Dictionary Learning for Sparse Representation: A Novel Approach

Mostafa Sadeghi*, Massoud Babaie-Zadeh, *Senior Member, IEEE*, Christian Jutten, *Fellow, IEEE*

***Abstract*—A dictionary learning problem is a matrix factorization in which the goal is to factorize a training data matrix, $\mathbf{Y}$, as the product of a dictionary, $\mathbf{D}$, and a sparse coefficient matrix, $\mathbf{X}$, as follows, $\mathbf{Y} \simeq \mathbf{D}\mathbf{X}$. Current dictionary learning algorithms minimize the representation error subject to a constraint on $\mathbf{D}$ (usually having unit column-norms) and sparseness of $\mathbf{X}$. The resulting problem is not convex with respect to the pair $(\mathbf{D}, \mathbf{X})$. In this letter, we derive a first order series expansion formula for the factorization, $\mathbf{D}\mathbf{X}$. The resulting objective function is jointly convex with respect to $\mathbf{D}$ and $\mathbf{X}$. We simply solve the resulting problem using alternating minimization and apply some of the previously suggested algorithms onto our new problem. Simulation results on recovery of a known dictionary and dictionary learning for natural image patches show that our new problem considerably improves performance with a little additional computational load.**

***Index Terms*—Dictionary learning, sparse representation, K-SVD, MOD.**

I. Introduction

Sparse and redundant representation modeling has been shown to be a powerful and efficient tool for signal analysis and processing [1]. The goal is to represent a given signal as a linear combination of some given basis functions in such a way that most of the representation's coefficients be equal to zero or have a small magnitude. More precisely, consider the signal $\mathbf{y} \in \mathbb{R}^n$ and the basis functions $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_K] \in \mathbb{R}^{n \times K}$. In this context, $\mathbf{D}$ is called a *dictionary* and each of its columns is called an *atom*. It is typically assumed that the dictionary is overcomplete, i.e. $K > n$. A sparse coding algorithm then seeks the sparsest representation, $\mathbf{x} \in \mathbb{R}^K$, such that $\mathbf{y} \simeq \mathbf{D}\mathbf{x}$. This model has received a lot of attention during the last decade, and a lot of work has been done to theoretically and experimentally investigate the efficiency of this model in different signal processing areas [1].

One crucial problem in a sparse representation-based application is how to choose the dictionary. There are many pre-specified dictionaries, e.g. Fourier, Gabor, Discrete Cosine Transform (DCT), and wavelet [2]. Though being simple and having fast computations, these *non-adaptive* dictionaries are not able to efficiently (sparsely) represent a given class of signals.

To address this problem, *dictionary learning* has been widely investigated during the last decade [2], [3]. In this approach, a dictionary is *learned* from some training signals belonging to the signal class of interest. It has been experimentally shown that these adaptive dictionaries outperform the non-adaptive ones in many signal processing applications, e.g. image compression and enhancement, and classification tasks [1], [4].

A dictionary learning algorithm uses a training data matrix, $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^L$, containing L signals from the particular class of signals at hand, and finds a dictionary, $\mathbf{D}$, in such a way that all training signals have a sufficiently sparse representation in it. More precisely, a typical dictionary learning algorithm solves the following problem:

$$\min_{\mathbf{D} \in \mathcal{D}, \mathbf{X} \in \mathcal{X}} \ \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2, \tag{1}$$

where $\|.\|_F$ is the Frobenius norm, and $\mathcal{D}$ and $\mathcal{X}$ are admissible sets of the dictionary and the coefficient matrix, respectively. $\mathcal{D}$ is usually defined as the set of all dictionaries with unit column-norms. $\mathcal{X}$ constrains the coefficient matrix to have sparse columns.

Note that the above problem is not convex with respect to the pair $(\mathbf{D}, \mathbf{X})$. Most dictionary learning algorithms attack this problem by iteratively performing a two-stage procedure: Starting with an initial dictionary, the following two stages are repeated several times,

1) Sparse representation:

$$\mathbf{X}^{(k+1)} = \underset{\mathbf{X} \in \mathcal{X}}{\operatorname{argmin}} \ \|\mathbf{Y} - \mathbf{D}^{(k)}\mathbf{X}\|_F^2, \tag{2}$$

2) Dictionary update:

$$\mathbf{D}^{(k+1)} = \underset{\mathbf{D} \in \mathcal{D}}{\operatorname{argmin}} \ \|\mathbf{Y} - \mathbf{D}\mathbf{X}^{(k+1)}\|_F^2. \tag{3}$$

Stage 1 is simply an ordinary sparse coding problem, in which the sparse representations of all training signals are computed using the current dictionary. Many sparse coding algorithms have been proposed that can be used to perform this stage [5]. The main difference between many dictionary learning algorithms is stage 2, in which the dictionary is updated to reduce the representation error of stage 1.

Method of Optimal Directions (MOD) [6] is one of the simplest dictionary learning algorithms which firstly finds the unconstrained minimum of $\|\mathbf{Y} - \mathbf{D}\mathbf{X}^{(k+1)}\|_F^2$ and then projects the solution onto the set $\mathcal{D}$. This leads to the following

M. Sadeghi and M. Babaie-Zadeh are with the Electrical Engineering Department, Sharif University of Technology, Tehran, Iran (e-mail: m.saadeghii@gmail.com; mbzadeh@yahoo.com).

C. Jutten is with the GIPSA-Lab, Department of Images and Signals, University of Grenoble and Institut Universitaire de France, France (e-mail: Christian.Jutten@inpg.fr).

This work has been partially funded by Iran National Science Foundation (INSF) under contract number 91004600, and EU project 2012-ERC-AdG-320684 CHESS.

closed-form expression[1]:

$$\mathbf{D}^{(k+1)} = \mathbf{Y}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}, \tag{4}$$

followed by normalizing the columns of D.

K-Singular Value Decomposition (K-SVD) [7] is another well-known algorithm, which has been very successful. In its dictionary update stage, only one atom is updated at a time. Moreover, while updating each atom, the non-zero entries in the associated row vector of $\mathbf{X}$ are also updated. This leads to a matrix rank-1 approximation problem which is then solved via performing a Singular Value Decomposition (SVD) operation.

In [8] the idea of fixing the support of $\mathbf{X}$ and updating its non-zero entries, along with atoms updating, has been extended to a more general case in which more than one atom along with the non-zero entries in their associated row vectors in $\mathbf{X}$ are updated at a time. In a similar work, [9] has derived an MOD-like algorithm that uses this idea. More precisely, the following problem has been proposed to be solved at stage 2 (see (3)):

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{X}} \ \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 \text{ subject to } \mathcal{S}(\mathbf{X}) = \mathcal{S}(\mathbf{X}^{(k+1)}), \tag{5}$$

where $\mathcal{S}(\mathbf{X})$ denotes the support of $\mathbf{X}$, i.e. the positions of its non-zero entries. To solve this problem, [9] proposed to use alternating minimization over $\mathbf{D}$ and $\mathbf{X}$. Minimizing (5) over $\mathbf{D}$ with a fixed $\mathbf{X}$ results in (4). Minimization of (5) over $\mathbf{X}$ with a fixed $\mathbf{D}$ decouples for each column of $\mathbf{X}$ and results in the following problems:

$$\forall i: \ \mathbf{x}_i = \underset{\mathbf{x}}{\operatorname{argmin}} \ \|\mathbf{y}_i - \mathbf{D}\mathbf{x}\|_2^2 \text{ subject to } \mathcal{S}(\mathbf{x}) = \mathcal{S}(\mathbf{x}_i^{(k+1)}). \tag{6}$$

By defining $\omega_i = \{j : \ \mathbf{x}_i(j) \neq 0\}$, (6) leads to the following solutions:

$$\forall i: \ \mathbf{x}_i(\omega_i) \leftarrow (\mathbf{D}_i^T\mathbf{D}_i)^{-1}\mathbf{D}_i^T\mathbf{y}_i, \tag{7}$$

where $\mathbf{D}_i$ consists of those columns of $\mathbf{D}$ that have been used in the representation of $\mathbf{y}_i$. Performing a few (e.g. 3) alternations between (4) and (7) gives a good result [9]. We henceforth refer to this algorithm as the *Multiple Dictionary Update* (MDU) algorithm.

In [10] a sequential algorithm, named as Sequential Generalization of K-means (SGK), has been proposed. This algorithm updates atoms of the dictionary sequentially, but unlike K-SVD and MDU, keeps the non-zero entries of the coefficient matrix intact. As explained in [10], "though K-SVD is sequential like K-means, it fails to simplify to K-means by destroying the structure in the sparse coefficients". This is due to performing SVD in K-SVD, which (unlike K-means) forces the atom-norms to be 1 and that the resulting coefficients are not necessarily 0 or 1 [10]. These problems, however, do not exist in SGK [10].

In this letter, we derive a new method for dictionary learning. The idea is to use a first order series expansion instead of the term $\mathbf{D}\mathbf{X}$. In this way, we obtain a new objective function that unlike the commonly used one, i.e. (1), is jointly convex with respect to $\mathbf{D}$ and $\mathbf{X}$. We simply solve the resulting problem using alternating minimization. We then apply MOD, MDU, and SGK onto our new problem. Experimental results on both synthetic and real data show that using our new problem results in a considerable improvement over the previous one, i.e. (1), with a little additional computational load.

The rest of the paper is organized as follows. In Section II we describe our proposed method in details. Then Section III presents the results of our simulations.

II. The proposed method

In this section, we derive a first order series expansion for the matrix-valued function $F(\mathbf{D},\mathbf{X}) = \mathbf{D}\mathbf{X}$ about a point $(\mathbf{D}_0,\mathbf{X}_0)$, and using it, we obtain a new dictionary learning problem. We then apply some of the previously suggested algorithms onto our new problem.

A. The new problem

Let write $\mathbf{D}$ and $\mathbf{X}$ as follows:

$$\begin{cases} \mathbf{D} = \mathbf{D}_0 + (\mathbf{D}-\mathbf{D}_0) \\ \mathbf{X} = \mathbf{X}_0 + (\mathbf{X}-\mathbf{X}_0) \end{cases}, \tag{8}$$

where $(\mathbf{D}-\mathbf{D}_0)$ and $(\mathbf{X}-\mathbf{X}_0)$ are small in the sense of Frobenius norm. We then substitute the above expressions into the function $F(\mathbf{D},\mathbf{X})$. Doing so we derive,

$$\begin{aligned} F(\mathbf{D},\mathbf{X}) = \mathbf{D}_0\mathbf{X}_0 + \mathbf{D}_0(\mathbf{X}-\mathbf{X}_0) + (\mathbf{D}-\mathbf{D}_0)\mathbf{X}_0 \\ + (\mathbf{D}-\mathbf{D}_0)(\mathbf{X}-\mathbf{X}_0) \end{aligned} \tag{9}$$

Neglecting the last term, whose Frobenius norm is upper-bounded by a small value[2], we obtain the following first order approximation for $F(\mathbf{D},\mathbf{X})$:

$$\tilde{F}(\mathbf{D},\mathbf{X}) = \mathbf{D}\mathbf{X}_0 + \mathbf{D}_0\mathbf{X} - \mathbf{D}_0\mathbf{X}_0. \tag{10}$$

Now, we use the above approximation instead of $\mathbf{D}\mathbf{X}$ in (1). We then derive the following new dictionary learning problem:

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{X}\in\mathcal{X}} \ \|\mathbf{Y}+\mathbf{D}_0\mathbf{X}_0-\mathbf{D}\mathbf{X}_0-\mathbf{D}_0\mathbf{X}\|_F^2. \tag{11}$$

Note that unlike (1), the objective function of the above problem is jointly convex with respect to $\mathbf{D}$ and $\mathbf{X}$.

In order for (11) to be a convex problem, in addition to its objective function, the constraint sets have to be convex, too. An example of such convex constraint sets would be $\mathcal{D} = \{\mathbf{D} : \ \forall i, \|\mathbf{d}_i\|_2^2 \leq 1\}$ and $\mathcal{X} = \{\mathbf{X} : \ \forall i, \|\mathbf{x}_i\|_1 \leq \tau\}$. To make sure that the approximation used in (9) remains valid, one may add the term $\lambda_1\|\mathbf{D}-\mathbf{D}_0\|_F^2 + \lambda_2\|\mathbf{X}-\mathbf{X}_0\|_F^2$ to the objective function of (11).

In this paper, to solve (11), we simply use alternating minimization. Moreover, at each alternation, we use the updated versions of $\mathbf{D}$ and $\mathbf{X}$ found at the previous alternation instead of $\mathbf{D}_0$ and $\mathbf{X}_0$. In other words, our problem becomes as follows[3]:

$$\left\{\mathbf{D}^{(k+1)},\mathbf{X}^{(k+1)}\right\} =$$

[1] We have dropped the superscript of $\mathbf{X}^{(k+1)}$ for simplicity.

[2] According to the submultiplicativity property of the Frobenius norm [11], we have $\|(\mathbf{D}-\mathbf{D}_0)(\mathbf{X}-\mathbf{X}_0)\|_F \leq \|\mathbf{D}-\mathbf{D}_0\|_F\|\mathbf{X}-\mathbf{X}_0\|_F$.

[3] Note the similarity of (12) and Newton's algorithm for minimization (neglecting the constraints): The cost function has been approximated by a quadratic term at the vicinity of the previous iteration.

$$\underset{\mathbf{D}\in\mathcal{D},\mathbf{X}\in\mathcal{X}}{\operatorname{argmin}} \quad \|\mathbf{Y}+\mathbf{D}^{(k)}\mathbf{X}^{(k)}-\mathbf{D}\mathbf{X}^{(k)}-\mathbf{D}^{(k)}\mathbf{X}\|_F^2. \qquad (12)$$

In order to minimize (12) over $\mathbf{X}$, we set $\mathbf{D}=\mathbf{D}^{(k)}$ in the objective function. In this way, (12) reduces to the stage 1 of the general dictionary learning problem, i.e. (2). Thus, our algorithm like most dictionary learning algorithms does not affect the sparse representation stage and any sparse coding algorithm can be used to perform this stage.

Stage 2, after substitution of $\mathbf{X} \leftarrow \mathbf{X}^{(k+1)}$ and setting $\mathbf{Z}=\mathbf{Y}+\mathbf{D}^{(k)}\mathbf{X}^{(k)}-\mathbf{D}^{(k)}\mathbf{X}^{(k+1)}$, reduces to the following problem:

$$\mathbf{D}^{(k+1)}=\underset{\mathbf{D}\in\mathcal{D}}{\operatorname{argmin}} \quad \|\mathbf{Z}-\mathbf{D}\mathbf{X}^{(k)}\|_F^2. \qquad (13)$$

B. The new MOD, MDU, and SGK

In what follows, we apply MOD, MDU, and SGK algorithms onto the above problem.

Solving (13) using MOD results in the following update formula for $\mathbf{D}$, in which we have dropped the superscript of $\mathbf{X}^{(k)}$ for simplicity:

$$\mathbf{D}^{(k+1)}=\mathbf{Z}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}, \qquad (14)$$

followed by normalizing the columns of $\mathbf{D}^{(k+1)}$.

To solve (13) using the MDU method, the dictionary update formula is exactly (14) but the update formula for the non-zero entries of $\mathbf{X}$ remains[4] as (7).

To apply the SGK method, problem (13) has to be solved sequentially for each column of $\mathbf{D}$. To update the ith column, $\mathbf{d}_i$, the following problem has to be solved:

$$\mathbf{d}_i^{(k+1)}=\underset{\mathbf{d}}{\operatorname{argmin}} \quad \|\mathbf{E}_i-\mathbf{d}\mathbf{x}_{[i]}^{(k)}\|_F^2 \text{ subject to } \|\mathbf{d}\|_2=1, \quad (15)$$

where $\mathbf{E}_i=\mathbf{Z}-\sum_{j\neq i}\mathbf{d}_j\mathbf{x}_{[j]}$ is the error matrix when $\mathbf{d}_i$ is removed, and $\mathbf{x}_{[i]}$ denotes the ith row of $\mathbf{X}$. Problem (15) results in

$$\mathbf{d}_i^{(k+1)}=\mathbf{E}_i(\mathbf{x}_{[i]}^{(k)})^T \qquad (16)$$

followed by a normalization. Note that in order to update each atom, the updated versions of other atoms are used to compute its associated error matrix.

III. Simulations

We compare the performance of our proposed problem and the previous one by performing two sets of experiments. The first experiment is the recovery of a known dictionary. The second experiment is on real data where the goal is to learn an overcomplete dictionary for natural image patches. For all algorithms, Orthogonal Matching Pursuit (OMP) [12] has been used as the sparse coding algorithm[5].

Our simulations were performed in MATLAB R2010b environment on a system with 2.13 GHz CPU and 2 GB RAM, under Microsoft Windows 7 operating system. As a rough measure of complexity, we will mention the run times of the algorithms.

A. Recovery of a known dictionary

Similar to [7], [10] we generated a random dictionary of size 20×50, with zero mean and unit variance independent and identically distributed (i.i.d.) Gaussian entries, followed by a column normalization. We then generated a collection of 1500 training signals, each as a linear combination of $s=3,4,5$ different atoms, with i.i.d. coefficients. White Gaussian noise with Signal to Noise Ratio (SNR) levels of 10, 20, 30, and 100 dB were added to these signals. For all algorithms, the exact value of s was given to OMP. Similar to [10], number of alternations between the two dictionary learning stages was set according to $\simeq 5s^2$. We applied all algorithms onto these noisy training signals, and compared the resulting recovered dictionaries to the generating dictionary in the same way as in [7]. It should be mentioned that as we saw in our simulations, using $\mathbf{X}^{(k+1)}$, the most recent update of $\mathbf{X}$, in (13) instead of $\mathbf{X}^{(k)}$ results in a better performance for this experiment. So, we used this alternative.

The final percentage of successful recovery (averaged over 30 trials), is shown in Table I (only the results of MOD, MDU, and New MOD have been reported here). To see the convergence behaviour of the algorithms, the successful recovery rate versus alternation number, for SNR = 30 dB, is shown in Fig. 1. The average running times of the algorithms are also shown in Table II.

With these results in mind, we conclude that our proposed problem results in much better convergence rate with only a little increase in the running time.

B. Dictionary learning for natural image patches

Similar to [9], we used a collection of seventeen well-known standard images, including Barbara, Cameraman, Jetplane, Lena, Mandril, and Peppers. A collection of 25,000, 8×8 patches from these images were extracted, 20,000 of which were used for training and the remaining 5,000 were used to test the reconstruction accuracy of the trained dictionary. The mean were subtracted from all image patches. These image patches were converted to column vectors of dimension $n=64$. Number of atoms in the dictionary was set to $K=3\times 64$ and $s=round(n/10)$ atoms were used to approximate each patch. As in [9], the dictionary was initialized with samples from the training signals. Root Mean Square Error (RMSE), defined as $\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F/(n\cdot L)$, was used to evaluate the reconstruction performance of the trained dictionaries.

The representation's RMSEs versus alternation number, for training and testing data are shown in Fig. 2. The average running times, with those of our proposed problem in parenthesise, are as follows, MOD: 219.80 (223.68), MDU: 564.90 (577.57), and SGK: 781.88 (794.75) seconds.

These results again emphasize on the advantage of our new problem over the previous one. This is very noticeable for "New SGK" that has achieved the best performance.

IV. Conclusion

In this letter we introduced a new problem for dictionary learning. Our idea is to use a first order series expansion

[4] Note that we must use $\mathbf{y}_i$'s in (7) not $\mathbf{z}_i$'s. This is because the coefficient matrix of "$\mathbf{Y}$" has been derived in the sparse representation stage not $\mathbf{Z}$.

[5] For OMP, we have used the OMP-Box v10 available at http://www.cs.technion.ac.il/~ronrubin/software.html. For the simulation performed in Subsection III-B we have used the complementary materials of [9] available at http://www.ieeexplore.ieee.org.

TABLE I
PERCENTAGE OF SUCCESSFUL RECOVERY.

SNR (dB)	Algorithm	$s=3$	$s=4$	$s=5$
10	MOD	85.40	77.27	7.73
	MDU	87.93	77.80	23.60
	New MOD	89.40	83.73	28.33
20	MOD	91.47	91.80	87.87
	MDU	91.93	90.40	91.73
	New MOD	94.07	94.13	93.07
30	MOD	88.87	92.20	91.60
	MDU	92.07	90.87	92.73
	New MOD	92.67	93.60	95.53
100	MOD	90.60	91.67	90.93
	MDU	91.73	93.00	93.27
	New MOD	94.00	94.87	93.80

TABLE II
AVERAGE RUNNING TIMES (IN SECOND). THOSE OF OUR PROPOSED PROBLEM ARE REPORTED IN PARENTHESES.

Algorithm	$s=3$	$s=4$	$s=5$
MOD	**2.12** (2.16)	**5.42** (5.52)	**10.31** (10.51)
MDU	**19.47** (19.93)	**41.40** (42.08)	**70.32** (71.61)
SGK	**3.83** (3.88)	**8.91** (9.00)	**15.73** (15.82)

instead of the dictionary-coefficient matrix product. We then solved the resulting problem using a simple alternating minimization algorithm. We experimentally showed that our proposed method considerably outperforms the previous one with a little additional cost. Applying other previously suggested dictionary learning algorithms to our proposed problem remains as our future works.

REFERENCES

[1] M. Elad, *Sparse and Redundant Representations*, Springer, 2010.

[2] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.

[3] I. Tosic and P. Frossard, "Dictionary learning: What is the right representation for my signal?," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[4] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 791–804, 2012.

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

[6] K. Engan, S. O. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," in *Proceedings of IEEE ICASSP*, 1999, vol. 5.

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

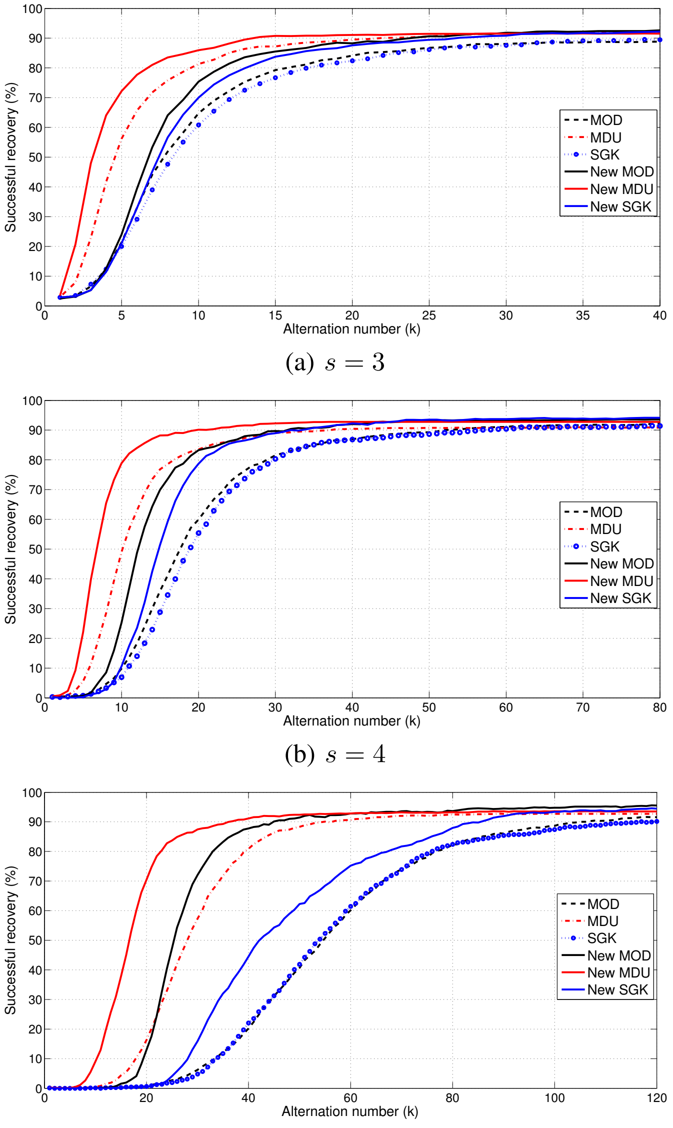
[8] W. Dai, T. Xu, and W. Wang, "Simultaneous codeword optimization (SimCO) for dictionary update and learning," *IEEE Trans. on Signal Proc.*, vol. 60, no. 12, pp. 6340–6353, 2012.

[9] L. N. Smith and M. Elad, "Improving dictionary learning: Multiple dictionary updates and coefficient reuse," *IEEE Signal Proc. Letters*, vol. 20, no. 1, pp. 79–82, 2013.

[10] S. K. Sahoo and A. Makur, "Dictionary training for sparse representation as generalization of K-Means clustering," *IEEE Signal Proc. Letters*, vol. 20, no. 6, pp. 587–590, 2013.

[11] G. H. Golub and C. F. Van Loan, *Matrix computations (3rd ed.)*, Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[12] J. A. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Info. Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.



(a) $s=3$

(b) $s=4$

(c) $s=5$

Fig. 1. Percentage of successful recovery versus alternation number for all algorithms at SNR = 30 dB.

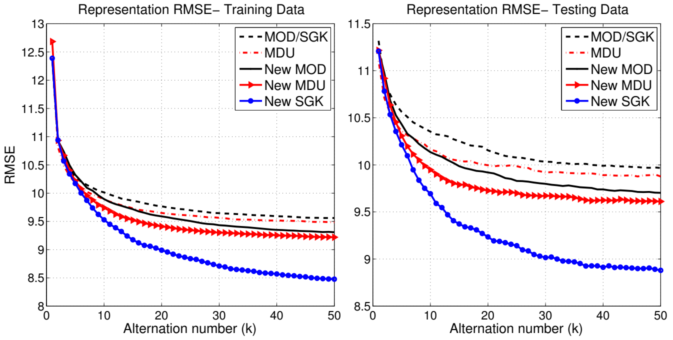


Fig. 2. RMSEs of the representations versus alternation number for training (left) and testing (right) data.