BILINEAR DICTIONARY UPDATE VIA LINEAR LEAST SQUARES

Qi Yu[†⋆] *Wei Dai*[⋆] *Zoran Cvetkovic*[♮] *Jubo Zhu*[†]

[†] College of Liberal Arts and Sciences, National University of Defense Technology, Changsha, China

[⋆] Department of Electrical and Electronic Engineering, Imperial College London, UK

[♮] Department of Informatics, King's College London, UK

ABSTRACT

Algorithms for dictionary learning aim to learn a dictionary under which training data have sparse representations. This paper addresses the dictionary update sub-problem, the goal of which is to update the dictionary and the corresponding sparse coefficients given a fixed sparsity pattern. It is a non-convex bilinear inverse problem, and hence challenging to solve. Inspired by a recent work by Ling and Strohmer, we re-formulate the dictionary update problem as a linear least squares problem, which is convex and easy to solve. Necessary bounds on the number of training samples required for a unique solution are derived when exact sparsity pattern is known. Further, for dictionary update with unknown sparsity patterns, an efficient iterative algorithm based on total least squares is developed. Embedding the new dictionary update procedure into an overall dictionary learning algorithm achieves better numerical performance compared to state of the art algorithms.

Index Terms— Bilinear inverse problem, dictionary learning, dictionary update, linear least squares, total least squares

1. INTRODUCTION

Methodologies of sparse signal representation have been used in a wide range of signal processing applications, including denoising [1, 2], deconvolution [3], super-resolution [4, 5], *etc.* They are rooted in the idea that natural signals tend to have sparse representation under certain bases/dictionaries. Hence, finding a dictionary under which a given data set can be represented in a sparse manner, has become a very active area of research [6, 7]. More formally, dictionary learning is the problem of finding a dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times l}$ such that the training samples in $\boldsymbol{Y} \in \mathbb{R}^{m\times n}$ can be written as $\boldsymbol{Y} = \boldsymbol{D}\boldsymbol{X}$ where the coefficient matrix $\boldsymbol{X} \in \mathbb{R}^{l\times n}$ is sparse. Clearly, this problem is a non-convex bilinear inverse problem, and therefore challenging to solve in general.

The bilinear dictionary learning problem is typically solved by alternating between two stages: sparse coding and dictionary update [6–11]. In the sparse coding stage, the goal is to find sparse representations $\boldsymbol{x}_i$ of training samples $\boldsymbol{y}_i$ under a given dictionary $\boldsymbol{D}$. For that purpose, scores of algorithms have been developed. They can be divided into two main categories. The first category consists of greedy algorithms, including orthogonal matching pursuit (OMP) [12], regularized orthogonal matching pursuit (ROMP) [13], and subspace pursuit (SP) [14]. In the second category, sparse coding is formulated as a convex optimization problem using ℓ_1-norm [15] and then solved via optimization techniques, *e.g.* the fast iterative shrinkage-thresholding algorithm (FISTA) [16]. Reviews of sparse recovery algorithms can be found in [17, 18].

The goal of the dictionary update stage is to refine the dictionary so that the training samples $\boldsymbol{y}_i$ can be better represented. In the probabilistic framework of dictionary updating, one may apply either maximum likelihood (ML) estimator [6] or maximum a posteriori (MAP) estimator [9], and then solve them by using gradient descent procedures. Using the same objective function as in [6], Engan et al. [8] proposed the method of optimal directions (MOD) where the sparse coefficients $\boldsymbol{X}$ are fixed and the dictionary update problem is cast as a least squares problem which can be solved efficiently. Several modifications of MOD were proposed in [19–21]. An approach alternative to MOD, where the sparse coefficients are fixed in the dictionary update stage, is to update both the dictionary and the sparse coefficients while fixing the support of non-zero coefficients, i.e., the sparsity pattern. In the famous K-SVD algorithm, developed by Aharon and Elad [7], the dictionary is updated iteratively, in particular, in each iteration one dictionary atom and its corresponding coefficients are updated simultaneously. A combination of K-SVD and MOD was proposed in [22]. More recently SimCO algorithm was developed [10] for updating simultaneously multiple (or all) dictionary atoms and the corresponding coefficients, of which K-SVD is a particular case.

This paper focuses on the problem of dictionary update, where both the dictionary and the sparse coefficients are updated given a sparsity pattern. Our main contributions are as follows. 1) Inspired by the approach in [23, 24], we re-cast this bilinear inverse problem as a linear least squares problem which can be solved efficiently and admits a unique solution (up to scaling factors), when the dictionary is either under-complete or complete and the given sparsity pattern is exact. 2) Two necessary conditions for the uniqueness of the solution (exact dictionary recovery), expressed as lower bounds on the size of the sampling set, are derived. The two bounds are asymptotically identical when the dimensions of the dictionary, the number of training samples, and the number of nonzero sparse coefficients approach infinity proportionally. 3) A practical algorithm is developed for scenarios where the sparsity pattern is not known. It is based on total least squares, and when embedded into the overall dictionary learning process it leads to better performance and faster convergence rate compared to state of the art.

This work is supported by China Scholarship Council and Royal Society International Exchanges 2017 Cost Share (with China)

2. DICTIONARY LEARNING AND BILINEAR DICTIONARY UPDATE PROBLEM

The overall dictionary learning problem is often formulated as:

$$\min_{\boldsymbol{D},\boldsymbol{X}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2, \text{ s.t. } \|\boldsymbol{x}_i\|_0 \leqslant k \ , \ \forall i \tag{1}$$

where $\boldsymbol{x}_i$ is the i-th column of $\boldsymbol{X}$, $\|\boldsymbol{x}_i\|_0$ denotes the number of non-zero elements, also known as sparsity level of $\boldsymbol{x}_i$, and $k \ll l$ is an integer upper bounding the sparsity level of $\boldsymbol{x}_i$.

Most dictionary learning algorithms alternate between two stages: sparse coding and dictionary update. In the sparse coding stage, the dictionary is fixed and columns of sparse coefficient matrix $\boldsymbol{X}$ are found by solving

$$\min_{\boldsymbol{x}_i} \|\boldsymbol{y}_i-\boldsymbol{D}\boldsymbol{x}_i\|_2^2, \text{ s.t. } \|\boldsymbol{x}_i\|_0 \leqslant k \ , \ \forall i \ , \tag{2}$$

using greedy algorithms, for example OMP [12] and SP [14].

The second step, dictionary update, is where various dictionary learning algorithms differ. For example, MOD method simply solves a least squares problem: $\min_{\boldsymbol{D}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2$, when coefficient matrix $\boldsymbol{X}$ is fixed from the first stage.

In many other methods, the dictionary update problem is defined as finding the dictionary and the corresponding sparse coefficients with a fixed sparsity pattern:

$$\min_{\boldsymbol{D},\boldsymbol{X}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2, \text{ s.t. } \mathscr{P}_{\Omega^c}(\boldsymbol{X})=\boldsymbol{0} \tag{3}$$

where Ω is the support set of sparse matrix $\boldsymbol{X}$, Ω^c denotes its complement, and $\mathscr{P}_{\Omega^c}(\cdot)$ is an operator that retrieves the values of the entries of $\boldsymbol{X}$ indexed by Ω^c.

One way to solve the bilinear inverse problem (3) is K-SVD, which updates one dictionary atom and the corresponding row of $\boldsymbol{X}$ at a time while fixing all the others:

$$\{\hat{\boldsymbol{d}}_{j_0},\hat{\boldsymbol{X}}_{j_0,:}\} = \arg\min_{\boldsymbol{d}_{j_0},\boldsymbol{X}_{j_0,:}} \| \left(\boldsymbol{Y}-\sum_{j\neq j_0}\boldsymbol{d}_j\boldsymbol{X}_{j,:}\right)-\boldsymbol{d}_{j_0}\boldsymbol{X}_{j_0,:}\|_F^2, \tag{4}$$

where $\boldsymbol{X}_{j,:}$ denotes the j-th row of $\boldsymbol{X}$. This is achieved by applying singular value decomposition (SVD) to the residue matrix $\boldsymbol{R}_{j_0}=\left(\boldsymbol{Y}-\sum_{j\neq j_0}\boldsymbol{d}_j\boldsymbol{X}_{j,:}\right)$. Another way to handle (3) is to update all the dictionary atoms in $\boldsymbol{D}$ simultaneously using gradient descent method; see SimCO [10].

The focus of this paper is to solve (3) using the linear least squares approach.

3. EXACT DICTIONARY RECOVERY

This section focuses on the case when the dictionary can be exactly recovered by assuming that the training samples in $\boldsymbol{Y}$ are generated from $\boldsymbol{Y}=\boldsymbol{D}_0\boldsymbol{X}_0$ where $\boldsymbol{D}_0$ is a tall or square matrix ($m \geqslant l$) and the sparsity pattern of $\boldsymbol{X}_0$ is given. For compositional convenience, we focus on the case where $\boldsymbol{D}_0$ is a square matrix, $\boldsymbol{D}_0 \in \mathbb{R}^{m\times m}$, as the same analysis is valid for a tall dictionary.

The bilinear inverse problem (3) is equivalent to

$$\begin{aligned} &\text{Find } \boldsymbol{D},\boldsymbol{X} \\ &\text{s.t. } \boldsymbol{Y}=\boldsymbol{D}\boldsymbol{X}, \mathscr{P}_{\Omega^c}(\boldsymbol{X})=0. \end{aligned} \tag{5}$$

Define $\boldsymbol{H}=\boldsymbol{D}^{-1}$. Then by using $\boldsymbol{H}\boldsymbol{Y}=\boldsymbol{X}$, the problem in (5) can be re-cast as:

$$\begin{aligned} &\text{Find } \boldsymbol{H},\boldsymbol{X} \\ &\text{s.t. } \begin{bmatrix}\boldsymbol{H} & \boldsymbol{X}\end{bmatrix}\begin{bmatrix}\boldsymbol{Y}\\ -\boldsymbol{I}\end{bmatrix}=\boldsymbol{0}, \text{ s.t. } \mathscr{P}_{\Omega^c}(\boldsymbol{X})=\boldsymbol{0}, \end{aligned} \tag{6}$$

which is a linear least squares problem with linear constraints. To avoid trivial solutions, further linear constraints $\sum_j h_{i,j}=1$, $\forall i$, are added in practice.

As solutions of bilinear inverse problems are not unique in general, we define a class of equivalent solutions in the context of unique dictionary recovery.

Definition 1 (Unique Recovery)**.** The bilinear dictionary update problem (5) admits a unique solution if all feasible solutions satisfy $\boldsymbol{D}=\boldsymbol{D}_0\boldsymbol{\Sigma}$ and $\boldsymbol{X}=\boldsymbol{\Sigma}^{-1}\boldsymbol{X}_0$, where $\boldsymbol{\Sigma}$ is an arbitrary non-singular diagonal matrix.

Based on the definition, a necessary sampling bound for unique recovery is given as folllows:

Theorem 1 (Rough Necessary Bound for Exact Dictionary Recovery)**.** *Assume that the number of nonzero elements in each column of $\boldsymbol{X}_0$ is $k_i, i=\{1,2,...,n\}$ and $\boldsymbol{D}_0$ is square and invertible. If bilinear problem (5) has unique solution, then $n \geqslant m+\frac{\sum_{i=1}^n k_i}{m}$.*

This theorem is proved by using the fact that the solution is unique only if the number of independent equations is at least the number of unknown variables.

Other than the rough bound given in Theorem 1, two more necessary bounds are also derived, which provide more insight into exact dictionary recovery. To proceed, for an arbitrary integer $k<l$ ($l=m$ for complete dictionaries), we define a Bernoulli-Gaussian distribution $BG(\frac{k}{l})$ of vectors.

Definition 2. A vector $\boldsymbol{x}$ satisfies the Bernoulli-Gaussian distribution $BG(\theta)$ with parameter $\theta \in [0,1]$, if $\boldsymbol{x}=\boldsymbol{\omega}\odot\boldsymbol{c}$, where $\boldsymbol{\omega}$ is an iid Bernoulli(θ) vector, and $\boldsymbol{c}\sim N(0,\boldsymbol{I})$ is a Gaussian random vector which is independent of $\boldsymbol{\omega}$.

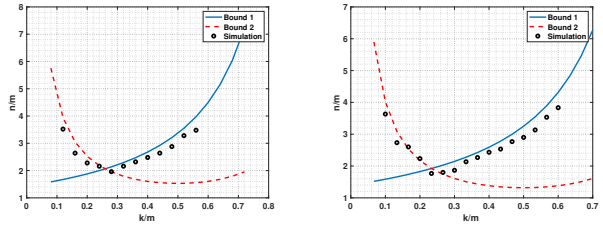
Theorem 2 (Fine Necessary Bounds for Exact Dictionary Recovery)**.** *For the dictionary update problem (5), suppose that $\boldsymbol{D}_0$ is square and invertible and columns of $\boldsymbol{X}_0$ satisfy iid Bernoulli-Gaussian distribution $BG(\frac{k}{m})$.*

For any given constant $\varepsilon \in (0,1)$, if the programming (6) recovers $\boldsymbol{D}_0$ and $\boldsymbol{X}_0$ with probability at least $(1-\varepsilon)$, the number of training samples $n \geqslant \max(n_1,n_2)$ where

$$\begin{aligned} n_1 &= \frac{\left[2(m-1)-\frac{m\cdot\log(\frac{\varepsilon}{m})}{2(m-k)}\right]+\sqrt{\left[2(m-1)-\frac{m\cdot\log(\frac{\varepsilon}{m})}{2(m-k)}\right]^2-4(m-1)^2}}{2\cdot(1-\frac{k}{m})} \\ n_2 &= \frac{\log\left(\frac{2m(m-1)}{\varepsilon}\right)}{\log\left(\frac{m^2}{k^2+m(m-k)}\right)}. \end{aligned} \tag{7}$$

Remark 1. When m is large, $n_1=O(C_1(\frac{k}{m},\varepsilon)\cdot m)$, $n_2=O(C_2(\frac{k}{m},\varepsilon)\cdot\log(m))$, where $C_1(\frac{k}{m},\varepsilon)$ and $C_2(\frac{k}{m},\varepsilon)$ are constants which depend on the sparsity $\frac{k}{m}$ and probability threshold ε.

Fig. 1 shows that the necessary bounds in Theorem 2 matches the numerical simulations even for relatively small m,k,n. In the simulations, Gaussian dictionaries $\boldsymbol{D}_0 \in \mathbb{R}^{m\times m}$ are used and the sparse coefficients are generated according to Definition 2.



(a) $m = 25$ (b) $m = 30$

Fig. 1: Comparison between the necessary bounds in Theorem 2 and numerical simulations. The two numbers n_1 and n_2 in Theorem 2 are calculated using $\varepsilon = 0.01$ and depicted as solid and dashed lines, respectively. The circles are simulated minimum sampling ratios n/m that achieve >99% recovery rate in 100 trials (i.e. 100% recovery rate).

In the asymptotic regime, the necessary bounds in Theorem 2 can be highly simplified.

Corollary 1 (Asymptotic Necessary Bound of Theorem 2)**.** *With the same settings in Theorem 2, assume that* $m,k,n \to \infty$ *with* $\frac{k}{m} \to \bar{k} \in \mathbb{R}^+$ *and* $\frac{n}{m} \to \bar{n} \in \mathbb{R}^+$*. If the programming (6) uniquely recovers the dictionary with probability arbitrary close to 1, then* $\bar{n} > \frac{1}{1-\bar{k}}$.

In the same asymptotic regime in Corollary 1, the rough necessary bound in Theorem 1 meets the fine bound in Corollary 1. Suppose that $k_i = k$ and $m,k,n \to \infty$ with $\frac{k}{m} \to \bar{k} \in \mathbb{R}^+$ and $\frac{n}{m} \to \bar{n} \in \mathbb{R}^+$. Divide both sides of the necessary bound in Theorem 1 by m. Then one has $\bar{n} \geq 1 + \bar{n}\bar{k}$ which is equivalent to $\bar{n} \geq 1/\left(1-\bar{k}\right)$.

We note that there were attempts in the literature [25–27] to analyze the sufficient bound on the number of training samples for exact dictionary learning of complete dictionaries. Their algorithms suggest a sampling complexity of $O(m^2\log(m))$ for complete dictionary learning problem with $O(m)$ non-zeros per column.

4. INEXACT DICTIONARY RECOVERY

This section focuses on cases where exact dictionary recovery cannot be guaranteed. The first is the case when the dictionary is complete or under-complete but the sparsity pattern of $\boldsymbol{X}_0$ contains errors. This case is important for the overall dictionary learning as the sparse coding stage may not be perfect. The second is the case of over-complete dictionaries, i.e., the number of the dictionary atoms is larger than their dimension. This case is also important as typical applications of dictionary learning involve over-complete dictionaries.

4.1. Under-Complete/Complete Dictionary

When the given sparsity pattern is inexact, there may not exist $\boldsymbol{D}$ and $\boldsymbol{X}$ s.t. $\boldsymbol{Y} = \boldsymbol{D}\boldsymbol{X}$ and $\mathscr{P}_{\Omega^c}(\boldsymbol{X}) = \boldsymbol{0}$. Hence, instead of finding a sparse representation to exactly match data $\boldsymbol{Y}$, consider finding a sparse representation that approximates data $\boldsymbol{Y}$ in some optimal way. In particular, consider the following optimisation problem:

$$\min_{\tilde{\boldsymbol{Y}},\boldsymbol{D},\boldsymbol{X}} \left\|\boldsymbol{Y}-\tilde{\boldsymbol{Y}}\right\|_F^2, \text{ s.t. } \tilde{\boldsymbol{Y}} = \boldsymbol{D}\boldsymbol{X},\ \mathscr{P}_{\Omega^c}(\boldsymbol{X}) = \boldsymbol{0},$$

or equivalently,

$$\min_{\tilde{\boldsymbol{Y}},\boldsymbol{H},\boldsymbol{X}} \left\|\boldsymbol{Y}-\tilde{\boldsymbol{Y}}\right\|_F^2, \text{ s.t. } \boldsymbol{H}\tilde{\boldsymbol{Y}} = \boldsymbol{X},\ \mathscr{P}_{\Omega^c}(\boldsymbol{X}) = \boldsymbol{0}. \tag{8}$$

We design an iterative process to solve this problem sub-optimally based on the total least squares method [28]. With a slight abuse of notations, denote the sparse coefficients at the beginning of each iteration by $\boldsymbol{X}$, and let $\boldsymbol{Y} = \tilde{\boldsymbol{Y}} + \boldsymbol{\Delta}_Y$ and $\boldsymbol{X} = \tilde{\boldsymbol{X}} + \boldsymbol{\Delta}_X$. In each iteration, we replace the objective function (8) with

$$\min_{\tilde{\boldsymbol{Y}},\boldsymbol{H},\tilde{\boldsymbol{X}}} \left\|\left[\boldsymbol{\Delta}_Y^T,\boldsymbol{\Delta}_X^T\right]\right\|_F^2, \text{ s.t. } \boldsymbol{H}\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{X}},\ \mathscr{P}_{\Omega^c}\left(\tilde{\boldsymbol{X}}\right) = \boldsymbol{0}. \tag{9}$$

In (9), the equality $\boldsymbol{H}\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{X}}$ implies that the columns of $\tilde{\boldsymbol{Y}}$ span the same m-dimensional subspace spanned by the columns of $\tilde{\boldsymbol{X}}$. And this subspace can be approximated by singular value decomposition (SVD) of the matrix $\left[\boldsymbol{Y}^T,\boldsymbol{X}^T\right]$. More specifically, let the SVD of $\left[\boldsymbol{Y}^T,\boldsymbol{X}^T\right]$ be

$$\begin{bmatrix}\boldsymbol{Y}^T & \boldsymbol{X}^T\end{bmatrix} = \begin{bmatrix}\boldsymbol{U}_Y & \boldsymbol{U}_X\end{bmatrix}\begin{bmatrix}\boldsymbol{\Sigma}_Y & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Sigma}_X\end{bmatrix}\begin{bmatrix}\boldsymbol{V}_{YY} & \boldsymbol{V}_{YX}\\ \boldsymbol{V}_{XY} & \boldsymbol{V}_{XX}\end{bmatrix}^T.$$

It follows that

$$\begin{aligned}\tilde{\boldsymbol{Y}}^T &= \boldsymbol{U}_Y\boldsymbol{\Sigma}_Y\boldsymbol{V}_{YY}^T,\\ \tilde{\boldsymbol{X}}^T &= \boldsymbol{U}_Y\boldsymbol{\Sigma}_Y\boldsymbol{V}_{XY}^T,\\ \boldsymbol{H}^T &= \boldsymbol{V}_{YY}\boldsymbol{V}_{XY}^T.\end{aligned}$$

In other words, given $\boldsymbol{Y}$ and $\boldsymbol{X}$, the SVD gives the optimal solution of

$$\min_{\tilde{\boldsymbol{Y}},\boldsymbol{H},\tilde{\boldsymbol{X}}} \left\|\left[\boldsymbol{\Delta}_Y^T,\boldsymbol{\Delta}_X^T\right]\right\|_F^2, \text{ s.t. } \boldsymbol{H}\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{X}}.$$

However, the obtained $\tilde{\boldsymbol{X}}$ may not have the required sparsity pattern. To enforce that, we simply set $\boldsymbol{X}$ such that

$$\mathscr{P}_{\Omega}(\boldsymbol{X}) = \mathscr{P}_{\Omega}\left(\tilde{\boldsymbol{X}},\right) \text{ and } \mathscr{P}_{\Omega^c}(\boldsymbol{X}) = \boldsymbol{0}.$$

With this updated $\boldsymbol{X}$, we are able to go to the next iteration until the iterative process converges or a preset number of iterations is reached. The above procedure is not guaranteed to find the global minima of either (8) or (9). However, our simulations show that it achieves good performance in practice.

4.2. Over-complete Dictionary

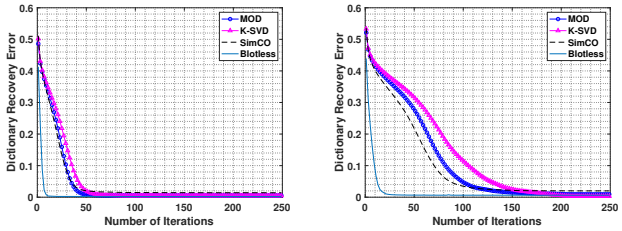
The above least squares approach cannot be directly applied to updating over-complete dictionaries. To observe that, consider $\boldsymbol{Y} = \boldsymbol{D}_0\boldsymbol{X}_0$. As $\boldsymbol{D}_0^\dagger\boldsymbol{D}_0 \neq \boldsymbol{I}$, it holds that $\boldsymbol{D}_0^\dagger\boldsymbol{Y} \neq \boldsymbol{X}_0$, so either (6) or (8) is not valid. To address this issue, a simple approach is to divide the whole dictionary into a set of sub-dictionaries each of which is either complete or under-complete, and then update these sub-dictionaries one-by-one whilst keeping all other sub-dictionaries and the corresponding coefficients fixed. We repeat this process for all sub-dictionaries and refer to this dictionary update algorithm as *Blotless*.

5. NUMERICAL TEST FOR DICTIONARY LEARNING

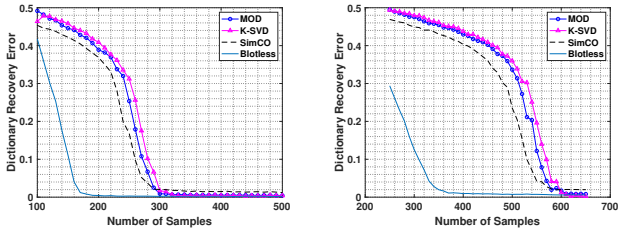
This section compares the performance of dictionary learning with different dictionary update methods. OMP [29] is adopted for sparse coding for simplicity. A direct comparison of dictionary update without involving sparse coding is omitted here due to the space constraint.

Numerical tests are conducted for both complete dictionary and over-complete dictionary learning using synthetic signals (simulations involving real data are omitted here due to space limitation). A Gaussian random matrix $\boldsymbol{D}_0 \in \mathbb{R}^{m\times l}$ is generated, and its column are normalized to a unit l_2 norm, to create a ground truth dictionary. Then, n data signals of dimension m are produced as columns of measurement matrix $\boldsymbol{Y}$. Each of the column vector is the linear combination of k random dictionary atoms in $\boldsymbol{D}_0$, and the corresponding coefficients are independently generated from the standard Gaussian distribution.

We define the mean dictionary recovery error using $R_{err} = \frac{1}{l}\sum_{p=1}^{l}\left(1-|\hat{\boldsymbol{d}}_p^T\boldsymbol{d}_{0_{j_p}}|\right)$, where $\hat{\boldsymbol{d}}_p$ is the p-th recovered atom and $j_p = \arg\max_{j\in J_p}\{|\hat{\boldsymbol{d}}_p^T\boldsymbol{d}_{0_j}| > 0.9\}, J_p = \{1,2,...,l\}\setminus\{j_1,j_2,...,j_{p-1}\}$, is the ground truth atom index which best matches $\hat{\boldsymbol{d}}_p$. This definition quantifies the difference between the learned dictionary and the ground truth dictionary used to generate training samples, taking possible permutations of dictionary atoms into consideration.



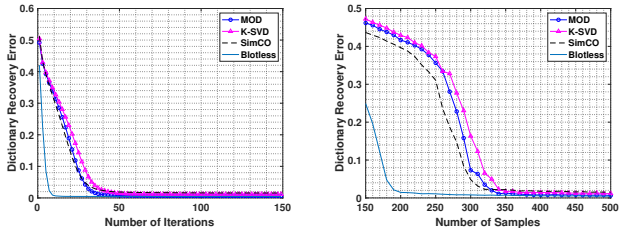
(a) $m = 64, k = 5, l = 64, n = 400$ (b) $m = 64, k = 5, l = 128, n = 600$



(c) $m = 64, k = 5, l = 64$, 150 iter. (d) $m = 64, k = 5, l = 128$, 150 iter.

Fig. 2: Comparison of different methods for the noise free case: results are averages of 100 trials

Numerical comparisons in the mean dictionary recovery error and convergence rate are given in Figures 2 and 3, corresponding to the noise free case and the noisy case, respectively. The benchmark algorithms that the Blotless is compared against are MOD, K-SVD and SimCO. The first row of Fig. 2 compares the convergence rate of different algorithms. Both complete and over-complete dictionaries are considered and the number of training samples are chosen to make sure that exact recovery is possible for all algorithms. Simulation results clearly indicate that Blotless converges much faster than the others. The second row of Fig. 2 compares the number of training samples needed for the exact dictionary recovery. The maximum number of iterations is chosen to be 150 according to the results in the first row of Fig. 2. Again Blotless is clearly the winner as it requires much fewer training samples for both complete and over-complete dictionary learning. Now we repeat the same numerical comparison in Fig. 3 for the noisy case by assuming that SNR=15dB, except that we only have the space in this conference paper to include the results for complete dictionary learning. Once again, Blotless is the clear winner in terms of both convergence rate and the number of training samples required for the exact recovery.



(a) $n = 500$ (b) number of iterations=150

Fig. 3: Comparison of methods for the noisy case: $m = 64, k = 5, l = 64$, SNR = 15dB, results are averages of 100 trials

Note that the minimum number of iterations does not necessarily mean the fastest run-time of an algorithm. Table 1 compares the run-time of different dictionary learning methods using Matlab 2018a on a Windows laptop with 8GB RAM and Core i5 7300HQ processor. The results presented in the table show that K-SVD runs faster when the sparsity level k is relatively small. This is not a surprise: K-SVD is based on SVD of matrices whose sizes depend on the sparsity levels of the rows of $\boldsymbol{X}$, while in Blotless the sizes of the matrices involved in SVD remain constants independent of the sparsity level. Hence, as results further show, Blotless runs the fastest as the number of nonzeros in the sparse coefficients increases.

Table 1: Comparison of the run-time (in seconds) of different dictionary learning methods (the noise free case). Results are shown with the same parameters $m = 64, l = 128, n = 800$ and the same stop criterion $R_{err} < 0.01$ (When $k = 8, 10$, SimCO converges without meeting the stop criterion).

	K-SVD	MOD	SimCO	Blotless
$k = 4$	**1.4973**	2.7961	26.8629	7.0811
$k = 6$	**4.3242**	10.5076	37.9067	13.4822
$k = 8$	51.4784	69.7387	$\sim$	**28.9088**
$k = 10$	132.7302	248.6604	$\sim$	**60.0417**

6. CONCLUSION

This paper focuses on solving the bilinear inverse problem of dictionary update with a given sparsity pattern. The main contributions include a least squares approach for the case where exact dictionary recovery is expected, necessary conditions for the exact recovery in terms of the minimum size of the sampling set, and a block total least squares (Blotless) approach for more practical cases where it is not clear whether exact dictionary recovery is possible. When embedding the new dictionary update process to dictionary learning, Blotless requires fewer training samples and converges faster.

7. REFERENCES

[1] Michael Elad and Michal Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[2] Licheng Liu, Long Chen, CL Philip Chen, Yuan Yan Tang, et al., "Weighted joint sparse representation for removing mixed noise in image," *IEEE transactions on cybernetics*, vol. 47, no. 3, pp. 600–611, 2017.

[3] Michael M Bronstein, Alexander M Bronstein, Michael Zibulevsky, and Yehoshua Y Zeevi, "Blind deconvolution of images using optimal sparse representations," *IEEE Transactions on Image Processing*, vol. 14, no. 6, pp. 726–736, 2005.

[4] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma, "Image super-resolution via sparse representation," *IEEE transactions on image processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[5] Qiqin Dai, Seunghwan Yoo, Armin Kappeler, and Aggelos K Katsaggelos, "Sparse representation-based multiple frame video super-resolution," *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 765–781, 2017.

[6] Bruno A Olshausen and David J Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607, 1996.

[7] Michal Aharon, Michael Elad, Alfred Bruckstein, et al., "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311, 2006.

[8] Kjersti Engan, Sven Ole Aase, and J Hakon Husoy, "Method of optimal directions for frame design," in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*. IEEE, 1999, vol. 5, pp. 2443–2446.

[9] Kenneth Kreutz-Delgado, Joseph F Murray, Bhaskar D Rao, Kjersti Engan, Te-Won Lee, and Terrence J Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural computation*, vol. 15, no. 2, pp. 349–396, 2003.

[10] Wei Dai, Tao Xu, and Wenwu Wang, "Simultaneous codeword optimization (simco) for dictionary update and learning," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6340–6353, 2012.

[11] Ivana Tosic and Pascal Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[12] Joel A Tropp and Anna C Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on information theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[13] Deanna Needell and Roman Vershynin, "Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit," *Foundations of computational mathematics*, vol. 9, no. 3, pp. 317–334, 2009.

[14] Wei Dai and Olgica Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE transactions on Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.

[15] Scott Shaobing Chen, David L Donoho, and Michael A Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[16] Amir Beck and Marc Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[17] Joel A Tropp and Stephen J Wright, "Computational methods for sparse solution of linear inverse problems," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 948–958, 2010.

[18] Zheng Zhang, Yong Xu, Jian Yang, Xuelong Li, and David Zhang, "A survey of sparse representation: algorithms and applications," *IEEE access*, vol. 3, pp. 490–530, 2015.

[19] Sven Ole Aase, John Hakon Husoy, JHHK Skretting, and Kjersti Engan, "Optimized signal expansions for sparse representation," *IEEE Transactions on Signal Processing*, vol. 49, no. 5, pp. 1087–1096, 2001.

[20] Karl Skretting, Kjersti Engan, JH Husoy, and Sven Ole Aase, "Sparse representation of images using overlapping frames," in *proceedings of the scandinavian conference on image analysis*, 2001, pp. 613–620.

[21] Kjersti Engan, Karl Skretting, and John Håkon Husøy, "Family of iterative ls-based dictionary learning algorithms, ils-dla, for sparse signal representation," *Digital Signal Processing*, vol. 17, no. 1, pp. 32–49, 2007.

[22] Leslie N Smith and Michael Elad, "Improving dictionary learning: Multiple dictionary updates and coefficient reuse," *IEEE Signal Processing Letters*, vol. 20, no. 1, pp. 79–82, 2013.

[23] Rémi Gribonval, Gilles Chardon, and Laurent Daudet, "Blind calibration for compressed sensing by convex optimization," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2713–2716.

[24] Shuyang Ling and Thomas Strohmer, "Self-calibration and bilinear inverse problems via linear least squares," *SIAM Journal on Imaging Sciences*, vol. 11, no. 1, pp. 252–292, 2018.

[25] Ju Sun, Qing Qu, and John Wright, "Complete dictionary recovery over the sphere i: Overview and the geometric picture," *IEEE Transactions on Information Theory*, vol. 63, no. 2, pp. 853–884, 2017.

[26] Ju Sun, Qing Qu, and John Wright, "Complete dictionary recovery over the sphere ii: Recovery by riemannian trust-region method," *IEEE Transactions on Information Theory*, vol. 63, no. 2, pp. 885–914, 2017.

[27] Niladri Chatterji and Peter L Bartlett, "Alternating minimization for dictionary learning with random initialization," in *Advances in Neural Information Processing Systems*, 2017, pp. 1994–2003.

[28] Ivan Markovsky and Sabine Van Huffel, "Overview of total least-squares methods," *Signal processing*, vol. 87, no. 10, pp. 2283–2302, 2007.

[29] Ron Rubinstein, "Software," `http://www.cs.technion.ac.il/~ronrubin/software.html`.