\beta}{2}\|\mathbf{Z}-\mathbf{D}\mathbf{X}\|_F^2 \text{ s.t } \|\mathbf{x}_i\|_0 \le \tau_0, i=1,\dots,L \tag{3.7}$$

which is equal to

$$\min_{\mathbf{X}} \frac{\beta}{2}\|\mathbf{Z}+\mathbf{\Lambda}/\beta-\mathbf{D}\mathbf{X}\|_F^2 \text{ s.t } \|\mathbf{x}_i\|_0 \le \tau_0, i=1,\dots,L. \tag{3.8}$$

It can be solved column by column as a sparse coding stage. The sparse coding part is implemented using BOMP [16], one of the extensions of OMP.

The $\mathbf{Z}$ subproblem is reformulated as

$$\min_{\mathbf{Z}} \|\mathbf{Y}-\mathbf{Z}\|_F^2 + \sum_{i=1}^{L}\langle \mathbf{\Lambda}_i, (\mathbf{Z}-\mathbf{D}\mathbf{X})_i\rangle + \frac{\beta}{2}\|\mathbf{Z}-\mathbf{D}\mathbf{X}\|_F^2. \tag{3.9}$$

It has the closed-form solution

$$\mathbf{Z} = (\beta\mathbf{D}\mathbf{X}+2\mathbf{Y}-\mathbf{\Lambda})/(2+\beta). \tag{3.10}$$

The $\mathbf{D}$ subproblem is

$$\min_{\mathbf{D}} \frac{\beta}{2}\|\mathbf{Z}+\mathbf{\Lambda}/\beta-\mathbf{D}\mathbf{X}\|_F^2. \tag{3.11}$$

To accelerate this step, we give an inexact solution in which the dictionary is updated column by column.

Let

$$\begin{aligned} \mathbf{E}_0 &:= \mathbf{Z}+\mathbf{\Lambda}/\beta-\mathbf{D}^{(k)}\mathbf{X}, \\ \xi_j &:= \mathbf{X}(j,:)\mathbf{X}(j,:)^T. \end{aligned} \tag{3.12}$$

The update can be written as

$$\mathbf{D}^{(k+1)}(:,j) = \mathbf{D}^{(k)}(:,j) + \mathbf{E}_0\mathbf{X}(j,:)^T/(\xi_j+\delta). \tag{3.13}$$

Here $\mathbf{D}(:,j)$ is the jth column of $\mathbf{D}$, $\mathbf{D}^{(k)}(:,j)$ is the jth column of $\mathbf{D}$ in the kth iteration, $\mathbf{X}(j,:)$ is the jth row of $\mathbf{X}$, and δ is a small value, such as 10^{-8}.

The $\mathbf{\Lambda}$ can be updated as

$$\mathbf{\Lambda}^{(k+1)} = \mathbf{\Lambda}^{(k)} + \gamma\beta(\mathbf{Z} - \mathbf{DX}). \tag{3.14}$$

The framework of our algorithm is called the ADMM for Dictionary Learning, or ADL.

Algorithm 2 ADMM for Dictionary Learning(ADL)

```
Require: Signal set $\mathbf{Y}$, initial dictionary $\mathbf{D} = \mathbf{D}^{(0)}$, initial lagrange multiplier matrix $\mathbf{\Lambda} = \mathbf{\Lambda}^{(0)}$, target
  sparsity $\tau_0$, number of iterations $K$, a small value $\delta$, parameter $\beta$.
Ensure: Dictionary $\mathbf{D}$ and sparse matrix $\mathbf{X}$ such that $\mathbf{Y} \approx \mathbf{DX}$.
  for $k = 1 \ldots K$ do
    Solve $\mathbf{x}_i := \arg\min_{\mathbf{x}} \|\mathbf{y}_i - \mathbf{Dx}\|_2^2$ subject to $\|\mathbf{x}\|_0 \leq \tau_0,\ \forall i \in \{1, \ldots, L\}$
    $\mathbf{Z} = (\beta\mathbf{DX} + 2\mathbf{Y} - \mathbf{\Lambda})/(2 + \beta)$
    Let $\mathbf{E}_0 := \mathbf{Z} + \mathbf{\Lambda}/\beta - \mathbf{DX}$
    for $j = 1 \ldots n$ do
      $\xi_j := \mathbf{X}(j,:)\mathbf{X}(j,:)^T$
      $\mathbf{D}(:,j) = \mathbf{D}(:,j) + \mathbf{E}_0\mathbf{X}(j,:)^T/(\xi_j + \delta)$
    end for
    Normalize $\mathbf{D}$
    Let $\mathbf{\Lambda} = \mathbf{\Lambda} + \gamma\beta(\mathbf{Z} - \mathbf{DX})$
  end for
```

Here we give some implementation details. Just like the K-SVD algorithm, ADL is susceptible to local minimum traps. The parameters β and $\mathbf{\Lambda}$ may influence the convergence of the algorithm. In our experiments, we simply designate $\beta = 1$, $\gamma = 0.618$, $\mathbf{\Lambda}^{(0)}$ to be all one matrix, and $\delta = 10^{-8}$. Other choices can be used according to the application.

3.3 Convergence Analysis

The convergence of the ADM and its variants for convex problems has been studied extensively. We state a well-known result below for completeness.

Theorem 3.1 *For the problem* (3.1), *if* $f : \mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$ *and* $g : \mathbf{R}^m \to \mathbf{R} \cup \{+\infty\}$ *are closed, proper, and convex. The unaugmented Lagrangian* L_0 *has a saddle point. Then, the residual,* $\mathbf{Ax} + \mathbf{By} - \mathbf{c}$, *convergence to* $\mathbf{0}$, *the objective convergence and the dual variable convergence.*

Unlike the convex case, when the objective becomes nonconvex , the convergence issue of ADMM remains largely open. For nonconvex problems, ADMM can converge to different (and in particular, nonoptimal) points, depending on the initial values $\mathbf{x}^{(0)}$, $\mathbf{y}^{(0)}$, and the parameter β. The papers [37] and [38] analyze the convergence of the ADMM for solving certain nonconvex problems. The papers [32–36],we have mentioned before, which apply ADMM to nonconvex problems, all give the convergence results under mild conditions.

On the other hand, empirical evidence suggests that our scheme seems to converge from any starting point. Even though we cannot provide a complete proof, the following theorem shows that ADL can be used to find an stationary point of (3.4) under some mild conditions. Although far from satisfactory, this result nevertheless provides some assurance on the reliability of the algorithm.

Theorem 3.2 *For the problem* (3.5), *let* $\mathbf{P} \triangleq \{\mathbf{X}, \mathbf{Z}, \mathbf{D}, \mathbf{\Lambda}\}$ *be a sequence generated by ADL that satisfies the condition* $\lim_{k\to\infty} \left(\mathbf{P}^{(k+1)} - \mathbf{P}^{(k)}\right) = 0$. *Then any accumulation point of* $\left\{\mathbf{P}_{k=1}^{(k)}\right\}_{k=1}^{\infty}$ *is a KKT point of the problem.*

Proof For a given $\mathbf{X}$, the constrained $\|\mathbf{x}_i\|_0 \leq \tau_0, i = 1, \ldots, L$ can be written as $(I_{\bar{J}_i})^T \mathbf{x}_i = 0$. J_i is an index set with $|J_i| = \tau_0$ such that the jth element of $\mathbf{x}_i$ is 0 for all $j \in \bar{J}_i$, where $\bar{J}_i = \{1, \ldots, n\} \backslash J_i$. $I_{\bar{J}_i}$ is a vector where the elements with index in $\bar{J}_i$ are 1 and others are 0. For more details, refer to [39] **Thm 2.1**.

Let $\{\hat{\mathbf{X}}, \hat{\mathbf{Z}}, \hat{\mathbf{D}}, \hat{\mathbf{\Lambda}}\}$ be the limit point of the sequence. Then we have $\hat{\mathbf{Z}} - \hat{\mathbf{D}}\hat{\mathbf{X}} = 0$ from the update of Lagrange multiplier matrix, $\mathbf{\Lambda}^{(k+1)} = \mathbf{\Lambda}^{(k)} + \gamma\beta(\mathbf{Z} - \mathbf{D}\mathbf{X})$.

For $\mathbf{Z}$ subproblem, it has the explicit solution

$$(2 + \beta)\hat{\mathbf{Z}} - \left(\beta\hat{\mathbf{D}}\hat{\mathbf{X}} + 2\mathbf{Y} - \hat{\mathbf{\Lambda}}\right) = 0. \tag{3.15}$$

As $\hat{\mathbf{Z}} - \hat{\mathbf{D}}\hat{\mathbf{X}} = 0$, hence

$$2\hat{\mathbf{Z}} - 2\mathbf{Y} + \hat{\mathbf{\Lambda}} = 0. \tag{3.16}$$

For $\mathbf{D}$ subproblem, it satisfies

$$(\hat{\mathbf{Z}} + \frac{\hat{\mathbf{\Lambda}}}{\beta})\hat{\mathbf{X}}^T - \hat{\mathbf{D}}\hat{\mathbf{X}}\hat{\mathbf{X}}^T = 0. \tag{3.17}$$

As $\hat{\mathbf{Z}} - \hat{\mathbf{D}}\hat{\mathbf{X}} = 0$, hence

$$\frac{\hat{\mathbf{\Lambda}}}{\beta}\hat{\mathbf{X}}^T = 0. \tag{3.18}$$

For the $\mathbf{X}$ subproblem, the Lagrange function can be written as

$$L(\mathbf{X}) = \|\mathbf{Z} + \mathbf{\Lambda}/\beta - \mathbf{D}\mathbf{X}\|_F^2 + \mathbf{\Lambda}_{\mathbf{X}}{}^T(I_{\bar{J}}^T\mathbf{X}). \tag{3.19}$$

The $\hat{\mathbf{X}}$ is at least the local minimal of the subproblem in our algorithm, so it is the KKT point of the subproblem. The KKT condition is

$$-\hat{\mathbf{D}}^T\left(\hat{\mathbf{Z}} + \frac{\hat{\mathbf{\Lambda}}}{\beta}\right) + \hat{\mathbf{D}}^T\hat{\mathbf{D}}\hat{\mathbf{X}} + I_{\bar{J}}\mathbf{\Lambda}_{\hat{\mathbf{X}}} = 0. \tag{3.20}$$

As $\hat{\mathbf{Z}} - \hat{\mathbf{D}}\hat{\mathbf{X}} = 0$, then

$$-\hat{\mathbf{D}}^T\frac{\hat{\mathbf{\Lambda}}}{\beta} + I_{\bar{J}}\mathbf{\Lambda}_{\hat{\mathbf{X}}} = 0. \tag{3.21}$$

Put the above equations together,

$$\begin{aligned} -\hat{\mathbf{D}}^T \frac{\hat{\boldsymbol{\Lambda}}}{\beta} + I_{\bar{J}} \boldsymbol{\Lambda}_{\hat{\mathbf{X}}} &= 0, \\ \frac{\hat{\boldsymbol{\Lambda}}}{\beta} \hat{\mathbf{X}}^T &= 0, \\ 2\hat{\mathbf{Z}} - 2\mathbf{Y} + \hat{\boldsymbol{\Lambda}} &= 0, \\ \hat{\mathbf{Z}} - \hat{\mathbf{D}}\hat{\mathbf{X}} &= 0. \end{aligned} \tag{3.22}$$

The Lagrange function of the original problem is

$$L(\mathbf{X}, \mathbf{Z}, \mathbf{D}, \boldsymbol{\Lambda}, \boldsymbol{\Lambda}_{\mathbf{X}}) = \|\mathbf{Y} - \mathbf{Z}\|_F^2 + \boldsymbol{\Lambda}^T (\mathbf{Z} - \mathbf{D}\mathbf{X}) + \boldsymbol{\Lambda}_{\mathbf{X}}{}^T (I_{\bar{J}}^T \mathbf{X}). \tag{3.23}$$

The above equations are just the KKT condition of the Lagrange function. This completes the proof. □

The convergence analysis of nonconvex dictionary learning model under some strict conditions is very difficult. We hope to dig deeper in our future work. In the following section, we do some numerical experiments to show the practicality of our algorithm.

4 Numerical Experiments

4.1 Synthetic Experiments

We first try the algorithm on synthetic signals to test whether this algorithm recovers the original dictionary that generated the data, and to compare its results with other reported algorithms. The experiment we describe includes the following steps:

- *Generation of the data to train on* A random matrix **D** (referred to later-on as the generating dictionary) of size 20×50 is generated with i.i.d uniformly distributed entries. Each column is normalized to a unit ℓ_2-norm. Then, 1500 data signals $\{\mathbf{y}_i\}_{i=1}^{1500}$ of dimension 20 are produced, each created by a linear combination of three different generating dictionary atoms, with uniformly distributed i.i.d coefficients in random and independent locations. White Gaussian noise with varying SNR is added to the resulting data signals. To test the speed of our algorithm, larger dictionaries and more signals are generated.
- *Applying ADL* The dictionary is initialized with 50 and more data signals. The coefficients are found using BOMP with a fixed number of 3, and other numbers of coefficients. The maximum number of iterations is set to 100.
- *Comparison to other reported works* We implement the K-SVD algorithm, and apply it on the same data, using BOMP with the same fixed number of coefficients, and initializing in the same way. We also implement another algorithm called block coordinate update (BCU) [40]. It solves the convex model of dictionary learning. We execute the K-SVD and BCU algorithm for the same number of iterations.
- *Results* The RMSE is compared to the ones from K-SVD and BCU, which is defined as $\sqrt{\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 / numel(\mathbf{X})}$. The $numel(\mathbf{X})$ is the number of elements in **X**. The

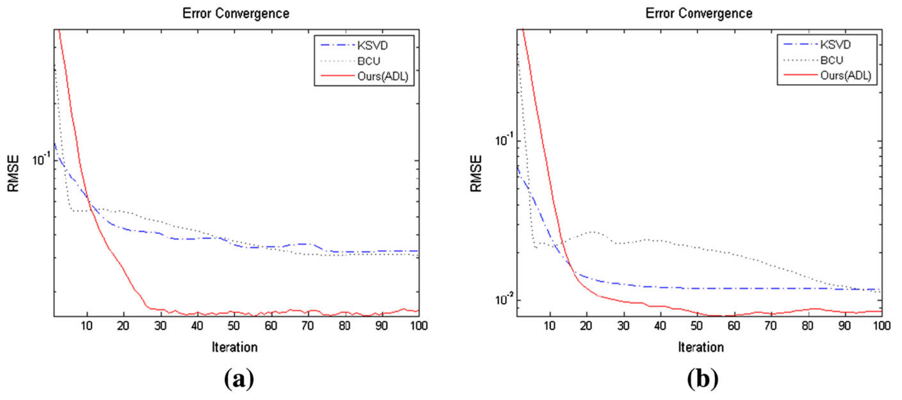

Fig. 1 RMSE comparison. **a D** : 20×50 $\tau_0 = 3$ with 1500 data signals; **b D** : 100×210 $\tau_0 = 5$ with 1500 data signals

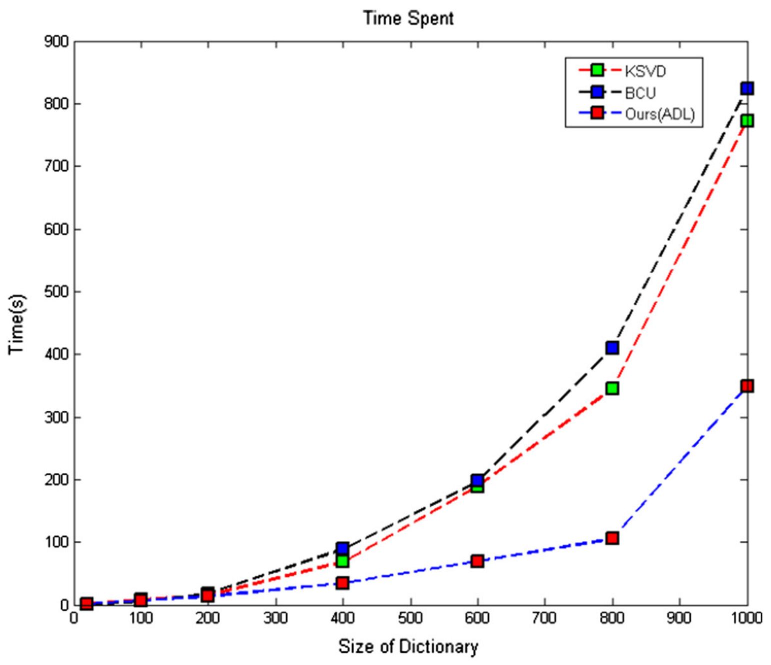

Fig. 2 Speed comparison:The *X-axis* represents the row number in the dictionary, and each row's number is doubled to obtain the corresponding column number. The number of data signals differs from 2000 to 4000. The sparsity is between 5 and 15

speed is also compared with K-SVD and BCU as the size of the dictionary grows. The figures below show the results.

Figure 1 shows that the RMSE obtained by our method is consistently smaller than the ones from K-SVD and BCU. More subsequent tests were conducted in order to prove this fact. Figure 2 shows that our method is faster than K-SVD and BCU, using

Table 1 Speed comparison

N	20	100	200	400	600	800	1000
Signal no.	2000	2000	2000	2000	2000	3000	4000
K-SVD (s)	1.50	7.84	15.26	68.24	188.78	344.71	772.56
BCU (s)	0.66	5.53	17.66	89.10	197.17	409.56	824.52
ADL (s)	1.29	7.27	13.69	34.74	69.15	105.20	349.52

the same number of iterations. When the size of the dictionary gets larger, it seems that our method is more efficient. Table 1 shows the corresponding set of numbers.

4.2 Application to Image Processing

We carried out several experiments on natural image data to try to show the practicality of our algorithm. All reported numerical simulations were implemented via MATLAB 8.0 on a 3.3 GHz Dual-Core Windows 7 PC with 8GB memory. This section is organized as follows. In the following subsection, we will first outline the bases learning details. Next, we will describe the image inpainting problem, and demonstrate the inpainting results. For more details about inpainting, see [28,30,41].

4.2.1 Bases Learning

Six images of natural scenes, shown in Fig. 3, are used as the training set for learning the basis matrix. Images were taken from a publicly available database[1] and converted to grayscale. The training images all measure 576×768 pixels. We then randomly extracted 18,000 patches, each measuring 8×8 pixels, from the six training images (3000 patches per training image) and organized them as columns in the $64 \times 180,00$ data matrix $\mathbf{Y}$. The mean value was subtracted from every patch: this is a very important preprocessing step. Then, the K-SVD, BCU and our method are used for basis learning. Patches of the size 16×16 pixels are also used in our paper. An important point is that the same patch sizes are used for these three bases, which allows fair comparison between them.

In our method and the K-SVD, the sparse coding stage was implemented by using the BOMP [16] algorithm with 5 nonzero coefficients of a solution. Figure 4 shows the basis vectors for the 8×8-pixel patches, which were learned by the K-SVD and the ADL algorithms.

4.2.2 Image Inpainting

We made use of the freely available MATLAB implementation of the Smoothed $\ell_0(SL_0)$ algorithm for image reconstruction, i.e., inpainting. In our simulations, the

[1] A. Olmos, F. A. A. Kingdom, McGill calibrated color image database, 2004. http://pirsquared.org/research/mcgilldb/.

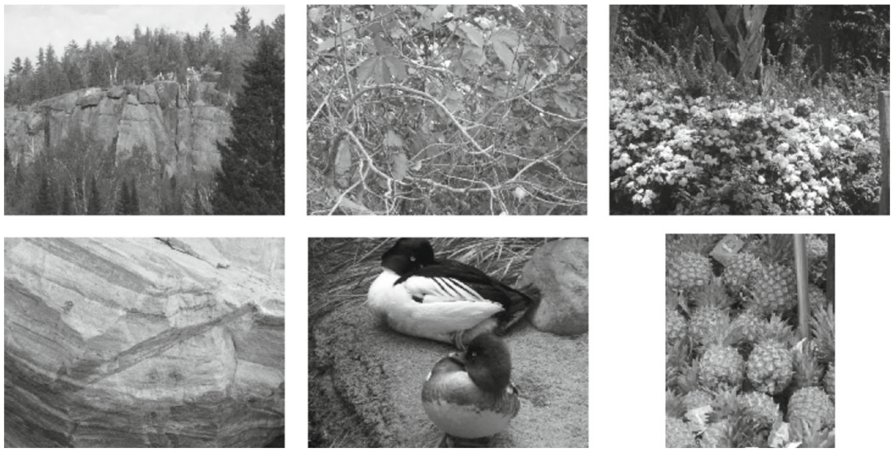

Fig. 3 Six images from the training set used for bases learning. Images were randomly selected from McGill calibrated color image database

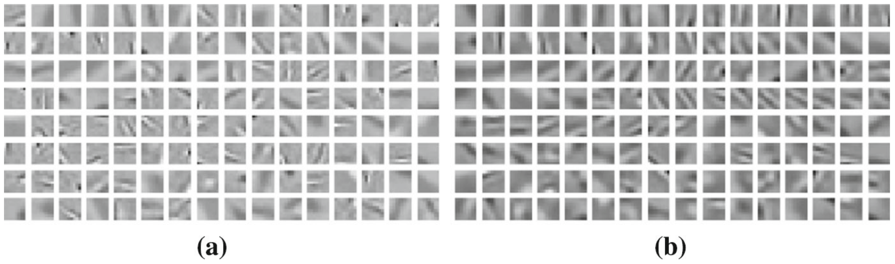

Fig. 4 128 matricized basis vectors of the size 8 × 8 pixels learned by ADL (**a**), and K-SVD (**b**). Basis vectors are columns of the learned basis matrix

SL_0 algorithm [42] worked better and was much faster than other approaches, especially those using ℓ_1 minimization. Prior to conducting the reconstruction, the mean value of the observed pixels in each patch was subtracted from the vector of the observed pixels and added back after the reconstruction. To prevent border effects, reconstruction was done with two rows, i.e., columns, of adjacent patches overlapping. After reconstruction, the overlapping regions were averaged.

For measuring the quality of the reconstructed images, we use the peak signal-to-noise ratio (PSNR). For comparison, we have also used the BCU [40] method, the DCT bases, and TV modeling inpainting method [43] for image inpainting. MATLAB implementation of the TV method is available at.[2] Default values of the parameters were used.

The six images shown in Fig. 5, also taken from[1], are used as the validation set. These images were reshaped to the size of 512 × 512 pixels. First, we randomly generated the missing pixels distribution. We repeated the inpainting experiment 10 times for every image in the validation set. Table 2 shows the detailed results from

[2] TV, http://www.imm.dtu.dk/~pcha/mxTV/.

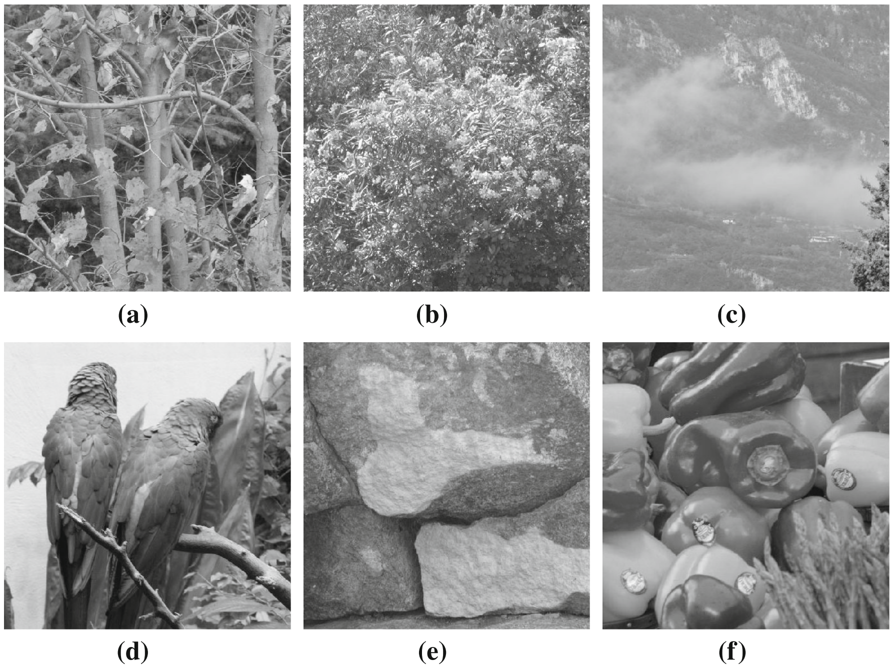

Fig. 5 Images used for validation purposes. Images were randomly selected from McGill calibrated color image database

Table 2 Inpainting results in terms of the PSNR metric for the overcomplete bases learned on patches of size 8×8 pixels

	ADL	K-SVD	BCU	DCT	TV
Fig. a	28.5 ± 0.01	27.0 ± 0.02	24.7 ± 0.03	24.2 ± 0.01	24.3 ± 0.01
Fig. b	22.2 ± 0.04	20.6 ± 0.04	21.0 ± 0.01	20.6 ± 0.09	21.2 ± 0.02
Fig. c	37.5 ± 0.04	36.1 ± 0.05	33.4 ± 0.08	32.9 ± 0.07	26.0 ± 0.01
Fig. d	34.6 ± 0.02	33.5 ± 0.07	28.9 ± 0.05	28.2 ± 0.01	25.5 ± 0.08
Fig. e	34.0 ± 0.04	32.5 ± 0.02	30.0 ± 0.08	29.5 ± 0.01	25.8 ± 0.05
Fig. f	38.8 ± 0.03	37.9 ± 0.09	31.9 ± 0.05	30.9 ± 0.04	25.8 ± 0.04
Mean	32.6 ± 0.03	31.3 ± 0.02	28.3 ± 0.06	27.7 ± 0.05	24.8 ± 0.01

inpainting the six natural images in the validation set using all mentioned methods. 50 % of the pixels from all six pictures were removed randomly from each image in the validation set. All bases were learned on patches measuring 8×8 pixels, wherein the number of atoms used in the training phase was 5. All bases were twice overcomplete, i.e., measuring 64×128. Numbers in the table stand for the mean values and standard deviations for the reconstructed images after 10 runs.

It is clear that our method outperformed the other methods. Results show that ADL performed better for the images with more details. The images recovered by ADL

Table 3 Inpainting results in terms of the PSNR metric for the overcomplete bases learned on patches of size 16 × 16 pixels

	ADL	K-SVD	BCU	DCT	TV
Fig. a	28.1 ± 0.03	26.1 ± 0.09	22.4 ± 0.09	22.1 ± 0.01	24.3 ± 0.01
Fig. b	22.4 ± 0.06	19.7 ± 0.06	19.8 ± 0.02	19.6 ± 0.04	21.2 ± 0.02
Fig. c	37.1 ± 0.08	35.4 ± 0.05	31.6 ± 0.01	31.1 ± 0.08	26.0 ± 0.01
Fig. d	33.5 ± 0.02	32.2 ± 0.01	25.4 ± 0.08	24.9 ± 0.02	25.5 ± 0.08
Fig. e	33.6 ± 0.04	31.9 ± 0.04	28.0 ± 0.08	27.5 ± 0.06	25.8 ± 0.05
Fig. f	37.4 ± 0.05	36.6 ± 0.06	27.8 ± 0.08	27.0 ± 0.07	25.8 ± 0.04
Mean	32.0 ± 0.06	30.3 ± 0.07	25.8 ± 0.09	25.4 ± 0.01	24.8 ± 0.01

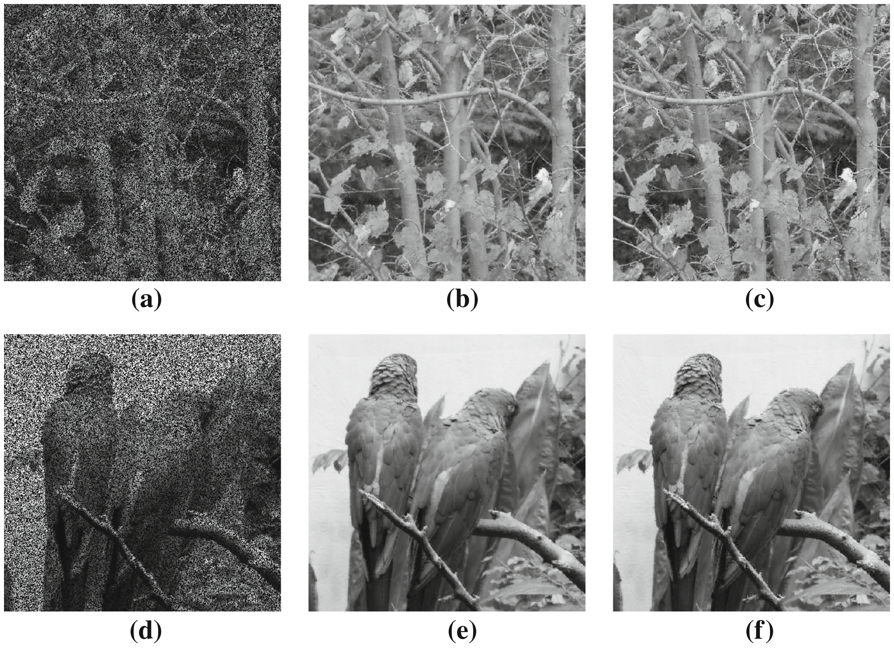

Fig. 6 Examples of two images from the validation set with 50 % of the pixels missing: (**a**) and (**d**). Inpainting of two degraded images using ADL learned basis: (**b**) and (**e**). Inpainting of two degraded images using K-SVD learned basis: (**c**) and (**f**). Both bases were learned on patches of size 8 × 8 pixels. **a** Damaged image; **b** PSNR:28.16; **c** PSNR:26.17; **d** Damaged image; **e** PSNR:33.57; **f** PSNR:32.14

dictionaries seemed smoother than the ones reconstructed by K-SVD dictionaries. For comparison, in Table 3, we also show results obtained by bases learned on 16×16-pixel patches, where 15 atoms were used in the training. It can be seen that the comparative performance of the two bases is similar. However, the algorithms performed better with smaller patches and smaller numbers of atoms in the training phase.

Figure 6 shows degraded and reconstructed versions of the two images from the validation set, whereupon 50 of the pixels were randomly removed from each image in the validation set.

Table 4 Inpainting results in terms of the PSNR metric for the overcomplete bases learned on patches of size 8 × 8 pixels with line patterns from missing pixels

	ADL	K-SVD	BCU	DCT	TV
Fig. a	38.1	35.03	37.48	37.06	37.17
Fig. b	32.18	27.53	32.68	32.48	32.77
Fig. c	47.94	45.68	47.07	46.62	42.12
Fig. d	40.55	38.46	40.73	40.30	39.66
Fig. e	43.51	40.64	42.67	42.31	39.81
Fig. f	44.84	44.43	44.23	43.43	40.96
Mean	41.19	38.63	40.81	40.37	38.58

Table 5 Inpainting results in terms of the PSNR metric for the overcomplete bases learned on patches of size 16 × 16 pixels with line patterns from missing pixels

	ADL	K-SVD	BCU	DCT	TV
Fig. a	37.43	34.74	35.66	35.29	37.17
Fig. b	31.21	28.53	31.80	31.63	32.77
Fig. c	47.15	45.98	45.38	44.88	42.12
Fig. d	38.62	37.58	38.00	37.47	39.66
Fig. e	42.59	40.43	41.23	40.75	39.81
Fig. f	42.54	42.07	41.05	40.10	40.96
Mean	39.92	38.22	38.86	38.36	38.58

We also include the results from the inpainting experiments with stronger structures for the missing pixels, namely, lines and text. Table 4 shows the results from inpainting images that were corrupted by line structures on 8 × 8 patches. The dictionary is the overcomplete 64 × 128 one used before. Table 5 shows the results on patches of size 16 × 16. It can be seen that our method outperformed other methods. Figure 7 shows two degraded and reconstructed images from the validation set, which are corrupted by line patterns from missing pixels.

We also show the results from the text inpainting. Table 6 shows the results from inpainting images that were corrupted by text structures on 8 × 8 patches. The overcomplete 64 × 128 dictionary is once again used. Table 7 shows results on patches of size 16 × 16. It can be seen that our method once again outperformed the other methods. Figure 8 shows two degraded and reconstructed images from the validation set.

5 Conclusion

In this paper, we present a new algorithm to solve the dictionary learning problem based on ADMM. We have shown that the dictionary learned by our method performs well for both synthetic and real images in applications such as inpainting and outperforms

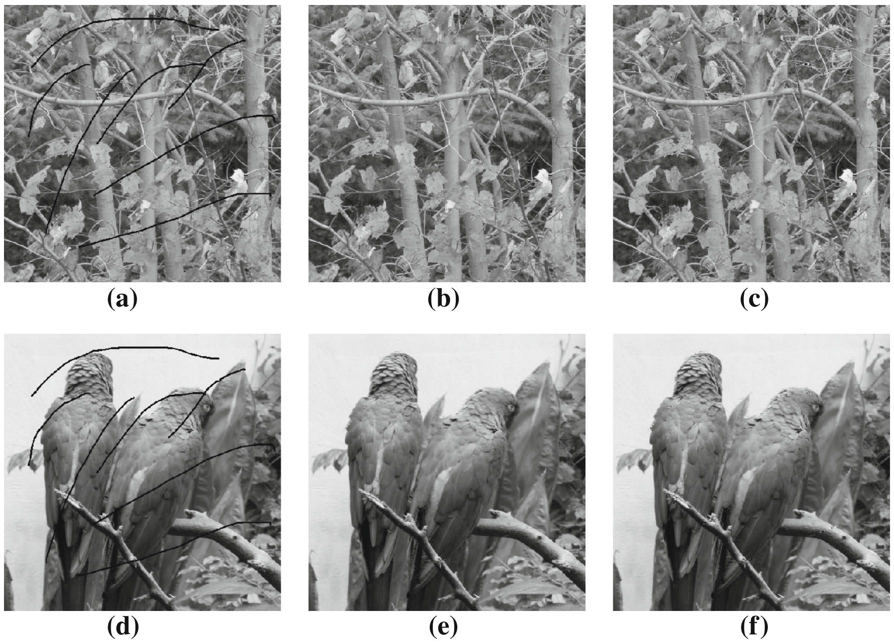

Fig. 7 Examples of two images from the validation set with line patterns from missing pixels: (**a**) and (**d**). Inpainting of two degraded images using ADL learned basis: (**b**) and (**e**). Inpainting of two degraded images using K-SVD learned basis: (**c**) and (**f**). Both bases were learned on patches of size 8 × 8 pixels. **a** Damaged image; **b** PSNR:38.10; **c** PSNR:35.03; **d** Damaged image; **e** PSNR:40.55; **f** PSNR:38.46

Table 6 Inpainting results in terms of the PSNR metric for the overcomplete bases learned on patches of size 8 × 8 pixels with text patterns from missing pixels

	ADL	K-SVD	BCU	DCT	TV
Fig a	32.47	30.36	31.42	31.03	31.42
Fig b	27.09	23.82	27.59	27.36	27.64
Fig c	43.69	42.33	42.15	41.67	35.36
Fig d	36.84	35.88	35.90	35.26	34.32
Fig e	38.38	36.34	37.09	36.56	34.34
Fig f	39.37	39.08	38.31	37.36	35.04
Mean	36.34	34.63	35.41	34.87	33.02

Table 7 Inpainting results in terms of the PSNR metric for the overcomplete bases learned on patches of size 16 × 16 pixels with text patterns from missing pixels

	ADL	K-SVD	BCU	DCT	TV
Fig a	32.09	31.12	29.59	29.19	31.42
Fig b	26.53	25.28	26.60	26.43	27.64
Fig c	43.27	42.29	40.55	39.97	35.36

Table 7 continued

	ADL	K-SVD	BCU	DCT	TV
Fig d	36.43	35.91	32.74	32.14	34.32
Fig e	37.52	36.39	35.36	34.79	34.34
Fig f	37.99	37.54	34.72	33.76	35.04
Mean	35.64	34.75	33.26	32.71	33.02

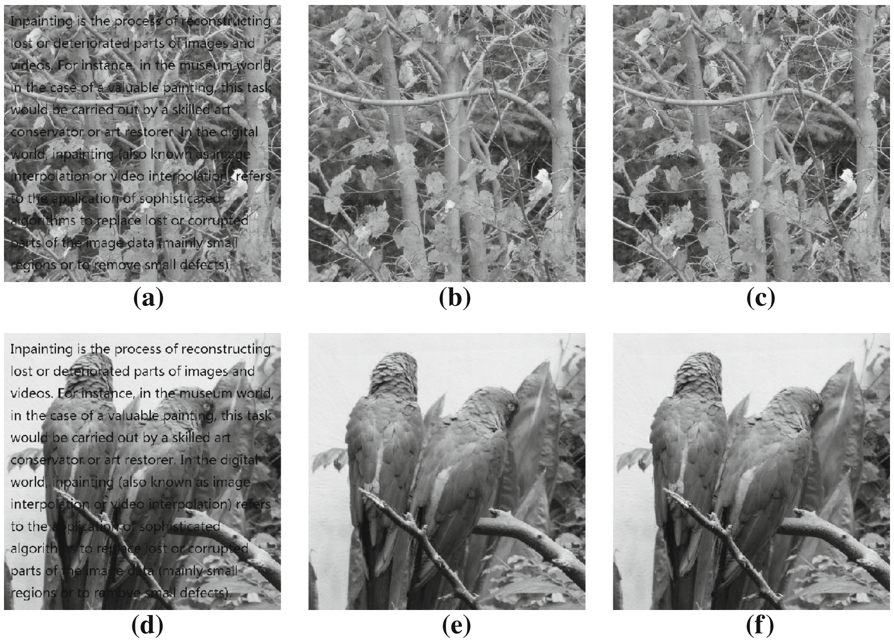

Fig. 8 Examples of two images from the validation set with the text patterns from missing pixels: (**a**) and (**d**). Inpainting of two degraded images using ADL learned basis: (**b**) and (**e**). Inpainting of two degraded images using K-SVD learned basis: (**c**) and (**f**). Both bases were learned on patches of size 8 $\times$ 8 pixels. **a** Damaged image; **b** PSNR:32.47; **c** PSNR:30.36; **d** Damaged image; **e** PSNR:36.84; **f** PSNR:35.88

the other mentioned methods. In addition, we tested numerous pictures to demonstrate that our method is more suitable for inpainting of images with greater details. We hope to provide some theoretical analysis of our algorithm in future work.

References

1. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learning **3**(1), 1–122 (2011)
2. Aharon, M., Elad, M., Bruckstein, A.: K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. Signal Process. IEEE Trans. **54**(11), 4311–4322 (2006)
3. Davis, G., Mallat, S., Avellaneda, M.: Adaptive greedy approximations. Constr. Approx. **13**(1), 57–98 (1997)

4. Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. Signal Process. IEEE Trans. **41**(12), 3397–3415 (1993)
5. Chen, S., Billings, S.A., Luo, W.: Orthogonal least squares methods and their application to non-linear system identification. Int. J. Control **50**(5), 1873–1896 (1989)
6. Davis, G.M., Mallat, S.G., Zhang, Z.: Adaptive time-frequency decompositions. Opt. Eng. **33**(7), 2183–2191 (1994)
7. Pati, Y.C., Rezaiifar, R., Krishnaprasad, P.: Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In: Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on IEEE, 1993, pp. 40–44
8. Tropp, J.A.: Greed is good: Algorithmic results for sparse approximation. Inf. Theory IEEE Trans. **50**(10), 2231–2242 (2004)
9. Tropp, J.A., Gilbert, A.C.: Signal recovery from random measurements via orthogonal matching pursuit. Inf. Theory IEEE Trans. **53**(12), 4655–4666 (2007)
10. Davenport, M.A., Wakin, M.B.: Analysis of orthogonal matching pursuit using the restricted isometry property. Inf. Theory IEEE Trans. **56**(9), 4395–4401 (2010)
11. Needell, D., Tropp, J., Vershynin, R.: Greedy signal recovery review. In: Signals, Systems and Computers, 2008 42nd Asilomar Conference on IEEE, 2008, pp. 1048–1050
12. Donoho, D.L., Tsaig, Y., Drori, I., Starck, J.-L.: Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. Inf. Theory IEEE Trans. **58**(2), 1094–1121 (2012)
13. Needell, D., Vershynin, R.: Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. Found. Comput. Math. **9**(3), 317–334 (2009)
14. Needell, D., Vershynin, R.: Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. Sel. Top. Signal Process. IEEE J. **4**(2), 310–316 (2010)
15. Needell, D., Tropp, J.A.: Cosamp: Iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmon. Anal **26**(3), 301–321 (2009)
16. Rubinstein, R., Zibulevsky, M., Elad, M.: Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. CS Tech. **40**, 1–15 (2008)
17. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM J. Sci. Comput. **20**(1), 33–61 (1998)
18. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
19. Van Den Berg, E., Friedlander, M.P.: Probing the pareto frontier for basis pursuit solutions. SIAM J. Sci. Comput. **31**(2), 890–912 (2008)
20. Becker, S., Bobin, J., Candès, E.J.: Nesta: A fast and accurate first-order method for sparse recovery. SIAM J. Imaging Sci. **4**(1), 1–39 (2011)
21. Donoho, D.L., Huo, X.: Uncertainty principles and ideal atomic decomposition. Inf. Theory IEEE Trans. **47**(7), 2845–2862 (2001)
22. Elad, M., Bruckstein, A.M.: A generalized uncertainty principle and sparse representation in pairs of bases. Inf. Theory IEEE Trans. **48**(9), 2558–2567 (2002)
23. Donoho, D.L., Elad, M.: Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization. Proc. Natl. Acad. Sci. **100**(5), 2197–2202 (2003)
24. Donoho, D.L., Elad, M., Temlyakov, V.N.: Stable recovery of sparse overcomplete representations in the presence of noise. Inf. Theory IEEE Trans. **52**(1), 6–18 (2006)
25. Tropp, J.A.: Just relax: Convex programming methods for identifying sparse signals in noise. Inf. Theory IEEE Trans. **52**(3), 1030–1051 (2006)
26. Engan, K., Aase, S.O., Husøy, J.H.: Multi-frame compression: Theory and design. Signal Process. **80**(10), 2121–2140 (2000)
27. Bruckstein, A.M., Donoho, D.L., Elad, M.: From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Rev. **51**(1), 34–81 (2009)
28. Elad, M., Aharon, M.: Image denoising via learned dictionaries and sparse representation. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, vol. 1, IEEE, 2006, pp. 895–900
29. Elad, M., Figueiredo, M.A., Ma, Y.: On the role of sparse and redundant representations in image processing. Proc. IEEE **98**(6), 972–982 (2010)
30. Elad, M.: Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing. Springer, New York (2010)

31. Rakotomamonjy, A.: Applying alternating direction method of multipliers for constrained dictionary learning. Neurocomputing **106**, 126–136 (2013)
32. Sun, D.L., Fevotte, C.: Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence. In: Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on IEEE, 2014, pp. 6201–6205
33. Ling, Q., Xu, Y., Yin, W., Wen, Z.: Decentralized low-rank matrix completion. In: Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on, IEEE, 2012, pp. 2925–2928
34. Shen, Y., Wen, Z., Zhang, Y.: Augmented lagrangian alternating direction method for matrix separation based on low-rank factorization. Optim. Methods Softw. **29**(2), 239–263 (2014)
35. Xu, Y., Yin, W., Wen, Z., Zhang, Y.: An alternating direction algorithm for matrix completion with nonnegative factors. Front. Math. China **7**(2), 365–384 (2012)
36. Wen, Z., Peng, X., Liu, X., Bai, X., Sun, X.: Asset allocation under the basel accord risk measures, Available at SSRN 2202845
37. Hong, M., Luo, Z.-Q., Razaviyayn, M.: Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems, arXiv preprint arXiv:1410.1390
38. Magnússon, S., Weeraddana, P.C., Rabbat, M. G., Fischione, C.: On the convergence of alternating direction lagrangian methods for nonconvex structured optimization problems, arXiv preprint arXiv:1409.8033
39. Lu, Z., Zhang, Y.: Sparse approximation via penalty decomposition methods. SIAM J. Optim. **23**(4), 2448–2478 (2013)
40. Xu, Y., Yin, W.: A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. SIAM J. Imaging Sci. **6**(3), 1758–1789 (2013)
41. Filipovi, M., Kopriva, I.: A comparison of dictionary based approaches to inpainting and denoising with an emphasis to independent component analysis learned dictionaries. Inverse Probl. Imaging **5**(4), 815–841 (2011)
42. Mohimani, G.H., Babaie-Zadeh, M., Jutten, C.: Fast sparse representation based on smoothed l0 norm. In: Independent Component Analysis and Signal Separation, Springer, 2007, pp. 389–396
43. Dahl, J., Hansen, P.C., Jensen, S.H., Jensen, T.L.: Algorithms and software for total variation image reconstruction via first-order methods. Numer. Algorithms **53**(1), 67–92 (2010)